Learning Temporal Sentiment from Business News

A Computational Approach

Xiubo Zhang

A dissertation submitted to the University of Dublin
for the degree of Doctor of Philosophy

School of Computer Science and Statistics
Trinity College Dublin
Ireland, Republic of
February, 2016

Declaration

I declare that this thesis has not been submitted as an exercise for a degree at this or any other university and it is entirely my own work.
I agree to deposit this thesis in the University's open access institutional repository or allow the Library to do so on my behalf, subject to Irish Copyright Legislation and Trinity College Library conditions of use and acknowledgement.

Signed,

Abstract

This thesis concerns is about the analysis of textual sentiment using computational approaches. Textual sentiment reflects the opinions and attitudes expressed through textual communications. With the advent of the Internet and World Wide Web, individuals are being exposed to an ever-growing amount of sentiment through various channels; such information can have powerful effect on how people make decisions. Because of this, computational approaches to sentiment analysis has attracted increasing interest from researchers in the past two decades. The goal of sentiment analysis is to determine whether a piece of text conveys a positive or negative evaluation on a certain topic. Being able to identify and comprehend sentiment expressed by the public at a large scale can be very useful to many applications. One such application is to use sentiment conveyed in business news to explain the behaviours of financial markets. In this thesis, I have proposed an extended model for describing the interactions between news sentiment and financial markets. This extended model recognises two different types of news sentiment: (i) the retrospective news sentiment, which refers to the sentiment that is associated with news stories that recount past events in the market; and (ii) the prospective news sentiment, which refers to the textual sentiment in business and financial news that is associated with speculations and projections about the future developments of the market. A method that can automatically extract sentiment-laden language patterns for the temporal sentiment classes was developed. The method models the texts in news articles as a mixture of lexical items distributions, where each distribution characterises one of the temporal sentiment classes. A supervised maximisation expectation algorithm was derived to infer these distributions given a corpus and a market performance measure as the target. The supervision was enforced by implementing different conditional probability distributions for returns given the underlying temporal sentiment class of the news article. The method was implemented by a system comprising two components, CiCui and TSMiner, which was used to conduct a case study on firm-level data. The results of the study suggested it is mainly changes in the market that leads to changes in the news sentiment, and retrospective news sentiment is in general more prevalent in business news. Also noted was that word dependencies seem to be able to capture prospective sentiment than unigrams. It has been shown that the method proposed in this thesis slightly outperforms content analysis methods used with either the GI or Loughran and McDonald's sentiment dictionary.

Summary

The work presented in this thesis contributed to the body of knowledge in the following areas:

1. Introduced the *notion* of *temporality* in the modelling of sentiment in business news. It was proposed in this thesis that news sentiment can be categorised into two classes: *retrospective news sentiment* and *prospective news sentiment*. *Retrospective news sentiment* reflects the general attitude in the news towards what had already happened in the market, whereas *prospective news sentiment* reflects the news' general attitude towards the potential future development of the market. These two types of news sentiment are collectively referred to as *temporal news sentiment* in this thesis.

2. Developed a *method* that automatically compiles a sentiment dictionary that captures the *temporal sentiment orientations* of lexical items from a corpus comprising arbitrary business news articles about a certain company, provided that some measure of the company's performance is supplied (e.g. the stock prices of the company).

3. Developed a *strategy* for evaluating the correlational performances of news sentiment extraction procedures in a cross-validation setting using meta-analysis techniques.

4. Created two *systems* that implemented the method and the strategy developed thereof:

   - a text analysis system called CiCui, which builds a *positional inverted index* for a corpus comprised of raw text documents;
   - a system that takes the index built by CiCui as input and learns the lexical item distributions for the temporal sentiment classes using a supervised expectation maximisation algorithm.

5. Conducted a *case study* that:

   (a) demonstrated the effectiveness of the method developed in this thesis in extracting sentiment orientations of the lexical items;

(b) assessed the impact of the introduction of sentiment temporality on the predictive performances of sentiment models;

(c) compared two lexical structures, the *unigrams* and the *word dependencies*, in terms of their effectiveness in representing temporal news sentiment;

(d) compared the performances of the news sentiment extracted using the method developed in this thesis with that extracted using content analysis methods.

6. The *main findings* of this thesis are:

(a) There is strong evidence that it were the changes in the market that led to the changes of sentiment in news.

(b) In general, the usage of language patterns associated with retrospective sentiment is more prevalent than that associated with prospective sentiment.

(c) Introducing temporality into the modelling of news sentiment did not confer significant improvements to the method's ability to predict firm-level stock price movements using news sentiment.

(d) Models trained with unigrams as features outperformed models trained with unigrams as features when predicting future firm-level stock price movements using sentiment found in business and financial news.

(e) Prospective sentiment orientations learnt for word dependencies are easier to verify compared to those learnt for unigrams.

(f) The sentiment models produced by the method developed in this thesis outperformed content-analysis-based methods used with either the General Inquirer or the Loughran and McDonald's sentiment dictionary when predicting firm-level stock price movements using sentiment found in business and financial news.

Acknowledgement

First of all, I would like to express my gratitude and special thanks to my supervisor Professor Khurshid Ahmad. Without his support and guidance, I would never have finished this thesis. I want to thank all my collogues — Dr. Stephen Kelly, Zeyan Zhao, Shane Finan, Dr. Daniel Isemann, Jason Cook, Dr. Aaron Gerow, Barry Redmond, and all my friends and families for their continued support throughout the course of my study.

Lastly, I am grateful to the financial support provided by Trinity College Dublin, Enterprise Ireland grant #CC-2011-2601-B: GRCTC, and EU FP7 grant #607691: Slandail.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Sentiment analysis is a topic that has attracted increasing research interest over the past decades. The Oxford Dictionary defines *sentiment analysis* as 'the process of computationally identifying and categorising opinions expressed in a piece of text, especially in order to determine whether the writer's attitude towards a particular topic, product, etc. is positive, negative, or neutral'[1]. It is an interdisciplinary field of study where techniques from computational linguistics are brought to analyse the expression and understanding of sentiment in written form, which is a process of psychological nature.

The analysis of sentiment in textual material pre-dated the advent of modern computers. the method of *content analysis* developed in the early 20th century, for example, was used to quantify attitude expressed in texts [Krippendorff, 2013, pg. 7]; though it was the recent advancements in computing mechanisms that had made it possible for the practice to be applied to large amounts of texts. Developments in computational theories such as natural language processing and approximate Bayesian inference have enabled computing systems to exploit complex and unstructured information in raw texts that were previously understandable only by humans. The insights gained through large-scale sentiment analyses on the ever increasing amount of information on the Web have powered various applications, ranging from the summarisation of product and service reviews [Pang et al., 2002, Turney, 2002, Turney and Littman, 2003, Gamon, 2004, Cui et al., 2006, e.g.] to the prediction of political polls [Kermanidis and Maragoudakis, 2013, Rill et al., 2014, e.g.].

Using news sentiment to explain price movements in financial markets is one such application that has gained much attention in recent years. The basic premise of the practice is that investors, being not fully rational, can have their decisions influenced

[1] `http://www.oxforddictionaries.com/definition/english/sentiment-analysis`, accessed in February, 2016.

by *investor sentiment*, which represents the general attitudes of investors towards the future development of the market. It follows that at least part of the price movements in financial markets could be attributed to this investor sentiment [Baker and Wurgler, 2006]. However, investor sentiment itself is a quantity that is quite difficult to measure directly.[2] It was nevertheless postulated that *news sentiment* — that is, the sentiment conveyed in the text of news articles (sometimes also referred to as the *tone* of the news) — could serve as a proxy to the intangible investor sentiment: optimistic news announcements may correlate with *positive* investor sentiment, while pessimism in news announcements may coincide with an increase of *negative* sentiment among the investors. The hope is that by applying sentiment analysis techniques on large volumes of business and financial news, one will be able to automatically extract and quantify news sentiment, which, as a proxy to the hidden investor sentiment, can in turn be used to explain the behaviours of the market (Figure 1.1).

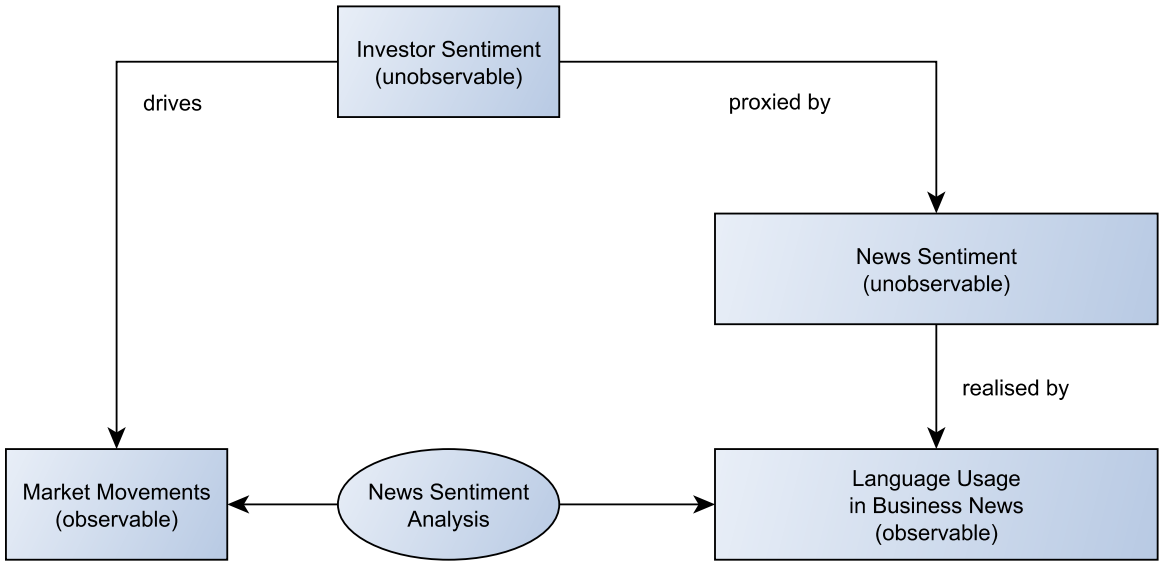


Figure 1.1: A schematic diagram illustrating the interactions between investor sentiment, news sentiment, and market movements as assumed in this thesis

The interactions between these variables in the real world is of course much more complicated; for example, there would exist a feedback link from news sentiment to investor sentiment, reflecting the fact that investors' beliefs can be influenced by current news sentiment as well. However, discussions on such effects are beyond the scope of this thesis, but may warrant further investigations.

A substantial body of literature has addressed the dynamics between news sentiment and the movements of the financial markets. Many studies from the finance literature used content analysis to gauge textual sentiment in the news, which mainly involves (i) looking up the sentiment connotations of individual words in a document from a

[2]Investor sentiment could be measured directly by sending out questionnaires to investors asking their opinions about the market.

sentiment dictionary such as the *General Inquirer*, (ii) aggregating the sentiment of the individual words into the daily level, and (iii) using the aggregated sentiment series as explanatory variables to regress against the target response, e.g. the returns of a stock or a market index. Studies from the computational linguistics discipline tend to take a different approach by employing statistical machine learning techniques: instead of consulting a dictionary for the sentiment orientations of the words in the text, an algorithm is developed to learn from historical data the impact each word's occurrences has on the future movements of the market. This approach essentially formulates the analysis of news sentiment as a supervised machine learning problem, where the inputs are the contents of the news announcements, and the targets are the directions of the market's movements.

Some previous studies have suggested that the impact of news sentiment on the market is not universal, but only becomes salient during specific time periods [Ahmad et al., 2015, Smales, 2015]; significant interactions between news sentiment and the stock market were identified during recessions at the market index level [García, 2013] as well as IPOs (Initial Public Offering) at the firm level [Koppel and Shtrimberg, 2004]. It was also understood that the sources from which the news sentiment was gathered played an important role — sentiment is often found where it is expected; Tetlock [2007] discovered that textual sentiment extracted from the editorial column *Abreast of the Market* on *The New York Times* had a significant impact on the returns of Dow Jones Industrial Average from the subsequent days; Loughran and McDonald [2011a] found some links between the usage of negative words in a company's 10-K forms[3] and the performance of the company's stock.

What previous studies have not addressed, however, is why textual sentiment in *arbitrary* business news has not contributed more, if not at all, to the price movements in the market. Is it because the market is so efficient outside times of turmoil that, as postulated by the *Efficient Market Hypothesis* [Fama, 1970], almost all the implications carried by news sentiment had already been absorbed before they could be investigated? Or is it because the sentiment signals were so distorted by the noise in the text that their impacts became undetectable?

One observation made in this thesis about business and financial news is that much of the information disseminated in the news is not 'new' — it is rather common to find chunks of text in financial news articles that mainly report on what had already happened in the market, e.g. fluctuations of market indices and stock prices. While such recounts are useful in providing contexts to the readers of the news, they themselves do not convey much information apart from what had already been told by the prices in the market (i.e. the *Efficient Market Hypothesis*). This repetition (i.e. news repeats the information entailed in prices) could become a source of noise that interferes with the analysis of news

[3] A Form 10-K is an annual report a company files that summarises its financial performances. The filing of the form is required by the U.S. Securities and Exchange Commission.

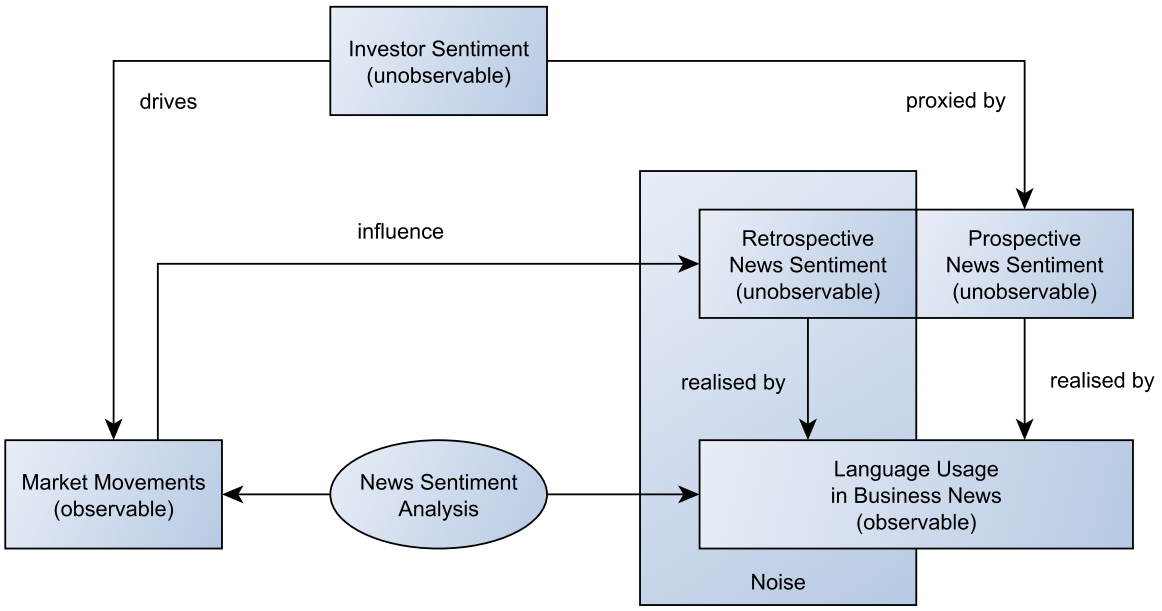


Figure 1.2: News sentiment as a mixture of prospective and retrospective sentiment

sentiment.

It is therefore proposed in this thesis that one of the reasons why traditional sentiment analysis methods have had difficulty detecting the effects of news sentiment on the market is due to the presence of *retrospective sentiment*: the *retrospective sentiment* reflects the evaluative stance business news articles take when reporting on what had happened in the market; its counterpart, the *prospective sentiment*, refers to the evaluative stance news articles take when discussing future development of the market. The introduction of the notions of *retrospectivity* and *prospectivity* brings a new *temporality* dimension to the classic positive-versus-negative model of news sentiment; these two types of sentiment will collectively be referred to as *temporal sentiment* in the rest of this thesis. It is argued in this thesis that a business news article can convey both types of temporal sentiment at the same time — the language usage in an article can be thought of as a mixture of the linguistic realisations of the two types of temporal sentiment (Figure 1.2). Traditional approaches to the analysis of news sentiment had sought to establish contemporaneous causal links from news sentiment to market movements; however, such links can be difficult to obtain during times periods when retrospective sentiment is prominent and prospective sentiment is weak, in which case the majority of the news sentiment will be driven by market movements, not the reverse; in other times, prospective sentiment becomes prevalent and its impact on the market becomes perceivable to traditional analysis approaches. By separating prospective news sentiment from its retrospective counterpart, it might be possible to construct a more accurate model for the interactions between news sentiment and the behaviour of the market.

The main objective of this thesis is to develop a computational method that can automatically learn from business news the lexicons used to express prospective and retrospective sentiment; this objective is approached by separately modelling the causal relationships between the two types of temporal sentiment and the behaviour of the market (Figure 1.3). A second objective is to investigate whether making the distinction between retrospectivity and prospectivity when modelling news sentiment improves to the method's ability to predict firm-level stock returns using news sentiment. The third objective is to explore the effectiveness of word dependencies serving as the *unit of meaning* when modelling news sentiment, and compare it with that of unigrams.

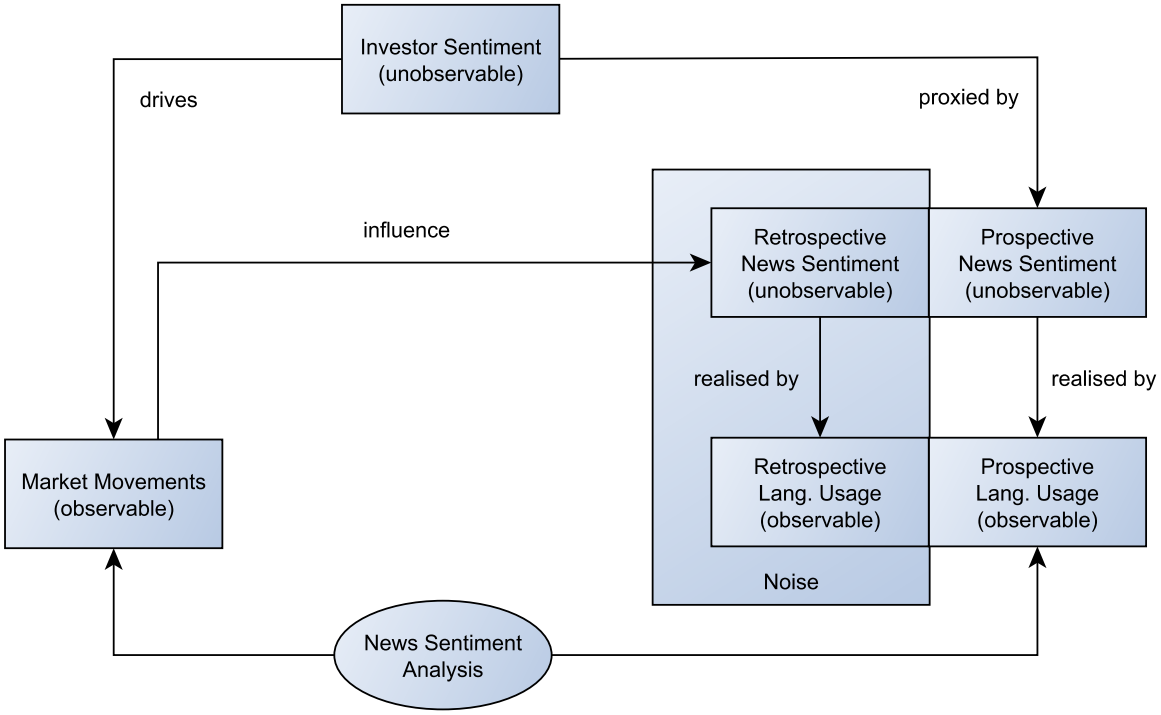


Figure 1.3: Establishing the link between the linguistic realisations of news sentiment and the movements of financial market while bypassing the 'noisy' retrospective sentiment

By distinguishing between the language patterns used to express prospective and retrospective sentiment, news sentiment analyses are able to bypass the 'noisy' retrospective sentiment patterns and work directly on the more informative prospective sentiment.

1.2 Research Questions

The principal question this thesis attempts to answer is:

- Can a method be developed to computationally distinguish the language patterns that realise retrospective and prospective news sentiment?

Suppose the answer to it is positive, then a second question is asked:

- Does distinguishing between retrospective and prospective news sentiment help improve the modelling of the interactions between news sentiment and the market more effectively than news sentiment?

As discussed in the previous section, a method that is able to computationally distinguish recounting language patterns from language patterns that prompt speculations may help separating the underlying prospective sentiment signals from retrospective sentiment signals. If such method can indeed be developed, it would be interesting to know whether making such distinction actually improves the extraction of news sentiment from arbitrary business news.

The third research question asked in this thesis is:

- Can word dependencies, as a unit of meaning, better capture sentiment than unigrams?

Noise may also arise during the analyses of sentiment due to the ambiguous nature of the lexical structures that serve as the unit of meaning. Some previous studies represented news sentiment based on word level connotations; in some cases, the different senses of the same word could convey distinct sentiment connotations (e.g. the word 'crude' in 'crude oil' no longer possess a negative undertone); in other cases, the sentiment orientation of a phrase or an expression cannot be inferred by simply combining the sentiment connotations of its constituent words — that is, the sentiment of the phrase cannot be compositionally computed. For example, if one consults the General Inquirer dictionary, the news title 'Microsoft earnings beat estimates' would be considered to bear negative sentiment by content analysis methods due to the word 'beat' being classified as negative in the dictionary. Some linguists have argued that single words by their own may be inadequate in representing meaning, and that lexical structures of higher orders should be used as the unit of meaning [Sinclair, 1998, Stubbs, 2009]; among the candidates proposed to substitute single words are *collocations* (co-occurrence of words), *colligations* (the co-occurrence of grammatical phenomena), and *semantic preferences* (the restriction of regular co-occurrence to items which share a semantic feature) [Sinclair, 1998]. In this thesis, I explore the use of an alternative lexical structure called *word dependency* as the unit of meaning when representing sentiment. According to *dependency grammar*, a sentence can be represented as a series of *dependency relations* (or *word dependencies*); each dependency relation describes the syntactical relation between a *dependent* word and another *governor* word [Kübler et al., 2009, pg. 2]. Recent developments in natural language processing have brought algorithms and tools that are capable of extracting *word dependencies* from raw texts [Klein and Manning, 2003], making it possible to explore the use of word dependency as the unit of meaning when analysing news sentiment.

1.3 Original Contributions and Findings

The principal original contribution this thesis makes to the overall body of knowledge is to have introduced the notion of *temporality* to the modelling of textual sentiment in business and financial news. In addition to the positive and negative news sentiment, this *temporality* dimension further divides news into two new categories: *retrospective* news, which mainly describes what had already happened on the market, and *prospective* news, which discusses and speculates on the future developments of the market. Through this new mechanism, this thesis attempted to explore the reason behind the lack of discernible impacts from news sentiment found in arbitrary business news on the movements of market. It is hypothesised in this thesis that news sentiment's influence over the market mainly channels through prospective sentiment; trying to model the influence without discounting the retrospective aspect of the sentiment could introduce noise that interferes with the process. It is hoped that by treating retrospective and prospective sentiment separately, one may be able to construct a more accurate model for news sentiment's effect on the financial market.

It is acknowledged that while the investigation of the prospective effect news sentiment has on the market is not new [Généreux et al., 2011, Li, 2010], nor is the notion of two-way feedbacks between news sentiment and the market [e.g. Mo et al., 2015], this thesis, to my best knowledge, represents the first attempt to separate the effect of retrospective news sentiment from that of the prospective news sentiment when examining the interactions between news sentiment and the behaviour of the market.

A second contribution made by this thesis consists in the development and subsequently the implementation of a method that computationally identifies linguistic realisations of temporal sentiment from raw news articles without manually labelled training sets. In this method, the overall sentiment of a news article is treated as a mixture of four types of temporal sentiment, i.e. *retrospective negative*, *retrospective positive*, *prospective negative*, and *prospective positive*. A probabilistic model was constructed to describe the interactions between the temporal sentiment about a company found in business news and the company's stock price. The parameters in this model, i.e. the distribution of the linguistic structures that realise the sentiment, were learnt using the *expectation maximisation* algorithm. The method was implemented by two systems built by the author of this thesis: the CiCui system which converts raw and unstructured textual data into a structured form, and the TSMiner system which takes the structured output from the CiCui system and carries out the calculations specified by the expectation maximisation algorithm developed in this thesis.

To address the research questions, a case study was conducted using a dataset comprised of news articles and the stock prices collected for 16 top-ranking publicly listed companies.

The case study first established the effectiveness of the method developed in this thesis in terms of its ability to learn the lexical item distributions for the four temporal sentiment classes. It was then discovered that, in business and financial news, retrospective sentiment is overall more prevalent than prospective sentiment; this implies that it is often the movements of the market that led to the the changes in news sentiment. Subsequent experiments found that introducing temporality into the modelling of news sentiment did not seem to have improved the predictive performance of the method when forecasting the daily returns of a company's stock with the textual sentiment found in the news stories pertinent to the company. Lastly, the thesis investigated the effectiveness of word dependencies serving as units of meaning when representing news sentiment compared to unigrams. The results suggest that using word dependencies as the textual features conferred no increase in the predictive performance of the method; however, it was noted that the prospective sentiment learnt for word dependencies are more interpretable than those learnt for unigrams.

1.4 Thesis Outline

While the three research questions above are all raised in a linguistic context, this thesis attempted to approach them from a computational point of view. Chapter 2 starts by surveying some of the previous studies on computational sentiment analysis in general; it then reviews related studies exploring news sentiment's impact on the financial market so that the work done in this thesis can be put into perspective. In Chapter 3, I first discuss the theoretical and mathematical backgrounds for the method developed in this thesis. A probabilistic model is then proposed to describe the interaction between *temporal sentiment* and the market at the firm level. This is followed by the derivation of the maximum likelihood estimators for the parameters in the model using the expectation maximisation algorithm. Chapter 4 describes the design and implementation of two software systems which perform the computations set out in Chapter 3. Using these two systems, Chapter 5 applies the method developed in Chapter 3 to conduct a case study on 16 publicly listed companies to explore the role of temporal sentiment in the interactions between business news and stock returns; the case study includes four experiments, each addressing a certain aspect of the research questions. Finally, Chapter 6 concludes the thesis and discussed areas in the method that could be improved in the future.

1.5 Publications

- Xiubo Zhang and Khurshid Ahmad. Affect Proxies and Ontological Change: A finance case study. In Sivaji Bandyopadhyay and Okumura Manabu, editors, *Pro-*

ceedings of the 2nd Workshop on Sentiment Analysis where AI meets Psychology, pages 99-114, 2012.

- Xiubo Zhang and Khurshid Ahmad. Ontology and Terminology of Disaster Management. In *DIMPLE: DIsaster Management and Principled Large-scale information Extraction Workshop Programme*, 2014.

Chapter 2

Background and Related Work

2.1 Introduction

In this chapter, I survey the literature on sentiment analysis in general as well as its applications in financial and business contexts. Previous studies on the interactions between news sentiment and the behaviour of the market have largely been motivated by two reasons: (i) to gain a better understanding of the role news sentiment plays in the financial market; and (ii) to exploit news sentiment as an instrument for managing risks in financial activities. Such inquiry therefore represents an interdisciplinary endeavour that exploits mainly two sets of techniques: text analytics methods developed in computational linguistics are used to model textual sentiment in business news, while statistical methods are brought to model the dynamics between news sentiment and the movements of the financial market. This division of interest resulted in a separated body of literature — researchers from the computational linguistic community tend to focus more on developing more accurate models for news sentiment, whereas studies from the finance community put emphasis on learning about the explanatory role of news sentiment in econometric models.

It can be argued that the method developed in this thesis belongs to the former category because its main goal is to find the distributions of lexical items for temporal sentiment. In achieving this goal, however, the method also explores the causal relationships between news sentiment and the movements of the market. This survey therefore starts by reviewing sentiment analysis related literature from the standpoint of computational linguistics (Section 2.2) and then moves on to examine related works in the finance literature that contemplate the relationships between news sentiment and market behaviours (Section 2.3).

2.2 A Computational Approach to Sentiment Analysis of Text

The analysis of textual sentiment, being a special form of text analysis, has benefited greatly from the rapid growth of computing capacities and the advances in natural language processing methodologies. Procedures such as probabilistic language modelling and statistical text classification have seen widespread use in sentiment analysis tasks. In this section, I set out to review some of the key concepts and techniques used in computational text analysis with an emphasis on their applications in sentiment analysis.

The process of computational sentiment analysis can be abstracted as a mathematical function — one that maps a piece of text to its *sentiment orientation* [1], a numeric vector that quantifies both the *direction* and the *magnitude* of the sentiment connotation the text evokes. This survey organises previous studies by how they had approached the input, mapping, and the output aspects of the sentiment analysis function.

Surveys of the literature have revealed two main paradigms of methods for sentiment analysis [Pang and Lee, 2008, Liu, 2012, Cambria et al., 2013, Medhat et al., 2014]. The first approach, which is referred to as the *pattern matching* method in this survey, treats input texts as streams of lexical items; an algorithm then searches the streams for sentiment-bearing patterns; results from the search are aggregated to form the overall sentiment orientation of the text using predefined rules and heuristics. The second approach formulates the analysis of sentiment as a supervised machine learning task: the input text is first transformed into document vectors; a statistical classifier is then trained using a set of labelled documents whose sentiment orientations are known (i.e. training set); the trained classifier can then be used to determine the sentiment orientations for unseen documents. A key difference between the two methods lies in their utilisation of external knowledge on sentiment — pattern-matching-based methods typically make use of *prior* information about the sentiment connotations of words found in pre-compiled sentiment dictionaries, whereas machine learning methods attempts to derive such information directly from the labelled training data. A comparison between the two paradigms is listed in Table 2.1.

Paradigm	Pattern Matching	Machine Learning
Input	lexical tokens and sentiment dictionaries	Labelled document vectors
Mapping	Scoring rules and heuristics	Statistical classifiers
Output	Sentiment orientation	Sentiment orientation

Table 2.1: Comparing two sentiment analysis method paradigms

[1] The concept has many names; it is sometime also referred to as *semantic orientation* [Wilson et al., 2009] or *sentiment polarity*, although the latter is mostly used to refer to only the direction of a sentiment orientation.

The remainder of this section is divided into two parts. The first part (Section 2.2.1) reviews techniques used by the pattern matching paradigm, with an emphasis on the development of sentiment dictionaries. The second part (Section 2.2.2) discusses some of the most common techniques and methods used for machine-learning-based sentiment analysis as well as their applications in sentiment analysis.

2.2.1 Sentiment Analysis as Pattern Matching

A key assumption made in sentiment analysis is that the writer's opinion and attitude towards a topic are reflected through her choice of words during writing[2] — that is, a writer prefers to use different sets of words when trying to deliver different attitudes; readers infer the author's attitude later by noting the word preferences.

In a sense, pattern-matching-based sentiment analysis methods attempt to mimic how human readers understand textual sentiment. Systems that implement such methods first identify the sentiment connotations of the individual words and phrases encountered in the input text; it then aggregates the orientations associated to the individual occurrences to form a summarising attitude expressed in the text according to some predefined rules and heuristics (Figure 2.1).

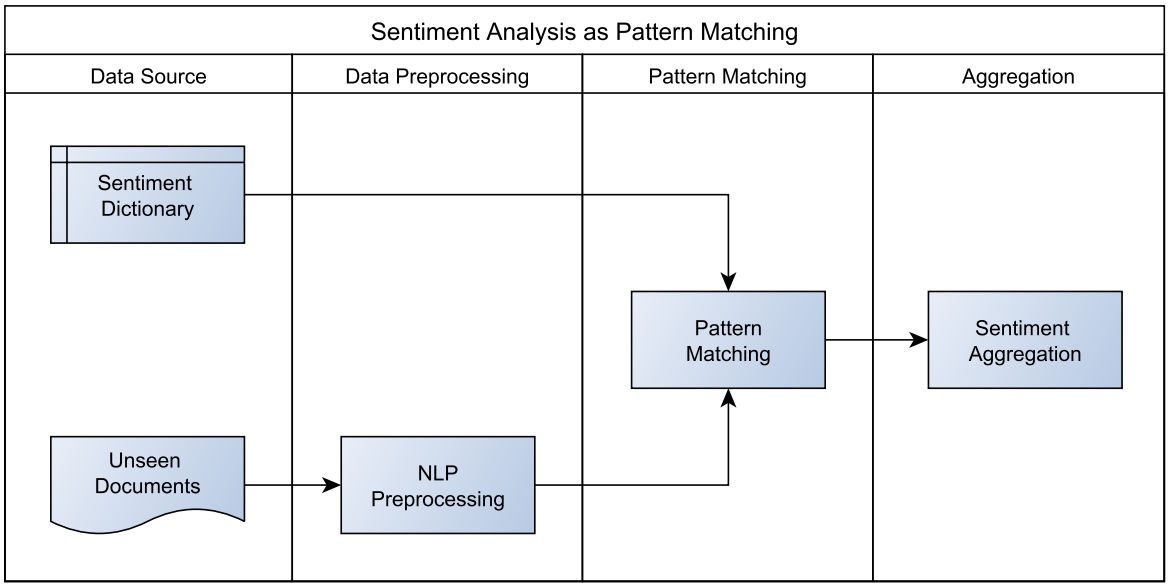


Figure 2.1: Sentiment analysis formulated as a pattern matching procedure

This communication of sentiment is made possible by the sharing between the authors and readers a common vocabulary for expressing opinions and attitudes; access to the

[2]As Sinclair [1965] pointed out: 'Any stretch of language has meaning only as a sample of an enormously large body of text; it represents the results of a complicated selection process, and each selection has meaning by virtue of all the other selections which might have been made, but have been rejected.'

vocabulary is thus vital for a heuristic-based system to be able to interpret the sentiment connotations of words and phrases it encounters. In sentiment analysis, such vocabulary is typically organised in the form of a *sentiment dictionary*. A sentiment dictionary maintains a mapping from certain lexical items (words, phrases, patterns, etc.) to their respective sentiment categories (i.e. 'positive' and 'negative'; a lexical item belongs to neither categories is sometimes considered as 'neutral').

A key challenge in sentiment analysis is therefore the compilation and utilisation of sentiment dictionaries. The literature related to this topic is reviewed in Section 2.2.1.1. One of the contributions of this thesis is a method that automatically infers the distributions of lexical items for different sentiment categories, which could assist the compilation of sentiment dictionaries.

The other aspect of heuristic-based sentiment analysis method is the aggregation of the sentiment orientations obtained from the subordinate structures into a summarising sentiment orientation for the encompassing lexical structure (i.e. sentence, paragraph, document, etc.), a topic which is reviewed in Section 2.2.1.3. This task can be challenging due to the non-compositional nature of the natural languages. This lack of compositionality means that one cannot derive the sentiment orientation of a sequence of lexical items by simply adding up the sentiment orientations of the constituent lexical items. As a result, efforts must be put to compose specialised linguistic rules that account for this 'non-compositionality'.

2.2.1.1 Sentiment Dictionaries

As discussed earlier, accesses to high quality sentiment dictionaries is crucial to heuristic-based sentiment analysis methods. The compilation of sentiment dictionaries has therefore attracted a considerable amount of research interest. There are generally two approaches to the compiling of sentiment dictionaries: (i) to have experts measure the sentiment connotations of the words through surveys and build the dictionaries manually, or (ii) to use an algorithm to automatically determine the sentiment orientations of the words.

The work described in Osgood [1957]'s work *The Measurement of Meaning* represents some of the earliest efforts in quantifying word sentiment. A psychological experiment was conducted in which participants were asked to rate words across a number of semantic scales. Using factor analysis, Osgood identified three semantic dimensions with which the affective connotations of a word can be characterised: (i) the 'Evaluative Factor' dimension, whose most representative scales include good–bad, optimistic–pessimistic, complete–incomplete, etc.; (ii) the 'Potency' dimension, which is associated with scales such as hard–soft, heavy–light, etc.; and (iii) the 'Oriented Activity' dimension, which is associated with scales such as active–passive, hot–cold, excitable–calm, etc.. In some sense, the 'Evaluative Factor' dimension can be considered to have encoded the sentiment

connotations of words to some extent.

Stone et al. [1966] later extended the work by Osgood and developed the *General Inquirer* system. The General Inquirer (abbreviated as GI hereafter) is a content analysis system that automatically codes and classifies texts using an affect dictionary and a set of disambiguation rules. The affect dictionary is essentially a generalised sentiment dictionary — it categories words and phrases with respect to not only their sentiment connotations, but also their *affect connotations*; example affect categories include positive versus negative (i.e. sentiment), activity versus passivity, pleasure versus pain, overstatement versus understatement, etc. The disambiguation rules specify how the affect of a word should be determined given its surrounding context. The affect dictionary came with the GI dictionary was obtained by merging two other dictionaries: the *Harvard IV-4* psychological dictionary, which incorporates Osgood's three semantic dimensions, and the *Lasswell value dictionary* [Namenwirth and Weber, 1987]. The GI dictionary covers 17788 words and 182 affect categories; four of the categories are directly related to sentiment: *Pstv*, *Ngtv*, *Positiv*, and *Negativ*. The *Pstv* and *Ngtv* categories were derived based on the word's position on the 'Evaluative Factor' scale. The *Positiv* and *Negativ* categories are improved versions of the *Pstv* and *Ngtv* categories respectively, and are often used in place of the *Pstv* and *Ngtv* categories.

The General Inquirer dictionary had been used in a number of studies as the source of sentiment orientations for words and phrases. Yi et al. [2003] included words filed under the *Positive*, *Negative*, and *Hostile* categories from the GI dictionary when constructing a combined sentiment lexicon from three existing lexical resources (GI, DAL, and WordNet). In a similar note, Wilson et al. [2009] referred to the GI dictionary when determining the prior sentiment orientations for single words, which were then used to infer the sentiment orientations of the enclosing phrases. Kennedy and Inkpen [2006] enhanced the sentiment word categories by taking into account *valence shifters*, which are modifier terms that can change the sentiment orientation of another word. Turney and Littman [2003] and Kamps et al. [2004] used the GI dictionary as an benchmark to evaluate the performances of their sentiment lexicon expansion algorithms. Tetlock [2007] and Tetlock et al. [2008] made use of the GI dictionary when measuring news' impact on financial market performances. Das and Chen [2007] exploited the difference between the counts of pessimistic and optimistic words as defined in the GI dictionary as a disambiguation method.

Several sentiment dictionaries were developed following the model of the GI dictionary while addressing some of its limitations. One such limitation is that the entries in the GI dictionary are defined over finite categorises — that is, the dictionary only records whether or not a word belongs to a certain category, but not the intensity of its membership [Ann and Khurshid, 2008]. Later sentiment dictionaries, such as Cynthia Whissell's Dictionary of Affective Language [Whissell et al., 1986] and Senti-WordNet [Baccianella et al., 2010],

quantify the intensity of the sentiment orientations on continuous scales, allowing more fine-grained representation of textual sentiment.

Another limitation of the GI dictionary and many other sentiment dictionaries is that they only contain affect definitions for unigrams (i.e. single words). While content analysis systems are in general agnostic to the type of the lexical structures they process (i.e. they are able to utilise dictionaries developed for lexical structures other than unigrams like bigrams or even multi-word phrases), most affect dictionaries were built at the unigram level; as such, it is sometimes difficult for content-analysis-based methods to resolve sentiment orientation for certain types of phrases due to the non-compositional nature of textual sentiment. The root of this limitation may lie in the inadequacy of unigram's ability to serve as the unit of meaning. Sinclair [1991, pg. 6] observed that meanings in language tend to manifest through phrases rather than single words. If Sinclair's proposition is correct, then textual sentiment, being a special form of meaning, will be better captured through lexical structures of higher orders.

Several later studies have explored the possibility of defining sentiment dictionaries for lexical structures other than unigrams. Popular alternatives considered include bigrams and n-grams [Pang et al., 2002, Dave et al., 2003, Matsumoto et al., 2005, Cui et al., 2006, Butler and Kešelj, 2009, Abbasi et al., 2008, Bespalov et al., 2011] and word dependencies [Wilson et al., 2009, Di Caro and Grella, 2013, Poria et al., 2014, Agarwal et al., 2015].

Lastly, the GI dictionary and some other psychological sentiment dictionaries were designed to measure the sentiment orientations of words appeared in the general language register, and may thus fail to reflect domain-specific connotations that become prominent only in certain *sublanguages* [Khurshid and Rogers, 2001, Li et al., 2014a]. A *sublanguage* is 'a specialized language or jargon associated with a specific group or context'[3]. The problem with sublanguages is particularly relevant to the analysis of news sentiment because business news often cover various specialised domains (e.g. aerospace, energy, motoring, finance, etc.). Later studies have attempted to address this problem by either introducing pre-defined domain-specific sentiment dictionaries [e.g. Loughran and McDonald, 2011a] or resorting to automatic expansion of sentiment dictionaries to construct ad hoc domain dictionaries, a technique elaborated in Section 2.2.1.2.

2.2.1.2 Automatic Compilation of Sentiment Dictionary

Compiling sentiment dictionaries can prove difficult. The quality of sentiment dictionary is dependent on the consensus it receives — the more people agree with the definitions a dictionary gives, the more reliable the dictionary becomes. For manually compiled dictionaries, there often exists a trade-off between quality and cost — considerable amount

[3] `http://www.oxforddictionaries.com/definition/english/sublanguage`, accessed in January 2016

of manual work is required to produce a sentiment dictionary of satisfactory quality; furthermore, verifying the quality of the dictionary requires additional resources. All these hinder the wider application of the pattern-matching-based method.

A number of studies have attempted to overcome the difficulties by introducing algorithms that are able to automatically derive the sentiment polarities for words based on their usages in a corpus. Such methods typically start with a list of 'seed words' whose sentiment orientations are taken as known; some pre-defined rules and heuristics are then applied to infer the sentiment polarities for the other words seen in the corpus. Hatzivassiloglou and McKeown [1997], in their pioneering work, noted that words linked by the conjunctions like 'and' tend to have similar sentiment orientations, while words linked by 'but' tend to possess opposite sentiment orientations. The authors then developed a logistic regression model that attempts to predict the types of the conjunctions, and used it to cluster adjectives into positive and negative groups. Kanayama and Nasukawa [2006] extended Hatzivassiloglou and McKeown [1997]'s method by introducing more sophisticated inference rules; the resulting sentiment pattern database were later used by [Yi et al., 2003]. Turney and Littman [2003] described a method that infers sentiment orientations of words based on their statistical association with a handful of seed words. The method evaluates the 'statistical distances' between the words using metrics such as point-wise mutual information (PMI), as well as the cosine distances between the word vectors produced by latent semantic analysis (LSA): words that are 'close' to each other in such metric space are considered to bear similar sentiment orientations. Also using the PMI metric, Yu et al. [2013] developed a 'contextual entropy model' to expand emotion words and their intensities from a list of seed words for classifying sentiment in stock market news. Kamps et al. [2004] and Hu and Liu [2004] both exploited the synonyms and antonyms information supplied in WordNet [Miller, 1995] to construct sentiment dictionaries from a list of seed word; Shaikh et al. [2016] extended this approach to creating sentiment lexicons for other language by employing Google's translation service.

Another approach to the automatic building of sentiment dictionaries involves the use of the weights/coefficients assigned to the features by statistical classifiers when learning sentiment form text. Bayesian text classifiers, for example, produce probability distributions over the words in the vocabulary for each sentiment class; the probabilities assigned to the words can be interpreted as the intensities of the words' sentiment in the corresponding sentiment category. The method developed in this thesis follows the Bayesian paradigm, and the distributions over words learnt for each of the temporal sentiment categories can be construed as sentiment dictionaries. It is worth nothing that while this method mostly applies to generative models; weights assigned to features (i.e. lexical items) in certain discriminative classifiers, support vector machines in particular, may also be interpreted as the contributions of the features make to the classification of

sentiment [Matsumoto et al., 2005].

2.2.1.3 Sentiment Classification via Aggregation Heuristics

Once the sentiment orientations of the underlying lexical items have been determined using a sentiment dictionary, the overall sentiment orientation of the encompassing structures can be acquired by aggregating the individual orientations from the subordinate lexical items.

Naive Classifier The simplest and perhaps most intuitive strategy for combining sentiment from individual lexical items is perhaps the so-called 'naive classifier' [Das and Chen, 2007]. In this strategy, each sentiment-bearing word or phrase is assigned a sentiment score (e.g. -1 for negative, 0 for neutral, and $+1$ for positive); the overall sentiment orientation of a sentence or document is then the sum or mean of the sentiment scores of its constituent words and phrases [Turney, 2002, Dave et al., 2003, Kim and Hovy, 2004, Hu and Liu, 2004, Das and Chen, 2007].

The drawback of this strategy is that it has difficulties handling non-compositionality, where the overall sentiment orientation of an phrase cannot be derived by simply adding up the sentiment orientations of its constituent words. One example of such phrases is the 'beat estimates' demonstrated in Section 1.2. Another example would be negations, where a word is modified with a negator such as 'not', 'hardly', etc.; consider the phrase 'not bad': the word 'bad' falls in the Negativ category of the GI dictionary; the word 'not' is categorised as neither positive nor negative, and is thus considered neutral by most sentiment dictionaries; a sentiment classifier that naively combines the sentiment orientations of the two words will classify the phrase as negative. For the sentiment orientation of this phrase to be classified correctly, the algorithm needs to be able to identify the negation, invert the contribution the adjective 'bad' makes to the overall sentiment orientation of the phrase, and conclude that the overall sentiment borne by the phrase is positive or at least non-negative. In other cases, adverbs could be used to modify the intensities of evaluative phrases (e.g. *quite* good, *very* bad); Polanyi and Zaenen [2006], Kennedy and Inkpen [2006] recognised such usages as *contextual valence shifters*. In these scenarios, the intensifiers (or diminishers) themselves do not invoke any affective connotation; they instead alter the intensity of the sentiment orientations of the words that follow them. The most problematic constructs are perhaps the metaphors — neither 'kick' nor 'bucket', for instance, are considered evaluative, yet the phrase 'kick the bucket' exhibits negative connotation. The interpretation of metaphors, however, is beyond the scope of traditional sentiment analysis, so I will not discuss this further in this thesis.

Heuristic-based Classifier Heuristic-based classifiers build on the basic principle of the naive classifiers discussed above, but introduce rules and heuristics that specify how the aggregations should be handled in certain scenarios. This strategy recognises the non-compositional nature of textual sentiment. When tasked to summarise the overall sentiment of a sentence or document, it combines the sentiment orientations of the subordinate words and lexical items following manually developed rules and heuristics. Yi et al. [2003] and Nasukawa and Yi [2003], for example, compiled dictionaries comprising hundreds of so-called *sentiment patterns*, which are essentially heuristics and rules that define how the sentiment of a certain phrase should be determined (e.g. '*some feature* prevents trouble' will be marked as a positive review). Liu [2010] developed a set of rules using BNF-like grammar for inferring sentiment orientations; an example rule is:

$$
\begin{array}{lcl}
\text{POSITIVE} & := & \text{P} \\
& | & \text{PO} \\
& | & \text{sentiment\_shifter N} \\
& | & \text{sentiment\_shifter NE}
\end{array}
$$

where P and PO represent atomic and compound positive sentiment expressions respectively, whereas N and NE represent atomic and compound negative sentiment expressions respectively; the grammar states that an expression is positive if it is an atomic positive expression or a compound positive expression or a negative expression (atomic or compound) modified by a sentiment shifter such as 'not' and 'hardly'. The previously mentioned contextual valence shifters by Polanyi and Zaenen [2006] operates on a similar fashion: (i) a sentiment-laden expression is initially assigned a sentiment value (e.g. 2 for the adjective 'clever'); (ii) if expression is modified by a intensifier (e.g. 'very') or a diminisher (e.g. 'hardly') then its sentiment value is scaled accordingly ('very clever' will yield a value of 3 while 'barely clever' will be assigned a value of 1); (iii) if the expression is modified by a negation (e.g. 'not clever'), then the sign of its value is inverted ('not clever' will have a sentiment value of -2).

2.2.2 Sentiment Analysis as Supervised Machine Learning

For the heuristic-based methods, the 'mapping' component of the sentiment analysis function is implemented through human expertises — the heuristics that direct the inferring of sentiment orientations are supplied by human experts. The gathering of such knowledge, be it the definitions for sentiment dictionaries or rules for sentiment aggregation, is a non-trivial endeavour. In contrast, for machine learning based methods, the mapping from lexical structures to their sentiment orientations are established by the algorithm itself with little to no human intervention.

Figure 2.2 illustrates the workflow of a typical machine-learning-based sentiment analysis. The process makes use of two sets of data: (i) the training documents, which could be product reviews or business news, and (ii) the targets of the learning, which could be the customer ratings or market movements. The running texts in the news are transformed into structured form and converted into numeric forms before they can be consumed by the machine learning algorithms; the conversion of textual documents into numeric vectors (i.e. *document vectorisation*) is addressed in Section 2.2.2.1. For classification tasks, continuous targets are mapped into discrete variables (e.g. numeric ratings are mapped into 'favourable' versus 'unfavourable' binary classes using a cut-off threshold); missing values are also treated at this time. Lastly, the target labels are assigned to the training documents — this step is relatively straightforward for some applications, e.g. product reviews, whereas for the learning of textual sentiment in financial news, it can involve an additional grouping of articles by dates when daily returns are used as the targets. The categorisation of classifiers (*discriminative* versus *generative* model) used in sentiment analysis are briefly discussed in Section 2.2.2.2. Once the training is complete, the sentiment orientations of unseen documents can be determined using the trained classifier.

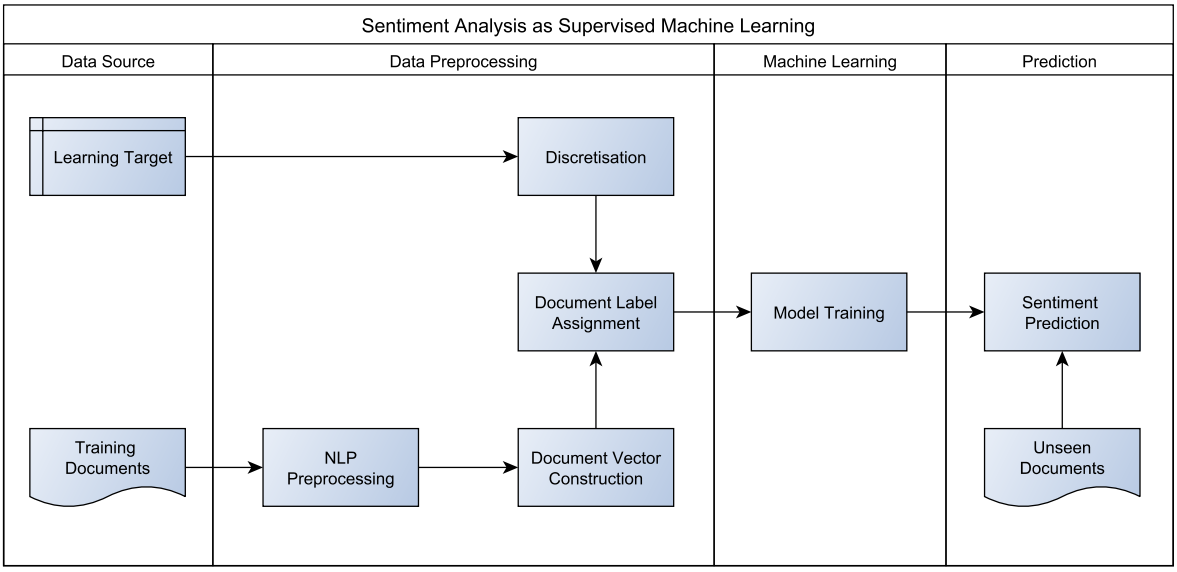


Figure 2.2: Sentiment analysis formulated as supervised machine learning

Machine-learning-based sentiment analysis inherits many of the key concerns seen in general machine learning tasks: feature representation (e.g. representing running texts as document vectors), feature selection (e.g. dropping infrequent words and stop words), choice of classifiers (e.g. discriminative versus generative models), and validations (e.g. setting up cross-validations). The remainder of this section reviews the literature that address some of these aspects in context of sentiment analysis. Section 2.2.2.1 first examines

the mathematical representations of textual materials; Section 2.2.2.2 then contemplates the different types of machine learning algorithms commonly used in sentiment analysis.

2.2.2.1 The Vector Space Model

Computational approaches to sentiment analysis require the input text be 'computable' — that is, the input textual material must be represented in such a way that an algorithm is able to evaluate its sentiment orientation automatically. To this end, the *de facto* practice is to represent the input text as a *document vector* using the *vector space model* [Salton et al., 1975]. In the vector space model, a piece of raw input text is treated as a sequence of lexical tokens; the set of all unique lexical tokens collected from the documents in a corpus constitutes the vocabulary of the corpus. Given a vocabulary of size V, a document d is represented by a V-dimensional document vector $d = (w_1, w_2, \ldots, w_i, \ldots, w_V)$, with w_i representing the weight assigned to the i-th entry in the vocabulary for document d.

As an example, the vectorised representation of the following sentence

> A rout in tech stocks — highlighted by LinkedIn Corp.'s massive drop after the business-oriented social network delivered a poor outlook — helped U.S. equities post their largest weekly drop in a month.[4]

is illustrated in Table 2.2.

Word	Weight	Word	Weight
drop	2	outlook	1
poor	1	post	1
massive	1	highlighted	1
stocks	1	social	1
network	1	tech	1
Corp.	1	equities	1
rout	1	weekly	1
delivered	1	U.S.	1
business-oriented	1	largest	1
helped	1	LinkedIn	1

Table 2.2: An example document vector

Stop words were removed from when vectorising the document.

The Bag-of-words Model In the above example (Table 2.2), the vocabulary is defined over single words (i.e. unigrams), and the weights used are the raw frequencies of the words'

[4]This is an excerpt taken from the news article *Nasdaq slammed as rout in tech drives stocks to largest weekly drop in a month* published on MarketWatch's website: `http://www.marketwatch.com/story/wall-street-gets-the-jitters-ahead-of-key-jobs-data-2016-02-05`

occurrences in the text. This particular configuration for representing document is also known as the *bag-of-words* model [Manning et al., 2008], and is a widely used document representation schema in text analysis literature.

An important property of the bag-of-words model is that the ordering of the words in the document becomes lost when it is represented as a document vector — so the phrase 'Microsoft beat estimation' will result in the same document vector as 'estimation beat Microsoft'. This is not necessarily true for vector space model in general though: if the vocabulary is defined for bigrams (i.e. two consecutive words), for example, the relative ordering of the words will be preserved to some extent. For sentiment analysis applications, the relative ordering of words can sometimes be important when negations and valance shifters are involved: the sentiment orientation of the phrase 'not good' (negative) will be different when the words 'not' and 'good' are considered separately; in such cases, the modification effects between the two words are destroyed when representing the text in the bag-of-words model. This drawback of the bag-of-words model has often been remedied by using alternative lexical features such as bigrams, n-grams, or word dependencies to represent input text [Pang et al., 2002, Dave et al., 2003, Matsumoto et al., 2005, Cui et al., 2006, Wilson et al., 2009, Bespalov et al., 2011, Di Caro and Grella, 2013, Poria et al., 2014, Agarwal et al., 2015].

Stop Words The vocabulary used to model the text may contain only a subset of all the unique words occurred in the corpus. A common practice when performing natural language processing tasks is the removal of *stop words* from the vocabulary before constructing document vectors. Stop words are often short *closed-class words* that serve mainly auxiliary and structural functions in languages. Typical stop words include the articles (e.g. *a*, *an*, and *the*), certain prepositions (e.g. *of*, *in*, *at*, etc.), certain conjunctions (e.g. *and*, *or*, *nor*, etc.), and sometimes auxiliary verbs such as the variants of *have* (i.e. *has*, *had*, etc.). There has yet been a standard stop word list — the words to be included in a stop word list will most likely vary depending on the application; for example, text classifications for Twitter messages might want to include a few special entries such as 'RT', which stands for 'retweet'; though there exist several popular stop word lists from which custom lists can be derived, such as the one in the R package `tm` [Feinerer et al., 2008] and the list provided in the Snowball `stemmer` project.

Negations The treatment of negations in machine learning settings has been discussed in a number of studies. It has been noted in the previous sections that the sentiment orientation of an phrase can be inverted when modified by a negation word such as 'not', 'hardly', 'little', etc.; however, it is often difficult to enforce special rules that account for such behaviour in statistical models. A common technique used to overcome this issue is to

replace negated words with a special tokens that are different from the original words [Pang et al., 2002, Das and Chen, 2007, Smailović et al., 2014]. For example, instead of being treated as two separate words, the phrase 'not bad' can be replaced with a special token 'NOT_bad'. This treatment is particularly suitable for machine learning methods because the negated tokens can simply be treated as a separate feature in the document vectors. Smailović et al. [2014] noted that applying negation replacement improved sentiment classification performance of their system significantly.

Alternative Weighting Schemas It is sometimes desirable to use weightings other than raw frequencies when constructing document vectors. A commonly used alternative is the *relative frequency*. The relative frequency of a term t in a document d is defined as the ratio between the raw frequency of the term in that document (freq_t) and the total number of terms in document d:

$$\text{r.freq}_t = \frac{\text{freq}_t}{\sum_{t \in d} \text{freq}_t}. \tag{2.1}$$

The advantage of using relative frequencies over raw frequencies is that the former is normalised relative to the total length of the document, so that the relatively less frequent words in a very long document will not overshadow the relatively more frequent words in a shorter document.

Another popular alternative is the tf-idf metric, which stands for *term frequency – inverse document frequency*. It is a weighting schema that weights words based on thier 'importance' within a corpus. Formally, the tf-idf is a function of three arguments: a term t, a document d, and a corpus D which contains the document d:

$$\text{tfidf}\,(t, d, D) = \text{tf}\,(t, d) \times \text{idf}\,(t, D)\,. \tag{2.2}$$

The *term frequency* $\text{tf}\,(t, d)$ is a function of both the term t and the document d; the function evaluates the prominence of term t in document d. A common choice for this function is the aforementioned *relative frequency*. The *inverse document frequency*, $\text{idf}\,(t, D)$, is a function of the term t and the corpus D; formally, it is defined as:

$$\text{idf}\,(t, D) = \log \frac{N}{\text{df}\,(t, D)} = \log \frac{N}{|\{d \in D \mid t \in d\}|}. \tag{2.3}$$

The *document frequency* function $\text{df}\,(t, D)$ returns the number of documents in corpus D that contain the term t; the *inverse document frequency*, being the logarithmic inverse of the relative document frequency, would approach zero as the document frequency of term t approaches N, i.e. the total number of documents in corpus D. The significance of the *inverse document frequency* was originally established by Jones [1988] — intuitively, the

idf component states that a word is more 'important' if its occurrences are concentrated within a small subset of the corpus, since its usage may be an sign of the presence of subtopics; from an information retrieval point of view, a word with high idf can be thought of to have yielded greater distinguishing power and thus contributes more to the relevancy of a document with a certain topic.

2.2.2.2 Learning and Predicting Sentiment with Statistical Classifiers

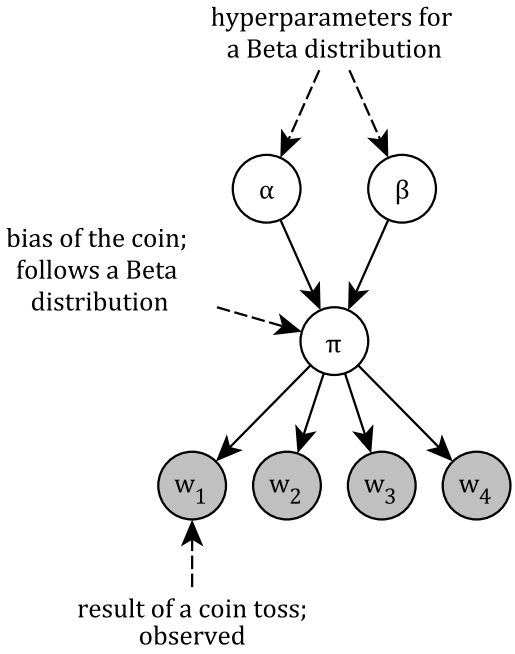
Traditionally, statistical classification models are generally divided into two categories: the *generative models* and the *discriminative models* [Bishop, 2007, Murphy, 2012, pg. 43 and 267 respectively]. While both kinds of models achieve the same goal, they represent two different schools of thought about the relationships between data and models. This distinction between generative and discriminative models also led to two different sets of approaches in sentiment analysis. The section discusses briefly the two different paradigms and their corresponding manifestations in the sentiment analysis literature.

Generative Models The generative models have their root in probability theory. Generative models assume the observable data were 'generated' or drawn from a joint probability distribution that is governed by a set of hidden parameters. To learn the model is essentially to obtain the estimates for the hidden parameters that govern the underlying distributions from which the observations were drawn. The application of the generative model requires some level of understanding on how the observed data could have been 'sampled' by the model from the distribution. In particular, the practitioners of this technique need to specify the relations between the hidden parameters as well as the probability distributions of the variables that generate the data. The model can be defined purely through mathematical description, as in the case of hierarchical Bayesian modelling [Gelman, 2014], or represented as a graph of interconnected random variables as in probabilistic graphical models [Koller and Friedman, 2009].

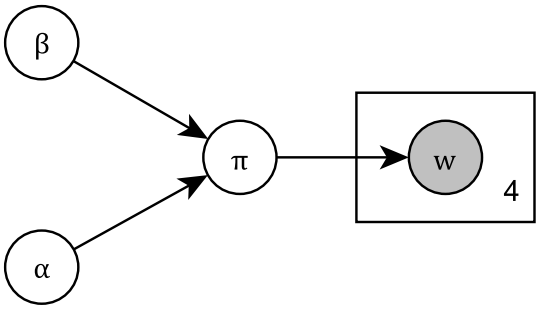
An example generative model for tossing a potentially biased coin is illustrated in Figure 2.3. In this example, it is assumed that a potentially biased coin is tossed for four times. The meaning of the elements in the graph is explained as follows:

- π denotes a random variable which follows a Beta distribution; it describes the probability of the coin showing a head after a toss (i.e. the bias of the coin).

- w_1 to w_4 are random variables that represent the actual outcomes of each tosses; these random variables follow Bernoulli distributions whose rates of successes are governed by π; the shades on the nodes indicate that the values for these random variables are observed and known.



(a) A graph depicting a generative model for tossing coins



(b) The same model represented using plate notation

Figure 2.3: An example generative model for coin toss

Conventionally in probabilistic graphical models, a circle represents a random variable; a shaded circle indicates the value of the variable is observed. The arrows specifies the governing relationship between the variables: the random variable at the destination end of the arrow depends on the variable at the source end. The plate notation exhibited in Figure 2.3b abbreviates Figure 2.3a by collapsing repeated structures into a box.

- The random variables α and β are hyper-parameters that govern the shape of the prior distribution of π.

Probabilistic models like this are often described in a *generative story*. For this example, the generative story starts from the top of the graph in Figure 2.3a. The random variables α and β characterise a Beta distribution which reflects the observer's prior belief on the biases of coins in general; it could then be assumed that the four outcomes w_1 to w_4 were

generated by tossing a particular coin whose bias is represented by the random variable of π four times; the observed values for the tosses become the data for this experiment. The model can be equivalently described with a full-joint distribution as:

$$p\left(w_1, w_2, w_3, w_4, \pi; \alpha, \beta\right) = \text{Beta}\left(\pi \mid \alpha, \beta\right) \prod_{i=1}^{4} p\left(w_i \mid \pi\right) \tag{2.4}$$

Given the data and the prior belief on the biases of coins in general, the goal of a learning algorithm is to infer the posterior distribution of π using Bayes' theorem:

$$p\left(\pi \mid w_1, w_2, w_3, w_4; \alpha, \beta\right) = \frac{p\left(\pi, w_1, w_2, w_3, w_4; \alpha, \beta\right)}{p\left(w_1, w_2, w_3, w_4; \alpha, \beta\right)} \tag{2.5}$$

$$= \frac{\text{Beta}\left(\pi \mid \alpha, \beta\right) \prod_{i=1}^{4} p\left(w_i \mid \pi\right)}{\int_0^1 \text{Beta}\left(\pi' \mid \alpha, \beta\right) \prod_{i=1}^{4} p\left(w_i \mid \pi'\right) \mathrm{d}\pi'} \tag{2.6}$$

Solving analytically for the posterior distribution, as will be demonstrated in later chapters, can prove difficult.

The naive Bayes classifier represents one of the simplest form of generative models. For naive Bayes classifiers, it is assumed that given the class of an observation, all its features are independently distributed. Technically, naive Bayes classifiers produce, for each observation (e.g. a document, or any object that is to be classified), a distribution describing the probability that the observation belongs to a certain class. Most applications of the classifier choose the class with the highest probability as the predicted class for an observation.

In text analysis, generative models had been adopted to model *topics* in texts. Hofmann [1999] proposed the *probabilistic latent semantic analysis* (PLSA); in PLSA, each topic is represented as a multinomial distribution of words; a document is thus modelled as a mixture of the words drawn from such distributions. Blei et al. [2003] later extended the idea of PLSA and developed a more sophisticated topic model called *latent Dirichlet allocation* (LDA); in LDA, each document is allowed to have its own topic distribution whereas in PLSA an overall distribution of topics is applied to all documents. From a mathematical point of view, the goal of topic modelling is to estimate the posterior distributions of the words for each topic as well as the posterior distributions of the topics for the documents given the input data.

The flexibility of the generative models has inspired a number of studies where sentiment was incorporated [Wu et al., 2006, Hu et al., 2007, Mei et al., 2007, Lin and He, 2009, Bespalov et al., 2011, Salter-Townshend and Murphy, 2014]; these methods generally extend the LDA model and introduced sentiment as additional random variables in the model.

Discriminative Models The discriminative models follow a rather different approach than the generative models. Instead of trying to 'understanding' the training data, it attempts to find a boundary that best separates instances of one class from the others. The basic principle of discriminative model is find a set of parameters for the model that will minimise the error of the classification. Examples of discriminative classifiers include Support Vector Machine (SVM), logistic regressions and artificial neural networks.

Discriminative classifiers, particularly SVM, have enjoyed widespread use in text analysis in the past two decades. SVM works by constructing a hyperplane in the feature space that optimally separates the two classes of instances; the hyperplane is optimal in the sense that it is the hyperplane that has the largest margin from the nearest data points. Previous research has suggested that SVM is particularly suitable for text classification tasks [Joachims, 1998]. A substantial number of studies used the SVM or other discriminative classifiers for sentiment analysis [Pang et al., 2002, Dave et al., 2003, Mullen and Collier, 2004, Koppel and Shtrimberg, 2004, Antweiler and Frank, 2004, Matsumoto et al., 2005, Prabowo and Thelwall, 2009, Wilson et al., 2009, Cecchini et al., 2010, Zhao et al., 2010, Bai, 2011, Généreux et al., 2011, Smailović et al., 2014, etc.].

One important difference that sets the generative classifier apart from its discriminative counterpart is that it requirement of a well defined statistical *language model* being imposed on the texts. A *language model* is 'a function that puts a probability measure over strings drawn from some vocabulary' [Manning et al., 2008, pg. 238]. In mathematical terms, it is required that:

$$\sum_{w \in V} p(w) = 1 \tag{2.7}$$

where w is a lexical unit in vocabulary V, and $p(w)$ is the probability the word w occurs in the texts. Assuming a probabilistic language model on the texts therefore implies that the vocabulary must contain homogeneous lexical items — for example, *uni-grams*, *bi-grams*, or, in general, *n-grams*. This constraint theoretically precludes the application of probabilistic classifiers in cases where the feature set is comprised of lexical units of different types (e.g. uni-grams mixed with patterns or n-grams of different orders) because the lexical units in such vocabularies will not form a proper probability distribution.

2.3 Textual Sentiment in Financial Contexts

The *Efficient Market Hypothesis* posited by Fama [1965a, 1970] states that financial markets are efficient, meaning that it reflects all publicly available information:

> In an efficient market, competition among the many intelligent participants leads to a situation where, at any point in time, actual prices of individual securities already reflect the effects of information based both on events that

> have already occurred and on events which, as of now, the market expects to take place in the future. [Fama, 1965b]

Furthermore, the *Random Walk Theory* suggested that:

> Most simply the theory of random walks implies that a series of stock price changes has no memory — the past history of the series cannot be used to predict the future in any meaningful way. The future path of the price level of a security is no more predict-able than the path of a series of cumulated random numbers. [Fama, 1965b]

The implication of these two theories is two-fold: (i) it should not be possible to predict future changes in the market based on historical 'patterns' (e.g. technical analysis); and (ii) it should not be possible to predict future changes in the market based on existing publicly available information, since they have already been absorbed by the market and reflected in the prices (i.e. the semi-strong form of the efficient market hypothesis [Fama, 1970]).

The recent advent of behavioural finance has posed a challenge to the classic assumption which states markets are efficient. Contrary to classic belief, the behavioural finance theory contends that traders and participants in the market are not fully rational. Investors sometimes over-react or under-react to news events and may not always make the optimal decision; furthermore, the presence of *herding behaviour* suggested the investors as a whole can be driven by emotions and behave irrationally [Shiller, 2000].

Most studies that explore the interactions between news sentiment and the market movements assume the market is at least weakly efficient, which implies that (i) the prices in the market had not incorporated all publicly available information; and (ii) sentiment of the investors can influence the the movements of the market; it is therefore hoped that the investor sentiment could be approximated using the textual sentiment conveyed in financial and business texts.

A large body of literature exists on the relationship between the evaluative sentiment found in business news and the performances of stocks on the market. The discussion in this section is formed mainly based on three literature reviews by Kearney and Liu [2014], Nassirtoussi et al. [2014], and Loughran and McDonald [2014]. Previous studies on the interactions between news sentiment and the market have generally concluded that news sentiment, be it expressed in news, corporate filings, or user generated content, does have an impact on future movements of the market, with Das and Chen [2007] and Kim and Kim [2014] being two exceptions.

Early studies on this subject that originated from financial backgrounds attempted to evaluate news sentiment independently from the texts. In these studies, the sentiment of a piece of text is often quantified using content analysis with the help of a sentiment

dictionary. The overall sentiment of a document is either determined using a naive classifier or represented as the number of positive and negative words (as defined in the sentiment dictionary) it contains. The sentiment representations are then aggregated to form a time series (e.g. daily, weekly, monthly, etc.) by aligning the articles with market data according to the publication dates of the articles. Finally, the sentiment series are inserted into regression models as independent variables with the market performance metric of interest acting as the dependent variable [e.g. Tetlock, 2007, Henry, 2006, Tetlock et al., 2008, Loughran and McDonald, 2011a, García, 2013, Ahmad et al., 2015]. The effects of news sentiment on the performance metric are captured by the estimated coefficients associated with the sentiment series.

Recent years have seen a surge of interest in the application of machine learning methods to the measuring of sentiment in business and financial text. Early studies such as [Antweiler and Frank, 2004, Das and Chen, 2007, Yu et al., 2013] took an indirect approach, where a machine learning algorithm is first used to classify postings from on-line message boards into *buy*, *hold*, or *sell* categories before evaluating the interactions between the classifications and the behaviour of the stock market. Wuthrich et al. [1998], Koppel and Shtrimberg [2004] and later Schumaker and Chen [2009b], Bollen et al. [2011], Généreux et al. [2011], Schumaker et al. [2012, etc.] attempted to establish a direct link between language usage in financial news and the movements of market. In these methods, performance measures of the market were used to automatically label the text documents. A common strategy for labelling documents was to link the contents of the documents with the contemporaneous stock or index returns (Figure 2.4) based on the times of their publication.

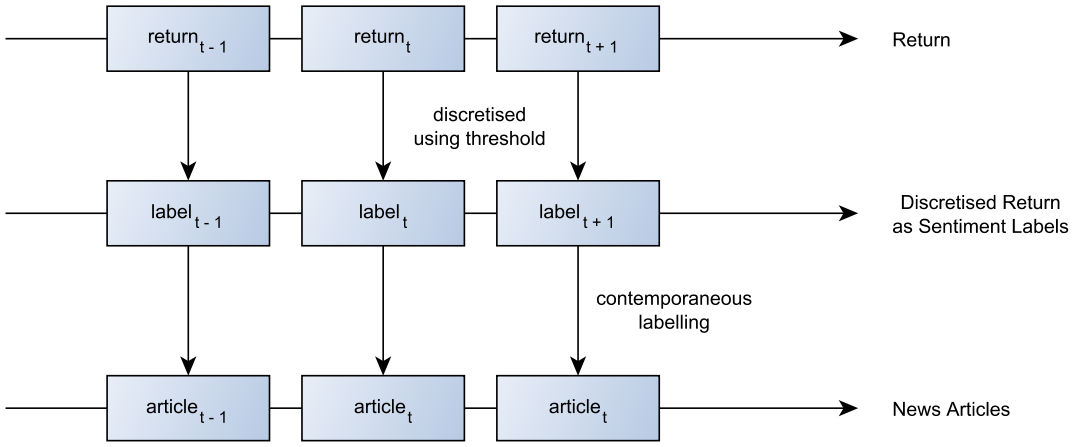


Figure 2.4: Labelling news articles with contemporaneous market returns

2.3.1 Document Labelling

While the performance of the market is often measured on continuous scales, it was noted that most machine learning based studies labelled the documents with discretised categories [Nassirtoussi et al., 2014]. Koppel and Shtrimberg [2004], Généreux et al. [2011] used manually picked threshold: (i) a news article is marked as positive if price of the corresponding stock rose 10 % or more on the next day of its publication; (ii) a news article is marked negative if the stock declined 7.8% or more on the next day after its publication. Zhai et al. [2007] combined text and technical market indicators and used SVM to predict the directions of future market movements; the classification outputs a binary category (-1 for it predicts a fall and 1 otherwise). Butler and Kešelj [2009] examined corporate annual reports and labelled them as either over or under performing the S&P 500 index based on the historical returns of the companies' stocks in the previous year. Hagenau et al. [2013] labelled financial news articles about a company based on the sign of the open-to-close return of its stock on the day the articles were published. On a similar note, Li et al. [2014b] discretised the open-to-close return of a company on day t into three categories (positive, negative, and neutral) by two thresholds which are symmetric around zero; the discretised return is then used to label the financial news articles released on day t accordingly; an interesting practice in their method is that the thresholds were calibrated on historical data, albeit the thresholds were still crisp boundaries. Bollen et al. [2011] made use of the OpinionFinder system [Wilson et al., 2005] to measure sentiments in Twitter messages; the output of the classification is a binary label indicating whether the input message is positive or negative.

Some exceptions in this regard are the studies that used *support vector regression* (SVR), a variant of SVM, as the classifier. In these cases, continuous performance measures such as daily returns can be used directly to label contemporaneous documents without the need to first discretise them. Schumaker and Chen [2008, 2009a,b], Schumaker et al. [2012] used their AZFinText system to extract news sentiment; the system used a SVR implementation in the Weka toolkit. Hagenau et al. [2013] used both SVM and SVR in their study to predict stock prices using financial news; they used classification accuracy to evaluate the binary classifications output by SVM and the R^2 measure to evaluate the performance of SVR-based models.

It is likely that the prevailing practice of using discretised returns to label documents is not by choice, but by the limitations in the existing implementations of the classifiers — support vector machine is by nature a binary classifier, and most existing naive Bayes classifier implementations produce categorical outputs. Labelling documents with discretised returns, however, may be suboptimal for three reasons:

1. Emotion is hardly a binary phenomenon. Using a binary output for sentiment

classification can lead to the omission of 'the rich, multi-dimensional structure of human mood' [Bollen et al., 2011].

2. The selection of the thresholds or cut-points for discretisation can introduce subjectivity.

3. With a threshold of 10%, a news article accompanied by a daily return of 10.1% will classified as positive, while another article accompanied by a daily return of 9.9% will be labelled as negative, which is arbitrary.

These observations partly motivated the work in this thesis. As will be demonstrated in later chapters, instead of using a threshold, the method developed in thesis models the correspondence between sentiment classes of a document and the accompanying returns using a probability distribution, which alleviates some of the problems mentioned above.

2.3.2 Sentiment Sources

Kearney and Liu [2014], Nassirtoussi et al. [2014] identified and compared three sources for textual sentiment in financial contexts:

Financial and Business News This represents perhaps the most intuitive source of news sentiment and is the main subject of study for this thesis. Financial and business news articles are widely available through various sources: newspapers, websites, newswires, etc. Popular sources include *The Wall Street Journal*, *New York Times* [Tetlock, 2007, Tetlock et al., 2008, García, 2013, Engelberg, 2008], and *The Financial Times* Ferguson et al. [2012].

Corporative Filings Corporative filings include reports and announcements disclosed by companies. Annual or interim reports such as 10-Ks and 10-Qs are popular subject of study [Feldman et al., 2008, Li, 2010, Loughran and McDonald, 2011a,b]. Other studies focused on earning announcements [e.g. Davis et al., 2011, Henry, 2006]. Such reports and announcements are released much less frequently than business news (e.g. quarterly or annually).

User Generated Contents These include discussions made by investors and Internet users about the market. Early studies by Antweiler and Frank [2004], Das and Chen [2007] looked at the postings on message boards such as Yahoo Finance. In recent years, social media has become a major source for user generated discussions; studies by Bollen et al. [2011], Ghiassi et al. [2013], Qasem et al. [2015] investigated the role of Twitter messages in the interactions between sentiment and market.

It has been noted by Kearney and Liu [2014] that financial and business news can be more credible than corporative disclosures because company managements who compose reports and make announcements have the incentive to publish only favourable content; on the other hand, corporative disclosures can contain insider information that is not available elsewhere, which can prove valuable. More importantly, it is observed that news articles are generally recount what has happened in the past rather than what might happen in the future, whereas corporative disclosures contains more forward-looking information [Li, 2010, Kearney and Liu, 2014]. This observation is particularly relevant to this work — it is hypothesised in this thesis that the news sentiment associated with recounts of past events should be modelled separately from the sentiment expressed in forward-looking news.

2.4 Summary

In this chapter, I reviewed some of the common themes in computational sentiment analysis. Two major schools of methods were recognised. First is the pattern-matching based methods, where externally compiled sentiment dictionaries are used to annotated texts and aggregation heuristics are used to derive the sentiment orientations of sentences and documents. Second is the machine learning based methods, where classification algorithms where used to mine and establish the latent relationships between textual features and sentiment.

Also surveyed were the methods previous studies used to investigate the interactions between news sentiment and the financial market. Both pattern-matching-based methods and machine-learning-based methods have seen widespread applications in this regard. Pattern-matching-based methods, when combined with regression analysis, produces relatively easy-to-interpret results. Overall, it has been noted that machine learning based sentiment analysis is becoming increasingly popular in this domain as it is free from the need of sentiment dictionaries, whose compilation can be difficult and expensive. While these approaches have proved successful, it was recognised that the machine learning based methods developed by previous studies have suffered from two problems:

1. Business and financial news, as noted by Kearney and Liu [2014] and others, tend to contain reports on what had already happened in the market. In this thesis, I identified news that is mainly driven by the changes in the market as *retrospective news*. It is reasonable to assume that retrospective news would provide limited insight into future development of the market because it conveys not much information beyond what had already been 'said' by the prices. Including such news, therefore, might 'dilute' the effect of forward-looking information in the news, preventing learning algorithms from discovering predictive associations.

2. When preparing the training dataset for machine learning algorithms, many previous studies used contemporaneous stock returns or index returns to label news documents. For most types of classifiers, the process involves discretising continuous returns into binary (i.e. positive versus negative) or ternary (i.e. positive, negative, and neutral) categories. This is often achieved by applying thresholds or by simply using the signs of returns to derive labels, which can be suboptimal and unintuitive.

3. Most studies have assumed a bag-of-words language model when representing textual content of the news, which can be limiting for sentiment analysis purposes.

The work done for this thesis is largely motivated by attempts to remedy some of these drawbacks identified in these studies — it is hoped that these problems can be mitigated by introducing a more sophisticated model for news sentiment.

Chapter 3

Methods

3.1 Introduction

This chapter outlines the method developed in this thesis. The method performs the following three tasks:

1. Given an arbitrary collection of business news on a company, learn the temporal sentiment orientations of the lexical items in the vocabulary, where the learning is supervised by the historical performances of the company's stock.

2. Once the estimates are obtained, evaluate the temporal sentiment distributions for unseen validation documents.

3. Given the predicted temporal sentiment distributions for unseen validation documents, evaluate how well the predictions are.

The method developed in this chapter belongs to the machine-learning family in terms of its place in the sentiment analysis methodology taxonomy — the sentiment orientations of the elemental lexical units are derived with statistical methods from historical data without resorting to any expert knowledge. It follows the general framework for machine-learning-based sentiment analysis outlined in Section 2.2, and is divided into three phases: learning, classification and evaluation, each corresponds to one of the aforementioned tasks (Figure 3.1).

The learning phase takes two inputs: (i) the inverted positional index database built from the training documents containing news articles on a company of interest, and (ii) the historical prices of the company's stock on some stock exchange. The construction of this index database is covered in Chapter 4. The historical prices of the company's stock were downloaded from Yahoo Finance using the `quantmod` library in R; five price series were retrieved for each company, namely the open, highest, lowest, close, and the adjusted close prices per trading day. The *adjusted close prices* supplied by Yahoo Finance are

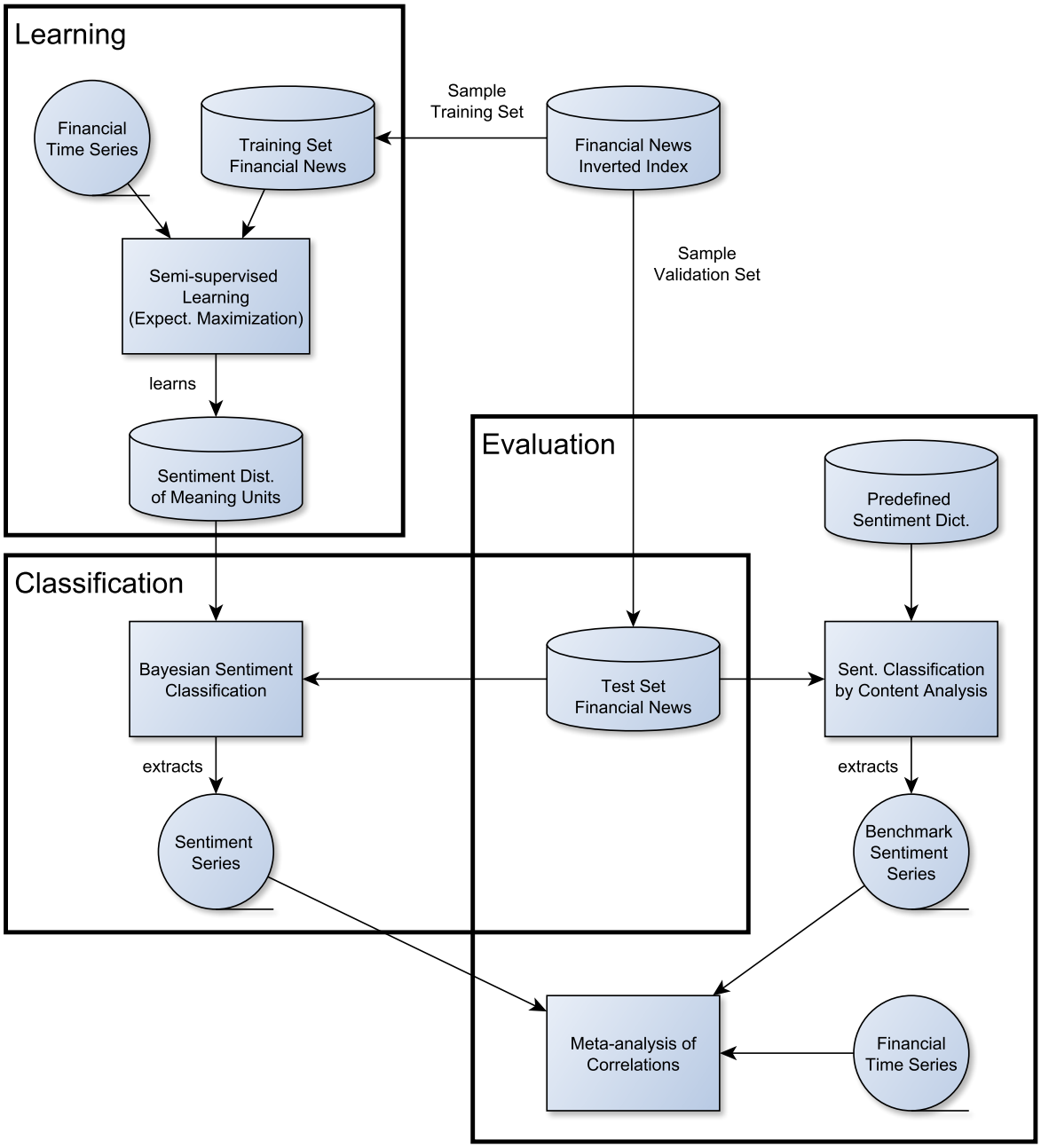


Figure 3.1: Flowchart for the method developed in this thesis

The diagram describes the procedures and resources involved in the method developed in this thesis. The entire process is divided into three sub-processes: (i) learning, where the probability distribution over the four sentiment classes is estimated for each lexical unit in the vocabulary and a sentiment dictionary is produced; (ii) classification, which evaluates the sentiment for any given piece of news using the distributions obtained thereof; (iii) evaluation, where the methodology's predictive performance is compared with other methods.

derived from the original close prices with corrections made to account for stock splits and dividends; it is worth noting that with Yahoo's correction, historical adjusted close prices will change every time a new split or dividend payout happens, therefore it is important to keep note of the date on which the price data are retrieved.

The notion of *temporality* in the context of sentiment analysis in business news has been introduced in Section 1.1 and will be further developed in Section 3.2. The method described in this chapter accounts for this temporal aspect of the sentiment by bringing one new dimension to the classification's target space: the temporality dimension — a piece of news can be categorised into two classes with respect to its temporality: a news article can be considered as *retrospective*, where the content of the news mainly concerns the event happened in the past, or *prospective*, where the article may inspire projections and speculations about the future of the company.

What follows is a brief introduction to the *expectation maximisation* algorithm, which the method developed in this thesis used to estimate the parameters in the proposed model. The algorithm is used to derive the maximum likelihood estimators for the word distributions for each of the temporal sentiment classes. The EM algorithm has been so adopted such that it can take advantage of the continuous nature of the stock returns that supervise the learning. The derivation of the EM algorithm is presented in Section 3.5. The sentiment orientations of unseen documents can be evaluated using standard Bayesian methods based solely on the text it contains (i.e. without market data).

This chapter aims to define a methodological framework for approaching the two tasks set forth earlier this section rather than to describe a system that implements the methods; instead, the implementation details of such a system is presented in Chapter 4.

3.2 Temporality and Sentiment in Business News

In this thesis, I argue that the need for distinguishing between retrospective and prospective sentiment arises due to the inadequacy in discretising equity returns to derive the class labels for the training set when developing statistical classifiers. Recall that in Section 2.2, it has been noted that the application of machine-learning-based sentiment analysis requires a training set — a collection of news articles whose true sentiment orientations have been provided. The advantage of using financial indicators to approximate news sentiment is that the returns would provide the information necessary for an algorithm to automatically derive the labels for the training set. Previous studies, as noted in Section 2.3, have attempted to discretise continuous returns into sentiment classes by applying pre-defined thresholds [Koppel and Shtrimberg, 2004, Généreux et al., 2011, Enric et al., 2014, Li et al., 2014b, Geva and Zahavi, 2014, Nassirtoussi et al., 2015] — news articles are labelled 'positive' if they are published on a day on which the return of the equity of interest is

above the 'positive threshold' and are labelled 'negative' if the return is below the 'negative threshold', and the articles published on a day whose associated stock return falls between the positive and negative thresholds are either discarded or marked 'neutral'.

The challenge here is that the discretisation procedure becomes less confident about its designation of sentiment classes as the return value approaches zero; the occurrence of a small return, regardless of its direction, could merely be the work of stochasticity in the financial market — in fact, the magnitudes of the aforementioned thresholds set by the practitioner of the discretisation reflect how confident she is when interpreting — the lower the threshold, the more ascertainable the relationship between the two quantities. Discounting articles accompanied by smaller returns also implies the assumption of simultaneity: there exists an contemporaneous response between news sentiment and the market's behaviour — if the return on a particular day is under the confidence threshold, it must be the case that the sentiment of the news released on that day is neutral. In reality, however, the assumption of simultaneity rarely holds: news that mirrors the market's movements are not uncommon, and the editorial columns in major business newspapers sometimes speculate on the future prospects of the market. Adding news temporality to the process relaxes the assumption, making it possible to accommodate the smaller returns without discarding articles — that is, news articles released on days with trivial returns (and would otherwise be discarded) are given a 'second chance' by allowing them to re-align with the returns from neighbouring days.

Distinguishing between the two types of temporality can also add to the robustness of the model against inconsistencies found in news data. It has been observed that in some cases the publication dates on news articles collected from popular news repositories can deviate from their actual release dates as found on the websites of their sources. In particular, the publication dates of certain news articles on *Financial Times* and *Wall Street Journal* as collected from the ProQuest and LexisNexis databases appear to be one-day late compared to the corresponding dates found on their respective websites. This discrepancy would affect the correctness of any analysis that requires proper alignment between the release dates of the news and the market indicators. By incorporating news temporality, the articles whose publication dates were wrongly registered would be treated as retrospective and are aligned with the returns from further back in time; in this way, impacts resulted from the inaccuracies can be mitigated. Another scenario which poses a similar challenge relates to the 'news updates', where a news article is in fact an updated or follow-up version of a previously published news piece. These articles are retrospective by nature as the events they describe had already happened — whatever impact they had on the market must have been absorbed in the prices.

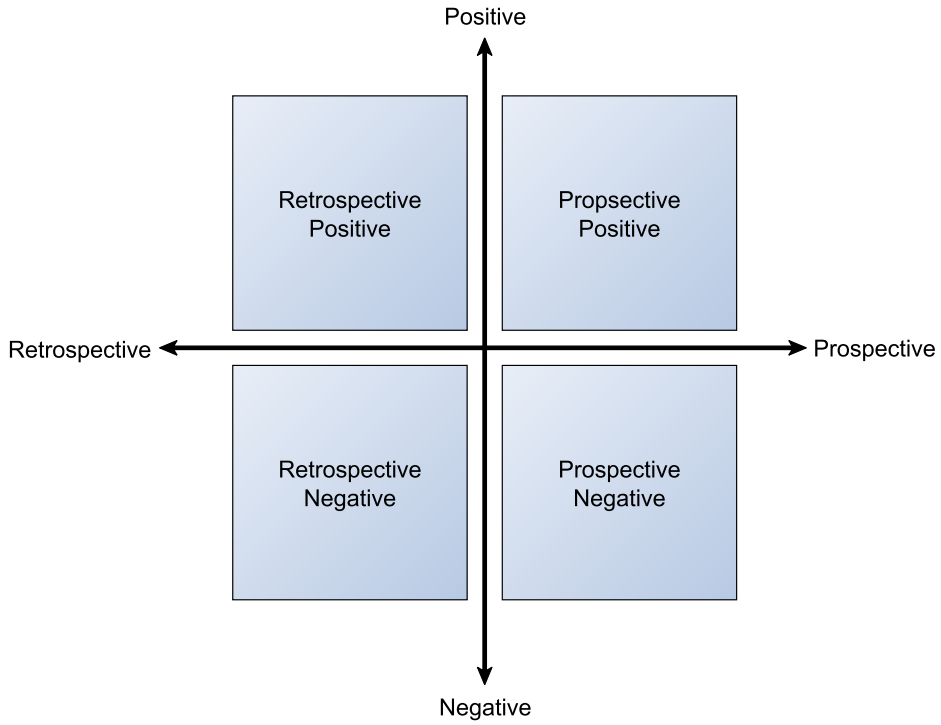


Figure 3.2: Target Space for Business News Classification

This diagram visualizes the target space for classifying business news after being extended by the added temporality dimension. The horizontal axis represents temporality while the vertical axis represents the classic sentiment dimension. The two axes therefore divide the document class space into four partitions, *retrospective negative*, *retrospective positive*, *prospective negative*, and *prospective positive*.

3.2.1 The Additional Dimension: Temporality

The addition of the temporality dimension to the modelling of news sentiment expands the target domain for news sentiment classification. Instead of categorising a piece of news article as either *positive* or *negative*, the article's class is now considered among four categories[1]: *retrospective negative*, *retrospective positive*, *prospective negative*, and *prospective positive* (Figure 3.2). The semantics of each category can be derived by combining the semantics of its sentiment and temporality components — a retrospective negative article on a certain company, for example, reports events happened before the publication of the article (retrospective) and is considered to have negative impacts on the company's performance (negative).

From a mathematical point of view, the expansion to the target class space represents an attempt to restore the original positions of the news articles in a sentiment space of higher dimensions. The model assumes a news article's accompanying return to be the projection of its position in a two-dimensional temporal sentiment space onto a one-

[1]Or six categories in the case with three-way sentiment classification, with the additional two categories being *retrospective neutral* and *prospective neutral*.

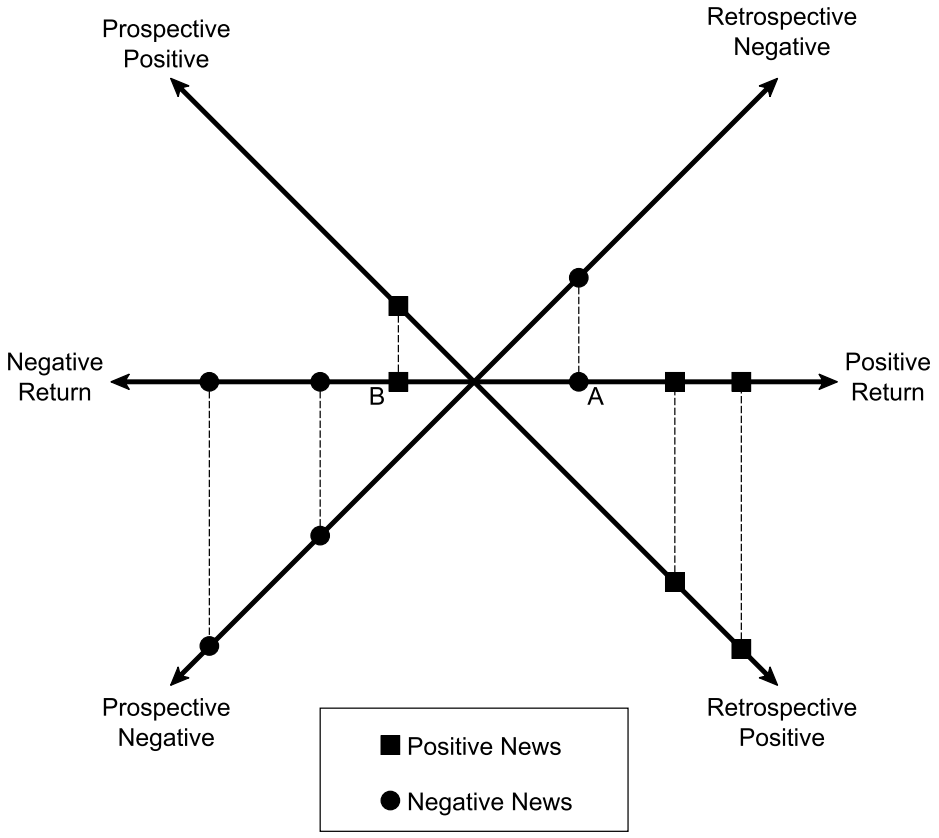


Figure 3.3: Projection from Temporal Sentiment Space to Return Scalar

This schematic diagram provides an intuitive visualisation of the effect the added temporal dimension has on the sentiment classification of business news. Square-shaped points represents instances of positively-toned news for a particular company while dot-shaped points represents negatively-toned news articles for the same company. The horizontal axis is a scale that measures the return of the said company's stock on the day the news article is published — a point positioned to the right-end of the scale, for example, represents an article published on a day when the return of the company's stock is more positive. The two diagonal axes each corresponds to one of the two sentiments decorated by the temporal modifier. What the diagram demonstrates is that news articles placed on the stock return scale are in fact projections from a point located in a space of higher dimension. An article discussing a major plummet seen in some company's stock price from yesterday (i.e. retrospective negative) could be published the next day when the return of the stock is positive (e.g. point A); in other scenarios, progress in a company's long-term strategic plan may fail to attract immediate interests from the investors, and its announcement may coincide with a minor drop in the company's stock price due to other reasons (e.g. point B). In other words, the stock returns accompanying the releases of news can be an inadequate indicator to the affective nature of the news articles.

dimensional return scale (Figure 3.3); like in most projections from a higher to a lower dimension, information is inevitably lost during the process. The goal for the methodology developed in this thesis, therefore, is to recover the lost information as much as possible by iteratively forming informed guess about news articles' true position in the original space.

3.2.2 News Temporality and Tenses Information

It can be tempting to equate news temporality with language tenses. The correspondence between the two are hard to ignore — accounting events in past tenses indicates the events happened in the past, while sentences composed in future tense often convey expectations; however, the correspondence between the two is not absolute. It is noted that language tenses are a phenomenon at the grammatical level and are thus dependent on the properties of the language; news temporality, on the other hand, exists independently of languages and is a phenomenon at the semantic level. One way to see this is to consider an example where a piece of retrospective news in English written in past tense; now imagine the same news is translated into Chinese where a formal notion of tense is lacking: the news would still be retrospective, but its tense has become undefined — the thought experiment exemplifies how the notion of temporality exists independently of language tense.

Despite the independence between tense and temporality, it is recognised in this thesis that the tense used to compose a sentence does hint at the temporality of the writing when tenses are defined for the language used. Past tenses and retrospectivity are closely related, while the correlation between future tense and prospectivity is much subtler — prospective news may be delivered by writings in both future and past tense. The signing of a contract, for example, is typically reported in past tense, but it may yield long-term effects that extend beyond the current time on the operations of the companies involved.

3.2.3 Modelling News Sentiment and Temporality

It is worth noting that the four temporal sentiment categories are *not* mutually exclusive. A single piece of news may address several issues in different parts of the article, each invoking a different type of temporal sentiment; as such, the article can be thought of to belong to more than one temporal sentiment categories. This is especially true for longer articles which typically discuss more than one aspects of a topic.

To reflect this 'mixture' nature of temporal sentiment in news, this thesis proposes that news articles be modelled as mixtures of lexical items drawn from four lexica, each characterising one of the four temporal sentiment categories. Such models have traditionally been approached from a probabilistic perspective — the overall probability of seeing a particular document given a language model is calculated as a weighted sum of the probabilities of seeing the article sampled from the lexical distribution for each of the temporal sentiment categories. Mathematically, the probability mass function for a mixture distribution made of finite components is described as:

$$p\left(x\right) = \sum_{k=1}^{K} \pi_k p\left(x \mid \boldsymbol{\theta}_k\right) \tag{3.1}$$

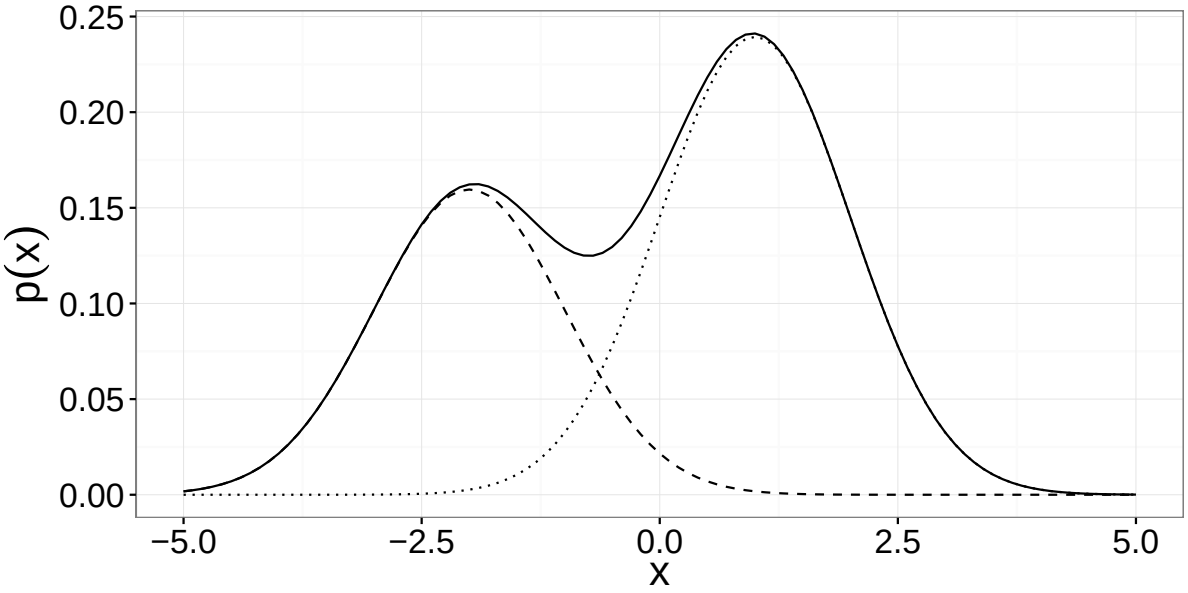


Figure 3.4: An example univariate mixture distribution comprised of two normal distributions

The curve in dashed line represents the p.d.f. (probability density function) for $\mathcal{N}(x \mid -2, 1)$, weighted by 0.4, and the curve in dotted line represents the p.d.f. for $\mathcal{N}(x \mid 1, 1)$, weighted by 0.6. The solid curve is the p.d.f. for the resulting distribution mixture: $p(x) = 0.4 \cdot \mathcal{N}(x \mid -2, 1) + 0.6 \cdot \mathcal{N}(x \mid 1, 1)$.

where x represents a sample from a random variable whose distribution is characterised by a mixture of K distributions; $\boldsymbol{\theta}_k$ denotes the parameters of the probability distribution for the k-th component. $p(x \mid \boldsymbol{\theta}_k)$ is the probability of drawing x from the distribution of the k-th component; π_k represents the k-th distribution's contribution to the mixture, and is subjected to the following constraint:

$$\sum_{k=1}^{K} \pi_k = 1. \tag{3.2}$$

An example univariate mixture distribution comprised of two normal distributions is illustrated in Figure 3.4.

When modelling text, x in Equation 3.1 represents a document, and k indexes the class the document belongs to. The probability of seeing a particular document given its document class (i.e. $p(x \mid \boldsymbol{\theta}_k)$) is often interpreted in a generative narrative, where each document is viewed as a sequence of lexical tokens and each token is drawn from a multinomial distribution defined over the vocabulary. For a document represented as a sequence of tokens $\mathbf{t} = (t_1, t_2, \ldots, t_N)$, the probability of seeing this particular sequence,

assuming independence between the tokens, can be defined as:

$$p\left(\mathbf{t} \mid \boldsymbol{\theta}\right) = \prod_{n=1}^{N} p\left(t_n \mid \boldsymbol{\theta}\right) \tag{3.3}$$

where N is the number of tokens in the document; $\boldsymbol{\theta}$ is the set of parameters governing the probability distribution from which the tokens in the text were drawn. When the tokens are drawn independently from a multinomial distribution, the set of parameters can be written as $\boldsymbol{\theta} = (p_{e_1}, p_{e_2}, \ldots, p_{e_V})$, with e_i denoting the i-th element in the vocabulary of the corpus which the document belong to and p_{e_i} denoting the probability of encountering the i-th lexical item in the vocabulary in the text; V is the size of the vocabulary of the corpus.

An alternative view to the formulation of the model is to treat each document as a frequency vector $(f_{e_1}, f_{e_2}, \ldots, f_{e_V})$, where f_{e_i} represents the raw frequency for the lexical entry e_i in the text (i.e. $\sum_{i=1}^{V} f_{e_i} = N$). For a document whose vector representation is $(f_{e_1}, f_{e_2}, \ldots, f_{e_V})$, the probability of observing a token $\mathbf{t}$ given the underlying parameters being $\boldsymbol{\theta}$ is:

$$p\left(\mathbf{t} \mid \boldsymbol{\theta}\right) = \prod_{i=1}^{V} p\left(e_i \mid \boldsymbol{\theta}\right)^{f_{e_i}}. \tag{3.4}$$

In the case where the token generation process follows a multinomial distribution whose parameters are given by $(p_{e_1}, p_{e_2}, \ldots, p_{e_V})$, the probability of observing a document represented as a frequency vector $(f_{e_1}, f_{e_2}, \ldots, f_{e_V})$ is:

$$p\left(f_{e_1}, f_{e_2}, \ldots, f_{e_V} \mid p_{e_1}, p_{e_2}, \ldots, p_{e_V}\right) = \prod_{i=1}^{V} {p_{e_i}}^{f_{e_i}}. \tag{3.5}$$

Equation 3.1 can now be rewritten as:

$$p\left(\mathbf{t}\right) = \sum_{k=1}^{K} \pi_k \prod_{n=1}^{N} p\left(t_n \mid \boldsymbol{\theta}\right) \tag{3.6}$$

$$= \sum_{k=1}^{K} \pi_k \prod_{i=1}^{V} p\left(e_i \mid \boldsymbol{\theta}\right)^{f_{e_i}} \tag{3.7}$$

$$= \sum_{k=1}^{K} \pi_k \prod_{i=1}^{V} {p_{k,e_i}}^{f_{e_i}} \tag{3.8}$$

where $\mathbf{t}$ is the token sequence for a document of interest; K is the total number of components in the mixture; π_k denotes the overall probability of encountering documents from the k-th class; p_{k,e_i} is the probability of generating the i-th element e_i when the document belongs to class k. When the lexical items being considered are unigrams, the

model is called 'mixture of unigrams' [Nigam et al., 2000], and had been commonly used for topic modelling before the more sophisticated Latent Dirichlet Allocation became popular. The topics are mapped to the components in the mixture and each topic's theme is reflected by the distribution of unigrams for that topic. It should be noted that the model itself is agnostic to the type of lexical items — it is possible to define 'mixture of bi-grams' or 'mixture of word dependencies', etc.

When exploited as a topic modelling tool, the number of the topics, K, does not have a definite value since the researcher would not know how many topics there are in advance. The practitioners of the method therefore will have to assign a value to K based on the his or her understanding of the application. In this case of this thesis, the number of components is fixed at exactly 4. Each of the four components will map to the temporal sentiment categories introduced in Section 3.1. Formally, a token sequence (i.e. document) modelled under this scheme can be expressed as the following:

$$\begin{aligned} p\left(\mathbf{t}\right) = \pi_{\text{retneg}} \prod_{i=1}^{V} {p_{\text{retneg},e_i}}^{f_{e_i}} + \pi_{\text{retpos}} \prod_{i=1}^{V} {p_{\text{retpos},e_i}}^{f_{e_i}} \\ + \pi_{\text{proneg}} \prod_{i=1}^{V} {p_{\text{proneg},e_i}}^{f_{e_i}} + \pi_{\text{propos}} \prod_{i=1}^{V} {p_{\text{propos},e_i}}^{f_{e_i}} \end{aligned} \tag{3.9}$$

with retneg, retpos, proneg and propos denoting 'retrospective negative', 'retrospective positive', 'prospective negative', and 'prospective positive' respectively.

During the training phase of the method where the actual texts are observed, both the elements in the vocabulary (i.e. $e_1, e_2, \ldots, e_V$) and the raw frequencies of the elements (i.e. $f_{e_1}, f_{e_2}, \ldots, f_{e_V}$) in the texts are known values, and the model is fully specified by the parameters. In this case, the parameter space consists of five distributions: (i) the prior distribution for the four temporal sentiment classes: π_{retneg}, π_{retpos}, π_{proneg} and π_{propos} (or $p\left(\mathbf{z}\right)$ according to the Bayesian view developed in subsequent sections); (ii) then, there are the four probability distributions of lexical items for the four temporal sentiment categories, where each distribution contains V parameters. The goal of the learning algorithm is to obtain the estimators for all the five distributions from historical data. Direct estimation of the parameters in mixture models can be difficult; fortunately, methods that are capable of iteratively obtaining these estimators are available. In the next section, I present a brief introduction to the *expectation maximisation* algorithm which is a framework for obtaining the maximum likelihood estimations for the parameters in models with latent structures.

3.3 Mixture Models and the EM Algorithm

This section aims to set forth the background knowledge necessary to obtain maximum likelihood estimates of the parameters in mixture distributions. I will first look at the challenges mixture models pose to the estimation of their parameters; a brief introduction to the *expectation maximisation* algorithm follows, which is a framework for developing iterative procedures which produce the maximum likelihood estimations for the parameters in mixture models. The expectation maximisation algorithm (EM algorithm) serves as the foundation for the methodologies being developed in this thesis. The discussion in this section largely follows that presented in [Bishop, 2007], but is complemented with additional details and comments emphasizing its relevancy to the application in this thesis.

In the machine learning and text mining literature, mixture models are often related with unsupervised learning — clustering in particular — where observations that share similar characteristics are grouped together. The parameters in such models can be obtained using the *maximum likelihood estimation* procedure; intuitively, this procedure seeks the particular configuration of the parameters that maximises the likelihood function of the model 'generating' the observations. Formally, the maximum likelihood estimation for $\boldsymbol{\theta}_{\mathrm{ML}}$ is given by:

$$\boldsymbol{\theta}_{\mathrm{ML}} = \underset{\theta}{\operatorname{argmax}}\, p\left(\mathbf{X} \mid \boldsymbol{\theta}\right) \tag{3.10}$$

where $\mathbf{X}$ denotes the observations; $\boldsymbol{\theta}$ represents the parameters that govern the behaviour of the model and $p\left(\mathbf{X} \mid \theta\right)$ is the *likelihood function*. Note that the likelihood function is a function of θ, not of the data $\mathbf{X}$. Solution to this optimisation problem can be obtained by taking the derivative of the likelihood function with respect to the parameters, setting it to zero, then solving for the roots. In cases where the observations in $\mathbf{X}$ are independent, the maximum likelihood estimator for $\boldsymbol{\theta}$ can be factorised as follows:

$$\boldsymbol{\theta}_{\mathrm{ML}} = \underset{\theta}{\operatorname{argmax}} \prod_{\mathbf{x} \in \mathbf{X}} p\left(\mathbf{x} \mid \boldsymbol{\theta}\right). \tag{3.11}$$

In practice, most problems that feature independent data samples can be translated into forms similar to the one described by Equation 3.11. Direct optimisation of a product of functions involves recursive applications of the chain rule, which can be cumbersome; it can also lead to numerical instability for even moderately large dataset because the product of a large number of probabilities can cause overflow. A widely used alternative that greatly simplifies the calculation exploits the fact that maximising the likelihood function is equivalent to maximising the logarithm of the likelihood function due to the

monotonic nature of the logarithm function:

$$\boldsymbol{\theta}_{\mathrm{ML}} = \underset{\boldsymbol{\theta}}{\operatorname{argmax}} \prod_{\mathbf{x}\in\mathbf{X}} p\left(\mathbf{x} \mid \boldsymbol{\theta}\right) \tag{3.12}$$

$$= \underset{\boldsymbol{\theta}}{\operatorname{argmax}} \ln \left\{ \prod_{\mathbf{x}\in\mathbf{X}} p\left(\mathbf{x} \mid \boldsymbol{\theta}\right) \right\} \tag{3.13}$$

$$= \underset{\boldsymbol{\theta}}{\operatorname{argmax}} \left\{ \sum_{\mathbf{x}\in\mathbf{X}} \log p\left(\mathbf{x} \mid \boldsymbol{\theta}\right) \right\} \tag{3.14}$$

which leads to a solution with better numerical stability as the algorithm is now operating with sums rather than products of small numbers.

For complex models, however, direct maximisation of the likelihood functions may not yield closed form estimators for the parameters. The proof for the correctness of the EM algorithm is not presented here but is attached as Appendix A. The procedures for the general EM algorithm is summarized as follows:

1. Choose an initial (usually random) settings for the parameters $\boldsymbol{\theta}^{\mathrm{old}}$.

2. The **E Step**: evaluate the posterior distribution $p\left(\mathbf{Z} \mid \mathbf{X}, \boldsymbol{\theta}^{\mathrm{old}}\right)$.

3. The **M Step**: compute $\boldsymbol{\theta}^{\mathrm{new}}$ as

$$\boldsymbol{\theta}^{\mathrm{new}} = \underset{\boldsymbol{\theta}}{\operatorname{argmax}}\, \mathcal{Q}\left(\boldsymbol{\theta}, \boldsymbol{\theta}^{\mathrm{old}}\right) \tag{3.15}$$

   where

$$\mathcal{Q}\left(\boldsymbol{\theta}, \boldsymbol{\theta}^{\mathrm{old}}\right) = \sum_{\mathbf{Z}} p\left(\mathbf{Z} \mid \mathbf{X}, \boldsymbol{\theta}^{\mathrm{old}}\right) \ln p\left(\mathbf{X}, \mathbf{Z} \mid \boldsymbol{\theta}\right). \tag{3.16}$$

4. Check for convergence of either the log-likelihood or the parameter values. The algorithm can be considered to have converged if the relative increase in the log-likelihood between two iterations is smaller than a pre-defined threshold[2]. If the convergence criterion (e.g. the relative increase in the log-likelihood is greater than the threshold) is not satisfied, then update

$$\boldsymbol{\theta}^{\mathrm{old}} \leftarrow \boldsymbol{\theta}^{\mathrm{new}} \tag{3.17}$$

   and go to step 2. If the algorithm successfully converges, exit the process.

The next section, I apply the EM algorithm to solve the parameters in the modelling of sentiment and temporality in business news using the procedures developed in this section.

[2]Other convergence criteria can be used, e.g., the relative changes in the parameters being smaller than a threshold.

3.4 Return-supervised Parameter Learning

In Section 3.2, I have demonstrated how temporality and sentiment in business news can be modelled as a mixture of four probability distributions over lexical items; also introduced in Section 3.3 was the *expectation maximisation* algorithm, which provides the means necessary for estimating the parameters in mixture models. In this section, I will contemplate the role of equity returns in the learning of the parameters for the temporal sentiment mixture model.

3.4.1 Parameter Learning for Standard Mixture Model

Before proceeding, it is helpful to recount the generative story associated with the mixture model proposed in Section 3.2.3. The original mixture model is illustrated in Figure 3.5 using the *plate notation*. The plate notation is a common method for representing the structures of probabilistic models. In plate notation, the circles represent random variables in the model, and the rectangles or plates specify the indices. If a circle is placed within a plate, the random variable represented by the circle is then indexed by the counter shown in the said plate. Plates can also intersect or nest; in such cases, if a circle lies within multiple plates, then the random variable it represents would be indexed by the counters specified by all the enclosing plates. Arrows between the circles represent conditional dependencies between the random variables — for example, an arrow pointing from node $\mathbf{X}$ to node $\mathbf{Y}$ would correspond to the conditional probability distribution $P(\mathbf{Y} \mid \mathbf{X})$.

The diagram shows that the generation process begins with $\boldsymbol{\pi}$, which represents the overall distribution of the topics in the corpus. For each of the N documents in the corpus, a topic is chosen from the multinomial distribution whose parameter is given by $\boldsymbol{\pi}$ and the binary vector representation of that topic is denoted by $\mathbf{z}_i$ (as discussed in the previous section). For each position j inside the document, a lexical item is again drawn from a multinomial distribution over the vocabulary in the corpus; parameters of this multinomial distribution are given by $\boldsymbol{\beta}_k$ where k is index of the active topic chosen for the position (i.e. the k-th component of $\mathbf{z}_i$ is 1). The shaded nodes, in this case t_{ij}, are observed variables, while the unshaded nodes represent unknown random variables to be estimated.

Estimation of the parameters using the EM algorithm typically begins with the specification of the $\mathcal{Q}\left(\boldsymbol{\theta}, \boldsymbol{\theta}^{\text{old}}\right)$ function. Recall that in Section 3.3, it had been defined:

$$\mathcal{Q}\left(\boldsymbol{\theta}, \boldsymbol{\theta}^{\text{old}}\right) = \sum_{\mathbf{Z}} p\left(\mathbf{Z} \mid \mathbf{X}, \boldsymbol{\theta}^{\text{old}}\right) \ln p\left(\mathbf{X}, \mathbf{Z} \mid \boldsymbol{\theta}\right) \tag{3.18}$$

For the the model described in Figure 3.5, the complete-data likelihood function,

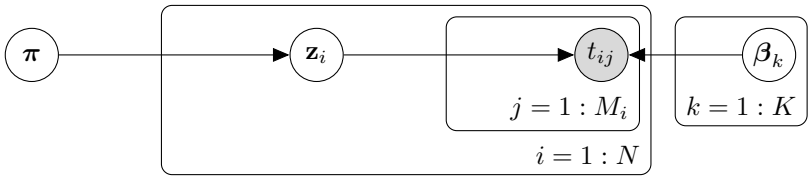


Figure 3.5: Standard mixture of lexical items.

This diagram describes the relationships between the variables in a standard topic mixture model for texts using the plate notation. In this diagram, $\mathbf{z}_i$ is the hidden 'topic' variable governing the generation of the lexical tokens (i.e. t_{ij}) for each position in document j. N is the total number of documents; M_i is the total number of running terms in document i.

$p(\mathbf{X}, \mathbf{Z} \mid \boldsymbol{\theta})$, can be written as:

$$p(\boldsymbol{\mathcal{D}}, \mathbf{Z} \mid \boldsymbol{\pi}, \boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_K) = p(\mathbf{Z} \mid \boldsymbol{\pi})\, p(\boldsymbol{\mathcal{D}} \mid \mathbf{Z}, \boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_K) \tag{3.19}$$

$$= \prod_{i=1}^{N} p(\mathbf{z}_i \mid \boldsymbol{\pi}) \prod_{j=1}^{M_i} p(t_{ij} \mid \mathbf{z}_i, \boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_K) \tag{3.20}$$

where $\boldsymbol{\mathcal{D}}$ represents the entire collection of documents in the corpus of interest. The probability distribution over the possible configurations of the hidden variable $\mathbf{Z}$ can be expressed through the use of Bayes' theorem:

$$p\left(\mathbf{Z} \mid \boldsymbol{\mathcal{D}}, \boldsymbol{\pi}^{\text{old}}, \boldsymbol{\beta}_1^{\text{old}}, \ldots, \boldsymbol{\beta}_K^{\text{old}}\right) = \frac{p\left(\boldsymbol{\mathcal{D}}, \mathbf{Z} \mid \boldsymbol{\pi}^{\text{old}}, \boldsymbol{\beta}_1^{\text{old}}, \ldots, \boldsymbol{\beta}_K^{\text{old}}\right)}{p\left(\boldsymbol{\mathcal{D}} \mid \boldsymbol{\pi}^{\text{old}}, \boldsymbol{\beta}_1^{\text{old}}, \ldots, \boldsymbol{\beta}_K^{\text{old}}\right)} \tag{3.21}$$

$$= \frac{p\left(\boldsymbol{\mathcal{D}}, \mathbf{Z} \mid \boldsymbol{\pi}^{\text{old}}, \boldsymbol{\beta}_1^{\text{old}}, \ldots, \boldsymbol{\beta}_K^{\text{old}}\right)}{\sum_{\mathbf{Z}} p\left(\boldsymbol{\mathcal{D}}, \mathbf{Z} \mid \boldsymbol{\pi}^{\text{old}}, \boldsymbol{\beta}_1^{\text{old}}, \ldots, \boldsymbol{\beta}_K^{\text{old}}\right)} \tag{3.22}$$

The algorithm can then commence as specified in Section 3.3.

Intuitively, the model attempts to capture the clustering of lexical items based on their co-occurrences in the documents — it is assumed that lexical items in the same document tends to be generated by a common set of themes found in the document, and lexical items which hallmark a topic are assigned higher probability in the distribution that characterise the said topic. According to this interpretation, the learning process is completely unsupervised because no information about the targets is presented in the model, and the estimation of the parameters is influenced only by the observed variable t, namely the lexical items in the documents. Without supervision, the themes or topics extracted will not reflect any of the four temporal sentiment categories defined in Section 3.2.

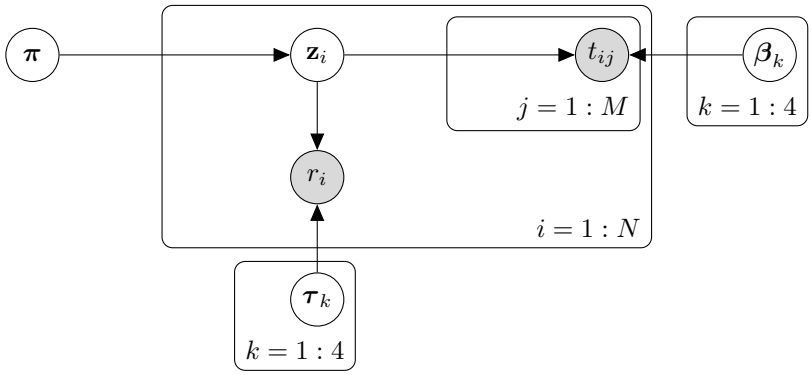


Figure 3.6: Mixture of lexical items supervised by concurrent returns.

3.4.2 Supervised Parameter Learning with contemporaneous Returns

Sentiment proxies are used to enforce supervision in the learning of the parameters; in this case, stock returns are used to approximate news sentiment. One option, as discussed earlier this chapter, is to approximate the sentiment carried in a news article with respect to a particular company with the daily return of the company's stock on the day the article is published. For the sake of clarity, I will begin with a strategy that considers only the sentiment aspect of the supervision first, and move to a more complex model that incorporate both sentiment and temporality when estimating the parameters. The integration of supervision over the learning of sentiment results in a model which is slightly more sophisticated than the unsupervised version and is illustrated by Figure 3.6.

Two new variables, r_i and $\boldsymbol{\tau}_k$, are introduced into the model. The variable r_i denotes the equity return associated with the i-th news article and constitutes part of the observations. $\boldsymbol{\tau}_k$ represents the set of parameters for the probability distribution of returns given that the underlying temporal sentiment category is k.

The generative story associated with this model now includes an additional process: once a temporal sentiment category $\mathbf{z}_i$ is chosen for a news article, the process generates the return for that article according to some distribution of returns. The particular distribution used for generating the returns varies based on the active temporal sentiment category, much like the case when drawing the lexical tokens — it would be more likely to see a positive return being generated if the underlying category come from one of the positive categories, namely the retrospective positive and prospective positive categories; likewise, if $\mathbf{z}_i$ is chosen to be either the retrospective negative or the prospective negative, the return generated will bias towards the negative side. In other words, each of the four temporal sentiment categories is assigned a probability distribution over the returns that is consistent with its semantics, and the return distribution together with the actual return observation supervise the learning of the parameters in this model.

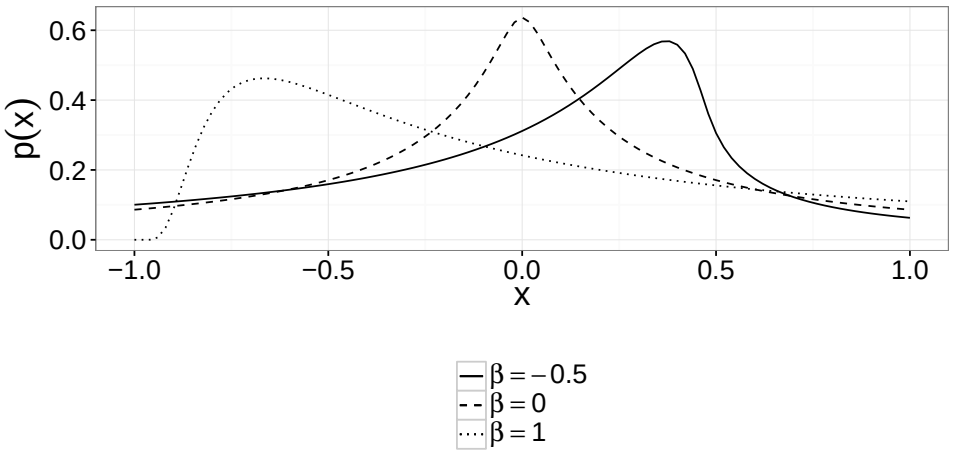
The selection of the family of the probability distributions over the daily returns is therefore crucial to the development of this method. Unfortunately, there has not been any conclusive evidence with respect to the shape of the distribution of returns when the market is being influenced by news sentiment, but previous research did suggest that the distribution of unconditional daily returns of stocks follows the *Stable Paretian Distribution* [Fama, 1965a]. A Stable Paretian Distribution, according to Fama [1965a], is 'any distribution that is stable or invariant under addition. That is, the distribution of sums of independent, identically distributed, stable Paretian variables is itself stable Paretian and, except for origin and scale, has the same form as the distribution of the individual summands'. The distribution is favoured for modelling returns because its property conforms with that of logarithmic equity returns — the return of a equity during a longer period of time can be regarded as a sum of returns for the shorter periods which constitute the longer interval[3].

The shape and properties of a stable distribution are controlled by four parameters [Nolan, a]: (i) the stability parameter α, which must be in the range $0 < \alpha \leq 2$; the stable distribution corresponds to a Gaussian distribution when $\alpha = 2$, and a Cauchy distribution when $\alpha = 1$. (ii) the skewness parameter β, which must be in the range $-1 \leq \beta \leq 1$; (iii) γ is a positive number that determines the scale of the distribution; (iv) a location parameter δ, which shifts the distribution to the right if $\delta > 0$, and left if $\delta < 0$. Further discussion on the theoretical properties of the distribution is beyond the scope of this thesis; instead, I will be focusing on applied aspects of the distribution.

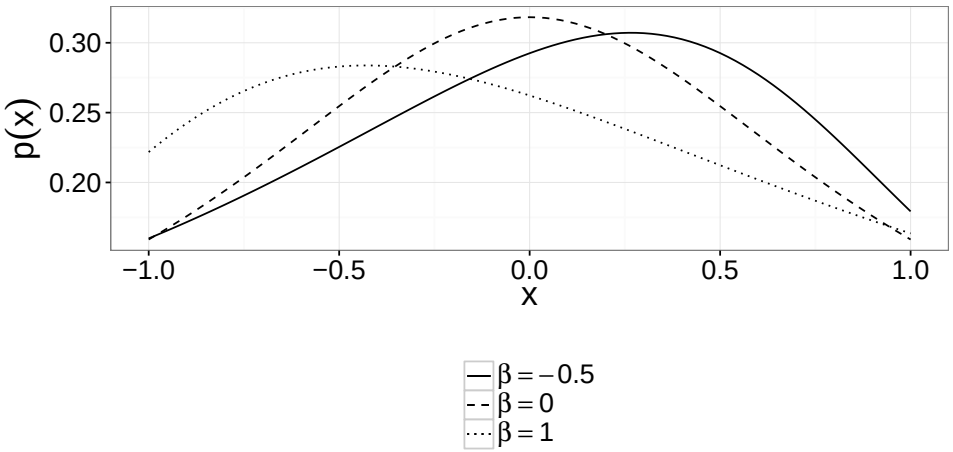
The stable distribution is particularly appealing when modelling asset returns for a practical reason: it allows for asymmetric skewness through the adjustment of the parameter β, which makes it suitable to modelling the biases seen in the return distribution for the temporal sentiment categories. The distribution is symmetric when $\beta = 0$, skewed towards the right if $\beta > 0$, and skewed towards the left when $\beta < 0$. A visualisation to stable distributions' probability density functions (p.d.f.) at various parameter configurations is presented in Figure 3.7. The p.d.f. of the stable distribution will be referred to as $\mathcal{S}(x \mid \alpha, \beta, \delta, \gamma)$ hereafter.

Unlike other parameters in this model, the parameters of the distributions for the returns given the temporal sentiment categories (i.e. $\boldsymbol{\tau}_{\text{rn}}$, $\boldsymbol{\tau}_{\text{rp}}$, $\boldsymbol{\tau}_{\text{pn}}$, $\boldsymbol{\tau}_{\text{pp}}$) shall remain unchanged during each iteration of the EM algorithm, and it is exactly by keeping these parameters constant that the learning process is made supervised — the algorithm would 'encourage' the remaining parameters (i.e. for the distribution over the categories as well as the distributions for the lexical items for each category) to be configured in such a way that it would generate texts and returns that show better 'affinity' with the actual

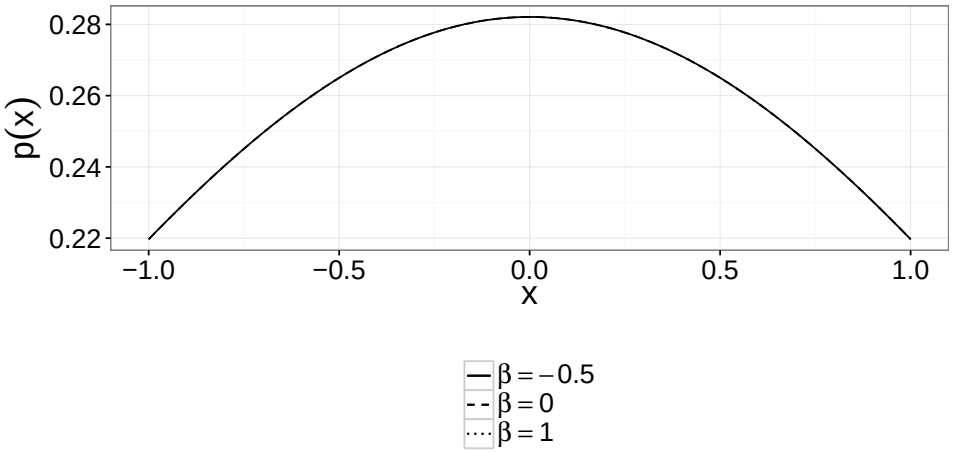
[3]Note that the stable distribution is not the only choice when modelling returns. Another distribution family that shares similar properties (i.e. fat-tailed and allows for asymmetric skewness) with the stable distribution is the Skewed Normal Distribution.



(a) The shapes of the stable distribution when $\alpha = 0.5$, $\delta = 0$, and $\gamma = 1.0$.



(b) The shapes of the stable distribution when $\alpha = 1$, $\delta = 0$, and $\gamma = 1.0$.



(c) The shapes of the stable distribution when $\alpha = 2$, $\delta = 0$, and $\gamma = 1.0$.

Figure 3.7: The probability density functions of the stable distribution over various configurations of the parameters.

observations, and penalise the configurations which are more likely to generate texts and returns that contradict with the actual observations.

Since the distributions of returns fulfil the role of the supervisor, the values for the parameters in these distributions have to be determined before the learning process can proceed. Ideally, only the value for the position parameter δ will have to be fixed — for example, a positive temporal sentiment category will be assigned a return distribution that is shifted to the positive side to reflect the intuition that it is more likely to see positive returns when positive sentiment prevails — and leave the estimations of α, β, and γ to the EM algorithm. In practice, however, the maximum likelihood estimators for the parameters in stable distribution do not yield closed form solutions and can only be approximated through numerical methods, making the application of the EM framework on these parameters impractical[4]. It is therefore necessary to specify the values for the parameters for each of the categories explicitly. For the method developed in this thesis, the parameter values were selected to reflect the following properties:

1. It should be more likely to observe moderate returns than extreme ones (for both positive and negative sentiment categories).

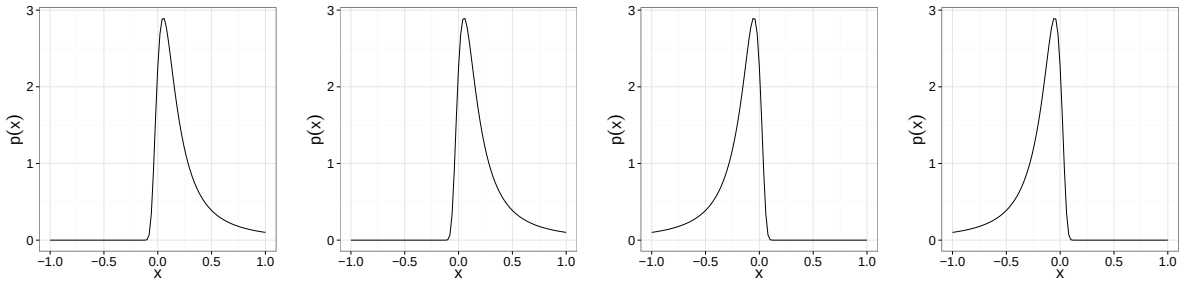
2. The distribution should accommodate extreme values with fat tails on the side consistent with the sentiment component of the category it is representing; that is, a fat tail on the positive side if the distribution is for positive categories and the opposite for negative categories.

3. It should be *possible* to observe small positive returns when the underlying category is one of the negative categories; the same applies for negative returns when the underlying sentiment category is positive. This is to account for the situation where a news article is seen published on a day with negative return while its text strongly suggests positive tone. In this case, the document should still have a good chance of being categorised into one of the positive categories.

4. The chance for observing negative returns when the underlying sentiment is positive should decrease rapidly as the magnitude of the return increases; the same applies for positive returns when the underlying sentiment is negative.

Figure 3.8 illustrates the probability density functions for the four return distributions with an example set of parameters selected following the aforementioned guidelines. Temporality was not considered when modelling the returns, so the retrospective and prospective return distributions within the same sentiment category share the same set of parameters.

[4]It may be possible to utilise the numerical ML estimators when maximizing the parameters during the M step when applying EM.



(a) Retrospective positive (b) Prospective positive (c) Retrospective negative (d) Prospective negative

Figure 3.8: Distributions of returns for the four sentiment categories without the temporality components

The p.d.f. for the return distributions associated with retrospective and prospective positive categories is generated with the following parameters: $\alpha = 0.9$, $\beta = 1.0$, $\delta = 0.1$, and $\gamma = 0.1$; the parameters for the negative categories are: $\alpha = 0.9$, $\beta = -1.0$, $\delta = -0.1$, and $\gamma = 0.1$. It is worth noting that when a positive return distribution is combined with a mirrored negative return distribution, the resulting distribution will be a symmetric stable distribution.

The EM algorithm for this iteration of the model operates much like the one developed for the standard mixture model but with a few modifications made to Equation 3.20 to incorporate the newly added return distributions:

$$p\left(\boldsymbol{\mathcal{D}}, \mathbf{R}, \mathbf{Z} \mid \boldsymbol{\pi}, \boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_K\right) = p\left(\mathbf{Z} \mid \boldsymbol{\pi}\right) p\left(\mathbf{R} \mid \mathbf{Z}\right) p\left(\boldsymbol{\mathcal{D}} \mid \mathbf{Z}, \boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_K\right) \tag{3.23}$$

$$= \prod_{i=1}^{N} p\left(\mathbf{z}_i \mid \boldsymbol{\pi}\right) p\left(r_i \mid \mathbf{z}_i\right) \prod_{j=1}^{M_i} p\left(t_{ij} \mid \mathbf{z}_i, \boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_K\right) \tag{3.24}$$

where $\mathbf{R}$ represents the series of returns from all documents (i.e. $r_1, r_2, \ldots, r_N$). Similarly, the posterior distribution for the configuration of the hidden variable $\mathbf{Z}$ needs to be updated to include the return variable:

$$p\left(\mathbf{Z} \mid \boldsymbol{\mathcal{D}}, \mathbf{R}, \boldsymbol{\pi}^{\text{old}}, \boldsymbol{\beta}_1^{\text{old}}, \ldots, \boldsymbol{\beta}_K^{\text{old}}\right) = \frac{p\left(\boldsymbol{\mathcal{D}}, \mathbf{R}, \mathbf{Z} \mid \boldsymbol{\pi}^{\text{old}}, \boldsymbol{\beta}_1^{\text{old}}, \ldots, \boldsymbol{\beta}_K^{\text{old}}\right)}{p\left(\boldsymbol{\mathcal{D}}, \mathbf{R} \mid \boldsymbol{\pi}^{\text{old}}, \boldsymbol{\beta}_1^{\text{old}}, \ldots, \boldsymbol{\beta}_K^{\text{old}}\right)} \tag{3.25}$$

$$= \frac{p\left(\boldsymbol{\mathcal{D}}, \mathbf{R}, \mathbf{Z} \mid \boldsymbol{\pi}^{\text{old}}, \boldsymbol{\beta}_1^{\text{old}}, \ldots, \boldsymbol{\beta}_K^{\text{old}}\right)}{\sum_{\mathbf{Z}} p\left(\boldsymbol{\mathcal{D}}, \mathbf{R}, \mathbf{Z} \mid \boldsymbol{\pi}^{\text{old}}, \boldsymbol{\beta}_1^{\text{old}}, \ldots, \boldsymbol{\beta}_K^{\text{old}}\right)} \tag{3.26}$$

It should be noted that the contributions made to the complete-data likelihood by the textual component of the observation (i.e. $p\left(\boldsymbol{\mathcal{D}} \mid \mathbf{Z}, \boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_K\right)$) overweight that made by the returns (i.e. $p\left(\mathbf{R} \mid \mathbf{Z}\right)$). This has profound implications on the outcomes from the parameter learning process; in particular, this means that the penalty a parameter configuration receives for generating 'wrong' returns tends to be much smaller compared with that received for generating 'wrong' texts. The difference between the two contributions depends on the number of terms in the textual component in the observations

— the more terms the text contains, the less the returns contribute to the likelihood. Another way to see this is to think the observed returns as 'pseudo-terms', only that the 'vocabulary' consists of an infinite number of such 'pseudo-terms'. According to this view, the temporal sentiment category of a document governs the generation of both the real terms and pseudo-terms in the dataset. However, as one can see from Equation 3.24, the real terms in most cases will outnumber the pseudo-terms due to the fact that $M_i \geq 1$ — the pseudo-terms' influence on the training process will be overwhelmed by that exerted by the real terms. In practical terms, the imbalance would cause the term distributions learnt from the dataset to mainly reflect the co-occurrences between the terms rather than their associations with the sentiment proxies, thus diminishing the effectiveness of the supervised learning.

One way to redress this issue is to adjust the weighting of the pseudo-terms to a scale that is comparable to that of the real terms — instead of generating exactly one pseudo-term per document, an equal number of real terms and pseudo-terms are generated for the same document. More specifically, the generative process is modified to produces M_i copies of the original pseudo-term for each document. Let $\mathbf{R}'$ denote the set of returns associated with the documents with their values replicated M_i times for each document, the updated version of the complete-data likelihood function now becomes:

$$p\left(\boldsymbol{\mathcal{D}}, \mathbf{R}', \mathbf{Z} \mid \boldsymbol{\pi}, \boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_K\right) = p\left(\mathbf{Z} \mid \boldsymbol{\pi}\right) p\left(\mathbf{R}' \mid \mathbf{Z}\right) p\left(\boldsymbol{\mathcal{D}} \mid \mathbf{Z}, \boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_K\right) \tag{3.27}$$

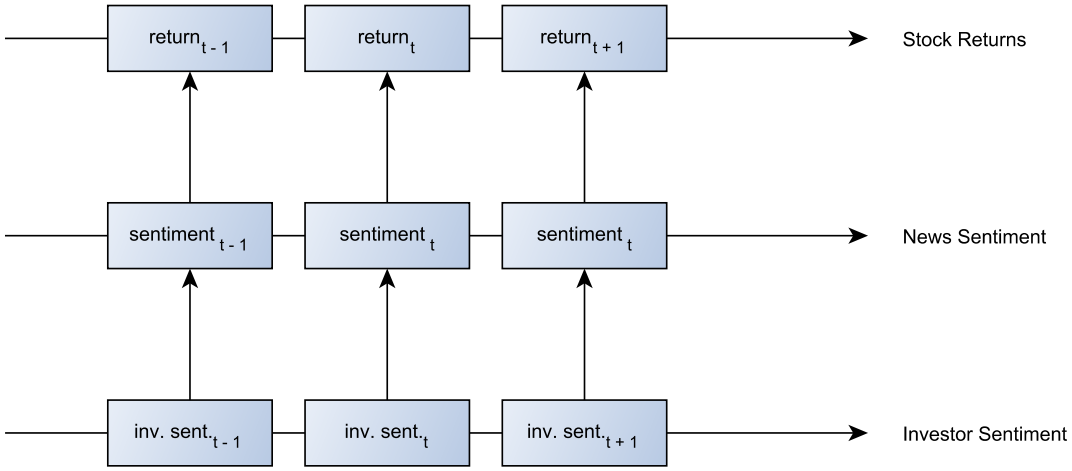
$$= \prod_{i=1}^{N} p\left(\mathbf{z}_i \mid \boldsymbol{\pi}\right) p\left(r_i \mid \mathbf{z}_i\right)^{M_i} \prod_{j=1}^{M_i} p\left(t_{ij} \mid \mathbf{z}_i, \boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_K\right) \tag{3.28}$$

In practice, the amount of the scaling on returns' contribution to the log-likelihoods does not have to match that of the texts. The optimal value for the weighting may have to be determined empirically.

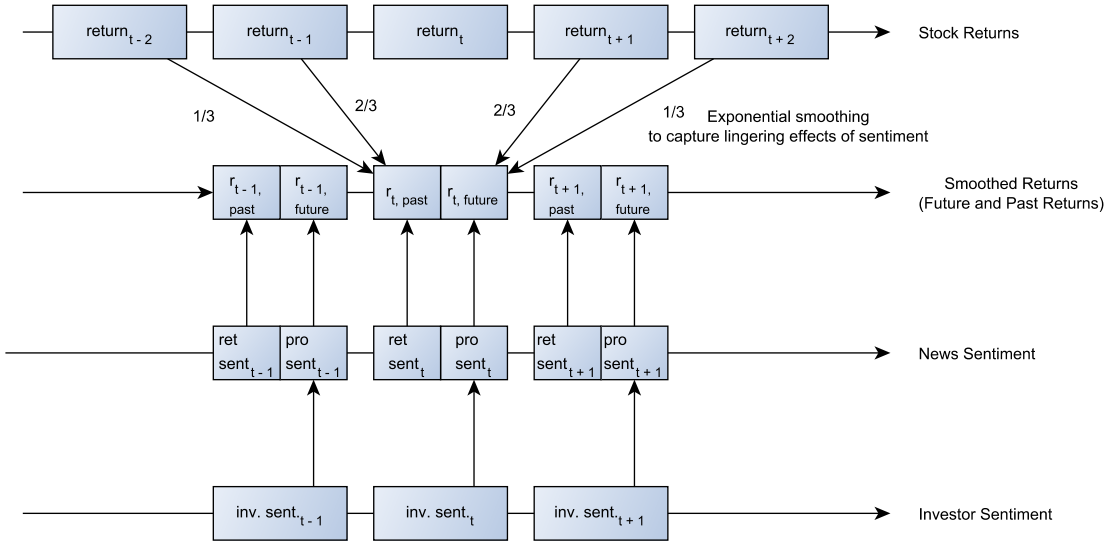
In the following section, I discuss the learning of the parameters for the temporal components of the model. This is achieved by evolving the supervision process described in this section to account for the asynchrony between news sentiment and its proxies.

3.4.3 Supervised Parameter Learning with Past and Future Returns

Having considered contemporaneous returns, I now move to address the temporal aspect of the model. In this case, it is necessary to consider not just the returns of the equity on the same day as the news is released, but also the returns before and after the publication of the news (Figure 3.9). The rationale is that the distributions for the *future returns* will be asymmetrically skewed if and only if the underlying temporality of the sentiment is



(a) Contemporaneous approach to the mapping between news articles and market returns

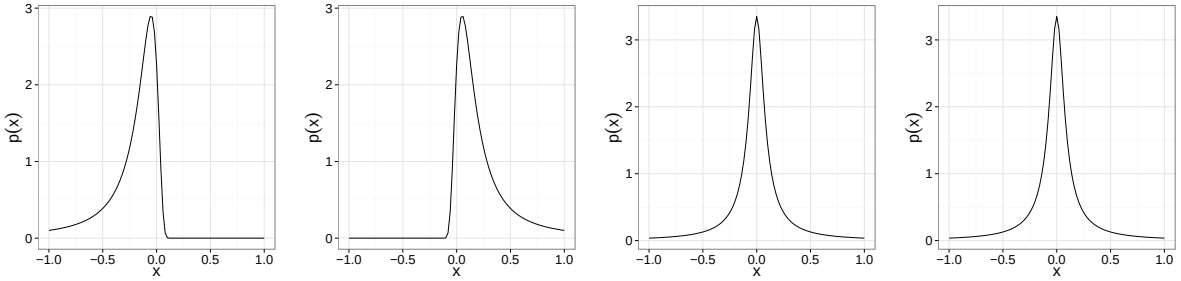


(b) The mapping approach proposed for this method

Figure 3.9: A comparison between the approaches to the mapping between news sentiment and returns

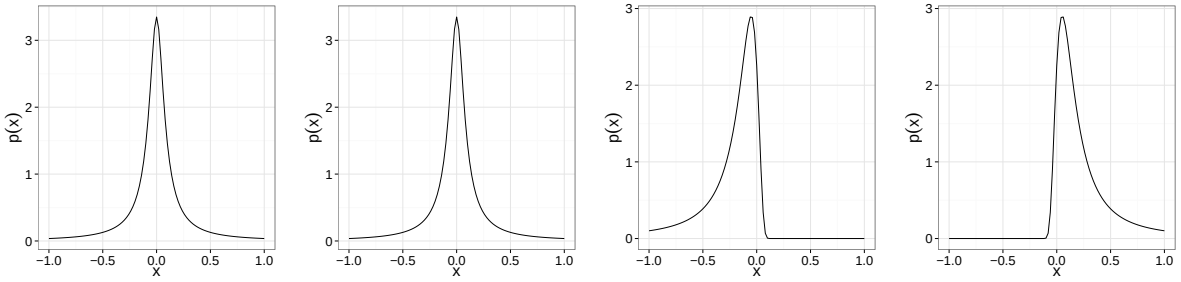
prospective; likewise, the asymmetry in the distributions of the *past returns* responds only to retrospective sentiments. In light of this, two sets of return distributions are developed, one for the past returns (Figure 3.10a) and the other for the future returns (Figure 3.10a).

It should be clarified that, at the conceptual level, the definition of prospective news sentiment only specify what it is (news sentiment that can potentially influence the future development of the market), but is not dependent on how the sentiment language patterns are learnt. As such, the fact that historical "future" stock market movements are used to help identifying language patterns that indicate historical prospective sentiment does not in any way imply that stock market movements *determine* or *define* prospective news sentiment — this precludes the possibility of circularities in the definition of prospective



(i) Retrospective negative (ii) Retrospective positive (iii) Prospective negative (iv) Prospective positive

(a) Example distributions of *past returns* for the four sentiment-temporality categories



(i) Retrospective negative (ii) Retrospective positive (iii) Prospective negative (iv) Prospective positive

(b) Example distributions of *future returns* for the four sentiment-temporality categories

Figure 3.10: Example distributions of *future* and *past* returns for each of the four sentiment temporality categories.

In Figure 3.10a, it is assumed that prospective sentiments impose no influence on the past returns; as a result, the return distributions are designed to be symmetric for prospective sentiment categories. Similarly, in Figure 3.10b, the distributions of future returns depend mostly on prospective sentiments. However, it may be argued that retrospective sentiments can have some influence on the returns from subsequent days, in which case the distributions of returns for the retrospective sentiments would skew slightly.

sentiment.

The updated structure of this new model is shown in Figure 3.11. Two return variables are generated for each document i instead of just one as the case in the previous iteration of the model, and their values are determined by the hidden variable $\mathbf{z}_i$ and the parameters $\boldsymbol{\tau}$ which governs the shape of the return distributions for each of the temporal sentiment categories. The complete-data likelihood function for this temporality-enabled model can

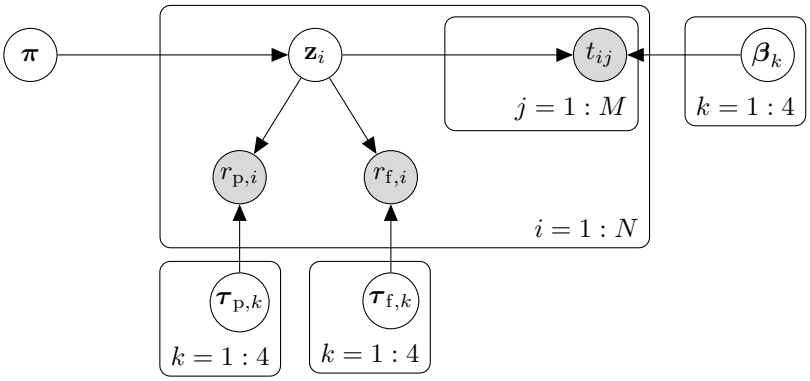


Figure 3.11: Mixture of lexical items supervised by past and future returns.

be written down along the same lines as in Equation 3.28:

$$p\left(\boldsymbol{\mathcal{D}}, \mathbf{R}', \mathbf{Z} \mid \boldsymbol{\pi}, \boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_K\right) = p\left(\mathbf{Z} \mid \boldsymbol{\pi}\right) p\left(\mathbf{R}' \mid \mathbf{Z}\right) p\left(\boldsymbol{\mathcal{D}} \mid \mathbf{Z}, \boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_K\right) \tag{3.29}$$

$$= \prod_{i=1}^{N} p\left(\mathbf{z}_i \mid \boldsymbol{\pi}\right) p\left(r_{\mathrm{f},i} \mid \mathbf{z}_i\right)^{M_i/2} p\left(r_{\mathrm{p},i} \mid \mathbf{z}_i\right)^{M_i/2} \tag{3.30}$$

$$\prod_{j=1}^{M_i} p\left(t_{ij} \mid \mathbf{z}_i, \boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_K\right) \tag{3.31}$$

Again, the weightings for the returns are rescaled so that their contributions to the supervision process become comparable to that of the texts.

Note that the two terminologies used to describe the concepts of 'past returns' and 'future returns' are carefully chosen. Notations like r_{t-1} and r_{t+1} have been deliberately avoided when specifying the model because such symbols may lead to the misconception that it were the returns at precisely $t-1$ and $t+1$ time that supervised the learning. Although one can certainly use the returns at $t-1$ and $t+1$ time as the past and future returns respectively for time t, they may not be the most appropriate choice. Ideally, the past and future returns, as proxies of news sentiment, should capture as much as possible the influence of news sentiment in terms of both its magnitude and span of effect. The former is reflected by the absolute values of the returns; the latter, however, is difficult to represent with a single return at a specific time. That said, it can be argued that the *Efficient Market Hypothesis* implies that it may be appropriate to represent the past return of an equity at time t with the return of the equity at time $t-1$ since most innovations before time $t-1$ would have already been absorbed in the prices, thus possessing limited value for news reporters. An appealing alternative scheme for constructing future returns r_{f} is to consider a weighted average of the returns after time t, namely:

$$r_{\mathrm{f}} = w_1 r_{t+1} + w_2 r_{t+2} + \cdots + w_N r_{t+N} = \sum_{i=1}^{N} w_i r_{t+i} \tag{3.32}$$

where $w_1, w_2, \ldots, w_n$ are the weights and N is the length of the averaging window. The selection of weights reflects the diminishing nature of new sentiment's impact on the market suggested again by the *Efficient Market Hypothesis*. One way to achieve this is to apply the so called *exponential moving average* (EMA) to the return series and use the smoothed returns for future returns r_{f}. Applying exponential moving average on the return series involves the calculation of a new smoothed return s_t, for each t in the series using the following formula (adapted from Roberts [1959]):

$$s_{t+n} = \alpha \cdot r_{t+n} + (1 - \alpha) \cdot s_{t+n+1} \tag{3.33}$$

$$s_{t+N} = r_{t+N} \tag{3.34}$$

where α is the smoothing coefficient that encodes the impact of the more recent returns on the smoothed value s_{t+n}. The closer α is to 1, the greater the influence assigned to the more recent returns on the resulted smoothed value s_{t+n}, and the stronger the diminishing effect. The choices of the values for α and N mostly rely on the practitioners' understanding of the model. In some cases, it may be desirable to capture the medium- and long-term relationships between prospective news sentiment and daily equity returns; Tetlock [2007] found statistically significant correlations between the negative news sentiment at day t and the equity returns at day t+4; Généreux et al. [2011]'s method reported the highest accuracy at $t + 2$. However, it is worth noting that as N increases, the marginal contributions made by returns from distant future diminish due to the exponential weighting. One can therefore use a relatively small N since it is simplifies the computation without compromising the effectiveness of the model too much.

In the next section, I am to present the derivation of the maximum likelihood estimators for the parameters in the final iteration of the model developed in this section using the EM algorithm described earlier this chapter. The parameters of interest are (i) the overall distribution of the four temporal sentiment categories, namely $\boldsymbol{\pi} = (\pi_{\mathrm{rn}}, \pi_{\mathrm{rp}}, \pi_{\mathrm{pn}}, \pi_{\mathrm{pp}})$ and (ii) the four vectors: $\boldsymbol{\beta}_{\mathrm{rn}}, \boldsymbol{\beta}_{\mathrm{rp}}, \boldsymbol{\beta}_{\mathrm{pn}}, \boldsymbol{\beta}_{\mathrm{pp}}$, with each vector characterising the distribution of lexical items for the corresponding temporal sentiment category.

3.5 Derivation of the Supervised EM Algorithm

The goal of this section is to present a detailed derivation of the return-supervised expectation maximisation algorithm for obtaining the maximum likelihood estimators for the parameters in the temporal sentiment model developed in the previous section. The practice follows the general EM framework outlined in Section 3.3 and Appendix A. The key procedures of the algorithm are summarised at the end of this section (page 66).

I start with the $\mathcal{Q}\left(\boldsymbol{\theta}, \boldsymbol{\theta}^{\mathrm{old}}\right)$ function, which is the target function for the optimisation

in the M step. For the sake of simplicity, the $\boldsymbol{\beta}$s are recoded with integer indices to enable easy indexing as follows:

$$\boldsymbol{\beta}_1 = \boldsymbol{\beta}_{\text{rn}} \tag{3.35}$$
$$\boldsymbol{\beta}_2 = \boldsymbol{\beta}_{\text{rp}} \tag{3.36}$$
$$\boldsymbol{\beta}_3 = \boldsymbol{\beta}_{\text{pn}} \tag{3.37}$$
$$\boldsymbol{\beta}_4 = \boldsymbol{\beta}_{\text{pp}} \tag{3.38}$$

where rn, rp, pn, and pp refer to *retrospective negative*, *retrospective positive*, *prospective negative*, and *prospective positive* respectively. Also, let $\mathbf{B}$ denote the collection of the $\boldsymbol{\beta}$s, namely:

$$\mathbf{B} = (\boldsymbol{\beta}_1, \boldsymbol{\beta}_2, \boldsymbol{\beta}_3, \boldsymbol{\beta}_4)\,. \tag{3.39}$$

I will use $\mathbf{s}_{\text{p}}$ and $\mathbf{s}_{\text{f}}$ to denote the two collections of parameters that govern the stable distributions modelling the past returns and future returns respectively:

$$\mathbf{s}_{\text{p}} = (\mathbf{s}_{\text{rn,p}}, \mathbf{s}_{\text{rp,p}}, \mathbf{s}_{\text{pn,p}}, \mathbf{s}_{\text{pp,p}}) = (\mathbf{s}_{1,\text{p}}, \mathbf{s}_{2,\text{p}}, \mathbf{s}_{3,\text{p}}, \mathbf{s}_{4,\text{p}}) \tag{3.40}$$
$$\mathbf{s}_{\text{f}} = (\mathbf{s}_{\text{rn,f}}, \mathbf{s}_{\text{rp,f}}, \mathbf{s}_{\text{pn,f}}, \mathbf{s}_{\text{pp,f}}) = (\mathbf{s}_{1,\text{f}}, \mathbf{s}_{2,\text{f}}, \mathbf{s}_{3,\text{f}}, \mathbf{s}_{4,\text{f}})\,. \tag{3.41}$$

Similar to $\mathbf{B}$, $\mathbf{S}$ is used to denote the collection of the $\mathbf{s}_{\text{p}}$ and $\mathbf{s}_{\text{f}}$. Using this notation, $\mathcal{Q}\left(\boldsymbol{\theta}, \boldsymbol{\theta}^{\text{old}}\right)$ can be written as follows:

$$\mathcal{Q}\left(\boldsymbol{\theta}, \boldsymbol{\theta}^{\text{old}}\right) = \sum_{\mathbf{Z}} p\left(\mathbf{Z} \mid \boldsymbol{\mathcal{D}}, \mathbf{R}', \boldsymbol{\pi}^{\text{old}}, \mathbf{B}^{\text{old}}, \mathbf{S}\right) \ln p\left(\boldsymbol{\mathcal{D}}, \mathbf{R}', \mathbf{Z} \mid \boldsymbol{\pi}, \mathbf{B}, \mathbf{S}\right) \tag{3.42}$$

Note that $\mathbf{S}$ in the posterior distribution of $\mathbf{Z}$ does not have the superscript $^{\text{old}}$ attached to it because the parameters for the return distributions take part in the supervision of the learning process and would remain unchanged during the iterations. Assuming independence between observations:

$$= \sum_{\mathbf{Z}} p\left(\mathbf{Z} \mid \boldsymbol{\mathcal{D}}, \mathbf{R}', \boldsymbol{\pi}^{\text{old}}, \mathbf{B}^{\text{old}}, \mathbf{S}\right) \sum_{n=1}^{N} \ln p\left(\mathbf{d}_n, \mathbf{r}_n, \mathbf{z}_n \mid \boldsymbol{\pi}, \mathbf{B}, \mathbf{S}\right) \tag{3.43}$$

where $\mathbf{r}_n$ denotes a vector containing both r_p and r_f.

$$= \sum_{\mathbf{Z}} p\left(\mathbf{Z} \mid \boldsymbol{\mathcal{D}}, \mathbf{R}', \boldsymbol{\pi}^{\text{old}}, \mathbf{B}^{\text{old}}, \mathbf{S}\right) \sum_{n=1}^{N} \ln p\left(\mathbf{z}_n \mid \boldsymbol{\pi}\right) p\left(\mathbf{d}_n, \mathbf{r}_n \mid \mathbf{z}_n, \mathbf{B}, \mathbf{S}\right) \tag{3.44}$$

$$= \sum_{\mathbf{Z}} p\left(\mathbf{Z} \mid \boldsymbol{\mathcal{D}}, \mathbf{R}', \boldsymbol{\pi}^{\text{old}}, \mathbf{B}^{\text{old}}, \mathbf{S}\right) \tag{3.45}$$

$$\sum_{n=1}^{N} \ln p\left(\mathbf{z}_n \mid \boldsymbol{\pi}\right) \mathcal{S}\left(r_{n,\mathrm{p}} \mid \mathbf{s}_\mathrm{p}\right) \mathcal{S}\left(r_{n,\mathrm{f}} \mid \mathbf{s}_\mathrm{f}\right) p\left(\mathbf{d}_n \mid \mathbf{B}\right) \tag{3.46}$$

Note that the actual value $p\left(\mathbf{z}_n \mid \boldsymbol{\pi}\right)$ takes depends on which of the K elements in $\mathbf{z}_n$ is 1. As a result, $p\left(\mathbf{z}_n \mid \boldsymbol{\pi}\right)$ can be rewritten as $\prod_{k=1}^{K} p\left(\mathbf{z}_{nk} \mid \boldsymbol{\pi}\right)^{z_{nk}} = \prod_{k=1}^{K} \pi_k^{z_{nk}}$ where $\mathbf{z}_{nk}$ is the instance of $\mathbf{z}_n$ whose k-th element is 1 and $z_{nk} = 1$ if and only if the k-th element of $\mathbf{z}_n$ is 1; similar rewrites apply to $\mathcal{S}\left(r_{n,\mathrm{p}} \mid \mathbf{s}_\mathrm{p}\right)$, $\mathcal{S}\left(r_{n,\mathrm{p}} \mid \mathbf{s}_\mathrm{p}\right)$ and $p\left(\mathbf{d}_n \mid \mathbf{B}\right)$:

$$= \sum_{\mathbf{Z}} p\left(\mathbf{Z} \mid \boldsymbol{\mathcal{D}}, \mathbf{R}', \boldsymbol{\pi}^{\text{old}}, \mathbf{B}^{\text{old}}, \mathbf{S}\right) \tag{3.47}$$

$$\sum_{n=1}^{N} \ln \prod_{k=1}^{K} \left\{\pi_k^{z_{nk}} \mathcal{S}\left(r_{n,\mathrm{p}} \mid \mathbf{s}_{k,\mathrm{p}}\right)^{z_{nk}} \mathcal{S}\left(r_{n,\mathrm{f}} \mid \mathbf{s}_{k,\mathrm{f}}\right)^{z_{nk}} p\left(\mathbf{d}_n \mid \boldsymbol{\beta}_k\right)^{z_{nk}}\right\} \tag{3.48}$$

$$= \sum_{\mathbf{Z}} p\left(\mathbf{Z} \mid \boldsymbol{\mathcal{D}}, \mathbf{R}', \boldsymbol{\pi}^{\text{old}}, \mathbf{B}^{\text{old}}, \mathbf{S}\right) \tag{3.49}$$

$$\sum_{n=1}^{N} \sum_{k=1}^{K} z_{nk} \left\{\ln \pi_k + \ln \mathcal{S}\left(r_{n,\mathrm{p}} \mid \mathbf{s}_{k,\mathrm{p}}\right) + \ln \mathcal{S}\left(r_{n,\mathrm{f}} \mid \mathbf{s}_{k,\mathrm{f}}\right) + \ln p\left(\mathbf{d}_n \mid \boldsymbol{\beta}_k\right)\right\} \tag{3.50}$$

The only two terms that are governed by $\mathbf{Z}$ are z_{nk} and $p\left(\mathbf{Z} \mid \boldsymbol{\mathcal{D}}, \mathbf{R}', \boldsymbol{\pi}^{\text{old}}, \mathbf{B}^{\text{old}}, \mathbf{S}\right)$, therefore they can be pushed into the summation over $\mathbf{Z}$:

$$= \sum_{n=1}^{N} \sum_{k=1}^{K} \left\{\ln \pi_k + \ln \mathcal{S}\left(r_{n,\mathrm{p}} \mid \mathbf{s}_{k,\mathrm{p}}\right) + \ln \mathcal{S}\left(r_{n,\mathrm{f}} \mid \mathbf{s}_{k,\mathrm{f}}\right) + \ln p\left(\mathbf{d}_n \mid \boldsymbol{\beta}_k\right)\right\} \tag{3.51}$$

$$\sum_{\mathbf{Z}} z_{nk} p\left(\mathbf{Z} \mid \boldsymbol{\mathcal{D}}, \mathbf{R}', \boldsymbol{\pi}^{\text{old}}, \mathbf{B}^{\text{old}}, \mathbf{S}\right) \tag{3.52}$$

Again, assuming independence between observations in the inner most summation over $\mathbf{Z}$ (3.52). A new symbol m is used as the index to avoid any confusion with n:

$$= \sum_{n=1}^{N} \sum_{k=1}^{K} \left\{ \ln \pi_k + \ln \mathcal{S}\left(r_{n,\mathrm{p}} \mid \mathbf{s}_{k,\mathrm{p}}\right) + \ln \mathcal{S}\left(r_{n,\mathrm{f}} \mid \mathbf{s}_{k,\mathrm{f}}\right) + \ln p\left(\mathbf{d}_n \mid \boldsymbol{\beta}_k\right) \right\} \tag{3.53}$$

$$\sum_{\mathbf{z}_1} \cdots \sum_{\mathbf{z}_{n-1}} \sum_{\mathbf{z}_n} \sum_{\mathbf{z}_{n+1}} \cdots \sum_{\mathbf{z}_N} z_{nk} \prod_{m=1}^{N} p\left(\mathbf{z}_m \mid \mathbf{d}_m, \mathbf{r}_m, \boldsymbol{\pi}^{\mathrm{old}}, \mathbf{B}^{\mathrm{old}}, \mathbf{S}\right) \tag{3.54}$$

The only term that are indexed by n and k inside the summation over $\mathbf{Z}$ is the one with $m = n$, i.e. $z_{nk} p\left(\mathbf{z}_n \mid \mathbf{d}_n, \mathbf{r}_n, \boldsymbol{\pi}^{\mathrm{old}}, \mathbf{B}^{\mathrm{old}}, \mathbf{S}\right)$. It can be separated from the other terms:

$$= \sum_{n=1}^{N} \sum_{k=1}^{K} \left\{ \ln \pi_k + \ln \mathcal{S}\left(r_{n,\mathrm{p}} \mid \mathbf{s}_{k,\mathrm{p}}\right) + \ln \mathcal{S}\left(r_{n,\mathrm{f}} \mid \mathbf{s}_{k,\mathrm{f}}\right) + \ln p\left(\mathbf{d}_n \mid \boldsymbol{\beta}_k\right) \right\} \tag{3.55}$$

$$\sum_{\mathbf{z}_n} z_{nk} p\left(\mathbf{z}_n \mid \mathbf{d}_n, \mathbf{r}_n, \boldsymbol{\pi}^{\mathrm{old}}, \mathbf{B}^{\mathrm{old}}, \mathbf{S}\right) \tag{3.56}$$

$$\sum_{\mathbf{z}_1} \cdots \sum_{\mathbf{z}_{n-1}} \sum_{\mathbf{z}_{n+1}} \cdots \sum_{\mathbf{z}_N} \prod_{m=1}^{n-1} p\left(\mathbf{z}_m \mid \mathbf{d}_m, \mathbf{r}_m, \boldsymbol{\pi}^{\mathrm{old}}, \mathbf{B}^{\mathrm{old}}\right) \tag{3.57}$$

$$\prod_{m=n+1}^{N} p\left(\mathbf{z}_m \mid \mathbf{d}_m, \mathbf{r}_m, \boldsymbol{\pi}^{\mathrm{old}}, \mathbf{B}^{\mathrm{old}}, \mathbf{S}\right) \tag{3.58}$$

$$= \sum_{n=1}^{N} \sum_{k=1}^{K} \left\{ \ln \pi_k + \ln \mathcal{S}\left(r_{n,\mathrm{p}} \mid \mathbf{s}_{k,\mathrm{p}}\right) + \ln \mathcal{S}\left(r_{n,\mathrm{f}} \mid \mathbf{s}_{k,\mathrm{f}}\right) + \ln p\left(\mathbf{d}_n \mid \boldsymbol{\beta}_k\right) \right\} \tag{3.59}$$

$$\sum_{\mathbf{z}_n} z_{nk} p\left(\mathbf{z}_n \mid \mathbf{d}_n, \mathbf{r}_n, \boldsymbol{\pi}^{\mathrm{old}}, \mathbf{B}^{\mathrm{old}}, \mathbf{S}\right) \tag{3.60}$$

$$\sum_{\mathbf{Z}\backslash\mathbf{z}_n} p\left(\mathbf{Z}\backslash\mathbf{z}_n \mid \boldsymbol{\mathcal{D}}\backslash\mathbf{d}_n, \mathbf{R}'\backslash\mathbf{r}_n, \boldsymbol{\pi}^{\mathrm{old}}, \mathbf{B}^{\mathrm{old}}, \mathbf{S}\right) \tag{3.61}$$

where $\mathbf{Z}\backslash\mathbf{z}_n$ means $\mathbf{Z}$ excluding $\mathbf{z}_n$; $\boldsymbol{\mathcal{D}}\backslash\mathbf{d}_n$ and $\mathbf{R}'\backslash\mathbf{r}_n$ are similarly defined. By the addition rule of probability, the summation $\sum_{\mathbf{Z}\backslash\mathbf{z}_n} p\left(\mathbf{Z}\backslash\mathbf{z}_n \mid \boldsymbol{\mathcal{D}}\backslash\mathbf{d}_n, \mathbf{R}'\backslash\mathbf{r}_n, \boldsymbol{\pi}^{\mathrm{old}}, \mathbf{B}^{\mathrm{old}}, \mathbf{S}\right)$ is equal to 1:

$$= \sum_{n=1}^{N} \sum_{k=1}^{K} \left\{ \ln \pi_k + \ln \mathcal{S}\left(r_{n,\mathrm{p}} \mid \mathbf{s}_{k,\mathrm{p}}\right) + \ln \mathcal{S}\left(r_{n,\mathrm{f}} \mid \mathbf{s}_{k,\mathrm{f}}\right) + \ln p\left(\mathbf{d}_n \mid \boldsymbol{\beta}_k\right) \right\} \tag{3.62}$$

$$\sum_{\mathbf{z}_n} z_{nk} p\left(\mathbf{z}_n \mid \mathbf{d}_n, \mathbf{r}_n, \boldsymbol{\pi}^{\mathrm{old}}, \mathbf{B}^{\mathrm{old}}, \mathbf{S}\right) \tag{3.63}$$

Among all K terms in summation $\sum_{\mathbf{z}_n} z_{nk} p\left(\mathbf{z}_n \mid \mathbf{d}_n, \mathbf{r}_n, \boldsymbol{\pi}^{\text{old}}, \mathbf{B}^{\text{old}}, \mathbf{S}\right)$, only one will have $z_{nk} = 1$, and all other terms are reduced to zero.

$$= \sum_{n=1}^{N} \sum_{k=1}^{K} \left\{ \ln \pi_k + \ln \mathcal{S}\left(r_{n,\text{p}} \mid \mathbf{s}_{k,\text{p}}\right) + \ln \mathcal{S}\left(r_{n,\text{f}} \mid \mathbf{s}_{k,\text{f}}\right) + \ln p\left(\mathbf{d}_n \mid \boldsymbol{\beta}_k\right) \right\} \tag{3.64}$$

$$p\left(\mathbf{z}_{nk} \mid \mathbf{d}_n, \mathbf{r}_n, \boldsymbol{\pi}^{\text{old}}, \mathbf{B}^{\text{old}}, \mathbf{S}\right) \tag{3.65}$$

Applying Bayes' theorem on $p\left(\mathbf{z}_{nk} \mid \mathbf{d}_n, \mathbf{r}_n, \boldsymbol{\pi}^{\text{old}}, \mathbf{B}^{\text{old}}, \mathbf{S}\right)$:

$$= \sum_{n=1}^{N} \sum_{k=1}^{K} \left\{ \ln \pi_k + \ln \mathcal{S}\left(r_{n,\text{p}} \mid \mathbf{s}_{k,\text{p}}\right) + \ln \mathcal{S}\left(r_{n,\text{f}} \mid \mathbf{s}_{k,\text{f}}\right) + \ln p\left(\mathbf{d}_n \mid \boldsymbol{\beta}_k\right) \right\} \tag{3.66}$$

$$\frac{p\left(\mathbf{d}_n, \mathbf{r}_n, \mathbf{z}_{nk} \mid \boldsymbol{\pi}^{\text{old}}, \mathbf{B}^{\text{old}}, \mathbf{S}\right)}{p\left(\mathbf{d}_n, \mathbf{r}_n \mid \boldsymbol{\pi}^{\text{old}}, \mathbf{B}^{\text{old}}, \mathbf{S}\right)} \tag{3.67}$$

$$= \sum_{n=1}^{N} \sum_{k=1}^{K} \left\{ \ln \pi_k + \ln \mathcal{S}\left(r_{n,\text{p}} \mid \mathbf{s}_{k,\text{p}}\right) + \ln \mathcal{S}\left(r_{n,\text{f}} \mid \mathbf{s}_{k,\text{f}}\right) + \ln p\left(\mathbf{d}_n \mid \boldsymbol{\beta}_k\right) \right\} \tag{3.68}$$

$$\frac{p\left(\mathbf{z}_{nk} \mid \boldsymbol{\pi}^{\text{old}}, \mathbf{B}^{\text{old}}, \mathbf{S}\right) p\left(\mathbf{d}_n, \mathbf{r}_n \mid \mathbf{z}_{nk}, \boldsymbol{\pi}^{\text{old}}, \mathbf{B}^{\text{old}}, \mathbf{S}\right)}{\sum_{j=1}^{K} p\left(\mathbf{d}_n, \mathbf{r}_n, \mathbf{z}_{nj} \mid \boldsymbol{\pi}^{\text{old}}, \mathbf{B}^{\text{old}}, \mathbf{S}\right)} \tag{3.69}$$

First note that $p\left(\mathbf{z}_{nk} \mid \boldsymbol{\pi}^{\text{old}}, \mathbf{B}^{\text{old}}, \mathbf{S}\right) = p\left(\mathbf{z}_{nk} \mid \boldsymbol{\pi}^{\text{old}}\right) = \pi_k^{\text{old}}$; $\mathbf{B}^{\text{old}}$ and $\mathbf{S}$ are dropped since $\mathbf{z}_{nk}$ only depends on $\boldsymbol{\pi}^{\text{old}}$ as is demonstrated in Figure 3.11. Also note that the term $p\left(\mathbf{d}_n, \mathbf{r}_n \mid \mathbf{z}_{nk}, \boldsymbol{\pi}^{\text{old}}, \mathbf{B}^{\text{old}}, \mathbf{S}\right)$ actually means the probability of seeing $\mathbf{d}_n, \mathbf{r}_n$ when the k-th category is active. The expression may then be rewritten as $p\left(\mathbf{d}_n, \mathbf{r}_n \mid \boldsymbol{\beta}_k^{\text{old}}, \mathbf{s}_{k,\text{p}}, \mathbf{s}_{k,\text{f}}\right)$; similar rewrites can be performed on $p\left(\mathbf{d}_n, \mathbf{r}_n, \mathbf{z}_{nj} \mid \boldsymbol{\pi}^{\text{old}}, \mathbf{B}^{\text{old}}, \mathbf{S}\right)$:

$$= \sum_{n=1}^{N} \sum_{k=1}^{K} \left\{ \ln \pi_k + \ln \mathcal{S}\left(r_{n,\text{p}} \mid \mathbf{s}_{k,\text{p}}\right) + \ln \mathcal{S}\left(r_{n,\text{f}} \mid \mathbf{s}_{k,\text{f}}\right) + \ln p\left(\mathbf{d}_n \mid \boldsymbol{\beta}_k\right) \right\} \tag{3.70}$$

$$\frac{\pi_k^{\text{old}} p\left(\mathbf{d}_n, \mathbf{r}_n \mid \boldsymbol{\beta}_k^{\text{old}}, \mathbf{s}_{k,\text{p}}, \mathbf{s}_{k,\text{f}}\right)}{\sum_{j=1}^{K} \pi_j^{\text{old}} p\left(\mathbf{d}_n, \mathbf{r}_n \mid \boldsymbol{\beta}_j^{\text{old}}, \mathbf{s}_{j,\text{p}}, \mathbf{s}_{j,\text{f}}\right)} \tag{3.71}$$

Recall that $\frac{\pi_k^{\text{old}} p\left(\mathbf{d}_n, \mathbf{r}_n | \boldsymbol{\beta}_k^{\text{old}}, \mathbf{s}_{k,\text{p}}, \mathbf{s}_{k,\text{f}}\right)}{\sum_{j=1}^{K} \pi_j^{\text{old}} p\left(\mathbf{d}_n, \mathbf{r}_n | \boldsymbol{\beta}_j^{\text{old}}, \mathbf{s}_{j,\text{p}}, \mathbf{s}_{j,\text{f}}\right)}$ corresponds to the *responsibility term* γ_{nk} introduced in Section 3.3. It is worth noting that γ_{nk} is defined in terms of the observations and the parameters from the previous iteration of the EM algorithm, and its value should therefore be treated as a constant in the M step. The final expression for the $\mathcal{Q}\left(\boldsymbol{\theta}, \boldsymbol{\theta}^{\text{old}}\right)$ function now becomes:

$$\mathcal{Q}\left(\boldsymbol{\theta}, \boldsymbol{\theta}^{\text{old}}\right) = \sum_{n=1}^{N} \sum_{k=1}^{K} \gamma_{nk} \left\{ \ln \pi_k + \ln \mathcal{S}\left(r_{n,\text{p}} \mid \mathbf{s}_{k,\text{p}}\right) + \ln \mathcal{S}\left(r_{n,\text{f}} \mid \mathbf{s}_{k,\text{f}}\right) + \ln p\left(\mathbf{d}_n \mid \boldsymbol{\beta}_k\right) \right\} \tag{3.72}$$

I now show the derivation of the expressions for updating the parameters of interest, namely π_k and $\boldsymbol{\beta}_k$. As discussed earlier, $\mathbf{s}_{k,\text{p}}$ and $\mathbf{s}_{k,\text{f}}$ are parts of the supervision process, and as such they will remain unchanged through out the learning process.

The target function $\mathcal{Q}\left(\boldsymbol{\theta}, \boldsymbol{\theta}^{\text{old}}\right)$ is optimised with respect to the mixture coefficient π_k under the linear constraint $\sum_{m=1}^{K} \pi_m = 1$. The constraint is enforced with a Lagrange multiplier: $\lambda \left(\sum_{m=1}^{K} \pi_m - 1 \right)$. Taking the derivative of $\mathcal{Q}\left(\boldsymbol{\theta}, \boldsymbol{\theta}^{\text{old}}\right) + \lambda \left(\sum_{m=1}^{K} \pi_m - 1 \right)$ with respect to π_k and set it to zero:

$$0 = \frac{\partial \mathcal{Q}\left(\boldsymbol{\theta}, \boldsymbol{\theta}^{\text{old}}\right) + \lambda \left(\sum_{m=1}^{K} \pi_m - 1 \right)}{\partial \pi_k} \tag{3.73}$$

$$0 = \frac{\partial \sum_{n=1}^{N} \sum_{k=1}^{K} \gamma_{nk} \left\{ \ln \pi_k + \ln \mathcal{S}\left(r_{n,\text{p}} \mid \mathbf{s}_{k,\text{p}}\right) + \ln \mathcal{S}\left(r_{n,\text{f}} \mid \mathbf{s}_{k,\text{f}}\right) + \ln p\left(\mathbf{d}_n \mid \boldsymbol{\beta}_k\right) \right\}}{\partial \pi_k} \tag{3.74}$$

$$+ \frac{\partial \lambda \left(\sum_{m=1}^{K} \pi_m - 1 \right)}{\partial \pi_k} \tag{3.75}$$

$$0 = \lambda + \sum_{n=1}^{N} \gamma_{nk} \frac{\partial \ln \pi_k}{\partial \pi_k} \tag{3.76}$$

$$0 = \lambda + \sum_{n=1}^{N} \frac{\gamma_{nk}}{\pi_k} \tag{3.77}$$

Mutiply both sides with π_k:

$$0 = \lambda \pi_k + \sum_{n=1}^{N} \gamma_{nk} \tag{3.78}$$

Such expression can be obtained for each possible k. Summing over k to solve for λ:

$$0 = \lambda \sum_{k=1}^{K} \pi_k + \sum_{n=1}^{N} \sum_{k=1}^{K} \gamma_{nk} \tag{3.79}$$

Note that $\sum_{k=1}^{K} \pi_k = 1$ and $\sum_{n=1}^{N} \sum_{k=1}^{K} \gamma_{nk} = N$:

$$\lambda = -N \tag{3.80}$$

Substituting λ back into 3.78 and solve for π_k:

$$\pi_k = \frac{\sum_{n=1}^{N} \gamma_{nk}}{N} \tag{3.81}$$

I now move to derive the ML estimators for updating $\boldsymbol{\beta}_k$. Recall that $\boldsymbol{\beta}_k$ consists of V parameters for the multinomial distribution from which the terms for the k-th temporal sentiment category are drawn; let parameters in $\boldsymbol{\beta}_k$ be denoted by $\alpha_{k,e_1}, \alpha_{k,e_2}, \ldots, \alpha_{k,e_V}$, where α_{k,e_j} is the probability for the j-th lexical entry e_j to occur in the text when the text is generated by the k-th temporal sentiment category. The $\mathcal{Q}\left(\boldsymbol{\theta}, \boldsymbol{\theta}^{\text{old}}\right)$ function listed in 3.72 now needs to be expanded in the same manner as in Equation 3.8:

$$\mathcal{Q}\left(\boldsymbol{\theta}, \boldsymbol{\theta}^{\text{old}}\right) = \sum_{n=1}^{N} \sum_{k=1}^{K} \gamma_{nk} \left\{ \ln \pi_k + \ln \mathcal{S}\left(r_{n,\text{p}} \mid \mathbf{s}_{k,\text{p}}\right) + \ln \mathcal{S}\left(r_{n,\text{f}} \mid \mathbf{s}_{k,\text{f}}\right) + \ln p\left(\mathbf{d}_n \mid \boldsymbol{\beta}_k\right) \right\} \tag{3.82}$$

$$= \sum_{n=1}^{N} \sum_{k=1}^{K} \gamma_{nk} \left\{ \ln \pi_k + \ln \mathcal{S}\left(r_{n,\text{p}} \mid \mathbf{s}_{k,\text{p}}\right) + \ln \mathcal{S}\left(r_{n,\text{f}} \mid \mathbf{s}_{k,\text{f}}\right) + \ln \prod_{j=1}^{V} {\alpha_{k,e_j}}^{f_{n,e_j}} \right\} \tag{3.83}$$

where f_{n,e_j} is the raw frequency of lexical item e_j occurring in the text of document n. Deriving the ML estimator for $\boldsymbol{\beta}_k$ is effectively equivalent to deriving the ML estimator for α_{k,e_j} with K additional linear constraints, i.e. $\sum_{j=1}^{V} \alpha_{1,e_j} = 1, \sum_{j=1}^{V} \alpha_{2,e_j} = 1, \ldots, \sum_{j=1}^{V} \alpha_{K,e_j} = 1$. Taking the derivative of the expanded $\mathcal{Q}\left(\boldsymbol{\theta}, \boldsymbol{\theta}^{\text{old}}\right)$ function together

with the Lagrange multipliers $\sum_{k=1}^{K} \lambda_k \left(\sum_{j=1}^{V} \alpha_{k,e_j} - 1 \right)$ with respect to α_{k,e_j}:

$$0 = \frac{\partial \mathcal{Q}\left(\boldsymbol{\theta}, \boldsymbol{\theta}^{\text{old}}\right) + \sum_{k=1}^{K} \lambda_k \left(\sum_{j=1}^{V} \alpha_{k,e_j} - 1 \right)}{\partial \alpha_{k,e_j}} \tag{3.84}$$

$$0 = \frac{\partial \sum_{n=1}^{N} \sum_{k=1}^{K} \gamma_{nk} \left\{ \ln \pi_k + \ln \mathcal{S}\left(r_{n,\text{p}} \mid \mathbf{s}_{k,\text{p}}\right) + \ln \mathcal{S}\left(r_{n,\text{f}} \mid \mathbf{s}_{k,\text{f}}\right) + \ln \prod_{j=1}^{V} {\alpha_{k,e_j}}^{f_{n,e_j}} \right\}}{\partial \alpha_{k,e_j}} \tag{3.85}$$

$$+ \frac{\partial \sum_{k=1}^{K} \lambda_k \left(\sum_{j=1}^{V} \alpha_{k,e_j} - 1 \right)}{\partial \alpha_{k,e_j}} \tag{3.86}$$

$$0 = \lambda_k + \sum_{n=1}^{N} \gamma_{nk} \cdot f_{n,e_j} \frac{\partial \ln \alpha_{k,e_j}}{\partial \alpha_{k,e_j}} \tag{3.87}$$

$$0 = \lambda_k + \sum_{n=1}^{N} \frac{\gamma_{nk} \cdot f_{n,e_j}}{\alpha_{k,e_j}} \tag{3.88}$$

Mutiply both sides with α_{k,e_j}:

$$0 = \lambda_k \alpha_{k,e_j} + \sum_{n=1}^{N} \gamma_{nk} \cdot f_{n,e_j} \tag{3.89}$$

The linear constraints can be exploited to rid of α_{k,e_j} by summing over V instead of K as in the case for π_k:

$$0 = \lambda_k \sum_{j=1}^{V} \alpha_{k,e_j} + \sum_{n=1}^{N} \sum_{j=1}^{V} \gamma_{nk} \cdot f_{n,e_j} \tag{3.90}$$

Making use of the fact that $\sum_{j=1}^{V} \alpha_{k,e_j} = 1$:

$$\lambda_k = - \sum_{n=1}^{N} \sum_{j=1}^{V} \gamma_{nk} \cdot f_{n,e_j} \tag{3.91}$$

Substituting λ back into 3.89 and solve for α_{k,e_j}:

$$\alpha_{k,e_j} \sum_{n=1}^{N} \sum_{j=1}^{V} \gamma_{nk} \cdot f_{n,e_j} = \sum_{n=1}^{N} \gamma_{nk} \cdot f_{n,e_j} \tag{3.92}$$

$$\alpha_{k,e_j} = \frac{\sum_{n=1}^{N} \gamma_{nk} \cdot f_{n,e_j}}{\sum_{n=1}^{N} \sum_{j=1}^{V} \gamma_{nk} \cdot f_{n,e_j}} \tag{3.93}$$

A key concept in interpreting the expression for updating α_{k,e_j} is the notion of 'soft counts' — whenever a lexical item e_j occurs in the n-th document, only a 'portion' of the occurrence is attributed to the k-th sentiment category, which is accounted by the responsibility term γ_{nk}. The term $\gamma_{nk} \cdot f_{n,e_j}$, therefore, represents the partial counts of the lexical item e_j's occurrences in the n-th document weighted by the k-th sentiment category's contribution to the responsibility term. The expression 3.93 can then be interpreted as the percentage of the total soft counts for e_j that are attributed to category k in all documents among the total soft counts for all the lexical items in the vocabulary that are attributed to category k in all documents.

The supervision enforced by the sentiment proxies is propagated through the responsibility terms γ_{nk} during the parameter updates. The better a temporal sentiment category explains a set of observations, the greater the category's responsibilities will be for that set of observations in the next iteration.

Once the updating expressions for π_k and α_{k,e_j} are derived, the EM algorithm can be formulated to obtain the ML estimators for the model developed in this chapter following the same structure as outlined in Section 3.3:

1. Choose an initial (usually random) settings for the parameters $\boldsymbol{\pi}^{\text{old}}$, and the four $\boldsymbol{\beta}^{\text{old}}$s, each containing V probabilities. Also choose a set of parameters for the distributions of past and future returns, namely $\mathbf{s}_{k,\text{p}}$ and $\mathbf{s}_{k,\text{f}}$.

2. The **E Step**: evaluate the responsibility term $\gamma_{nk} = \frac{\pi_k^{\text{old}} p\left(\mathbf{d}_n, \mathbf{r}_n | \boldsymbol{\beta}_k^{\text{old}}, \mathbf{s}_{k,\text{p}}, \mathbf{s}_{k,\text{f}}\right)}{\sum_{j=1}^{K} \pi_j^{\text{old}} p\left(\mathbf{d}_n, \mathbf{r}_n | \boldsymbol{\beta}_j^{\text{old}}, \mathbf{s}_{j,\text{p}}, \mathbf{s}_{j,\text{f}}\right)}$ where

$$p\left(\mathbf{d}_n, \mathbf{r}_n \mid \boldsymbol{\beta}_k^{\text{old}}, \mathbf{s}_{k,\text{p}}, \mathbf{s}_{k,\text{f}}\right) = \mathcal{S}\left(r_{n,\text{p}} \mid \mathbf{s}_{k,\text{p}}\right) \mathcal{S}\left(r_{n,\text{f}} \mid \mathbf{s}_{k,\text{f}}\right) p\left(\mathbf{d}_n \mid \boldsymbol{\beta}_k^{\text{old}}\right) \tag{3.94}$$

$$= \mathcal{S}\left(r_{n,\text{p}} \mid \mathbf{s}_{k,\text{p}}\right) \mathcal{S}\left(r_{n,\text{f}} \mid \mathbf{s}_{k,\text{f}}\right) \prod_{j=1}^{V} {\alpha_{k,e_j}}^{f_{n,e_j}} \tag{3.95}$$

3. The **M Step**: compute

$$\pi_k^{\text{new}} = \frac{\sum_{n=1}^{N} \gamma_{nk}}{N} \tag{3.96}$$

$$\alpha_{k,e_j}^{\text{new}} = \frac{\sum_{n=1}^{N} \gamma_{nk} \cdot f_{n,e_j}}{\sum_{n=1}^{N} \sum_{j=1}^{V} \gamma_{nk} \cdot f_{n,e_j}} \tag{3.97}$$

as described earlier in this section.

4. Check for convergence of either the log-likelihood or the parameter values. The log-likelihood for the incomplete-dataset is computed as:

$$\mathcal{L}\left(\boldsymbol{\pi}, \mathbf{B}, \mathbf{S} \mid \boldsymbol{\mathcal{D}}, \mathbf{R}', \mathbf{S}\right) = \ln p\left(\boldsymbol{\mathcal{D}}, \mathbf{R}' \mid \boldsymbol{\pi}, \mathbf{B}, \mathbf{S}\right) \tag{3.98}$$

$$= \sum_{n=1}^{N} \ln \sum_{k=1}^{K} \pi_k^{\text{new}} \mathcal{S}\left(r_{n,\text{p}} \mid \mathbf{s}_{k,\text{p}}\right) \mathcal{S}\left(r_{n,\text{f}} \mid \mathbf{s}_{k,\text{f}}\right) p\left(\mathbf{d}_n \mid \boldsymbol{\beta}_k^{\text{new}}\right). \tag{3.99}$$

The convergence can be verified by comparing the improvement of the log-likelihood relative to the last iteration with a pre-defined threshold — if the improvement is lower than the threshold for a sufficient number of iterations, the process is considered to have converged. If the convergence criterion is not satisfied, then update the parameters:

$$\pi_k^{\text{old}} \leftarrow \pi_k^{\text{new}} \tag{3.100}$$

$$\alpha_{k,e_j}^{\text{old}} \leftarrow \alpha_{k,e_j}^{\text{new}} \tag{3.101}$$

$$\boldsymbol{\beta}_k^{\text{old}} \leftarrow \left(\alpha_{k,e_1}^{\text{old}}, \ldots, \alpha_{k,e_V}^{\text{old}}\right) \tag{3.102}$$

and go to step 2. If the algorithm successfully converges, exit the process.

The methods described in this section tackle the learning of the temporal sentiment model as outlined at the beginning of this chapter. The degree and direction of the influence a particular lexical item e_j's occurrences has on the sentiment proxy when encountered in news text is quantified by its probability assigned in the four lexical item distributions, i.e. α_{k,e_j}. The next section addresses the evaluation of the sentiment and temporal orientation of a new document based solely on the document's textual content.

3.6 Evaluating the Sentiment and Temporality for Unseen Documents

Once the estimates for the parameters are obtained, new documents' coordinates in the temporal sentiment space can be evaluated. For this task, the quantity of interest is

$$p\left(\mathbf{z}_n \mid \mathbf{d}_n, \boldsymbol{\pi}, \boldsymbol{\beta}_1, \boldsymbol{\beta}_2, \boldsymbol{\beta}_3, \boldsymbol{\beta}_4\right) \tag{3.103}$$

that is, the probability distribution for document $\mathbf{d}_n$'s temporal sentiment given the parameter estimates obtained from the previous sections. It can be seen from the above equation that the evaluation of the probabilities relies on the information contained in the

texts of the documents alone, and no sentiment proxies such as equity returns are involved in the process. The probabilities can be evaluated with the Bayes' theorem:

$$p\left(\mathbf{z}_n \mid \mathbf{d}_n, \boldsymbol{\pi}, \boldsymbol{\beta}_1, \boldsymbol{\beta}_2, \boldsymbol{\beta}_3, \boldsymbol{\beta}_4\right) = \frac{p\left(\mathbf{z}_n, \mathbf{d}_n \mid \boldsymbol{\pi}, \boldsymbol{\beta}_1, \boldsymbol{\beta}_2, \boldsymbol{\beta}_3, \boldsymbol{\beta}_4\right)}{p\left(\mathbf{d}_n \mid \boldsymbol{\pi}, \boldsymbol{\beta}_1, \boldsymbol{\beta}_2, \boldsymbol{\beta}_3, \boldsymbol{\beta}_4\right)} \tag{3.104}$$

$$= \frac{p\left(\mathbf{z}_n, \mathbf{d}_n \mid \boldsymbol{\pi}, \boldsymbol{\beta}_1, \boldsymbol{\beta}_2, \boldsymbol{\beta}_3, \boldsymbol{\beta}_4\right)}{\sum_{k=1}^{K} p\left(\mathbf{z}_{nk}, \mathbf{d}_n \mid \boldsymbol{\pi}, \boldsymbol{\beta}_k\right)} \tag{3.105}$$

For each document, evaluating Equation 3.105 produces a vector consisting of four probabilities, each describing the overall probability of one of the four temporal sentiment categories. In a document classification setting where exactly one category is assigned to a single document, the category with the highest probability assigned to it is typically chosen as the category for the document.

An alternative application of the probabilities mined this way, as has been briefly mentioned at the beginning of Section 3.1, is to collate them into temporal sentiment time series. More specifically, the probabilities for each of the temporal sentiment categories associated with the documents published on the same day may be aggregated to form the news temporal sentiment orientations for that day. This view to the temporal sentiment orientations is particularly useful when evaluating the predictive performances of the model. The parameters of the model are estimated using a training dataset where both the texts and sentiment proxies are available; the trained model, with its parameters learnt, is then used to evaluate the temporal sentiment orientations of the new documents in the test set; the resultant orientations are grouped into four times series, each quantifies the intensity and polarity of the fluctuations of one of the temporal sentiment. The linear correlation between each temporal sentiment series extracted from the test dataset and the sentiment proxies — especially the future equity returns — can be used to measure how well the temporal sentiment in the news predict the trending of the sentiment proxies.

3.7 Summary

In this chapter, I first introduced the notion of news temporality in the context of sentiment analysis of business news. Traditional sentiment analysis of business news typically positions a news document on a one-dimensional scale on which the document's evaluative stance is measured; the practice is often followed by an attempt to establish a predictive correspondence between the extracted news sentiment and the performances of equities, usually for the purpose of explaining the movements of the market. In this thesis, I propose the addition of a new temporality scale to the original single-dimensional sentiment scale. The added temporality dimension measures the retrospectivity/prospectivity of the narratives in business news — news that recounts the events already happened to the

market is considered as retrospective news while news that speculates on or hints at the possible developments of events in the future in the market is considered as prospective news.

The introduction of this new scale is motivated by both theoretical and practical reasons. The Efficient Market Hypothesis implies that all publicly (and privately if the strongest form of the hypothesis is assumed) available information must have already been incorporated into the market and reflected in the current prices on the market. It may therefore be beneficial to distinguish retrospective news from its prospective counterparts since such news mainly accounts for events whose impact has already been absorbed into the prices and thus yields limited predictive power. The practical reason concerns with the use of equity returns as sentiment proxies to label training dataset when approaching the analysis of sentient as a supervised machine learning task. Relying solely on contemporaneous equity returns when labelling training data could lead to misclassification of documents' sentiment orientation. The addition of the news temporality dimension makes it possible for the documents to be aligned with past or future returns, which would accommodate the delays between the release of information and market's responses and alleviate the issues thereof to some extent.

The modelling of sentiment and temporality in business news is then presented. The textual content in news articles is treated as a mixture of lexical items drawn from four term distributions, each characterises the linguistic realisations for one of the four temporal sentiment categories. Obtaining the maximum likelihood estimators for mixture models' parameters, however, proves to be a challenging task. The EM algorithm is introduced to overcome this difficulty; the algorithm works by iteratively approaching a local maximum in the log-likelihood function of the model while producing the maximum likelihood estimators of the parameters as side products.

The standard mixture model together with the EM algorithm alone are not enough for tackling the task of learning temporal sentiment orientations of business news. The learning process will have to be supervised by some form of extrinsic sentiment proxies — in this case, the daily returns of the related company's stock on the market. Additional structures were put in place to accommodate the past and future returns; The designs of the distributions for the two types of returns reflected the fact that past returns are associated mainly with retrospective news while future returns are governed by prospective news. The extended model was subjected to analysis and the maximum likelihood estimators for the parameters in the model were derived through the EM algorithm. The implementation of the methods detailed above will be presented in Chapter 4.

Chapter 4

System Implementation

4.1 Introduction

In this chapter, I present the design and implementation of a system which (i) realises the methods described in Chapter 3, and (ii) provides the infrastructure for evaluating and benchmarking the method's performance. This system consists of two components. The first component is CiCui, a general purpose text analysis platform I had developed over the course of this study; the CiCui system serves as the entry point for textual data — it collects news documents from online sources, transforms the unstructured text in those documents into structured form using natural language processing techniques and creates an *inverted index* for the structured texts with a relational database management system. The second component I created is the TSMiner system (Temporal Sentiment Miner), which is a data analytic workflow built within a third-party data analytic environment called KNIME; the TSMiner implements the calculations in the methods described in Chapter 3 as well as the evaluation of the method's performances described in Chapter 5. Figure 4.1 gives an overview of the system's architecture, illustrating the input and the output of the system as well as the interconnection between the two components. The main tasks of the system are:

1. To fit the model developed in Chapter 3 on training data, namely to obtain the maximum likelihood estimates for the parameters in the model given the term-document matrix and the related historical equity returns;

2. To evaluate the sentiment-temporal orientations for unseen documents given the model parameter estimates obtained from the training process;

3. To compare the predictive performance of the method developed in this thesis with those of other benchmarking methods.

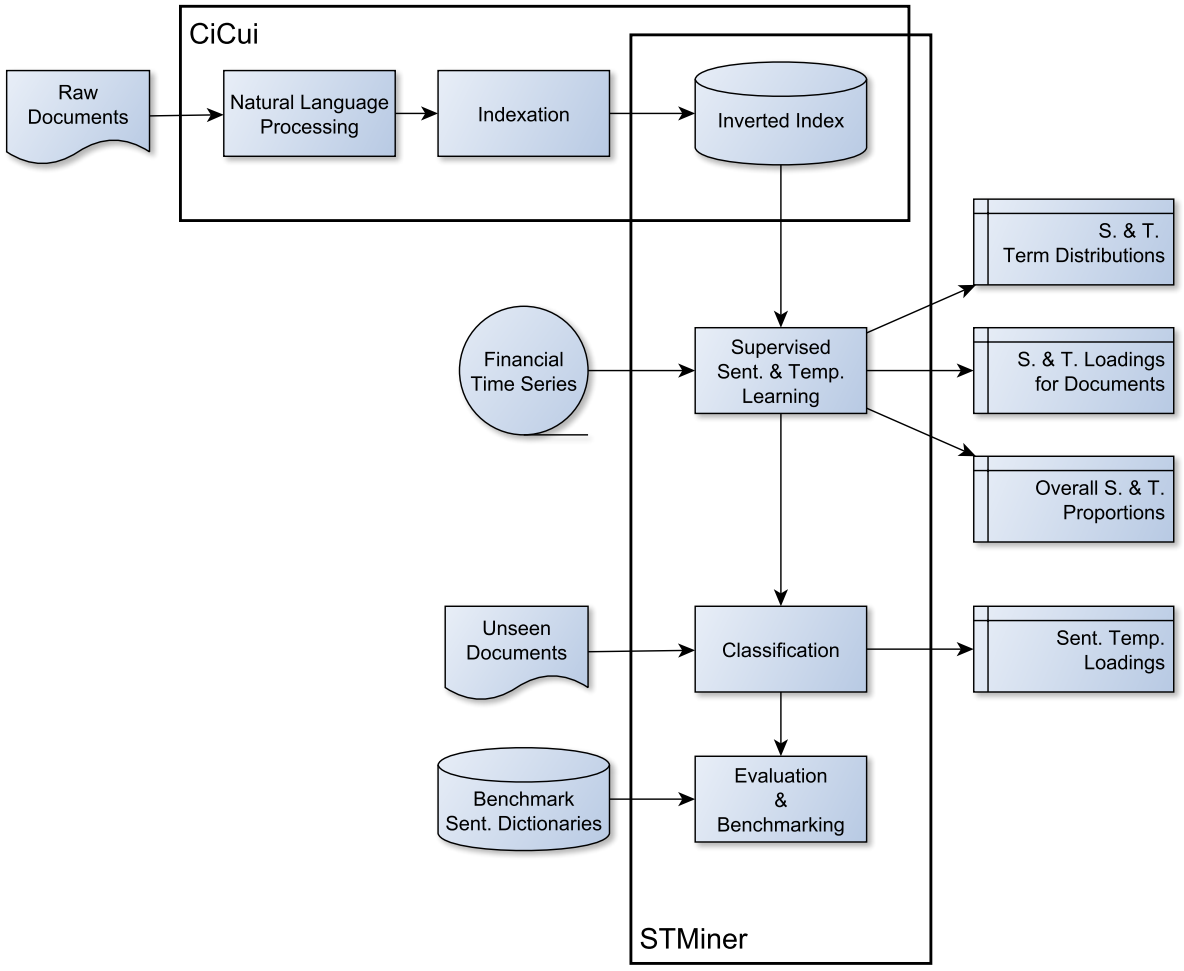


Figure 4.1: System Architecture

This diagram gives an overview of the architecture of the system presented in this chapter. The system as a whole takes three inputs: (i) News articles about a particular company of interest, retrieved as unstructured texts from online news repositories or the original websites; (ii) A sentiment proxy for the company; in this case the returns of the company's stock on a exchange, where the returns are calculated using historical Open-Close-High-Low data found on Yahoo Finance; (iii) External sentiment dictionaries which are used by some of the benchmarking methods when evaluating the performances of the system. The principal outputs of the system include: (i) Four term lists, each containing the distribution of the terms for one of the four temporal sentiment categories; (ii) The probabilities for the documents to belong to each of the four categories; (iii) The overall proportions of the document classes in the corpus; (iv) The probabilities for an unseen document to belong to each of the four categories as classified by the trained model. As can be seen from the overlap between the two systems' boundaries, the two components form a coalition through the sharing of the inverted index — the TSMiner component utilizes the inverted index produced by the CiCui system as the textual input for model training and validation.

4.2 Text Preprocessing with the CiCui System

The goal of the text preprocessing phase is to extract from unstructured input texts the information needed for computing the quantities used in the return-regularised EM algorithm presented in Section 3.5. More specifically, one needs to calculate three quantities during each iteration of the algorithm: the responsibility term γ_{nk} in the E step and the

expressions for updating π_k and α_{k,e_j} in the M step. Observe that the evaluation of the expressions 3.95, 3.96 and 3.97 requires the following information be extracted from the text:

The vocabulary a set containing all the lexical items that will be entertained by the analysis, i.e. $\{e_1, e_2, \ldots, e_V\}$.

The size of the vocabulary namely V.

The total number of documents which is the quantity N used when calculating π_k^{new}, as defined in Equation 3.96.

The term-document frequency matrix which captures f_{n,e_j}. The rows in the matrix represents the documents and its column space represents the vocabulary; the entry whose coordinate in the matrix is (n, j) corresponds to the number of times e_j occurred in document $\mathbf{d}_n$.

Among the four quantities, the size of the vocabulary V can be derived from the vocabulary once it is constructed; the vocabulary is essentially the column space of the term-document frequency matrix. The total number of documents N can be obtained by counting the number of rows in the term-document frequency matrix. So the key computation task in the implementation of the EM algorithm is to construct the term-document matrix. In the CiCui system, the construction of the term-document frequency matrix is implemented as a SQL query issued against the inverted index database. The discussion in this section shall present the various steps that lead to the building of the inverted index.

Before diving deeper into the details, it is helpful to clarify what constitute a vocabulary in this thesis. Note that prior to this point through out the thesis, the phrase 'lexical items' and sometimes 'terms' have been used in placed of 'words' when referring to the basic constituting units in a vocabulary. Specifying the method this way makes it agnostic with regard to the actual lexical items used. In this thesis, two choices for lexical items are contemplated: the *unigrams* and the *word dependencies*. The inverted index created by the CiCui system covers the postings for both unigrams and word dependencies, which makes it easier to produce term-document frequency matrix for either lexical item choices.

4.2.1 Preprocessing Workflow

The flowchart in Figure 4.2 illustrates the workflow for creating term-document frequency matrices from raw texts. News articles in various formats are imported into the system using specialised importers and converted into CiCui's custom XML format; this XML

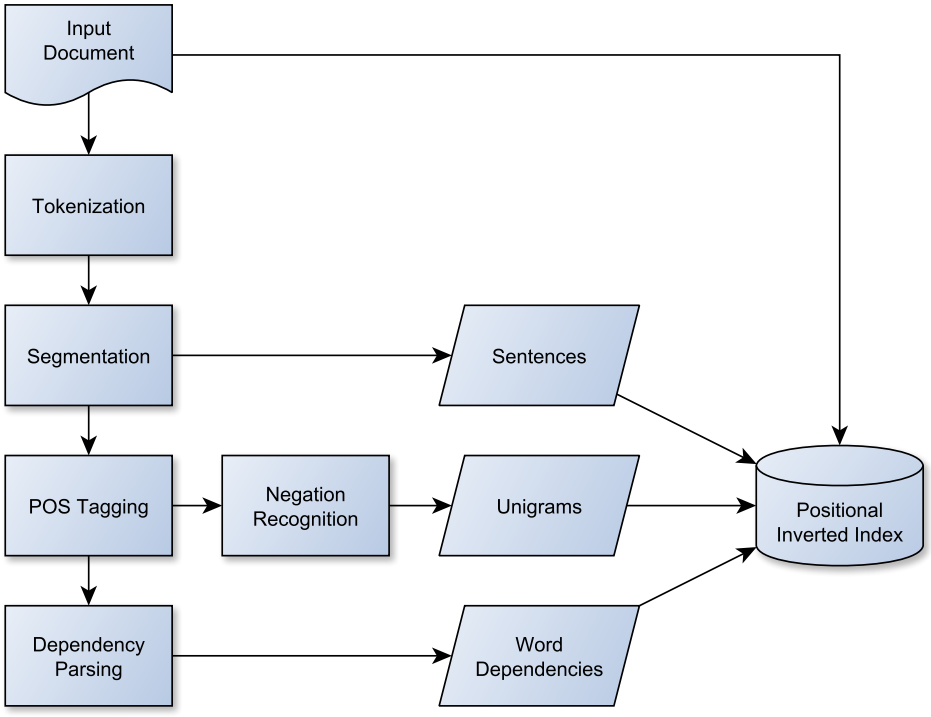


Figure 4.2: Text Preprocessing Workflow

format records the title, source, link, publication date and the actual text of an article (an example document is shown in 4.3).

A number of natural language processing tasks are carried out to convert the texts into a more structured form before the lexical items can be registered with the index. The Running texts are first subjected to tokenisation and sentence segmentation where word and sentence boundaries are recognised. Each token is then tagged with part-of-speech information based on its surrounding tokens using sequential modelling techniques such as the hidden Markov model or conditional random fields. Lemmatisation[1], which strips a word of its inflections and reduces it to its root form, is applied to the tokens to extract their lemmas — e.g. the word 'better' is reduced to 'good' and 'rose', 'risen' and 'rises' are all mapped to 'rise'; by grouping words by their semantics, the procedure effectively reduces the dimensionality of the resulting term-document matrix, which helps alleviating the over-fitting problem in the subsequent learning of the model's parameters. Lastly, the token stream is put through a process called *dependency parsing* (using the parser developed by Klein and Manning [2003]), which extracts relational dependencies between words based on the concept of *dependency grammar*. Each dependency instance contains

[1]An alternative procedure is the *stemming* process. Stemming removes the inflections from a word by either looking it up in a dictionary or stripping it from suffixes that often hallmark inflections, such as -ed, -ing, -ly, etc. The disadvantage of this method compared with lemmatisation is that it does not take contextual information into consideration; this could cause over-stemming — the proper name *Boeing*, for example, would be converted into *Boe* due to the -ing suffix.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<article>
<attributes>
<key>company</key>
<value>company</value>
</attributes>
<content>Boeing Co. is on track to outsell Airbus for the first time
   since 2006, reporting that it had booked more than 1,000 net new
   orders this year. (more text ...) </content>
<id>840e3954-126e-4ce7-bf0e-bd7babdff4f4</id>
<link>http://www.wsj.com/articles/
   SB10001424127887324894104578106671355062146</link>
<publishedDate>2012-11-08T00:00:00Z</publishedDate>
<source>Wall Street Journal (Online)</source>
<title>Boeing Is on Track to Outsell Airbus</title>
</article>
```

Figure 4.3: An Example Article in CiCui's XML Format

Most tags in CiCui's custom XML format are self-explanatory ('title', 'source', 'publishedDate'). The 'id' tag contains a unique UUID (universally unique identifier) string which is automatically generated and assigned to the article as a global identifier. The 'attributes' tag can contain multiple key-value style attributes; it is introduced to accommodate situational information which does not consistently present in all documents.

the *governor*, the *dependent*, and the*typed relation* between the governor and the dependent; in addition, the parser also gives the token positions of the governor and the dependent in the sentence, without which the entry cannot be uniquely identified. The CiCui system internally delegates all these tasks to the Stanford CoreNLP tool-kit [Manning et al., 2014]; CiCui enhances the efficiency of the procedures by wrapping the executions in a pipeline that enables parallel processing.

After these steps, the unigrams tokens are subjected to additional filtering to reduce noise before they are put into the index. Punctuation and tokens containing digits are omitted. Part-of-speech tags belonging to the same family are collapsed: `NNS` and `NNPS` which represent the plural forms of nouns and proper nouns, for example, will be mapped to their singular counterparts (`NN` and `NNP`) respectively; this procedure, not unlike the lemmatisation process, aims to consolidate tokens whose semantics are the similar.

One additional treatment is the recognition of negations. The reversal effect negation has on the sentiment orientation of expressions has been noted by a number of previous studies [Dave et al., 2003, Yi et al., 2003, Kennedy and Inkpen, 2006, Wilson et al., 2009, Smailović et al., 2014]. In the CiCui system, the negations are accounted for using manually crafted rules. A token modified by expressions like 'not', 'not going to', 'not seem to', etc. is indexed as a separate token that is different from the original; the token

'omitted' in the expression 'not to be omitted', for example, will be considered as a new 'not_omitted' token.

4.2.2 Inverted Positional Index

Once the preprocessing completes, an inverted positional index for the corpus is built. While an inverted index maintains a mapping from individual lexical items to the enclosing lexical structure (e.g. sentences, documents, etc.) in which they occurred, an inverted *positional* index records the additional pieces of information with respect to the positions of the token at which the lexical items occurred in the enclosing structures.

The CiCui system manages the index in a relational database using the H2 Database Engine[2]. The schema for the database is presented in Figure 4.4. Some of the key tables to the analysis include:

DOCUMENT A table that registers the meta-information about documents indexed during the pre-processing: the title, the source, the URL link (if available) and the publication date. A unique ID string is automatically generated and assigned to each document. The full text of the documents are not stored in this table but can be reconstructed from the texts indexed in `SENTENCE` table.

DICTIONARY The `DICTIONARY` table, as its name suggests, keeps a catalogue of all the distinct tokens encountered in the texts. The index records the original inflected form as occurred in the text, the lemma and the part-of-speech tag of each of the tokens in the text if it has not been recorded already. Again, a unique identifier is assigned to each entry in the dictionary.

SENTENCE The sentences segmented during the pre-processing are kept in this table. The full text for each sentence is retained so that reconstruction of the original document is possible.

WORD_POSTING and SENTENCE_POSTING These two tables together serve as the 'ledger' for the tokens in the corpus: for each token in a sentence, the `WORD_POSTING` table registers the ID of the token, the ID of the sentence, and the token-level position at which the token appeared in the sentence; similarly, the `SENTENCE_POSTING` table, for each sentence, records the ID of the sentence, the ID of the document as well as the relative position of the sentence in the document.

DEPENDENCIES This table essentially keeps the postings for word dependencies, much like the `WORD_POSTING` and `SENTENCE_POSTING` tables. The IDs for the governor and the dependent tokens together with their token positions are associated with the

[2] `http://www.h2database.com/`, accessed in February 2016

sentences. Note that the primary key for this table contain all the attributes because it is possible for the same governor-dependent-relation triplet to occur multiple times in a sentence at different token positions.

The arrows that link the tables represent foreign key constraints; in practice, however, these constraints are honoured but not enforced[3] due to performance considerations. Note that not all tables were used by the analyses performed in this thesis — the `TERMINOLOGY_DICTIONARY` and `TERMINOLOGY_POSTING` tables are used to index multi-word expressions and named entities; the `CONTENT_ANALYSIS_DICTIONARY` and `CONTENT_ANALYSIS_POSTING` tables record of the occurrences of terms and expressions as defined in some supplied dictionaries; the `META_INFO` maintains a key-value map for storing configurations and parameters for the indexation.

4.2.3 Constructing Term-Document Frequency Matrix

After the indexation is completed, the term-document frequency matrix can be constructed by issuing the following SQL query against the index database:

```
SELECT D.TEXT AS TEXT,
        D.LEMMA AS LEMMA,
        D.POS AS POS,
        DOC.ID AS ARTICLE_ID,
        DOC.TITLE AS ARTICLE_TITILE,
        FORMATDATETIME(DOC.PUBLISHED_DATE,
            'YYYY-MM-DD') AS DATE,
        CAST(COUNT(*) AS DOUBLE) AS
            WEIGHT
FROM DICTIONARY AS D
INNER JOIN WORD_POSTING AS WP
INNER JOIN SENTENCE_POSTING AS SP
INNER JOIN DOCUMENT AS DOC
WHERE D.ID = WP.WORD_ID
AND WP.SENTENCE_ID = SP.SENTENCE_ID
AND DOC.ID = SP.DOCUMENT_ID
AND ( D.POS LIKE 'VB%' OR D.POS LIKE 'NN' OR D.
    POS LIKE 'JJ%')
GROUP BY WP.WORD_ID, SP.DOCUMENT_ID;
```

The above query produces a term-document matrix in sparse form, capturing the relations between major open-class words (i.e. nouns, verbs and adjectives) and the documents in which they occurred. An example output is illustrated in Figure 4.1. It is

[3]Efforts were made in the indexation process to ensure the consistency between the keys, but no actual table constraints were put in place in the database.

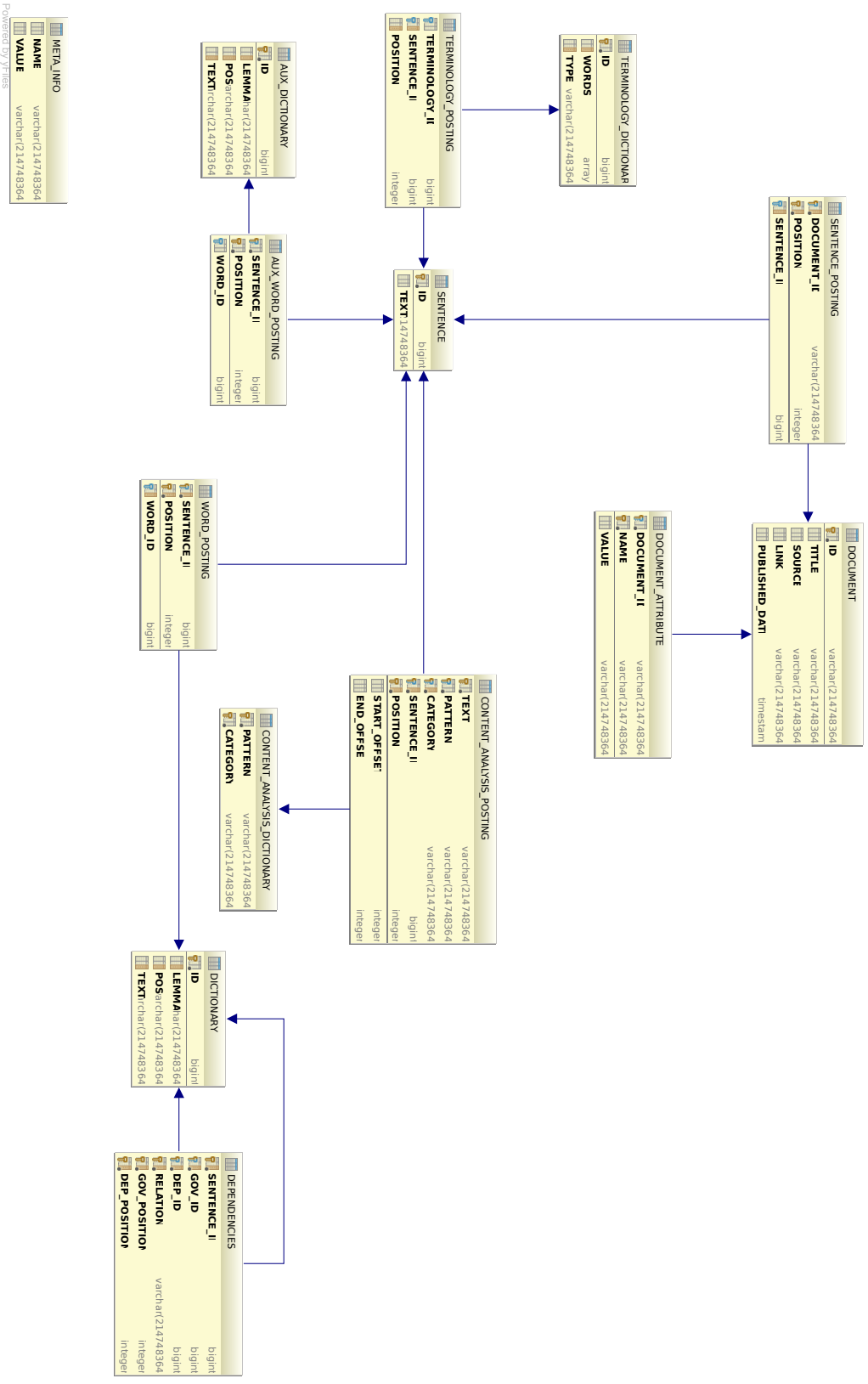


Figure 4.4: Schema for the Inverted Positional Index Database

Word ID	Text	Lemma	POS	Article ID	Pub. Date	Weight
2045	long-term	long-term	JJ	5a5878a3-03e3-4c86-979f-7098ee9779c3	31/08/2006	1
2169	state	state	NN	d376b3f1-638a-4885-91a4-088d049960d3	26/09/2013	1
14199	hole	hole	NN	e7ac88f2-c5de-4334-ac65-71de1578138e	16/10/2008	2
938	seen	see	VBN	9c4a4dbb-8701-4b48-9b11-3053c3d48837	19/12/2012	1
2628	payment	payment	NN	6f74a3d1-809d-4231-a39f-c44431f3605c	08/12/2012	5
9188	attempted	attempt	VBD	7846e03d-af69-480a-bee3-dbc0e27273dd	12/08/2009	1
545	moved	move	VBD	c216345f-fa4b-4b7e-84e0-c1c34858e514	24/05/2008	1
5444	jpmorgan	jpmorgan	NN	d3df4a25-16b2-4f82-9142-cbf027b1358f	15/06/2011	4
5580	told	tell	VBN	34df6229-eda8-47a7-b35e-924e4c4d949e	15/05/2012	1
6494	participant	participant	NN	9d3631fc-130b-4c3b-95de-e2a50df238e3	30/01/2007	1

Table 4.1: Example term-document matrix in sparse form produced by CiCui

In the most succinct form, the term-document matrix contains only three columns, namely `WORD_ID`, `ARTICLE_ID`, and `WEIGHT`, which record the IDs of the tokens, the IDs of the articles and the number of times said tokens occurred in said documents. The other columns in this example are included to improve readability. The `ARTICLE_TITLE` column is hidden to save page space.

convenient for the program to produce and parse the term-document matrix in this sparse form, but it also has its problems; in particular, the `ARTICLE_ID` column will not register 'empty' documents that contain no lexical items (e.g. unigrams or word dependencies) of interest[4]. One simple solution is to ignore such documents because they do not contain any information of interest. Another options is to retain these 'empty' documents, but set the frequencies of all lexical item in these documents to zero, leaving the stock market returns associated with these documents their sole contributions to the training process. One can see from Equation 3.97 that the stock price returns affect the estimation of lexical item distributions only through the responsibility term γ_{nk}; for a document n that is 'empty', the frequency for a lexical entry e_j for n is zero, i.e. $f_{n,e_j} = 0$; as a result, the returns associated to 'empty' documents cannot affect the estimation of $\alpha^{\text{new}}_{k,e_j}$s during the learning process. For the distribution of the temporal sentiment classes, however, it can be seen from Equation 3.96 that the estimation of π^{new}_k can actually be affected if 'empty' documents are ignored — for example, when there exists a tendency for documents from a certain temporal sentiment class are more likely to be 'empty'.

For the implementation in this thesis, I chose the former solution based on the observation that (i) the presence of empty document does not affect the estimation of the distributions of lexical items for the temporal sentiment categories, and (ii) such 'empty' documents are rare in general and their impacts should be minimal compared to the other 'non-empty' documents.

[4]If it is decided that closed class words are to be ignored, then a document containing only closed class words will still be considered 'empty'.

4.3 Model Learning and Evaluation

4.3.1 Brief Introduction to the KNIME Analytic Platform

The second component of the system, i.e. TSMiner, is implemented in the KNIME Analytic Platform[5]. Built upon the Eclipse application framework and Java, KNIME is a 'modern data analytics platform that allows you to perform sophisticated statistics and data mining on your data to analyse trends and predict potential results. Its visual workbench combines data access, data transformation, initial investigation, powerful predictive analytics and visualisation'[6].

There are several advantages using the KNIME platform to build analyses compared to starting from scratch:

- The platform provides the facilities to import and manage data from a variety of sources (e.g. database, structured files, etc.) in an uniformed data structure. The platform's infrastructure takes care of most of the rudimentary but critical data management tasks (e.g. off-heap on-demand data serialisation and de-serialisation), and also makes it easy to enable concurrent parallel processing when needed.

- It is possible to have data analytic procedures developed in heterogeneous languages collaborating in the same workflow. For example, one may combine scripts written in Java, R and Python when composing analyses.

- In KNIME, common data manipulation tasks such as joining, grouping, pivoting, and rule-based filtering, etc., are all packaged in atomic units called *nodes*, and the platform offers a large collection of readily usable nodes (Figure 4.5). The use of modular computation components significantly reduces development time.

Nodes can be connected with each other to form a workflow; data are represented as structured tables and passed along the paths that link the nodes. In general, a node has two sets of ports: a set of input ports which accept input data tables from other nodes, and a set of output ports to which the outputs of this node can be accessed. Each node type defines a certain operation on the data it accepts from the input ports; the output resulted from the operation is then passed to all the nodes that are connected to its output ports. Nodes also provide configuration interfaces through which the behaviours of the nodes can be fine-tuned. An sample workflow and its output is demonstrated in Figure 4.6.

Equipped with this new tool, I now proceed to describe the implementation of the algorithms developed in this thesis using the KNIME analytic platform in the following subsections.

[5]Version 2.12.0 of KNIME was used for the implementations described in this thesis.

[6]`https://www.knime.org/knime`, accessed in February 2016

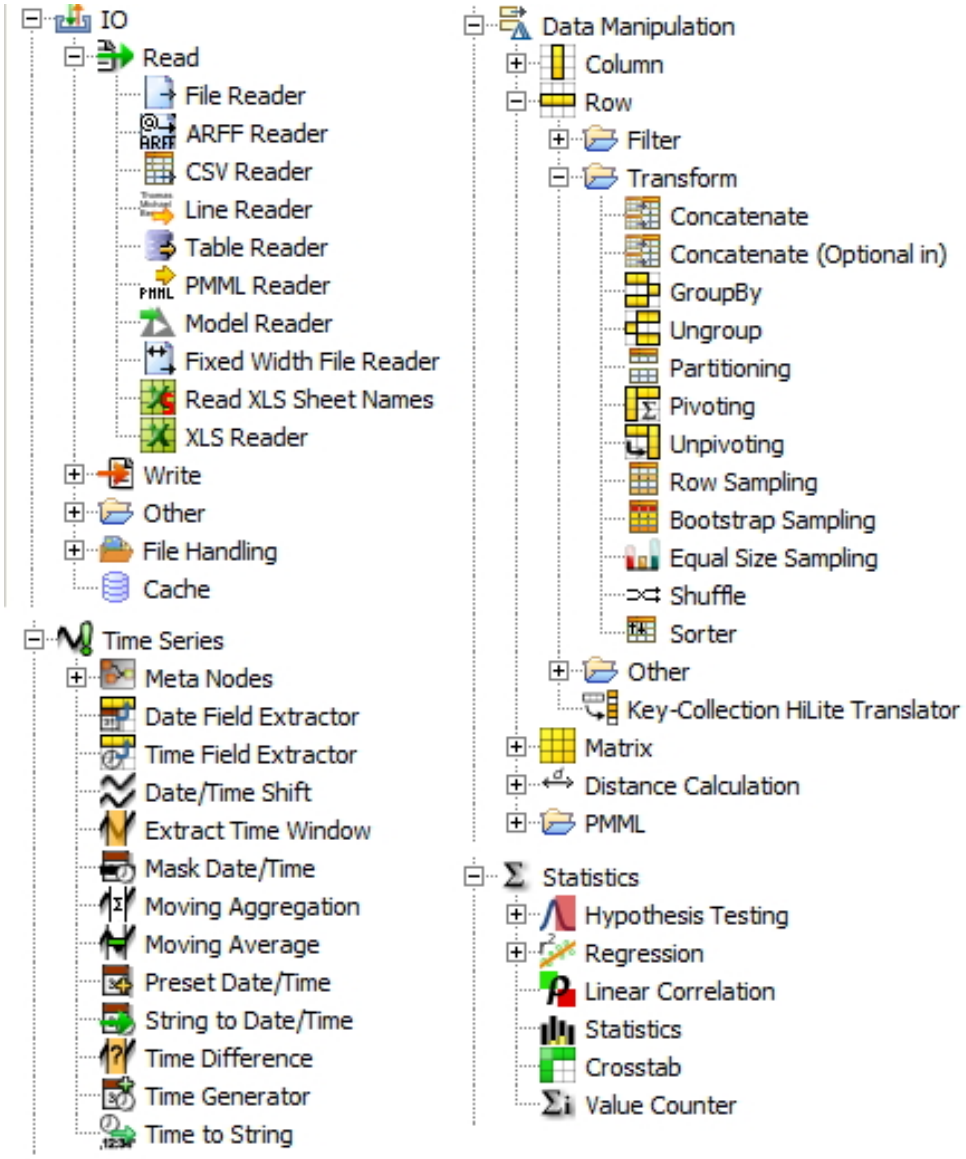
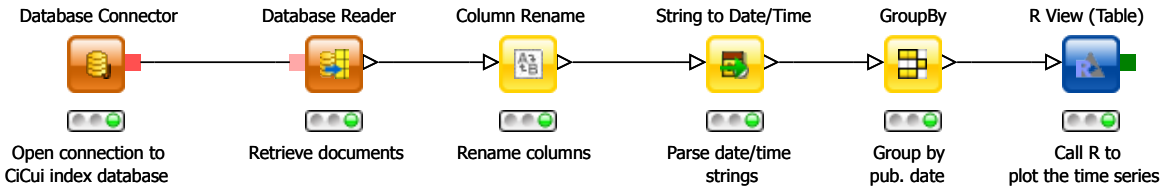


Figure 4.5: Some node collections offered by the KNIME platform

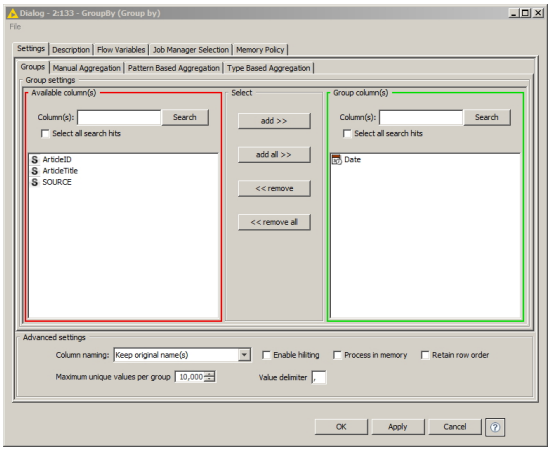
4.3.2 Design of the Workflow

4.3.2.1 Imputing Missing Interday Returns

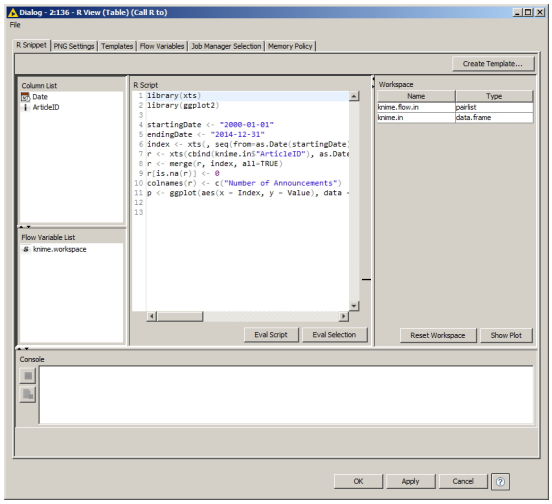
The calculation of the daily logarithmic returns to be used as the sentiment proxies during the learning phase can prove to be a challenge. The raw time series returned by `quantmod` functions are *irregular*, meaning there exist periods during which the prices are not available (e.g. weekends and public holidays); a direct consequence of this is that the daily returns calculated from these prices are also irregular. Sometimes, news articles are seen to be published on days when no returns are available, leaving them unaccounted for by sentiment proxies. For firm-level analysis, in general, textual data are not as abundant and thus more 'valuable' than price data, therefore it may be desirable to keep news articles which do not have accompanying sentiment proxies by assigning them imputed



(a) Sample workflow



(b) Configuring the GroupBy node



(c) Configuring the R script node

Figure 4.6: An example KNIME workflow

This example workflow (4.6a), when executed, opens an inverted index database produced by the CiCui system, retrieves the meta-information on the documents stored in the database, count the number of the documents released on each day (i.e. number of announcements) and plot the time series in R language using the ggplot library. Figure 4.6b and 4.6c showcase the configuration interfaces for the GroupBy node and the R scripting node.

returns.

Imputing missing equity returns itself can be complicated. Consider an example price series with N consecutive observations; the close prices are denoted by $p_1, p_2, \ldots, p_N$ and the interday logarithmic returns are denoted by $r_2, \ldots, p_N$ where $r_t = \log p_t - \log p_{t-1}$ — note that r_1 is not available (i.e. missing) since at least two close prices are needed to compute the interday returns. If the the close prices $p_t, p_{t+1}, \ldots, p_{t+n}$ are all absent, then there will be $n+1$ return values missing in the series (i.e. $r_t, r_{t+1}, \ldots, r_{t+n+1}$). The following imputing options were considered:

1. Substitute missing returns with the empirical mean of the observed returns, namely $r_t = r_{t+1} = \cdots = r_{t+n+1} = \frac{\sum_{i=2}^{t-1} r_i + \sum_{i=t+n+2}^{N} r_i}{N-n-1}$.

2. Substitute the missing returns with log differences between the two nearest close prices, namely $r_t = r_{t+1} = \cdots = r_{t+n+1} = \log p_{t+n+1} - \log p_{t-1}$.

3. Apply linear interpolation on the missing close prices, and calculate the missing returns based on the interpolated prices: a line is first fitted between p_{t-1} and p_{t+n+1}

by solving the α and β in the following system:

$$p_{t+n+1} = \alpha\,(t+n+1) + \beta \tag{4.1}$$

$$p_{t-1} = \alpha\,(t-1) + \beta \tag{4.2}$$

which leads to

$$\alpha = \frac{p_{t+n+1} - p_{t-1}}{n+2} \tag{4.3}$$

$$\beta = \frac{(n+1)\,p_{t-1} - p_{t+n+1}}{n+2} \tag{4.4}$$

for any day m that has $t \leq m \leq t+n$, the interpolated price p_m becomes:

$$p_m = \frac{p_{t+n+1} - p_{t-1}}{n+2} m + \frac{(n+1)\,p_{t-1} - p_{t+n+1}}{n+2}. \tag{4.5}$$

The calculations of the imputed values for the missing returns may then proceed as normal.

Using the mean of the observed values to replace the missing values (i.e. option 1) is a commonly used scheme for imputation in many applications; it has the benefit of not changing the mean of the returns after the imputation. However, as have been noted, equity return series are generally considered to have zero unconditional means — the method is therefore equivalent to replacing missing returns with close-to-zero returns. While statistically sound, adopting this method would defeat the purpose of the imputation as articles would be marked as if they are 'neutral' in sentiment, thus providing little new information.

The rationale behind option 2 is that the imputed sentiment proxies should reflect the news' contributions to the changes in prices, and since no trading was commenced during the period from t to $t+n$, whatever news released during that period will only have its effect manifested in the price fluctuations on the next trading day (i.e. day $t+n+1$). Under this scheme, the news article released from day t to day $t+n+1$ are attributed with all the change in prices from p_{t-1} to p_{t+n+1}. In contrast, option 3 spreads the change in prices across the $n+2$ days where the interday returns were not available; each article published on those days are attributed $\dfrac{1}{n+2}$ of the changes in price between time t and time $t+n$.

For imputing missing sentiment proxies, method 2 and 3 are equally applicable. However, the effects they have on the autocorrelation properties of the imputed series can differ greatly. More specifically, adopting option 2 introduces a significant amount of autocorrelation into the imputed series. This is not unexpected since consecutive duplicated returns are inserted into the series. Figure 4.7 compares the autocorrelation functions for three

return series. In 4.7c, the autocorrelation function considers only the correlations between non-missing returns and it shall serve as the baseline for the comparison; as can be seen in the diagram, the return series show slightly significant autocorrelation at first four lags. Figure 4.7a shows that the adoption of option 2 inflates the autocorrelations in the return series, while in 4.7b, calculating missing returns from interpolated close prices results in only a minor increase in the autocorrelations.

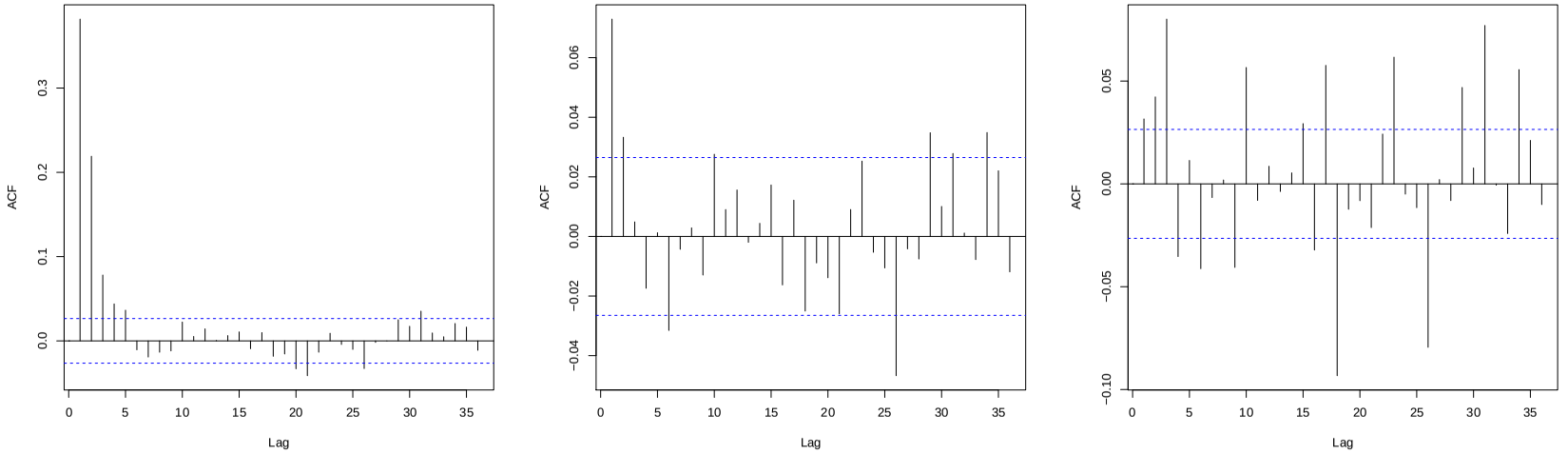
The increase in autocorrelations itself does not affect the training of the model, but may pose a problem when evaluating the methods' performances with econometrics techniques that involve autoregressive components (ARIMA and some GARCH models, for example), because greater than normal autocorrelations would trick such models into believing returns from previous times can predict the future returns.

Another factor to consider when choosing between the treatments for missing returns is the *weekend effect* (also known as the *Monday effect*, the *day-of-the-week effect*), which states that stock returns on Mondays (i.e. difference between the close price on Friday and the close price on the subsequent Monday) tend to be significantly lower than that on other weekdays [French, 1980]. When option 2 is adopted, all articles released during non-trading days are equally responsible for the full amount of the effect; in the case of option 3, the weekend effect's influence is diluted by the non-trading days. To account for the greater price changes around the non-trading days, it is decided in this implementation that missing returns in the training phase are treated with option 2, and option 3 is used when analysing the performances of the method.

4.3.2.2 Setting Up Cross-validation

Next, a cross-validation is set up so that the method can be tested multiple times and a more reliable conclusion can be drawn about the its performance. The cross-validation procedure tests how well the method generalises to new data. During a cross-validation process, the model of interest is repeatedly trained and tested on two separate datasets (i.e. the *training set* and the *test set*), with the training set and test set being changed each time; the accuracies and recalls of the model during the test phases are averaged to form a more reliable measure of the model's performance. By repeating the experiments multiple times

A typical cross-validation design used in machine learning literature is the k-fold cross-validation; a k-fold cross-validation first divides the complete dataset into k sub-datasets; the validation process then iterates through each sub-dataset, and uses the current sub-dataset as the test set while combining the remaining $k-1$ sub-datasets as the training set. One drawback of the k-fold cross-validation is that it can be difficult to obtain statistically significant conclusions during the test phases when the amount of the data available is relatively small or when k is large. In my method, an alternative



(a) ACF for interday returns with missing values padded by the next available returns (option 2)

(b) ACF for interday returns calculated from linear interpolated adjusted close prices (option 3)

(c) ACF for interday returns with missing values removed (baseline)

Figure 4.7: Comparison among the autocorrelation functions of the interday return series for The Boeing Company as treated by three different imputing methods.

The logarithmic returns used by the ACF were calculated from the adjusted close prices of The Boeing Company's stock on the New York Stock Exchange; the price series covers the time period from 2000-01-01 to 2014-12-31.

cross-validation scheme called the *repeated random sub-sampling validation* is implemented. A repeated random sub-sampling validation behaves much like the k-fold cross-validation except that the split between the training set and the test set is random in each iteration of the validation; with this cross-validation method, one is able to specify a fixed ratio between the size of the training set and the test set and repeat validations as many time as necessary, with each iteration receiving a reasonably sized test set. The disadvantage of this repeated random sub-sampling validation is that the some data points may never appear in the training/test sets due to the randomness in the splitting of the complete dataset.

Precautions are taken to prevent 'cheating' during the cross-validation process that have presented in some previous studies [Koppel and Shtrimberg, 2004, Généreux et al., 2011]. Features selection were performed strictly within the training set.

The results obtained from each fold of the cross-validation procedure are aggregated into a single metric using the technique of *meta-analysis*, which will be elaborated in Section 4.3.4.

4.3.2.3 Preparing Past and Future Returns

Within the cross-validation framework, the news articles in the training set will be joining with the past and future returns whose definitions were discussed in Section 3.4.3. To obtain the past return series, the imputed return series is first lagged forward by 1 day so that today's news is matched with yesterday's return; the exponential moving average is calculated from the lagged series using KNIME's Moving Average node; the coefficient α is defined by $\frac{2}{k+1}$ where k is the size of the averaging window. In this case, k is set to 1 for the calculation of past returns, which is equivalent to using yesterday's return as today's past return. For future returns, the original return series must be sorted in descending order before the moving average transformation can be applied since the MA node provided by KNIME support only backwards moving averaging; k is set to 3 when calculating the exponential moving average for the future returns.

4.3.2.4 Calculation of the Responsibility Term

Each iteration in the EM algorithm implementation begins with a SQL-style join between the lexical item postings and the probabilities of those lexical items in all four temporal sentiment categories from the previous iteration (i.e. the $\boldsymbol{\beta}^{\text{old}}s$); this process essentially looks up the probabilities distributions for the lexical items and attaches them to the postings list. An example for this procedure is illustrated by Figure 4.8. During the first iteration of the algorithm where probability distributions for the lexical items are unavailable, a set of randomly initialised distributions are used instead.

Dependency Tuple	Article ID	Pub. Date	Weight
agree,pay,xcomp	e20288a7-ac0a-4d5b-a1bd-f4b1db93b730	17/03/2005	5
agree,pay,xcomp	03a4a368-6521-4de7-b3bf-dc7de42571c4	08/07/2011	4
allow,borrow,xcomp	2a1a45ea-9b31-4a3e-b4ab-270d3d4fe6e3	16/04/2002	4
beat,forecast,dobj	426eeeae-d61f-43be-836b-6e987a1275dd	14/04/2000	4
buy,bond,dobj	8e2f1ca5-eb42-47f5-b8f2-1deba65d4e46	13/12/2011	4
buy,share,dobj	9e1f8342-1915-40bf-b5d4-62a2b028e98f	22/04/2006	4
buy,warrant,dobj	2d8b53bb-3d2d-4211-b0fe-03e3cd677986	10/07/2009	4
change,divisor,nsubj	83ce3834-5c0c-4ceb-9f9e-1fd17d5cb288	02/01/2001	10
charge,individual,dobj	b66d7b19-280f-41d9-bce2-37d264dc32b4	12/11/2012	4
decline,comment,xcomp	564918da-5f3b-4851-b302-e52c977357aa	19/09/2002	7
raise,money,dobj	0f38c316-6608-41c9-b1ef-df87c9a0e7be	27/06/2012	5
report,result,dobj	eaad4000-b2e0-4258-9718-eb5ff2b0fdc3	19/10/2000	6
rise,cent,dobj	e1b97041-52ad-47dc-a3af-5f4737ae7306	09/01/2003	5
sell,convertible,dobj	b5101da6-324c-45e0-94f9-04887e06195b	02/03/2000	6
sell,share,dobj	414f4ddc-9d5c-422e-83cc-f40451dcec75	13/09/2000	8
take,position,dobj	9d3631fc-130b-4c3b-95de-e2a50df238e3	30/01/2007	6

(a) Sample word dependency postings

Dependency Tuple	retneg	retpos	proneg	propos
agree,pay,xcomp	0.0005	0.0005	0.0010	0.0012
allow,borrow,xcomp	0.0000	0.0001	0.0000	0.0002
beat,forecast,dobj	0.0001	0.0002	0.0002	0.0002
buy,bond,dobj	0.0001	0.0001	0.0002	0.0001
buy,share,dobj	0.0004	0.0001	0.0003	0.0003
buy,warrant,dobj	0.0000	0.0000	0.0001	0.0001
change,divisor,nsubj	0.0005	0.0000	0.0000	0.0000
charge,individual,dobj	0.0000	0.0000	0.0001	0.0000
decline,comment,xcomp	0.0030	0.0018	0.0035	0.0028
raise,money,dobj	0.0002	0.0000	0.0004	0.0002
report,result,dobj	0.0006	0.0008	0.0003	0.0006
rise,cent,dobj	0.0008	0.0021	0.0003	0.0004
sell,convertible,dobj	0.0000	0.0000	0.0002	0.0000
sell,share,dobj	0.0006	0.0000	0.0001	0.0003
take,position,dobj	0.0002	0.0001	0.0002	0.0003

(b) Sample probability distributions for word dependencies

Dependency Tuple	retneg	retpos	proneg	propos	Article ID	Pub. Date	Weight
sell,convertible,dobj	0.0000	0.0000	0.0002	0.0000	b5101da6-324c-45e0-94f9-04887e06195b	02/03/2000	6
beat,forecast,dobj	0.0001	0.0002	0.0002	0.0002	426eeeae-d61f-43be-836b-6e987a1275dd	14/04/2000	4
sell,share,dobj	0.0006	0.0000	0.0001	0.0003	414f4ddc-9d5c-422e-83cc-f40451dcec75	13/09/2000	8
report,result,dobj	0.0006	0.0008	0.0003	0.0006	eaad4000-b2e0-4258-9718-eb5ff2b0fdc3	19/10/2000	6
change,divisor,nsubj	0.0005	0.0000	0.0000	0.0000	83ce3834-5c0c-4ceb-9f9e-1fd17d5cb288	02/01/2001	10
allow,borrow,xcomp	0.0000	0.0001	0.0000	0.0002	2a1a45ea-9b31-4a3e-b4ab-270d3d4fe6e3	16/04/2002	4
decline,comment,xcomp	0.0030	0.0018	0.0035	0.0028	564918da-5f3b-4851-b302-e52c977357aa	19/09/2002	7
rise,cent,dobj	0.0008	0.0021	0.0003	0.0004	e1b97041-52ad-47dc-a3af-5f4737ae7306	09/01/2003	5
agree,pay,xcomp	0.0005	0.0005	0.0010	0.0012	e20288a7-ac0a-4d5b-a1bd-f4b1db93b730	17/03/2005	5
buy,share,dobj	0.0004	0.0001	0.0003	0.0003	9e1f8342-1915-40bf-b5d4-62a2b028e98f	22/04/2006	4
take,position,dobj	0.0002	0.0001	0.0002	0.0003	9d3631fc-130b-4c3b-95de-e2a50df238e3	30/01/2007	6
buy,warrant,dobj	0.0000	0.0000	0.0001	0.0001	2d8b53bb-3d2d-4211-b0fe-03e3cd677986	10/07/2009	4
agree,pay,xcomp	0.0005	0.0005	0.0010	0.0012	03a4a368-6521-4de7-b3bf-dc7de42571c4	08/07/2011	4
buy,bond,dobj	0.0001	0.0001	0.0002	0.0001	8e2f1ca5-eb42-47f5-b8f2-1deba65d4e46	13/12/2011	4
raise,money,dobj	0.0002	0.0000	0.0004	0.0002	0f38c316-6608-41c9-b1ef-df87c9a0e7be	27/06/2012	5
charge,individual,dobj	0.0000	0.0000	0.0001	0.0000	b66d7b19-280f-41d9-bce2-37d264dc32b4	12/11/2012	4

(c) The table resulted from a joining between 4.8a and 4.8b on `Dependency Tuple`

Figure 4.8: Joining lexical item postings with their probability distributions over the temporal sentiment categories

Once the two tables are joined together, lexical items from the same documents are grouped together, after which the corresponding future and past returns as calculated in 4.3.2.3 are attached to each article.

With the aggregated table in place, I then proceed to evaluate the responsibility terms for the temporal sentiment categories as outlined in the E step in Section 3.5. Recall that the responsibility from the k-th category to the n-th document is

$$\gamma_{nk} = \frac{\pi_k^{\text{old}} p\left(\mathbf{d}_n, \mathbf{r}_n \mid \boldsymbol{\beta}_k^{\text{old}}, \mathbf{s}_{k,\text{p}}, \mathbf{s}_{k,\text{f}}\right)}{\sum_{j=1}^{K} \pi_j^{\text{old}} p\left(\mathbf{d}_n, \mathbf{r}_n \mid \boldsymbol{\beta}_j^{\text{old}}, \mathbf{s}_{j,\text{p}}, \mathbf{s}_{j,\text{f}}\right)} \tag{4.6}$$

where

$$p\left(\mathbf{d}_n, \mathbf{r}_n \mid \boldsymbol{\beta}_k^{\text{old}}, \mathbf{s}_{k,\text{p}}, \mathbf{s}_{k,\text{f}}\right) = \mathcal{S}\left(r_{n,\text{p}} \mid \mathbf{s}_{k,\text{p}}\right) \mathcal{S}\left(r_{n,\text{f}} \mid \mathbf{s}_{k,\text{f}}\right) p\left(\mathbf{d}_n \mid \boldsymbol{\beta}_k^{\text{old}}\right) \tag{4.7}$$

$$= \mathcal{S}\left(r_{n,\text{p}} \mid \mathbf{s}_{k,\text{p}}\right) \mathcal{S}\left(r_{n,\text{f}} \mid \mathbf{s}_{k,\text{f}}\right) \prod_{j=1}^{V} \alpha_{k,e_j}{}^{f_{n,e_j}} \tag{4.8}$$

The calculation of this quantity proves particularly challenging due to risk of arithmetic underflow — more specifically, the product of large quantities of probabilities (in this case $p\left(\mathbf{d}_n, \mathbf{r}_n \mid \boldsymbol{\beta}_k^{\text{old}}, \mathbf{s}_{k,\text{p}}, \mathbf{s}_{k,\text{f}}\right)$) can become so small that its value cannot be represented accurately by the standard floating point notation used in modern computers. Arithmetic underflow during the calculation of posterior probabilities is a common difficulty faced by Bayesian methods. A widely adopted strategy to tackle the problem is to preform the multiplications in logarithmic space and then convert the result back to linear space — with the logarithmic function applied, products of probabilities become sums of logarithmic probabilities, which is much safer to handle by computer programmes; the workaround bypasses the issue by delaying any direct operations on small probabilities in linear scale as much as possible until the magnitude of the calculations become manageable when converted back in linear scale.

To demonstrate, consider the responsibility term γ_{nk} presented above. Observe that while the individual terms in the numerator and denominator can be very small, the fraction itself represents a normalized probability and thus can be handled safely in linear scale, which indicates there might be an opportunity to transform the ratio in such a way that the underflow-prone calculations can be evaluated in logarithmic spaces before transforming the intermediate results back to linear spaces. To leverage this observation, rewrite the responsibility term as:

$$\gamma_{nk} = \exp\left\{\log \frac{\pi_k^{\text{old}} p\left(\mathbf{d}_n, \mathbf{r}_n \mid \boldsymbol{\beta}_k^{\text{old}}, \mathbf{s}_{k,\text{p}}, \mathbf{s}_{k,\text{f}}\right)}{\sum_{j=1}^{K} \pi_j^{\text{old}} p\left(\mathbf{d}_n, \mathbf{r}_n \mid \boldsymbol{\beta}_j^{\text{old}}, \mathbf{s}_{j,\text{p}}, \mathbf{s}_{j,\text{f}}\right)}\right\} \tag{4.9}$$

$$= \exp\left\{-\log \sum_{j=1}^{K} \frac{\pi_j^{\text{old}} p\left(\mathbf{d}_n, \mathbf{r}_n \mid \boldsymbol{\beta}_j^{\text{old}}, \mathbf{s}_{j,\text{p}}, \mathbf{s}_{j,\text{f}}\right)}{\pi_k^{\text{old}} p\left(\mathbf{d}_n, \mathbf{r}_n \mid \boldsymbol{\beta}_k^{\text{old}}, \mathbf{s}_{k,\text{p}}, \mathbf{s}_{k,\text{f}}\right)}\right\} \tag{4.10}$$

$$= \exp\left\{-\log \sum_{j=1}^{K} \exp\left\{\log \frac{\pi_j^{\text{old}} p\left(\mathbf{d}_n, \mathbf{r}_n \mid \boldsymbol{\beta}_j^{\text{old}}, \mathbf{s}_{j,\text{p}}, \mathbf{s}_{j,\text{f}}\right)}{\pi_k^{\text{old}} p\left(\mathbf{d}_n, \mathbf{r}_n \mid \boldsymbol{\beta}_k^{\text{old}}, \mathbf{s}_{k,\text{p}}, \mathbf{s}_{k,\text{f}}\right)}\right\}\right\} \tag{4.11}$$

The inner most logarithm

$$\log \frac{\pi_j^{\text{old}} p\left(\mathbf{d}_n, \mathbf{r}_n \mid \boldsymbol{\beta}_j^{\text{old}}, \mathbf{s}_{j,\text{p}}, \mathbf{s}_{j,\text{f}}\right)}{\pi_k^{\text{old}} p\left(\mathbf{d}_n, \mathbf{r}_n \mid \boldsymbol{\beta}_k^{\text{old}}, \mathbf{s}_{k,\text{p}}, \mathbf{s}_{k,\text{f}}\right)} \tag{4.12}$$

can be evaluated safely as additions of negative real numbers instead of multiplications of less-than-one probabilities without risking underflow, resulting in better numerical

stability:

$$\gamma_{nk} = \exp\Bigg\{ -\log \sum_{j=1}^{K} \exp \Bigg\{ \log \pi_j^{\text{old}} + \log \mathcal{S}\left(r_{n,\text{p}} \mid \mathbf{s}_{j,\text{p}}\right) + \log \mathcal{S}\left(r_{n,\text{f}} \mid \mathbf{s}_{j,\text{f}}\right) \tag{4.13}$$

$$+ \left(\sum_{m=1}^{V} f_{n,e_m} \log \alpha_{j,e_m} \right) - \log \pi_k^{\text{old}} - \log \mathcal{S}\left(r_{n,\text{p}} \mid \mathbf{s}_{k,\text{p}}\right) - \log \mathcal{S}\left(r_{n,\text{f}} \mid \mathbf{s}_{k,\text{f}}\right) \tag{4.14}$$

$$- \left(\sum_{m=1}^{V} f_{n,e_m} \log \alpha_{k,e_m} \right) \Bigg\} \Bigg\} \tag{4.15}$$

Technically it is possible to compute the inner most fraction 4.12 without using the $\exp\{\log \dots\}$ transformation — instead of computing the numerator and the denominator as a whole first and then process the division, one can breakdown the whole fraction into products of smaller fractions, each of which contains a numerator and a denominator that are of comparable magnitude. While this approach is feasible, special care has to be taken to ensure the computation is carried out in correct orders, making the exp-log transformation a more appealing option.

4.3.2.5 Smoothing Lexical Item Probability Distributions

Another issue that requires special attention is the smoothing of the probability distributions that characterise the lexical items for each temporal sentiment category. For very large corpora, the relative frequencies for some of the rarely occurring lexical items can become so small that their probabilities in the distributions are practically zero. Such near-zero probabilities can cause trouble when evaluating γ_{nk} as in Equation 4.15: the logarithms for near-zero probabilities are numerically unstable. In practice, taking logarithms of the relative frequencies of rare lexical item produces a `NaN` (which stands for *Not-a-Number*) value in most numerical computation languages, and even worse the `NaN` values would then also poison any subsequent calculations.

One solution to this issue is to apply *smoothing* to the probability distributions for the lexical items in the vocabulary. Smoothing of lexical item distributions is often employed to deal with the sparseness in data that arises when modelling texts with n-grams [Manning and Schütze, 1999, pg. 199]. Such sparseness is a side product of corpus sampling: a corpus can be considered as a 'sample' drawn from a population which contains all the texts humans have ever written; due to the stochastic nature of sampling, some rare lexical items will be inevitably overlooked by the process. As a result, the training corpus will not cover every possible language usage, and models trained from such language samples will have problem dealing with unseen usages. The smoothing process assigns estimated probabilities to lexical items that were not encountered during the training phases. The rationale behind such practice is to transfer some of the probability mass that was assigned

to frequent lexical items to the rare ones in an attempt to better represent their usages in the population texts when only incomplete samples (i.e. corpora) are available.

A widely adopted method for smoothing lexical item frequencies is Laplace smoothing (also called add-one smoothing). This procedure operates by adding 1 to each and all lexical items' raw frequencies as occurred in the corpus. Let f_{e_j} denote the raw frequency of lexical item e_j in the entire corpus, the maximum likelihood estimator α_{e_j} for the probability of a lexical item e_j occurring in the text when no smoothing is applied is

$$\alpha_{e_j} = \frac{f_{e_j}}{\sum_{m=1}^{V} f_{e_m}}; \tag{4.16}$$

the smoothed probability α'_{e_j} is given by

$$\alpha'_{e_j} = \frac{f_{e_j} + 1}{\sum_{m=1}^{V} f_{e_m} + V} \tag{4.17}$$

where the V is the size of the vocabulary. By adding 1 to each lexical item's frequency, Laplace smoothing assumes each lexical item occurred at least once in the corpus; an alternative interpretation of the assumption is that each lexical item occurred at least $\frac{1}{N}$ times in each of the N documents. Figure 4.9 illustrates the effect of Laplace smoothing on the distribution of frequencies over single words. Other more sophisticated smoothing techniques include Good-Turing smoothing [Good, 1953] and Witten-Bell smoothing [Witten and Bell, 1991]. The implementation of these methods can be complicated and it has been suggested that applying them resulted in classification performances inferior than Laplace smoothing [Dave et al., 2003].

For the temporal sentiment mixture model developed in this thesis, the smoothed probability for lexical item e_j in the distribution charactering the k-th category is defined similarly to the expression given above. In particular, α'_{k,e_j}, namely the smoothed version of α_{k,e_j}, can be written as

$$\alpha'_{k,e_j} = \frac{1 + \sum_{n=1}^{N} \gamma_{nk} \cdot f_{n,e_j}}{\sum_{m=1}^{V} \left(1 + \sum_{n=1}^{N} \gamma_{nk} \cdot f_{n,e_m}\right)} \tag{4.18}$$

$$= \frac{1 + \sum_{n=1}^{N} \gamma_{nk} \cdot f_{n,e_j}}{V + \sum_{n=1}^{N} \sum_{m=1}^{V} \gamma_{nk} \cdot f_{n,e_m}} \tag{4.19}$$

The expression 4.19 essentially adapts estimator given by 4.17 to use *soft counts* — instead of making the assumption that each lexical item occurred at least $\frac{1}{N}$ times in each document, the smoothed estimator 4.19 entertains only the portion of occurrences for which the k-th category is responsible.

An alternative and perhaps more general solution is to incorporate a *prior* distribution

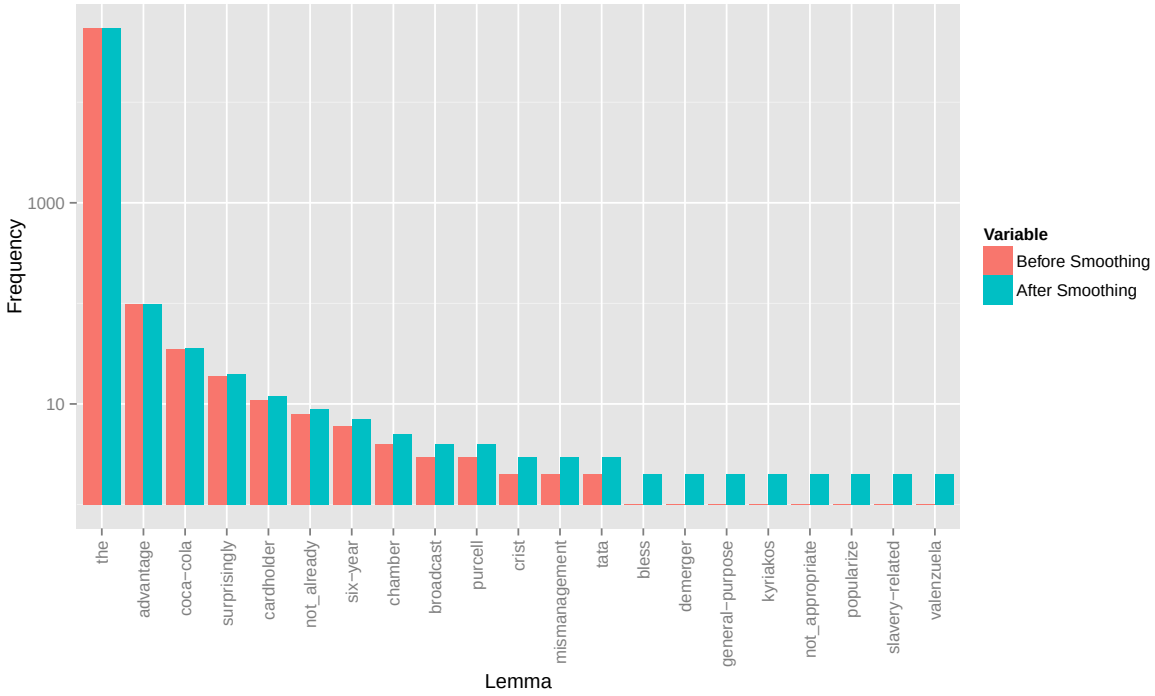


Figure 4.9: Lexical item frequency distribution: before and after smoothing

This chart visualises the effect Laplace smoothing has on the frequencies distributions over lexical items. The horizontal axis (i.e. Lemma) displays 21 lemmas sampled from the vocabulary of a corpus; the words in the vocabulary were first sorted according to their raw frequencies in the text in descending order; the 21 lemmas were obtained by sampling every 1000th record from the sorted vocabulary. The vertical axis is transformed to logarithm scale to better illustrate the differences between the frequencies before and after the smoothing.

for the lexical items' probabilities of occurrence in the texts; this solution effectively leads to the derivation of the *maximum a posteriori* (referred to as MAP hereinafter) estimator for the lexical item distributions instead of the *maximum likelihood* estimator. The MAP of a parameter returns the parameter value that maximises the p.d.f. of the posterior probability distribution, that is, the conditional distribution of the parameter after observing new data. Formally:

$$\boldsymbol{\theta}_{\text{MAP}} = \underset{\boldsymbol{\theta}}{\operatorname{argmax}}\, P\left(\boldsymbol{\theta} \mid \mathbf{X}\right) \tag{4.20}$$

$$= \underset{\boldsymbol{\theta}}{\operatorname{argmax}}\, \frac{P\left(\mathbf{X} \mid \boldsymbol{\theta}\right) P\left(\boldsymbol{\theta}\right)}{P\left(\mathbf{X}\right)} \tag{4.21}$$

$$\propto \underset{\boldsymbol{\theta}}{\operatorname{argmax}}\, P\left(\mathbf{X} \mid \boldsymbol{\theta}\right) P\left(\boldsymbol{\theta}\right) \tag{4.22}$$

where $P\left(\mathbf{X} \mid \boldsymbol{\theta}\right)$ is the usual likelihood function for observing $\mathbf{X}$, and $P\left(\boldsymbol{\theta}\right)$ is prior distribution for the parameter $\boldsymbol{\theta}$.

The derivation of the MAP version of the EM algorithm is discussed in detail in Appendix A. Some changes have to be made to both the E and the M steps to accommodate

the prior distributions for the $\boldsymbol{\beta}$s. In particular, the responsibility term γ_{nk} is evaluated as the following during the E step:

$$\gamma_{nk} = \frac{\pi_k^{\text{old}} p\left(\mathbf{d}_n, \mathbf{r}_n \mid \boldsymbol{\beta}_k^{\text{old}}, \mathbf{s}_{k,\text{p}}, \mathbf{s}_{k,\text{f}}\right) p\left(\boldsymbol{\beta}_k^{\text{old}}\right)}{\sum_{j=1}^{K} \pi_j^{\text{old}} p\left(\mathbf{d}_n, \mathbf{r}_n \mid \boldsymbol{\beta}_j^{\text{old}}, \mathbf{s}_{j,\text{p}}, \mathbf{s}_{j,\text{f}}\right) p\left(\boldsymbol{\beta}_j^{\text{old}}\right)} \tag{4.23}$$

where $p\left(\boldsymbol{\beta}_k^{\text{old}}\right)$ represents the p.d.f. of the prior distribution for $\boldsymbol{\beta}_k^{\text{old}}$; the $\mathcal{Q}\left(\boldsymbol{\theta}, \boldsymbol{\theta}^{\text{old}}\right)$ function to be maximised in the M step becomes

$$\mathcal{Q}\left(\boldsymbol{\theta}, \boldsymbol{\theta}^{\text{old}}\right) = \sum_{\mathbf{Z}} p\left(\mathbf{Z} \mid \boldsymbol{\mathcal{D}}, \mathbf{R}', \boldsymbol{\pi}^{\text{old}}, \mathbf{B}^{\text{old}}, \mathbf{S}\right) \ln\left\{p\left(\boldsymbol{\mathcal{D}}, \mathbf{R}', \mathbf{Z} \mid \boldsymbol{\pi}, \mathbf{B}, \mathbf{S}\right) p\left(\mathbf{B}\right)\right\}. \tag{4.24}$$

In theory, one may choose any valid distribution over $\boldsymbol{\beta}$ to be $p\left(\boldsymbol{\beta}_k\right)$, yet the popular choice is the *Dirichlet distribution*. The Dirichlet distribution is a family of continuous multivariate probability distributions over vectors comprised of K random variables, $\mu_1, \ldots, \mu_K$, subjecting to the following constraint:

$$0 \leq \mu_k \leq 1 \text{ and } \sum_{k=1}^{K} \mu_k = 1. \tag{4.25}$$

The Dirichlet distribution is parameterised by another vector consisting of K positive reals: $\boldsymbol{\alpha} = (\alpha_1, \ldots \alpha_K)$. Formally, the p.d.f. for a Dirichlet distribution is given by

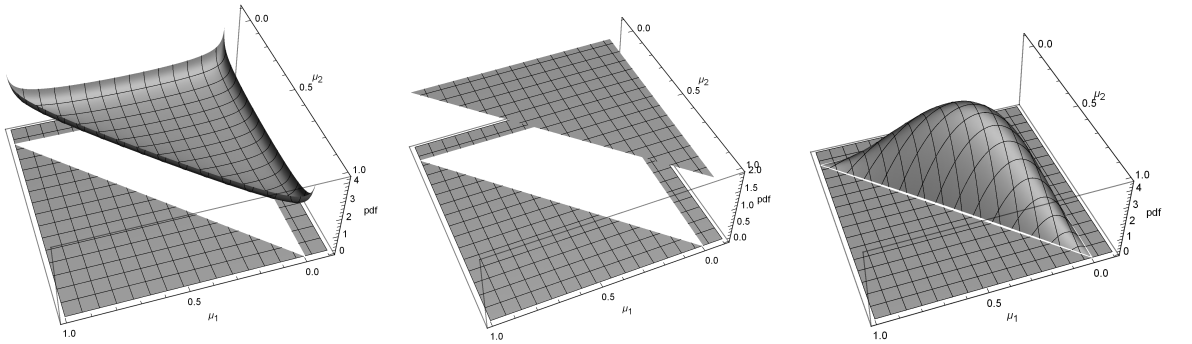
$$\text{Dir}\left(\boldsymbol{\mu} \mid \boldsymbol{\alpha}\right) = \frac{\Gamma\left(\sum_{k=1}^{K} \alpha_k\right)}{\sum_{k=1}^{K} \Gamma\left(\alpha_k\right)} \prod_{k=1}^{K} \mu_k^{\alpha_k - 1} \tag{4.26}$$

where $\Gamma\left(\cdot\right)$ denotes the Gamma function:

$$\Gamma\left(t\right) = \int_0^{\infty} x^{t-1} e^{-x} dx \tag{4.27}$$

which is an extension of the factorial function for real numbers, that is, $\Gamma\left(t\right) = t \cdot \Gamma\left(t-1\right)$ for any real number t.

Intuitively, the Dirichlet distribution defines a distribution over multinomial distributions — in other words, a sample drawn from a Dirichlet distribution effectively represents a multinomial distribution. Its parameter $\boldsymbol{\alpha}$, therefore, determines how probable it is to draw a multinomial distribution of a certain type. For example, when $\alpha_1 = \alpha_2 = \cdots = \alpha_K < 1$, it is more probable to draw from the Dirichlet distribution uneven multinomial distributions in which one component's probability dominates the others; with $\alpha_1 = \alpha_2 = \cdots = \alpha_K > 1$ and increasing, it is more and more probable to draw multinomial distributions that assign even probabilities to each of its K components (Figure 4.10).



(a) $\boldsymbol{\alpha} = (0.9, 0.9, 0.9)$ (b) $\boldsymbol{\alpha} = (1.0, 1.0, 1.0)$ (c) $\boldsymbol{\alpha} = (2.0, 2.0, 2.0)$

Figure 4.10: The p.d.f. of the Dirichlet distribution at various parameter configurations

These plots illustrates the p.d.f. for Dirichlet distributions with $K = 3$ at different parameter configurations. For clarity, the same value is used for all three components in the parameter vector $\boldsymbol{\alpha}$. Note that when $\boldsymbol{\alpha} = (1.0, 1.0, 1.0)$, all multinomial distributions, even or uneven, are assigned equal probabilities. With $\alpha_k < 1$, it is more probable to encounter multinomial distributions with extreme probabilities assigned to one single component, and when $\alpha_k > 1$, it is more probable for the Dirichlet distribution to generate multinomial distributions with even probabilities assigned among its components.

One of the most appealing reasons for choosing the Dirichlet distribution as the prior for the $\boldsymbol{\beta}$s is that the Dirichlet distribution constitutes the *conjugate prior* for the multinomial distributions that govern the generation of the lexical items. A probability distribution $p(\boldsymbol{\theta})$ is considered a conjugate prior to a likelihood function $p(\mathbf{X} \mid \boldsymbol{\theta})$ if the posterior distribution $p(\boldsymbol{\theta} \mid \mathbf{X})$ is from the same distribution family as the prior distribution.

Choosing a distribution that conjugates with the likelihood function as the prior distribution has the benefit of leading to closed-form solutions for the maximum a posteriori estimators:

$$0 = \frac{\partial \mathcal{Q}\left(\boldsymbol{\theta}, \boldsymbol{\theta}^{\text{old}}\right) + \sum_{k=1}^{K} \lambda \left(\sum_{j=1}^{V} \mu_{k,e_j} - 1\right)}{\partial \mu_{k,e_j}} \tag{4.28}$$

$$0 = \frac{\partial \sum_{n=1}^{N} \sum_{k=1}^{K} \gamma_{nk} \left\{ \ln \prod_{j=1}^{V} \mu_{k,e_j}^{f_{n,e_j}} + \ln \frac{\Gamma\left(\sum_{j=1}^{V} \alpha_j\right)}{\sum_{j=1}^{V} \Gamma(\alpha_j)} \prod_{j=1}^{V} \mu_{k,e_j}^{\alpha_j - 1} \right\}}{\partial \mu_{k,e_j}} \tag{4.29}$$

$$+ \frac{\partial \sum_{n=1}^{N} \sum_{k=1}^{K} \gamma_{nk} \left\{ \ln \pi_k + \ln \mathcal{S}\left(r_{n,\text{p}}, r_{n,\text{f}} \mid \mathbf{s}_{k,\text{p}}, \mathbf{s}_{k,\text{f}}\right) \right\}}{\partial \mu_{k,e_j}} \tag{4.30}$$

$$+ \frac{\partial \sum_{k=1}^{K} \lambda_k \left(\sum_{j=1}^{V} \mu_{k,e_j} - 1\right)}{\partial \mu_{k,e_j}} \tag{4.31}$$

$$0 = \lambda_k + \sum_{n=1}^{N} \gamma_{nk} \frac{\partial \ln \mu_{k,e_j}^{f_{n,e_j} + \alpha_j - 1}}{\partial \mu_{k,e_j}} \tag{4.32}$$

$$0 = \lambda_k + \sum_{n=1}^{N} \gamma_{nk} \left(f_{n,e_j} + \alpha_j - 1\right) \frac{\partial \ln \mu_{k,e_j}}{\partial \mu_{k,e_j}} \tag{4.33}$$

Note that 4.33 is of similar form as 3.87, only that $f_{n,e_j} + \alpha_j - 1$ replaced f_{n,e_j} inside the summation term. It can be shown that the resulting MAP estimator is given by the following expression:

$$\mu_{k,e_j} = \frac{\sum_{n=1}^{N} \gamma_{nk} \cdot \left(f_{n,e_j} + \alpha_j - 1\right)}{\sum_{n=1}^{N} \sum_{j=1}^{V} \gamma_{nk} \cdot \left(f_{n,e_j} + \alpha_j - 1\right)} \tag{4.34}$$

By comparing 4.19 with 4.34, it can be seen that Laplace smoothing is in fact a special case for this approach — it is equivalent to applying a Dirichlet prior whose parameters comply with the following constraint:

$$\alpha_1 = \alpha_2 = \cdots = \alpha_V = 2. \tag{4.35}$$

Note that applying an uninformed prior (i.e. $\alpha_1 = \alpha_2 = \cdots = \alpha_V = 1$) is equivalent to applying no smoothing, in which case μ_{k,e_j} reduces to the maximum likelihood estimator.

4.3.2.6 Calculating Log-Likelihood with the Log-Sum-Exp Technique

As mentioned earlier, the algorithm shall be considered converged when the relative increase in the log-likelihood (i.e. $p\left(\mathbf{X} \mid \boldsymbol{\theta}\right)$) between two iterations of the algorithm falls below a set threshold. The computation of the log-likelihood involves the evaluation of the following:

$$\ln p\left(\mathbf{X} \mid \boldsymbol{\theta}\right) = \ln \sum_{\mathbf{Z}} p\left(\mathbf{X}, \mathbf{Z} \mid \boldsymbol{\theta}\right) \tag{4.36}$$

$$= \ln \sum_{\mathbf{Z}} \prod_{n=1}^{N} p\left(\mathbf{x}_n, \mathbf{z}_n \mid \boldsymbol{\theta}\right) \tag{4.37}$$

The computation of the likelihoods from the individual components (i.e. $p\left(\mathbf{X}, \mathbf{Z} \mid \boldsymbol{\theta}\right) = \prod_{n=1}^{N} p\left(\mathbf{x}_n, \mathbf{z}_n \mid \boldsymbol{\theta}\right)$) poses a challenge similar to that encountered during the evaluation of the responsibility terms — for any non-trivial-sized dataset, numerical underflow can occur when calculating products of a large amount of probabilities. Unlike the responsibility terms, it is not possible to transform the products to sums in logarithmic space and have them cancel each other out before converting them back into linear space as was done for the responsibility terms. The safe evaluation of the likelihoods requires an alternative method called the *log-sum-exp* technique.

The log-sum-exp technique states that the following transformation applies:

$$\log \sum_{n=1}^{N} e^{x_n} = \log \sum_{n=1}^{N} e^{x_n - a} \cdot e^{a} \tag{4.38}$$

$$= a + \log \sum_{n=1}^{N} e^{x_n - a} \tag{4.39}$$

where a can be any real number. The rationale behind the technique is that by subtracting a from the x_n, the magnitudes of the exponents on e are reduced to such a level that the term $e^{x_n - a}$ can be evaluated safely without overflowing. One common choice for a for this purpose is to let $a = \max(x_1, \ldots, x_N)$.

Utilising the transformation described above, 4.37 can be re-written as:

$$\ln p(\mathbf{X} \mid \boldsymbol{\theta}) = \ln \sum_{\mathbf{Z}} \prod_{n=1}^{N} p(\mathbf{x}_n, \mathbf{z}_n \mid \boldsymbol{\theta}) \tag{4.40}$$

$$= \ln \sum_{\mathbf{Z}} e^{\ln \prod_{n=1}^{N} p(\mathbf{x}_n, \mathbf{z}_n \mid \boldsymbol{\theta})} \tag{4.41}$$

$$= a + \ln \sum_{\mathbf{Z}} e^{-a + \sum_{n=1}^{N} \ln p(\mathbf{x}_n, \mathbf{z}_n \mid \boldsymbol{\theta})} \tag{4.42}$$

with a being the maximum value among all possible $\sum_{n=1}^{N} \ln p(\mathbf{x}_n, \mathbf{z}_n \mid \boldsymbol{\theta})$ configurations relative to $\mathbf{Z}$. After the transformation, $\ln p(\mathbf{x}_n, \mathbf{z}_n \mid \boldsymbol{\theta})$ can be evaluated fairly easily for each document within logarithmic space, and by subtracting a, it is possible to calculate $e^{-a + \sum_{n=1}^{N} \ln p(\mathbf{x}_n, \mathbf{z}_n \mid \boldsymbol{\theta})}$ without risking numerical underflow.

4.3.3 Model Comparison with Linear Correlation

Once the training of the model is complete, the model is ready to be evaluated on the test set. The evaluation of the method developed in this thesis mainly focuses on its predictive performances when classifying unseen news articles with respect to the four temporal-sentiment categories.

Traditionally, the performance of a classifier is evaluated using metrics like accuracy and error rate. Such metrics are best illustrated with the help of a *confusion matrix*; in its simplest form, a confusion matrix tabulates the number of cases in the test set that were correctly classified by the model along side with the number of cases that were misclassified. The accuracy metric measures the percentage of the cases that were classified correctly among all cases when the classifier of interest is applied. The higher the accuracy from the evaluation phase the greater the predictive power of the model when dealing with unseen data. The error rate is simply the complement of accuracy, namely: error rate $= 1 -$ accuracy.

One limitation with accuracy/error-rate based evaluations is that it only applies to scenarios where the domain of the classification target is discrete. For regression models where the target is continuous, the *mean squared error* (MSE) is often used in lieu of accuracy/error-rate to evaluate the quality of the models. The MSE for predictor is defined as

$$\text{MSE} = \frac{\sum_{i=1}^{N} (\hat{y}_i - y_i)^2}{N - 2} \tag{4.43}$$

where $\hat{y}_i$ is the prediction from the model while the y_i is the corresponding observed value in the test set. The MSE measures the average 'deviation' of the predicted values from the actual observations — the greater the MSE, the less accurately the model predicts out-of-sample inputs.

In the case of this thesis, however, neither of the above two metrics are directly applicable due to the semi-supervised nature of the algorithm. The predicted probabilities over the temporal sentiment categories for each article constitute four continuous variables, which makes evaluation with accuracy/error-rate inapplicable. In many applications where Bayesian classifiers are involved, the probability distributions assigned to each case are often 'discretised' into a single outcome as a way to circumvent the problem. Usually, the most probable class for a observation case, that is, the class assigned the highest probability by the classifier, is chosen as the classification outcome for that particular case. However, even with discretisation, the lack of actual temporal-sentiment series which would serve as the golden standard still precludes the use of both the accuracy/error-rate and the MSE as evaluation measures — the only extrinsic references available are the past and future returns, which are merely proxies to the news' actual temporal sentiment orientations, not the true labels.

To overcome these difficulties, one may use the correlation between the return series and the temporal sentiment probability series as a measure for the predictive performance of the model. The simplest form of such correlation measure is the Pearson's linear correlation coefficient. The linear correlation between two variables X and Y measures the direction and strength of the linear relationship between the two variables. Mathematically, the coefficient, often denoted by $\rho_{X,Y}$, is defined as

$$\rho_{X,Y} = \frac{\text{cov}\,(X, Y)}{\sigma_X \sigma_Y} \tag{4.44}$$

where σ_X and σ_Y are the variances of X and Y respectively; $\text{cov}\,(X, Y)$ is the covariance between X and Y:

$$\text{cov}\,(\text{X}, \text{Y}) = \text{E}\left[(X - \text{E}\left[X\right])\,(Y - \text{E}\left[Y\right])\right]. \tag{4.45}$$

In reality where only sampled observations are available for X and Y, the linear

correlation coefficient between the two datasets r_{xy} can be estimated as:

$$r_{X,Y} = \frac{\sum_{n=1}^{N}(x_n - \bar{x})(y_n - \bar{y})}{\sqrt{\sum_{n=1}^{N}(x_n - \bar{x})}\sqrt{\sum_{n=1}^{N}(y_n - \bar{y})}} \tag{4.46}$$

where $x_1, x_2, \ldots, x_N$ and $y_1, y_2, \ldots, y_N$ are the samples for X and Y respectively while $\bar{x}$ and $\bar{y}$ are the sample means calculated for $x_1, x_2, \ldots, x_N$ and $y_1, y_2, \ldots, y_N$ respectively.

Specific to this case, X would be the temporal sentiment probabilities associated with each of the new documents about a certain company, and Y would be the daily returns of the company's stock; the calculation of the former was elaborated in Section 3.6, and the preparation of the latter was described in Section 4.3.2.3.

It is worth noting that an alternative approach to using Pearson's linear correlation coefficient is to fit a simple linear regression model to the datasets (i.e. $Y = \beta_1 X + \beta_0$) and use the fitting of the model (e.g. R^2 or MSE) to compare the predictive performances of different models. Statistically, this should produce results equivalent to that acquired by using Pearson's linear correlation coefficient. For example, the mean squared error is defined as:

$$\text{MSE} = \frac{\sum_{i=1}^{N}(\hat{y}_i - y_i)^2}{N-2}. \tag{4.47}$$

By the definition of simple linear regression, the following is also true:

$$\hat{y}_i = \frac{\text{cov}(X,Y)}{\text{cov}(X,X)}x_i + \bar{y} - \frac{\text{cov}(X,Y)}{\text{cov}(X,X)}\bar{x} \tag{4.48}$$

$$\hat{y}_i = \frac{\text{cov}(X,Y)}{\text{cov}(X,X)}(x_i - \bar{x}) + \bar{y} \tag{4.49}$$

$$= \frac{\text{cov}(X,Y)}{\sqrt{\text{var}(X)}\sqrt{\text{var}(Y)}}\frac{\sqrt{\text{var}(Y)}}{\sqrt{\text{var}(X)}}(x_i - \bar{x}) + \bar{y} \tag{4.50}$$

$$= r_{XY}\frac{\sigma_Y}{\sigma_X}(x_i - \bar{x}) + \bar{y} \tag{4.51}$$

Substitute $\hat{y}$ into the definition of MSE gives:

$$\text{MSE} = \frac{\sum_{i=1}^{N}\left(r_{XY}\frac{\sigma_Y}{\sigma_X}(x_i - \bar{x}) + \bar{y} - y_i\right)^2}{N-2} \tag{4.52}$$

$$= \frac{\sum_{i=1}^{N} r_{XY}^2\frac{\sigma_Y^2}{\sigma_X^2}(x_i - \bar{x})^2 + \sum_{i=1}^{N}(\bar{y} - y_i)^2 - 2\sum_{i=1}^{N} r_{XY}\frac{\sigma_Y}{\sigma_X}(x_i - \bar{x})(y_i - \bar{y})}{N-2} \tag{4.53}$$

$$= \frac{r_{XY}^2\sigma_Y^2(N-1)}{N-2} + \frac{(N-1)\sigma_Y^2}{N-2} - \frac{2(N-1)\sigma_Y^2 r_{XY}^2}{N-2} \tag{4.54}$$

$$= \frac{(N-1)}{(N-2)}\sigma_Y^2\left(1 - r_{XY}^2\right) \tag{4.55}$$

In other words, the MSE for a simple linear regression model fitted with independent variable X and dependent variable Y is essentially a function of Pearson's linear correlation coefficient and the variance of Y. Since I will be comparing the models on the same set of test data, the σ_Y^2 (i.e. the variance of the return series) will be the same for all the models. The only factor that affects the MSE therefore is r_{XY}, i.e. the linear correlations between X and Y — the larger the magnitude of r_{XY}, the lower the mean squared error. Also, the expression for the coefficient β_1 can also be written as a function of r_{XY} and the ratio between σ_Y and σ_X, so the direction (i.e. sign) of the correlation coefficient is consistent with the regression coefficient.

One issue that requires some consideration when calculating the linear correlations is the treatment of incomplete observations where the temporal sentiment probabilities are missing. It can be the case that no news are announced for a company of interest during the majority of the days in the test set. Since the calculation of the linear correlation coefficient requires that no missing values is present in the dataset, one has to either discard the incomplete cases from the dataset or impute the missing values. For this case study, the missing values in the series were imputed using the same method as discussed in Section 4.3.2.1.

4.3.4 Benchmarking with Meta-analysis

The set-up of cross-validation using repeated random sub-sampling validation was discussed in Section 4.3.2.2. The cross-validation procedure is conducted for each of the models to be compared. For every model, the procedure produces k correlation coefficients, one from each fold; these correlation coefficients are then combined into a single aggregated correlation using a technique called *correlation meta-analysis* [McDonald, 2015, pg. 264]. The aggregated correlations are used to compare and benchmark the performances of the models.

Meta-analysis is a collection of statistical methods and techniques that summarises the *effect sizes* (e.g. linear correlations, mean differences, or any measure of a phenomenon's strength) obtained from multiple experiments into a aggregated effect size. The rationale behind meta-analysis techniques follows that of the classical statistics: each experiment is considered to have been performed against dataset sampled from the population; the effect sizes estimated from the multiple samples (for each experiment) would follow a distribution, with the true effect size of the population being the mean of the sample effect size distribution. This enables statistical inference on the distributional properties of the effect sizes from the samples as to how close they are to the true effect size of the population.

Before applying meta-analysis techniques, it is necessary to distinguish between two types of models: the fixed effect models and the random effect models. A 'fixed effect'

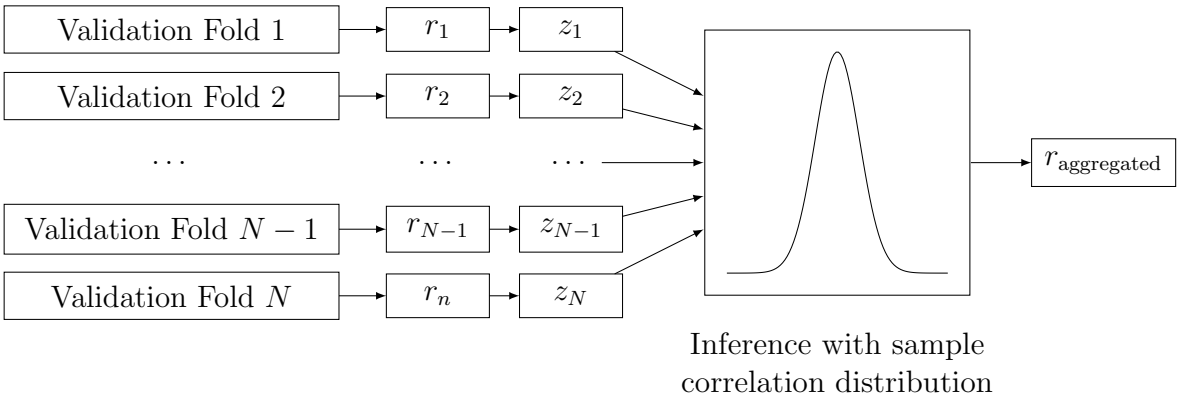


Figure 4.11: Schematic diagram illustrating the principle of correlation meta-analyses for fixed effect models

Each fold of the cross-validation procedure produces a sample correlation coefficient, i.e. r_n; these correlation coefficients are then subjected to the *Fisher z-transformation*, which turn them into z-scores, i.e. $z_n = \frac{1}{2}\ln\frac{1+r_n}{1-r_n}$. The transformed z_n would approximately follow a normal distribution with their mean being a function of the true population correlation ($\hat{z} = \frac{1}{2}\ln\frac{1+\rho}{1-\rho}$); the standard error of the mean is a function of the sample size N (standard error $= \frac{1}{\sqrt{N-1}}$). The mean and variance of z_n can then be used to make statistical inferences on the true population correlations.

model assumes that the samples gathered from the different studies were all drawn from the same distribution governed by a single set of parameters, and a 'random-effects' model assumes that parameters that governed the underlying distributions from which the samples were drawn themselves follow some distribution [Higgins et al., 2009]. The difference becomes relevant when meta-analysis is applied in clinical and medical meta-reviews since studies of homogeneous nature are relatively rare in these fields. For this case, the set-up of the experiments can be considered a fixed effect model since each fold operates on a subset of a single population comprised of news articles, and the procedures followed by each fold are exactly the same. The general principle of a meta-analysis on linear correlation coefficients is illustrated by Figure 4.11.

The meta-analysis on the correlation coefficients obtained from each fold is performed using the `metacor` package in R. The package offers routines that test the null hypothesis that the true correlation coefficient for the population does not differ significantly from zero against the alternative hypothesis that the true correlation coefficient differs from zero. The output of the procedure includes both the aggregated r_{XY} from the sample correlation coefficients and the associated p-value for the test. These statistics are collected for all benchmarked models and presented as the result of the evaluation process.

4.4 Summary

The design and implementation of TSMiner, a software system which realises the methods developed in Chapter 3, is described in detail. TSMiner is comprised of two components: (i) a self-contained text analysis system called CiCui; and (ii) a workflow developed in the data analytic platform KNIME. The first component, CiCui, makes use of the natural language processing functions provided by the Stanford CoreNLP toolkit to preprocess (e.g. tokenisation, POS-tagging, dependency parsing, etc.) the raw running text in the news articles and transform them into structured format. The results are stored in a relational database to form an inverted positional index. The second component was built on top of KNIME, an open-source data analysis platform that offers visualised programming, modularised computation routines and integrated access to functionalities from heterogeneous computation environments such as R, Python and Java; TSMiner leverages this interoperability and was able to utilise several statistical functions that are available exclusively from R. The KNIME workflow that constitutes the second component connects to the index database created by the CiCui system and carries out the learning and the benchmarking of the models.

Some of the practical issues emerged from the learning and evaluation of the temporal sentiment model have been addressed in the later sections of the chapter. These issues include the treatment of missing stock prices in the dataset, numerical stabilities when calculating responsibility terms, smoothing of frequencies for lexical items that did not present in the test set, and the calculation of log-likelihood using log-sum-exp technique when testing convergence of the EM algorithm.

The predictive performance of the model is evaluated based on how much the temporal sentiment series outputted by the model correlates (linearly) with the future/past returns. A repeated random cross-validation schema was set up to reduce the uncertainty from single-fold validation: multiple training-testing cycles are conducted on repeatedly randomly split dataset and the Pearson's linear correlation coefficients calculated from each fold are aggregated into a single correlation coefficient using meta-analysis techniques.

In the next chapter, I present a case study conducted using the methods developed in Chapter 3 and its implementation described in this chapter.

Chapter 5

Case Study and Evaluation

5.1 Introduction

In this chapter, I present a case study through which the learning method and its implementation developed in Chapter 3 and Chapter 4 respectively are put to test on real world data. The case study aims to answer the following three questions:

1. Is the supervised EM algorithm developed in this thesis *effective*? i.e., can it learn models that are able to capture the correlations present in the data?

2. Can stock price movements be better explained by introducing the distinction between prospective and retrospective news sentiment?

3. Do word dependencies better reflect news sentiment than unigrams?

Four experiments were designed to attempt to answer these questions. In these experiments, the questions were first formulated into statistical hypotheses; the learning algorithm developed in Chapter 3 was then configured to produce the data required to test the hypotheses. The testing of the statistical hypotheses proposed in the experiments is based on the *correlational performances* achievable by the different configurations of the models over the dataset collected for this case study; the correlational performance of a model configuration is measured by how well the predictions the model makes on news sentiment can correlate with stock price movements for the test set in a cross-validation set-up.

The absolute effectiveness of the learning algorithm (i.e. 'Can the algorithm actually learn any association from the data at all?') is evaluated by comparing the correlational performances between two models: one is trained over the original dataset and the other over a *shuffled dataset*; this helps to determine whether the algorithm developed in this thesis is able to find correlations from the dataset. The relative effectivenesses of the different model configurations and linguistic features with respect to each other were also

evaluated by forming comparisons between the correlational performances achieved by the corresponding model configurations. Benchmarking models against baselines and each other was chosen over human evaluation because proper surveys that would produce valid results can be difficult to design given the resources available with the amount of data involved in the study; yet it is admitted that the lack of human benchmarking is a weak point to the evaluation of this thesis.

The rest of the chapter is organised as follows. Section 5.2 outlines the designs of the experiments conducted in the case study as well as the rationales behind their designs. Section 5.3 and 5.4 describes the news and market data used for this case study — how they were collected, and the pre-processing procedures that were applied to them before they were used in the experiments. Section 5.5 displays the results from the experiments and discusses their implications. Section 5.6 concludes the case study and summarises the findings.

5.2 Experiment Design

In this case study, four experiments were designed to investigate the questions set out earlier in this Chapter:

1. Experiment 1 sets out to demonstrate the effectiveness of the algorithm developed in this thesis by showing it is capable of establishing significant associations between language patterns seen in business news and firm-level stock prices movements. Since the goal of the experiment is to verify the algorithm's learning capability rather than the effects of temporality on the modelling of news sentiment, no sentiment temporality will be modelled in this experiment.

2. With the effectiveness of the algorithm established by Experiment 1, Experiment 2 will attempt to answer the second research question by examining the behaviours of the extended models where sentiment temporality is introduced.

3. To address the third research question, Experiment 3 tests whether word dependencies, when used as lexical features, can better capture news sentiment than unigrams when used with the method developed in this thesis.

4. Finally, experiment 4 evaluates the correlational performances from using words defined in two sentiment dictionaries as linguistic features: the *General Inquirer* affect dictionary, and the *Loughran and McDonald Sentiment Word Lists*. Their correlational performances will be compared to those achieved by the method developed in this thesis.

Exp. #	Method	Linguistic Feature	Temporality
1	Supervised EM	Word dependencies	Excluded
2	Supervised EM	Word dependencies	Included
3	Supervised EM	Single Words	Included
4	Content Analysis	Single Words	Included

Table 5.1: Summary of experiment designs

A summary of the configurations for the experiments are listed in Table 5.1.

The effectiveness of the EM learning algorithm proposed in Chapter 3 was evaluated by benchmarking its correlational performance with that achieved by a *baseline design*. More specifically, for a model learnt by the algorithm on the original *ordered dataset* in which the business news and the prices movements were temporally aligned, the *baseline design* for the model is created by training the same model on a *shuffled dataset*. The shuffled dataset is created by independently shuffling the date-time information attached to the stock returns as well as the publication dates associated with the news articles; the shuffling should destroy any correlation that had previously existed between the news sentiment and the price movements became lost. Comparing the correlational performance the method achieved with the ordered dataset with that achieved over the shuffled dataset enables the evaluation of the absolute effectiveness of the learning method — that is, it answers the question: 'if there exists correlations in the data, can the method find it'? The rationale behind this strategy is illustrated by the following grid:

Learning algorithm effective?	Relationship between news sentiment and market movements exist?	Correlational performance from models trained over ordered dataset significantly better than that from models trained over shuffled dataset?
Yes	Yes	Yes
Yes	No	No, due to correlations being destroyed by shuffling
No	Yes	No, due to ineffective learning algorithm
No	No	No, due to both destroyed correlation and ineffective algorithm

As a result, if one can demonstrate that the correlational performances obtained by models trained with the ordered dataset is significantly greater than that obtained with models trained with the shuffled dataset, it can then be argued that (i) there must exist certain correlations between news sentiment and movements of the market, and (ii) the learning algorithm used is effective in learning correlations between news sentiment and

market movements.

The correlational performances for each of the model configurations were measured and benchmarked using the meta-correlations obtained through the meta-analysis of the correlations between the models' predictions and the actual price movements during the validation phases of the cross-validation procedures as described in Section 4.3.4.

The main lexical units investigated in this case study are word dependencies and unigrams. Experiment 1 and 2 use word dependencies produced by the Stanford CoreNLP package as text features; Experiment 3 and 4 also evaluate models that use unigrams (i.e. single words) as the lexical unit for capturing sentiment and temporality.

The dependency grammar parser in the Stanford CoreNLP package recognises about 50 grammatical relations. Including all of them when preparing the vocabularies for the learning, however, would be inappropriate from both a theoretical and a practical point of view. Many of the dependency relations can be considered as *closed class relations* (analogous to *closed class words*). Examples include *determiner* (abbreviated as *det*), a type of dependencies that captures the relations between a determiner article (i.e. *a(n)*, *the*) and its corresponding head word, or *auxiliary* (*aux*), which connects non-main verbs (e.g. between 'have' and 'done' in sentences with perfect tenses). In practice, a vocabulary comprised of unfiltered word dependencies would often requires the algorithm to estimate tens of thousands of parameters, a number too large for the amount of data available to this case study; it is therefore decided that only word dependencies containing verbs are to be included in the feature vocabulary.

For this case study, the vocabulary was limited to the following five grammatical relations: *nsubj*, *nsubjpass*, *dobj*, *xcomp*, and *ccomp*. Their meanings are explained as follows:

nsubj represents the *nominal subject* relation. For example, in the sentence 'the stock of the company fell 5 percent', the word 'stock' is the nominal subject of 'fell'. Such a relation is noted in this section as a triplet, following the notation: '*dependent governor relation*' (e.g. 'stock fell nsubj').

nsubjpass refers to *passive nominal subject* relation, which is the counterpart of the *nsubj* relation for passive voices.

dobj is for the *direct object* relation. The dependent of this relation is usually a verb, and its governor typically a noun. The relation states that the governor word is the direct object of the dependent verb. For example, the pattern 'technology use dobj' can emerge from a scenario like '[the cellphone] use next-generation cellphone technology [. . .]'.

ccomp denotes *clausal complement*. The manual of the Stanford dependency parser states that: 'A clausal complement of a verb or adjective is a dependent clause with

an internal subject which functions like an object of the verb, or adjective'. The definition of the relation is rather complex. An example of the relation from the manual is 'I am certain that he did it', which gives the dependency 'certain did ccomp'.

xcomp denotes *open clausal complement*. An open clausal complement of a verb or an adjective is defined as 'a predicative or clausal complement without its own subject.'. An example of this relation is 'pay agree xcomp', which translates to 'agree to pay'.

Lemmatised base-forms of the words were used to represent both the *dependent* and *governor* in the relation triplet. Inflections were stripped from the words so that the tense information that would give hints about the temporality of the language patterns are removed. This is to ensure that the categorisation of the language patterns' temporality is derived only from the supervising returns rather than the tenses of the language patterns.

A list of stop words was used to further filter the word dependencies from entries that do not convey significant meaning. Word dependencies that have the following words as either its dependent or governor were excluded from the analysis: 'he', 'she', 'who', 'it', 'that', 'which', 'they', 'we', 'i', 'what', 'this', 'you', 'said', 'say', 'says', 'have', 'has', 'is', 'rights', 'reserved'. After this treatment, secondary patterns like 'he said nsubj', 'rights reserved nsubjpass' (as in 'all rights reserved'), etc. were excluded from the analysis. Additionally, a low-pass filter was applied to remove entries that are too rare from the vocabulary before further analysis; the low-pass filter removed any dependency that occurred less than 16 times in the corpus.

5.3 The News Corpus

The corpus used in the case study includes business news articles on 16 companies. Some basic information about the news corpus is summarised in Table 5.2. The documents were systematically retrieved from the ProQuest database using a custom CiCui plug-in. The search to the ProQuest database was set to cover articles published between 1st Jan, 2000 to 31st Dec, 2014. The articles were collected from a set of selected sources which represent some of the most established news agencies to ensure the quality of the text retrieved. A breakdown of the corpus by the sources from which the articles are retrieved is presented in Table 5.3. Once collected, the documents were preprocessed and transformed into an positional inverted index using the CiCui system, as detailed in Section 4.2.1.

The diachronic property of the corpus, namely, the news-flow, is summarised in Table 5.4; the annual number of articles published has seen a minor increase in the recent years, while the average length of the articles remained largely unchanged. The news-flows vary greatly among individual firms (Figure 5.1). It would seem companies

in manufacturing and IT sectors (e.g. Microsoft, Ford, Apple, etc.) tend to attract more media coverage than those operating in financial sectors (e.g. BRK-B, WFC).

The volume of news articles published varies depending on the day of the week (Table 5.5). On average, more articles were published during weekdays than on weekends. That being said, about 10% of the articles are published on weekends. Among the weekdays, the number of articles published increases with each day passing starting from Monday before dropping on Friday. The average length of the articles is slightly longer for those published on Saturdays, which could be due to the articles published on the weekend editions being written in a different style than the regular editions[1].

5.4 The Equity Data

The EM algorithm for learning news sentiment developed in this thesis requires that target returns be provided to enforce the supervision during the training phase. In a contemporaneous setting where the temporal aspects are omitted (as in Experiment 1), the algorithm makes use of one set of returns to supervise the learning of sentiment. Where the temporality in sentiment is included in the model, two sets of supervising returns are needed: a future return series that supervises the learning of prospective sentiment, and a past return series that supervises the learning of retrospective sentiment.

For this case study, interday logarithmic returns (sometimes also called 'total return') of the companies' stocks were used to derive the supervising returns series. Such returns were calculated from linearly interpolated adjusted close prices. Interpolated adjusted close prices were used to derive the returns so that non-trading days were also covered with return data — as having been shown in Section 5.3, a considerable amount of news was published on weekends (and potentially other non-trading days); the textual data for these days would otherwise be wasted if raw returns were used. The preparation of the interday return series as well as the subsequent calculation of future and past returns followed the discussion in Section 3.4.3. The logarithmic returns were calculated as log-differences between consecutive stock prices:

$$r_t = \log(p_t) - \log(p_{t-1}) \tag{5.1}$$

where r_t is the stock return at time t and p_t is the stock price at time t.

The market data collected for this study spans from 1st January, 2000 to 31st December, 2014, which coincides with the dates covered by the corpus. Table 5.6 presents some descriptive statistics for the past and future returns calculated from the interday returns

[1]The Financial Times and The Wall Street Journal each has a weekend edition, which are published on Saturdays.

Company	Ticker	Exchange	Query	Sector of Operation	Token Count	Document Count
Microsoft	MSFT	NASDAQ	(TI(Microsoft))	Service	3750621	7185
Ford Motor	F	NYSE	(TI(Ford))	Manufacturing	2417832	4825
The Boeing Company	BA	NYSE	(TI(Boeing))	Manufacturing	1999682	4048
General Electric	GE	NYSE	(TI(GE) OR TI("General Electric"))	Manufacturing	1790566	3796
JPMorgan Chase & Co.	JPM	NYSE	(TI(JPMorgan) OR TI("J.P. Morgan"))	Finance	1606152	3284
Apple	AAPL	NASDAQ	(TI(Apple) AND ORG(Apple))	Service	1556253	3056
Wal-Mart	WMT	NYSE	(TI(Wal-Mart))	Service	1605889	3005
Intel	INTC	NASDAQ	(TI(Intel))	Manufacturing	1285736	2841
IBM	IBM	NYSE	(TI(IBM))	Service	1037185	2330
Verizon	VZ	NYSE	(TI(Verizon))	Service	843249	1847
Hewlett-Packard	HPQ	NYSE	(TI(HP) OR TI("Hewlett-Packard"))	Manufacturing	715938	1749
Exxon	XOM	NYSE	(TI(Exxon) OR TI(Mobil) OR TI(ExxonMobil))	Energy	751685	1645
Procter & Gamble Co.	PG	NYSE	(TI("Procter & Gamble") OR TI(P&G))	Manufacturing	719206	1463
Shell	RDSB.L	London	(TI(Shell) AND ORG(Shell))	Energy	625625	1383
Berkshire Hathaway	BRK-B	NYSE	(TI(Berkshire))	Finance	409649	821
Wells Fargo	WFC	NYSE	(TI("Wells Fargo"))	Finance	284332	679
					21399600	43957

Table 5.2: Summary of the composition of the corpus used in the case study

This table lists the key parameters used when assembling the corpus for this case study. The 'Query' column displays the queries issued to the ProQuest database for retrieving the news articles for the companies. The entries in this table are sorted in descending order by the 'Document Count' column. Two points to note: (i) There are two stocks for Berkshire Hathaway: an Class A share (BRB-A) and a Class B share (BRB-B). The BRB-A stock is priced at about $197660 per share at the time of writing; the higher price of the stock attracts investors who focuses more on long-term returns. The BRB-B stocks is priced at a much lower rate and is offered as a more flexible instrument for small investors. BRB-B is chosen for this case study because it is supposedly traded more frequently than its A-class counterpart due to the lower share price. (ii) For companies whose name could be ambiguous (e.g. Shell and Apple), an additional criterion is imposed using the ORG keyword (which stands for 'ORGANISATION') to help disambiguating the semantics of the names — the inclusion of such constraints ensures only the articles whose title refers the queried term as a certain organisation name are returned by the search. Using the ORG keyword alone, however, would not enforce as strong a constraint as that imposed by the TI keyword for an article can qualify for an ORG query when it only trivially mention the said company in the text; so it is necessary to use the TI constraint.

Source	Document Count	Word Count
Wall Street Journal, Eastern edition	15419	8725163
Financial Times	12789	5094826
FTcom	7739	3475880
Wall Street Journal Online	7001	3798937
Business Week	616	7
The Economist	281	248853
The Economist Online	112	55934

Table 5.3: Corpus breakdown by source

The abnormally low word count for *Business Week* was due to the fact that ProQuest registers only the URL to the original article on *Business Week*'s website, not the actual full text. This issue should not affect any subsequent analyses though as they do not depend on this information on sources in any way. The number of articles published on traditional paper media has been decreasing over recent years; the online versions of the publication titles were included to compensate for this trend.

Year	Document Count	Word Count	Average Length
2000	2596	1203010	463.41
2001	2579	1140137	442.08
2002	2864	1233839	430.81
2003	2389	1193901	499.75
2004	2704	1345511	497.60
2005	3247	1547145	476.48
2006	3312	1584828	478.51
2007	2569	1282985	499.41
2008	2817	1368715	485.88
2009	2278	1091570	479.18
2010	3368	1614979	479.51
2011	3430	1693585	493.76
2012	3180	1632756	513.45
2013	3512	1879716	535.23
2014	3112	1586923	509.94

Table 5.4: Corpus breakdown by year

Weekday	Document Count	Word Count	Average Length
Monday	6065	3005201	495.50
Tuesday	8018	3824034	476.93
Wednesday	8921	4315949	483.80
Thursday	8676	4212548	485.54
Friday	7861	3820003	485.94
Saturday	3085	1656462	536.94
Sunday	1331	565403	424.80

Table 5.5: Corpus breakdown by day of week

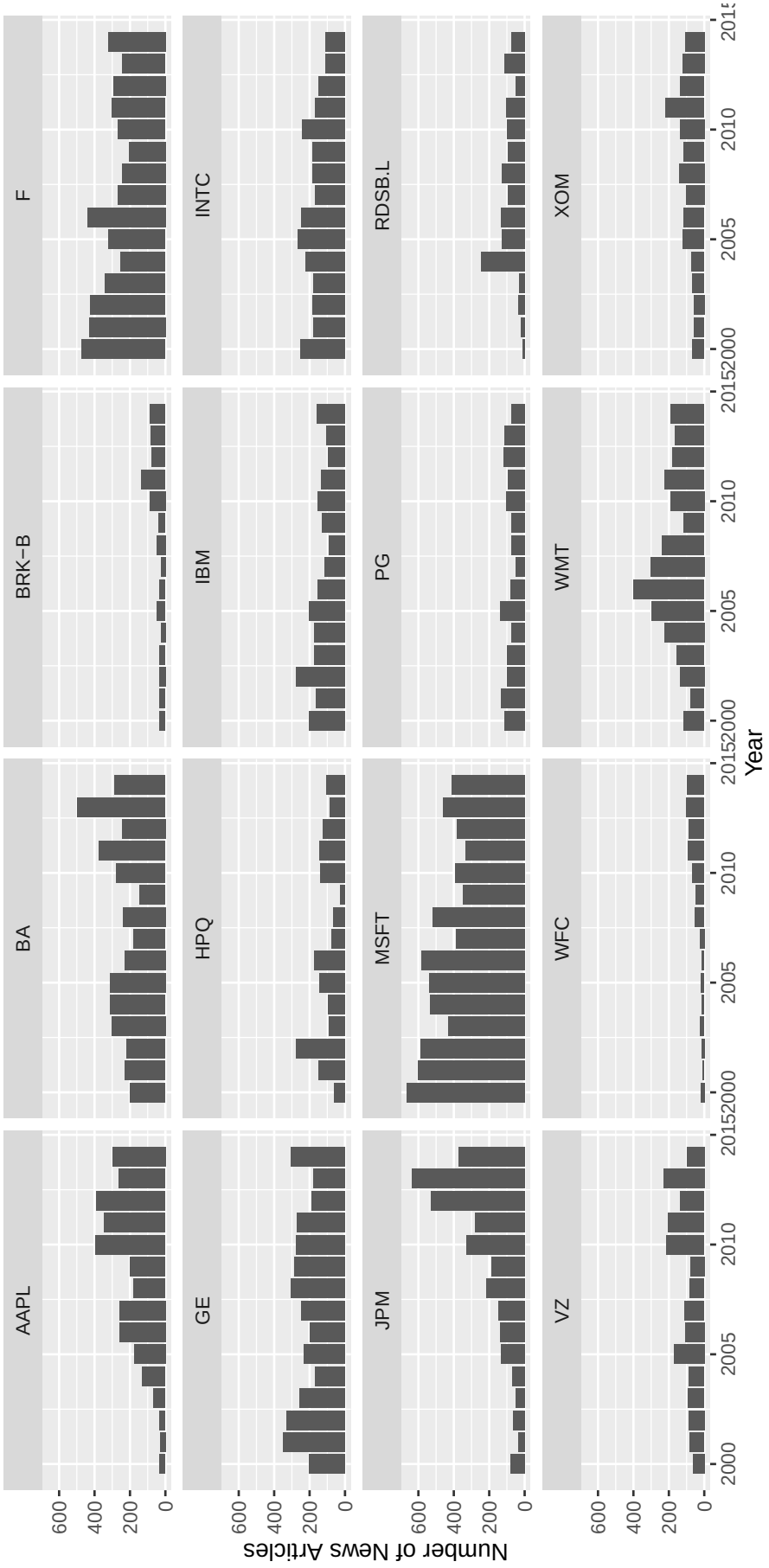


Figure 5.1: News-flow by company

for each of the companies covered in the corpus. The difference between the distributional properties of these two types of returns are minimal due to the relative small windows used by the smoothing procedure. The average returns over this time period for all companies are close to 0, with the highest found for Hewlett-Packard (about 0.001 for both future and past returns) and the lowest found for the Ford Motor Company (at around -0.002 for both types of returns). The standard deviation of the returns of most companies fluctuates between 0.01 and 0.02, with the exception of Ford, whose returns yield a standard deviation of about 0.023. The standard deviation of historical returns are sometimes used as a measure of volatility. The *skewness* statistic measures the asymmetry of a probability distribution: a negative skewness indicates that the left tail of the distribution is longer while a positive skewness indicates the right tail of the distribution is longer. The *kurtosis* of the series measures the 'peakness' of its distribution; distributions with large kurtosis will have a more pointy peak. The past and future returns for Berkshire Hathaway and Royal Dutch Shell exhibit exceptional skewness and kurtosis. Excess skewness and kurtosis are often attributed to extreme outlier [Taylor, 2007, pg. 69].

5.5 Results and Discussion

5.5.1 Experiment 1: Non-temporal Sentiment and Stock Returns

Before presenting the results from the temporality-enabled models, it can be helpful to study the behaviours of non-temporal sentiment classification models trained using the methods described in Section 3.4.2. By presenting the analysis of such models, it is hoped that the reader is convinced that the supervised EM algorithm is indeed able to identify associations between the use of language in business news and the movement of firm-wise stock prices to some extent. The discussion in this section begins with an investigation into the correlational performances of the models learnt by the algorithm, which is followed by an examination of the sentiment connotations of the language patterns extracted through the learning procedure. Observations and insights that would become useful for subsequent analyses will be noted during the process.

For the purpose of this experiment, no smoothing was applied to the returns, so the model is trained on interpolated interday returns. The weighting of the returns during the training phase is so configured that returns' contribution to the learning is equal to that provided by the textual features.

	Past Return							Future Return						
Ticker	n	$\bar{r}$	s	min	max	$skew.$	$kurt.$	n	$\bar{r}$	s	min	max	$skew.$	$kurt.$
AAPL	5475	0.00062	0.018	-0.53	0.12	-5.02	141.78	5477	0.00060	0.018	-0.55	0.11	-5.54	169.09
BRK-B	5475	0.00027	0.009	-0.09	0.13	1.04	18.97	5477	0.00026	0.009	-0.10	0.13	1.14	20.73
BA	5475	0.00027	0.012	-0.08	0.10	-0.10	5.54	5477	0.00027	0.012	-0.07	0.11	-0.06	5.50
XOM	5475	0.00022	0.010	-0.11	0.09	-0.64	14.53	5477	0.00022	0.010	-0.11	0.09	-0.19	13.51
F	5475	-0.00005	0.018	-0.22	0.21	-0.34	22.52	5477	-0.00006	0.018	-0.20	0.22	-0.31	20.85
GE	5475	-0.00004	0.012	-0.12	0.14	-0.10	11.80	5477	-0.00005	0.012	-0.11	0.12	0.04	11.11
IBM	5475	0.00009	0.010	-0.12	0.10	-0.10	12.92	5477	0.00009	0.010	-0.12	0.09	-0.02	12.52
INTC	5475	0.00002	0.015	-0.19	0.14	-0.68	13.95	5477	0.00001	0.015	-0.19	0.15	-0.63	14.11
JPM	5475	0.00013	0.016	-0.18	0.15	0.43	16.19	5477	0.00012	0.016	-0.15	0.17	0.71	16.29
MSFT	5475	0.00002	0.012	-0.10	0.14	0.11	12.01	5477	0.00001	0.012	-0.10	0.15	0.13	12.13
PG	5475	0.00017	0.009	-0.27	0.07	-5.90	165.07	5477	0.00016	0.009	-0.28	0.06	-6.16	186.27
RDSB.L	5342	0.00009	0.011	-0.10	0.10	-0.32	12.09	5344	0.00008	0.011	-0.08	0.12	0.32	13.40
VZ	5475	0.00010	0.010	-0.09	0.11	0.15	9.98	5477	0.00009	0.010	-0.10	0.09	0.11	9.29
WMT	5475	0.00009	0.010	-0.09	0.08	-0.14	10.73	5477	0.00008	0.010	-0.09	0.08	-0.13	10.53
WFC	5475	0.00026	0.015	-0.16	0.21	1.42	32.50	5477	0.00025	0.015	-0.14	0.22	1.39	30.32
HPQ	5475	0.00001	0.015	-0.18	0.12	-0.29	12.18	5477	0.00000	0.015	-0.16	0.11	-0.28	10.81

Table 5.6: Descriptive statistics for past and future returns series

n, $\bar{r}$, s, min, max , $skew.$, $kurt.$ are the sample size, mean, standard deviation, minimum, maximum, skewness and kurtosis of the corresponding return series. All statistics are computed using the `psych` package in R. The skewness and kurtosis are calculated as skewness $= \left(\frac{n-1}{n}\right)^{3/2} \frac{m_3}{m_2^{3/2}}$ and kurtosis $= \left(\frac{n-1}{n}\right)^{3/2} \frac{m_4}{m_2^2}$ respectively, where m_2, m_3, and m_4 are the second, third, and forth moment of the return series.

5.5.1.1 Correlational Performances

I shall first look at the correlational performances of the models to gain overall understanding of the behaviour of the learning algorithm. The box plots shown in Figure 5.2 summarise the correlational performances from an experiment where the learning algorithm was tasked to find associations between language usage seen in company business news on day $t = 0$ (which I will refer to as 'today' or 'current day' for the ease of interpretation) and the interday logarithmic returns of their respective stocks, lagged at various days ranging from -5 to 5. The two plots are displayed side-by-side to facilitate easy visual comparison; columns in the sub-plot summarises the results obtained using the ordered and shuffled dataset respectively. Negative lags indicate the returns in the future are aligned with current day's language patterns; similarly, with positive lags, the past returns are aligned with today's language patterns; when the lag is zero, the model attempts to find contemporaneous associations between the language patterns and the interday returns on day t. At each lag, the meta-analysis procedure is applied to each of the 16 companies; the meta-correlations between the sentiment series and the market returns produced from the meta-analysis are summarised and plotted as one box component in the box plot.

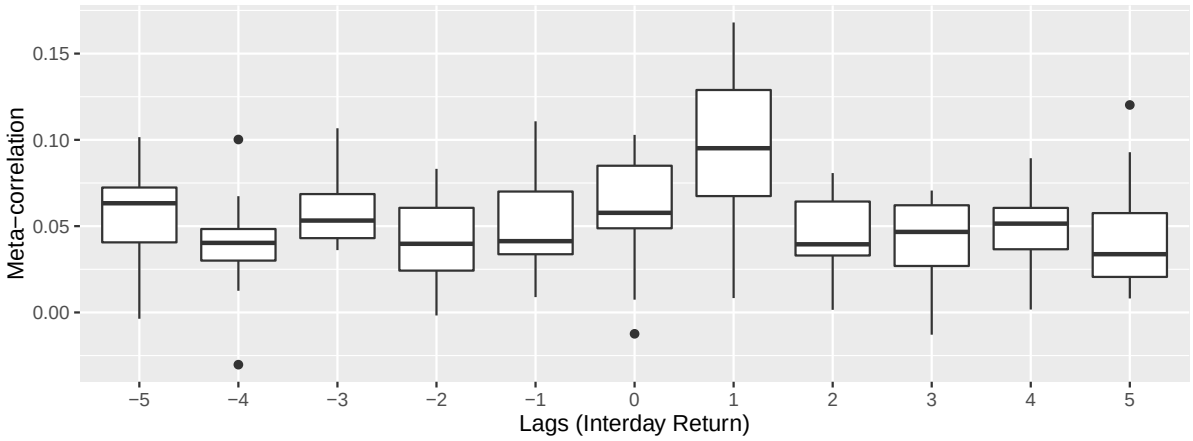
The results from the experiment using the ordered dataset is illustrated by the box plot in Figure 5.2a, while the benchmark results based on the shuffled dataset are summarised by Figure 5.2b. The first thing to note is that the models were able to consistently achieve positive correlations when predicting returns in both default and randomised settings. Among the meta-correlations for the ordered dataset, a noticeable peak can be identified at lag 1. Further tests showed that for certain lags, applying the method to the ordered dataset had resulted in better correlation performances than when it was applied to the shuffled dataset; the differences in means are greatest at 0.05 level of significance for lags -5, -3, 0, and 1; weaker differences are also observed for lag -2 and 2 (Table 5.7).

Lag	-5	-4	-3	-2	-1	0	1	2	3	4	5
t-stat.	3.07	-0.85	3.05	1.01	1.53	2.09	3.48	1.54	-0.37	0.91	0.41
p-value	0.00	0.80	0.00	0.16	0.07	0.02	0.00	0.07	0.64	0.18	0.34

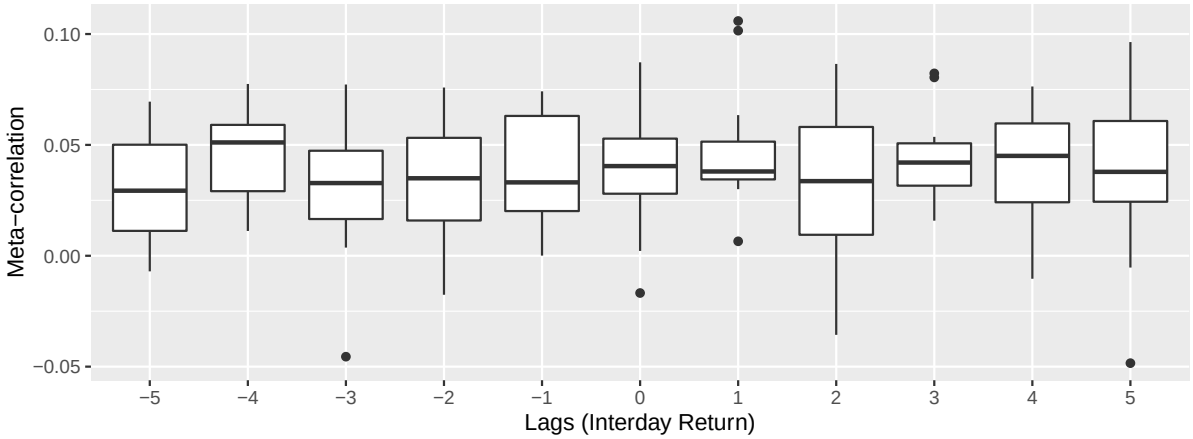
Table 5.7: Testing for significant differences between the means of meta-correlations achieved by the supervised EM algorithm on the ordered and the shuffled dataset

The table summarises the results from the one-side t-tests that were designed to test whether the EM algorithm developed in Chapter 3 is able to achieve better correlational performances on the ordered dataset than on the shuffled dataset. One test is performed for each of the 11 lags shown in 5.2. In each test, the null hypothesis is that the mean of the meta-correlations achieved by the EM algorithm with the ordered dataset is no greater than that achieved with the shuffled dataset. the 'Lag' row indicates the number of days the return series was lagged for each test; The 't-stat' and 'p-value' rows display the t-statistic and p-value for the tests respectively.

It is important to note that the positive correlation found by the algorithm between the



(a) Meta-correlations produced from the ordered dataset



(b) Meta-correlations produced from the randomised setting

Figure 5.2: Box plots summarising the correlational performances of the two non-temporal sentiment models trained with the default and randomised settings respectively.

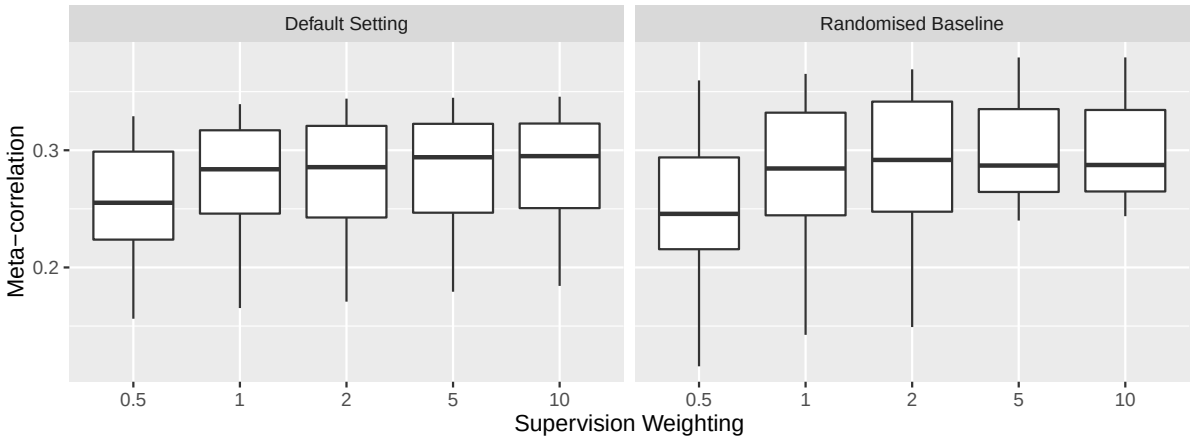
Box plots such as these two are used across the case study to summarise the correlational performances of the models. Each box in the plot summarises a group of 16 meta-correlations, one for each of the 16 companies covered by the corpus. The upper and lower whisks attached to each box indicates the maximal and minimal meta-correlations for that group respectively. The body of the box indicates the second and third quantiles of the meta-correlation distribution. The horizontal line in the middle indicates the median of the meta-correlations. The ticks on the x-axis indicates how many days the interpolated interday return series was lagged to form the future return series, which is in turn used to train the model.

language patterns and the price movements does *not* necessarily entail that such relation truly exists — the algorithm can be very 'aggressive' when set to find associations when none existed. A typical vocabulary in the experiments can contain hundreds of patterns — the weights of which but one can all vary freely; with this many degrees of freedom, the algorithm will almost surely be able to find a distribution of lexical items that leads to positive correlational performances; this theory is supported by the fact that the average meta-correlations achievable by the algorithm on the shuffled dataset are statistically different from zero for all lags.

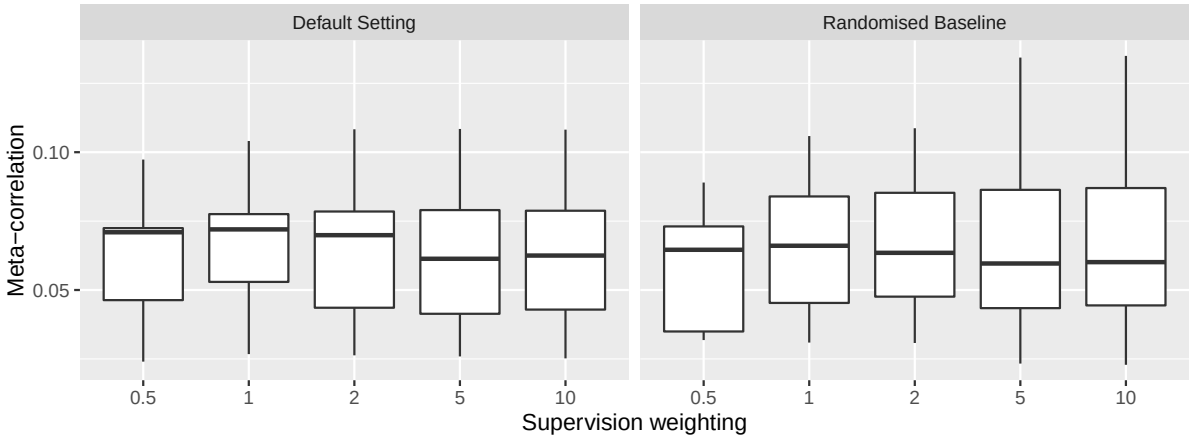
The presence of background correlation is likely a symptom of over-fitting, which is an unfortunate weakness of maximum likelihood estimation. One factor that affects the magnitude of this 'background' correlation is the weighting of the return series that is used to supervise the learning process. Recall that in Section 3.4.2, it was noted that the returns in the model can be thought of as 'pseudo-terms', and the strength of supervision from the returns can be affected by adjusting weighting of such pseudo-terms. During the learning process, the distribution of the lexicon items will evolve in such a way that positive language patterns are more likely to co-occur with positive pseudo-terms (i.e. positive returns) while negative language patterns are more aligned with negative pseudo-terms (i.e. negative returns). As a result, the greater the weightings assigned to the returns (i.e. pseudo-terms), the higher the background correlation the learning algorithm will be able to achieve (Figure 5.3). It should become clear that model evaluation based solely on association measures can be inadequate because of the aggressiveness of maximum likelihood estimators — algorithm is effective in learning *some* associations from the data, but it is difficult to tell whether such associations are attributable to the interactions between news sentiment and the market. It is therefore important to examine the actual distributions of the language patterns learnt for each sentiment classes by the algorithm; in order to gain a better understanding of the behaviour.

With the above observation in mind, I now attempt to interpret the outstanding positive correlations around lag 1 and 0. Intuitively, the peak indicates that the correlation between the interday returns of companies from 'yesterday' and the language patterns seen 'today' is, on average, statistically stronger than the 'background' correlations achievable when the learning algorithm was applied to the shuffled dataset. Such excess suggests that the learning algorithm was able to find a stronger-than-average association between stock price movements and language usages at this particular configuration; in other words, it must be the case that there truly exists some correlation between the two quantities which cannot be explained by the aggressiveness of the maximum likelihood methods for the learning algorithm to pick up.

The peak in the correlations provides some empirical evidence to the existence of a link between the textual news sentiment and stock price movements in the dataset; it is worth noting that the timing also provides important information: the fact that the peak is situated at lag 1 suggests it is the market movements from 'yesterday' that have influenced current day's change in news sentiment — that is, the sentiment featured in the news is mostly retrospective. In comparison, no abnormal meta-correlation is immediately recognisable in the baseline results generated from the shuffled dataset.



(a) Meta-correlations from the training phase



(b) Meta-correlations from the testing phase

Figure 5.3: The impact of supervision weighting on the performances of learnt model

These plots compare the meta-correlations obtained when models learnt with varying supervision weighting were tasked to predict the returns in the training set and test set respectively. Each sub-plot is divided into two columns; the left sub-plot summarises the meta-correlations produced using the ordered dataset while the sub-plot to the right is for results under the randomised setting. Figure 5.3a shows that when the learnt sentiment was used to predict lag 1 interday returns in the training set, the average correlation achievable by the algorithm increases as the supervision weighting increases for both the default and the randomised setting. The fact that the increase is seen in both the default and randomised settings implies that it must be the background correlation that is responding to the change. The marginal increase in the average background correlation declines as the magnitude of weighting increases, which may indicate that weightings greater than 1.0 is actually excessive and the model over-fitted the data; this is confirmed by what is shown in Figure 5.3b, where the meta-correlations when predicting returns increases when the weighting moves from 0.5 to 1, but decreases as the weighting becomes larger.

5.5.1.2 Distribution of Language Patterns

As stated in the previous section, the correlational performance of the model alone can be unreliable when determining the effectiveness of the learning algorithm due to its aggressiveness in finding associations even when little or none existed; it is therefore necessary to examine the distributions of the language patterns extracted for the sentiment

classes.

In this experiment where the temporal aspects of the sentiment are omitted, each fold of the cross-validation procedure would output two language pattern distributions from its training phase: one for the positive sentiment class and the other for the negative sentiment classes. Each cross-validation procedure contains 35 folds, and the procedure is repeated for each of the 16 companies surveyed in this experiment. It is impractical to examine and compare over all 1120 distributions; instead these distributions were aggregated into two 'meta-distributions', one for each of the two sentiment classes. This aggregation is a two-step process:

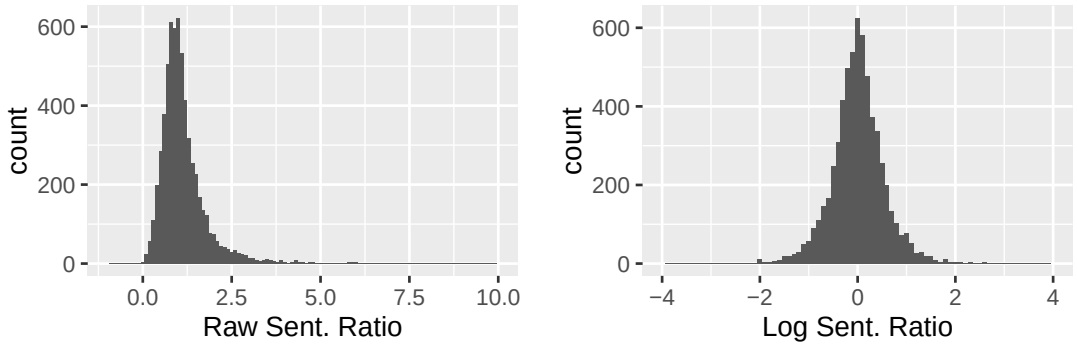
1. The first step is to obtain two aggregated distributions (one for positive and negative classes each) for each company, which involves merging the 35 distributions produced by the cross-validation procedure for each sentiment class. For this matter, the aggregated distributions for both the positive and negative vocabularies were constructed by averaging over the 35 sets of distributions collected during the folds of the cross-validation process. Where a language pattern is absent in a training set, its probability in the vocabulary for that fold is treated as 0. Since the probabilities associated with the entries in each distribution by definition sum to 1, the averages of the probabilities distributions over the same set of entries will sum to 1 as well.

2. The second step is to further aggregated the positive and negative distributions from each company into two summarising distributions. The aggregation method differs slightly from that used to aggregate the distributions within each company: for every entry that is indexed in all the corpora, instead of simply taking the average of the probabilities gathered for all the companies, an weighted mean was calculated from the probabilities, with the frequencies of the entry in the firm-wise distributions as weights. The weighted average approach is adopted to alleviate the issue where the frequencies of certain patterns occurring in one or two corpora are quite low, but for those occurrences the patterns would yield relatively extreme probabilities compared to their presences in the other corpora. Weighting vocabulary entries by their frequencies prevents such outlier occurrences from distorting the resulting distributions.

To facilitate the comparison of the sentiment polarities between the language patterns, a *sentiment score* was computed for each pattern by taking the log difference between its aggregated positive probability and its aggregated negative probability, namely:

$$\text{sent}_{e_j} = \log p\left(e_j \mid \text{pos}\right) - \log p\left(e_j \mid \text{neg}\right). \tag{5.2}$$

The measure is essentially the logarithm of the ratio between an entry's probability in the distribution for positive sentiment and its probability in the negative sentiment distribution.



(a) Raw sentiment ratios (b) Logarithmic sentiment ratios

Figure 5.4: Histograms depicting the distributions for raw and logarithmic sentiment ratios

The metric is positive for a language pattern that is relatively more likely to appear in positive context than in negative context, and is negative when the reverse is true. The distribution of the raw ratio itself is highly skewed, with about half of the entries falling between 0 and 1 and the other half falling between 1 and ∞ (Figure 5.4a); applying logarithm to the raw ratios balances the distribution of the metric into a symmetric shape (Figure 5.4b).

Table 5.10 and 5.11 display the 20 most frequent salient positive and negative language patterns mined by the algorithm from the ordered dataset using two different sets of parameters respectively; the former shows the lexical items extracted when lag-1 interday returns where used to supervise the learning while the latter depicts the lexical items extracted using lag--1 returns. To evaluate how well the language pattern distributions extracted by the algorithm capture the connotations of the patterns, I manually examined the lists in the two tables and marked each entry into three categories:

- An entry is marked with a ✓ if I could identify a reason as to why the language pattern could be said to bear the sentiment or temporality score assigned to it by the algorithm.

- An entry is marked with a × if I considered the sentiment or temporality score assigned to the pattern by the algorithm is wrong — that is, I could identify a reason as to why the language pattern could be said to bear a sentiment or temporality score that is *opposite to* the one assigned by the algorithm.

- An entry is marked with a ? if the sentiment connotation of the language pattern cannot be sufficiently determined without further context, and thus the algorithm's scoring of the pattern can be neither validated nor invalidated.

The manual evaluations for Table 5.10 and Table 5.11 are tallied in Table 5.8. The tallies suggest that:

	Lag 1 (Table 5.10)		Lag −1 (Table 5.11)	
Eval.	Negative	Positive	Negative	Positive
✓	8	15	7	7
?	9	5	11	13
×	3	0	2	0

Table 5.8: The tallied results for the manual evaluation of the word dependencies displayed in Table 5.10 and Table 5.11

1. The sentiment orientations of the word dependencies were classified correctly more often than incorrectly.

2. The algorithm identified positive word dependencies more reliably than negative word dependencies as it made fewer mistakes assigning scores to positive word dependencies.

3. The algorithm identified retrospective sentiment more reliably than prospective sentiment as it produced fewer 'uncertain' (?) entries for Table 5.11 than for Table 5.10.

It appears that the sentiment polarities inferred for most of the word dependencies shown in Table 5.10 align with common perceptions about their sentiment connotations. In particular, word dependencies such as 'share/stock/index/average/sale fall/drop' dominated negative vocabularies, while 'stock/index/share rise/jump/gain' along with 'sign deal', 'expect to rise', 'rise/add *some* per cent' are all found in the positive vocabulary, which largely conforms with the general expectation on their sentiment connotations in business contexts. It also appears that many of these expressions can be construed to encode retrospective sentiment even though their tense had been erased during lemmatisation.

As a comparison, Table 5.11 displays the most frequent word dependencies that are sentiment-salient extracted using the same experiment setting as that used for 5.10, but the supervising returns were constructed by lagging the original return series by -1; in other words, the algorithm attempted to associate the current day's language usage with the next day's price movements. The first thing to note in this list is perhaps the absence of the retrospective expressions that were once captured in Table 5.10 (e.g. phrases containing 'rise' and 'fall'). While some of the patterns may be associable to their corresponding sentiment classes, such associations for many of the patterns may not be immediately obvious to human examiners.

Table 5.12a shows the 20 most frequent language patterns whose sentiment classes were classified differently (i.e. the signs of the two sentiment scores contradict) by models trained with the original dataset and by models trained using the shuffled dataset respectively. Similar to what I did before, I examined the entries in the table and marked into three categories:

Eval.	Count
og.	12
?	6
sf.	2

(a) Ordered dataset versus shuffled dataset

Eval.	Count
P.	10
?	7
F.	3

(b) Past return versus future return

Table 5.9: The tallied results for the manual evaluation of the word dependencies displayed in Table 5.12a and Table 5.12b

- An entry is marked with *og.* if I consider the classification given to it by the model trained over the original dataset is correct.
- An entry is marked with *sf.* if I consider the classification given to it by the model trained over the shuffled dataset is correct.
- An entry is marked with a $\times$ if the sentiment connotation of the language pattern cannot be sufficiently determined without further context, and thus I could not determine whether a model's classification is correct or incorrect.

Table 5.12b shows the 20 most frequent language patterns whose sentiment classes were classified differently by models trained with lag-1 past return as the supervising series and by models trained with lagged--1 future return as the supervising series. Again, I marked each of the entries in the table into three categories:

- An entry is marked with *P.* if I consider the classification given to it by the model supervised by the past returns is correct.
- An entry is marked with *F.* if I consider the classification given to it by the model supervised by the future returns is correct.
- An entry is marked with a $\times$ if the sentiment connotation of the language pattern cannot be sufficiently determined without further context, and thus I could not determine whether a model's classification is correct or incorrect.

The results from the above two evaluations are tallied in Table 5.9. Table 5.9a suggests that when sentiment class assignments differ, the classifications made by the models trained over the original dataset are more likely to be correct than that made by those produced by the models trained with the shuffled dataset, providing evidences that the learning algorithm performs better than uninformed guessing. Table 5.9b suggests that when a language pattern is assigned different sentiment orientations by the model trained to find retrospective sentiment and the model trained to identify prospective sentiment, it is more likely that the pattern is in fact retrospective. This may indicate that prospective

patterns are in general much rarer than retrospective patterns (i.e. the *prior probability* of prospective sentiment class is lower than that of retrospective sentiment class). It is also observed from Table 5.9b that the magnitudes of the sentiment scores assigned to the salient retrospective expressions are generally greater than that of assigned to their prospective counterparts; this may imply that prospective patterns are generally harder to identify.

5.5.1.3 Summary

The principal goal of the discussion in this subsection is to show (i) there exists certain causal links between the language usages in business news about companies and the returns of their stocks on the market, and (ii) the supervised EM algorithm developed in this thesis is effective. The assumption behind the design of this experiment is that if a model can be trained to use business news to predict stock returns better than random guesses, then it must be because some association between news and returns truly exists, and the model can be considered to have captured the sentiment polarities or orientations of the language patterns of the news in terms of their effects on the movements of stock prices.

In practice, it was demonstrated that the supervised EM learning algorithm behaves aggressively when tasked to find associations — that is, the algorithm will find some association even when none actually exist in the data. As a result, when fed with shuffled datasets where the dates are shuffled (thus any causal link between news and returns presumably destroyed), the learning algorithm is still able to consistently find distributions of language patterns that significantly better predict returns than guessing randomly. Consequently, one must be careful when interpreting the correlations or other metrics of association produced by the models. Assertions on the existence of an association should only be made when the model trained is able to perform better than the 'background performances', which can be achieved solely through the aggressiveness of maximum-likelihood estimators.

With the above observation in mind, subsequent analyses revealed that for the dataset collected for this case study, lexical distributions obtained by models supervised with the stock returns from the previous day significantly outperformed the baseline in predicting the stock returns for the current day. Further examination of the distributions of vocabularies for each sentiment classes learnt reaffirmed the link. It may be argued that the findings provide additional evidences for the Efficient Market Hypothesis, which states that all publicly available information about the market are already reflected in the prices; however, the other results from the experiment seem to suggest that certain language patterns correlate significantly with price movements in the future.

dep	gov	rel	Freq.	Sent.	Eval.
point	fall	dobj	353.86	-0.97	✓
index	fall	nsubj	341.20	-0.77	✓
share	drop	nsubj	245.06	-0.76	✓
share	fall	nsubj	2154.40	-0.58	✓
share	sell	dobj	389.46	-0.55	✓
people	familiar	nsubj	230.14	-0.50	?
stock	fall	nsubj	556.23	-0.50	✓
challenge	be	nsubj	215.49	-0.48	✓
company	seek	nsubj	211.09	-0.46	?
product	develop	dobj	174.00	-0.45	?
income	report	dobj	390.60	-0.44	×
software	develop	dobj	175.69	-0.42	?
revenue	fall	nsubj	659.20	-0.36	✓
business	run	dobj	321.43	-0.35	?
be	expect	xcomp	185.86	-0.35	?
money	make	dobj	719.23	-0.35	×
revenue	expect	dobj	263.11	-0.33	?
plan	be	nsubj	185.26	-0.33	?
business	buy	dobj	233.80	-0.33	?
income	rise	nsubj	354.83	-0.30	×

(a) Most frequent salient negative word dependencies

dep	gov	rel	Freq.	Sent.	Eval.
point	rise	dobj	392.77	1.15	✓
index	rise	nsubj	386.83	1.03	✓
average	rise	nsubj	251.43	1.02	✓
term	not_disclose	nsubjpass	244.63	0.83	?
cent	advance	dobj	261.54	0.76	✓
cent	rise	dobj	1800.00	0.69	✓
company	raise	nsubj	270.40	0.67	✓
stock	rise	nsubj	480.03	0.61	✓
forecast	raise	dobj	330.63	0.61	✓
cent	gain	dobj	798.91	0.57	✓
cent	add	dobj	364.23	0.56	✓
share	rise	nsubj	1889.14	0.55	✓
company	look	nsubj	229.06	0.38	?
share	jump	nsubj	267.43	0.37	✓
maker	report	nsubj	254.29	0.36	?
report	expect	xcomp	314.17	0.34	?
revenue	grow	nsubj	315.71	0.33	✓
expectation	beat	dobj	485.91	0.32	✓
deal	sign	dobj	322.20	0.30	✓
share	buy	dobj	636.60	0.30	?

(b) Most frequent salient positive word dependencies

Table 5.10: 20 most frequent sentiment-salient word dependencies extracted from the ordered dataset trained with lag 1 interday returns

For this table, a word dependency is considered sentiment-salient if its absolute sentiment score is greater than 0.3.

dep	gov	rel	Freq.	Sent.	Eval.
dividend	pay	dobj	439.71	-0.38	✓
time	take	dobj	401.46	-0.31	?
product	sell	dobj	369.94	-0.34	?
point	fall	dobj	353.86	-0.31	✓
index	fall	nsubj	341.20	-0.54	✓
stock	buy	dobj	319.89	-0.93	?
revenue	expect	dobj	263.11	-0.44	?
boeing	be	nsubj	187.71	-0.35	?
do	need	xcomp	176.43	-0.32	?
product	develop	dobj	174.00	-0.45	?
message	send	dobj	160.86	-0.38	?
evidence	be	nsubj	158.23	-0.32	?
price	pay	dobj	155.43	-0.35	✓
revenue	decline	nsubj	151.63	-0.67	✓
patent	infringe	dobj	146.40	-0.43	✓
opportunity	be	nsubj	143.89	-0.32	×
expectation	miss	dobj	138.09	-0.34	✓
news	come	nsubj	137.60	-0.37	?
expectation	meet	dobj	135.29	-0.58	×
result	be	nsubj	134.74	-0.41	?

(a) Most frequent salient negative word dependencies

dep	gov	rel	Freq.	Sent.	Eval.
sale	grow	nsubj	403.66	0.39	✓
share	sell	dobj	389.46	0.51	?
stake	take	dobj	333.94	0.47	?
share	trade	nsubj	321.80	0.30	?
price	raise	dobj	308.09	0.34	✓
executive	become	xcomp	285.51	0.32	?
estimate	beat	dobj	272.17	0.84	✓
plant	build	dobj	272.09	0.43	✓
cent	advance	dobj	261.54	0.39	✓
version	offer	dobj	248.69	0.39	?
lawsuit	file	dobj	231.49	0.51	?
people	familiar	nsubj	230.14	0.35	?
people	employ	dobj	205.80	0.34	?
close	expect	xcomp	201.74	0.37	?
job	create	dobj	192.11	0.46	✓
company	spend	nsubj	186.40	0.57	?
fee	pay	dobj	167.60	0.33	?
production	boost	dobj	158.40	0.43	✓
use	plan	xcomp	157.37	0.36	?
share	own	dobj	156.11	0.45	?

(b) Most frequent salient positive word dependencies

Table 5.11: 20 most frequent sentiment-salient word dependencies extracted from the ordered dataset trained with lag -1 interday returns

Similar to Table 5.10, A word dependency is considered sentiment-salient if its absolute sentiment score is greater than 0.3.

dep	gov	rel	$S_{og.}$	$S_{sf.}$	Freq.	Eval.
share	fall	nsubj	-0.58	0.15	2152.21	og.
cent	rise	dobj	0.69	-0.38	1800.00	og.
pay	agree	xcomp	0.22	-0.49	882.67	og.
cent	gain	dobj	0.57	-0.33	798.53	og.
business	sell	dobj	-0.00	0.33	534.69	?
expectation	beat	dobj	0.32	-0.15	485.91	og.
stock	rise	nsubj	0.61	-0.04	480.03	og.
stake	sell	dobj	-0.13	0.35	429.83	?
list	top	dobj	0.08	-1.38	428.69	og.
cent	drop	dobj	0.06	-0.36	425.63	sf.
the	see	dobj	0.06	-1.49	414.34	?
stake	buy	dobj	-0.26	0.41	406.89	?
stock	track	dobj	0.04	-1.52	400.89	?
point	rise	dobj	1.15	-0.36	392.77	og.
income	report	dobj	-0.44	0.05	388.11	sf.
index	rise	nsubj	1.03	-0.25	386.44	og.
cent	add	dobj	0.56	-0.29	364.23	og.
income	rise	nsubj	-0.30	0.07	354.44	?
forecast	raise	dobj	0.61	-0.06	329.77	og.
deal	sign	dobj	0.30	-0.22	321.43	og.

(a) 20 most frequent word dependencies whose sentiment were classified differently by models trained over the original (Og.) versus the shuffled (Sf.) dataset

dep	gov	rel	$S_{P.}$	$S_{F.}$	Freq.	Eval.
share	rise	nsubj	0.55	-0.00	1889.14	P.
cent	gain	dobj	0.57	-0.03	798.91	P.
money	make	dobj	-0.35	0.20	719.23	F.
share	buy	dobj	0.30	-0.23	636.60	?
expectation	beat	dobj	0.32	-0.21	485.91	P.
stock	rise	nsubj	0.61	-0.25	480.03	P.
point	rise	dobj	1.15	-0.08	392.77	P.
income	report	dobj	-0.44	0.01	390.60	?
share	sell	dobj	-0.55	0.51	389.46	?
index	rise	nsubj	1.03	-0.14	386.83	P.
product	sell	dobj	0.24	-0.34	369.60	?
cent	add	dobj	0.56	-0.02	364.23	P.
income	rise	nsubj	-0.30	0.17	354.83	F.
report	expect	xcomp	0.34	-0.12	314.17	?
executive	become	xcomp	-0.17	0.32	285.51	?
plant	build	dobj	-0.08	0.43	272.09	F.
company	raise	nsubj	0.67	-0.13	270.40	P.
share	jump	nsubj	0.37	-0.08	267.43	P.
maker	report	nsubj	0.36	-0.19	254.29	?
average	rise	nsubj	1.02	-0.21	251.43	P.

(b) 20 most frequent word dependencies whose sentiment were classified differently by models trained with the past (P.) versus the future (F.) returns

Table 5.12: Comparing word dependencies whose sentiment were classified differently under various training settings

5.5.2 Experiment 2: Temporal Sentiment in Business News and Stock Returns

In Experiment 1, I have demonstrated the effectiveness of the supervised EM algorithm for learning textual sentiment from business news and stock price movements. In this experiment, I explore the temporal aspects of news sentiment by adopting the extended model introduced in Section 3.4.3. The design of this experiment mimicked that of the previous one, only this time incorporating temporalities into the sentiment model. The temporal aspect of the sentiment is introduced by splitting each of the original two sentiment classes into two, one representing retrospective sentiment and the other representing prospective sentiment. To keep the notations concise, the temporal sentiment class names in the equations are abbreviated as the following hereafter:

retneg representing *retrospective negative*
retpos representing *retrospective positive*
proneg representing *prospective negative*
propos representing *prospective positive*

As in Experiment 1, I first inspected the performances achieved by temporality-enabled models in cross-validation settings. As having been discussed in Section 3.4.3, the incorporation of the temporal aspects of the news sentiment requires that a past and a future return series be provided to enforce the supervision functions of the learning procedure. The past and the future return series used in this study were constructed from interpolated interday logarithmic returns of the stocks of the companies of interest. The interday returns were smoothed with weighted averaging; the past return for time t: $r_{t,\text{past}}$ was calculated by taking the average of the two returns at time $t-1$ and $t-2$, weighted exponentially:

$$r_{t,\text{past}} = \frac{2^1}{2^1+2^0} r_{t-1} + \frac{2^0}{2^1+2^0} r_{t-2} \tag{5.3}$$

$$= \frac{2}{3} r_{t-1} + \frac{1}{3} r_{t-2}. \tag{5.4}$$

Similarly, the future return at time t is calculated as

$$r_{t,\text{future}} = \frac{2^1}{2^1+2^0} r_{t+1} + \frac{2^0}{2^1+2^0} r_{t+2} \tag{5.5}$$

$$= \frac{2}{3} r_{t+1} + \frac{1}{3} r_{t+2}. \tag{5.6}$$

5.5.2.1 Correlational Performances

Similar to what were done for the previous experiment, a set of box plots (Figure 5.5) is used to summarise the correlational performances of the models when predicting market

returns. It is immediately recognisable that the algorithm was able to consistently learn language patterns that could be used to produce sentiment predictions that correlate significantly with both past and future returns (shown via the 'retneg'/'retpos' versus 'Past Return' sub-plots and the 'proneg'/'propos' versus 'Future Return' sub-plots, respectively). It is also observed that the retrospective positive sentiment correlates with past returns stronger than retrospective negative sentiment, though no such differences were observed among the prospective sentiment series and future returns.

To test whether the introduction of temporality helps improve the correlational performances of the models, a series of two-sample t-tests were conducted to compare the means of meta-correlations obtained from this experiment and those acquired from the previous experiment where no temporality was included in the model. The results for the tests are summarised in Table 5.13. The p-values suggested that none of the null hypotheses for the four tests could be rejected; in other words, a model that tries to capture sentiment and temporality at the same time does not predict future or past returns significantly better (nor worse) than a sentiment model which is trained to predict future or past returns separately.

Return Type	Sent.-Temp. Class	t-stat.	p-value
Past.Return	retneg	-1.17	0.25
Past.Return	retpos	0.24	0.81
Future.Return	proneg	-0.04	0.97
Future.Return	propos	-0.39	0.70

Table 5.13: Testing for significant differences between the means of meta-correlations achieved by sentiment models with and without the temporal aspect

The table summarises the results for the four t-tests that were used to determine whether introducing temporality into the sentiment learning model helps improve its correlational performance. Four tests were conducted, the result for each was summarised by one row in the table. The 'Return Type' and 'Sent.-Temp. Class' columns indicate from which two series the meta-correlations were calculated. The 't-stat' and 'p-value' columns display the t-statistic and p-value for the test respectively. For each test, the null hypothesis is that there is no difference between the correlational performances achievable by a mixture model that exploits sentiment and temporality at the same time (as in Experiment 2) and that by a sentiment model that is trained to predict past and future returns separately (as in Experiment 1).

An anomaly in the correlation patterns is noted when the learning was supervised by zero-lag returns: the model was able to achieve better performance predicting past returns using prospective sentiment, and also when predicting future returns using retrospective sentiment (Figure 5.5); in other words, if the current day's price movements were used to supervise the learning of prospective sentiment, then the resultant language patterns also correlate with 'yesterday"s returns (the same is true for retrospective sentiment and future returns). This counter-intuitive behaviour is likely a result of strong autocorrelations seen in news coverages — it is often the case that an event is covered by the media for several

days, a period during which key language patterns are reused; this may have made it difficult for the algorithm to distinguish between prospective and retrospective language patterns as their usage become mixed and distorted.

5.5.2.2 Distribution of Language Patterns

The aggregated probability distribution for the language patterns for each of the temporal sentiment classes were computed following the same procedure used in Experiment 1, except that four classes (i.e. 'retrospective negative', 'retrospective positive', 'prospective negative', and 'prospective positive') were included instead of the original two. The summaries of the resultant distributions are tabulated in Table 5.15. An additional *temporality score* is calculated for each entry in the vocabularies due to the incorporation of temporalities into the sentiment (shown in the table as column E_S). This score is defined as:

$$\begin{aligned}\text{temp}_{e_j} = &\log\left\{p\left(e_j \mid \text{proneg}\right) + p\left(e_j \mid \text{propos}\right)\right\} \\ &- \log\left\{p\left(e_j \mid \text{retneg}\right) + p\left(e_j \mid \text{retpos}\right)\right\}. \end{aligned} \tag{5.7}$$

An lexical entry will yield a positive temporality score if its sentiment, be it positive or negative, is more indicative of the market movements in the future than of what had happened in the past; a negative temporality score suggests the related lexical entry is mostly used to described the past incidences on the market.

At this point, it is worth noting that standalone evaluations of the algorithm's assignments of the temporality scores without in-depth domain knowledge can prove difficult. The main problem here is that it is often possible to find explanations that justify the assigning of both prospective and retrospective sentiment for many lexical patterns in the vocabulary. For example, the phrase 'analysts expect' could be understood as emphasising either the analysts' act of expecting, hence an expression indicating retrospective sentiment, or the details of their expectation, which leads to prospective interpretations of its potential impact on the future movements of the market. In such cases, the evaluation method employed in the previous experiment could not be properly applied to evaluate temporality score assignments because many of the patterns would have to be classified into the 'uncertain' category.

To overcome this problem, I decided to verify the temporality score assignments by way of contradiction — that is, instead of approaching the difficult problem of evaluating the temporality score assignment of a language pattern directly, I would first assume the temporality score the algorithm assigned to that language pattern were in fact correct, and then attempt to determine whether an explanation could be found to justify the *sentiment* assignment of the said pattern; the temporality class originally proposed by

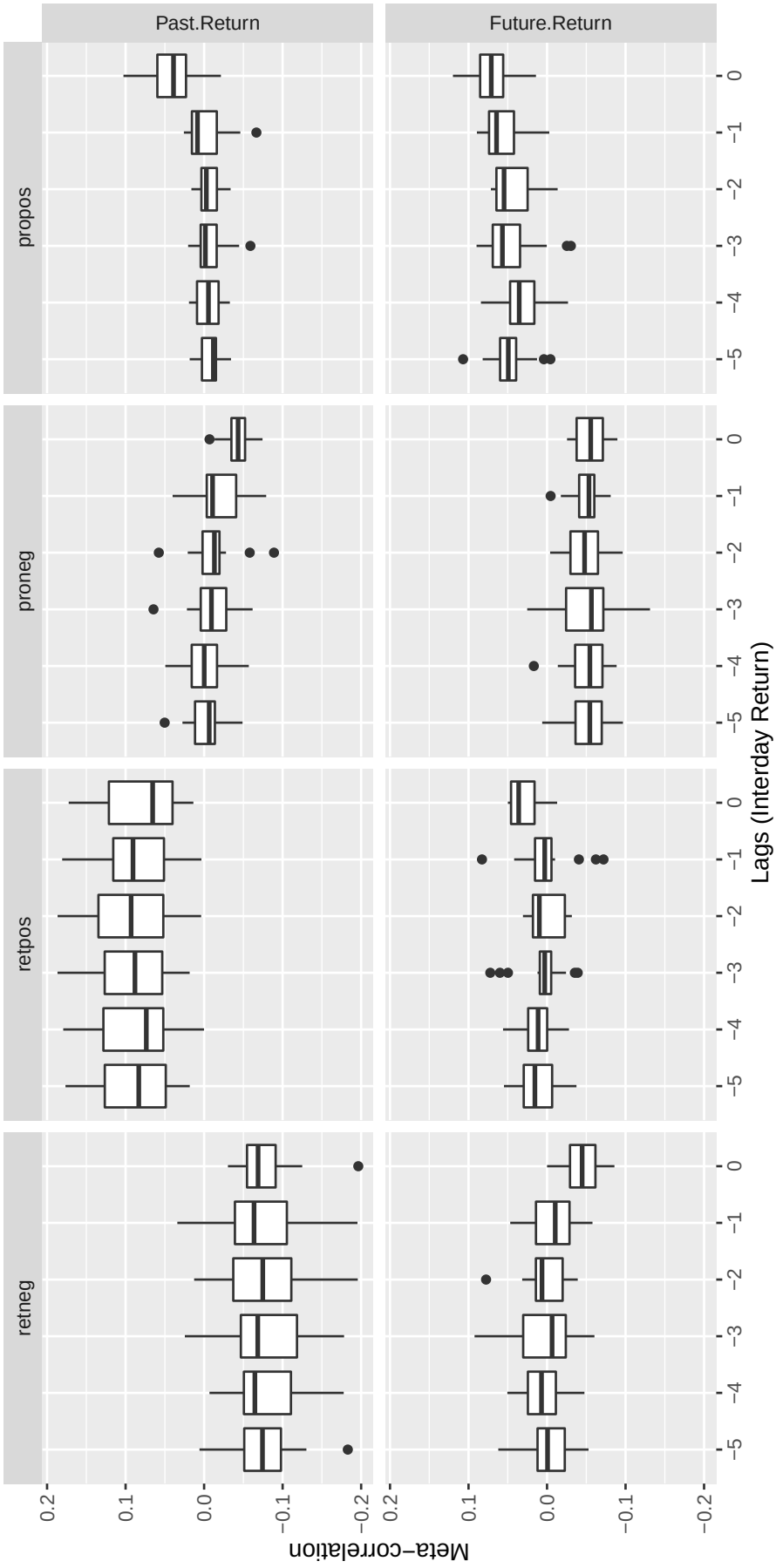


Figure 5.5: Summary of the correlational performances of the temporal sentiment models (testing phase)

In each sub-plot, the x-axis indicates how many days lagged used for calculating future returns. The future returns in this experiment were calculated with lag -1 interday returns. Lag 1 interday returns was used to compute past returns. Each sub-plot summarises the meta-correlations between a sentiment series and a return series; the sentiment class is indicated by the column the sub-plot is in, while the return type is indicated by the row the sub-plot resides.

the algorithm would be considered (i) correct if such an explanation can be found; (ii) incorrect if an explanation that would justify the opposite sentiment assignment can be identified; (iii) neither correct nor incorrect if no explanation could be found to support either the current sentiment assignment or the opposite sentiment assignment. The rationale behind this strategy is the observation that evaluating sentiment is much easier a task than to directly evaluate temporality. To implement this strategy, I introduced two additional *conditional sentiment scores* as the main devices used to evaluate the algorithm's classifications of a pattern's sentiment given its temporality class: the *prospective sentiment scores* and the *retrospective sentiment scores*. These two scores are defined in a similar manner as the sentiment scores previously introduced in Experiment 1 (Eq. 5.2), but were adapted to be based on only the prospective positive/negative and retrospective positive/negative scores respectively. More specifically:

$$\text{sent} \mid \text{pro}_{e_j} = \log p\left(e_j \mid \text{propos}\right) - \log p\left(e_j \mid \text{proneg}\right) \tag{5.8}$$

$$\text{sent} \mid \text{ret}_{e_j} = \log p\left(e_j \mid \text{retpos}\right) - \log p\left(e_j \mid \text{retneg}\right). \tag{5.9}$$

Table 5.15 displays the 20 most frequent word dependencies extracted by the algorithm that yielded outstanding temporality scores for each of the retrospective and prospective sentiment classes respectively. Appropriate conditional sentiment scores were computed for the outstanding retrospective and prospective patterns respectively. The Eval. columns present the manual evaluations I made towards the correctness of the conditional sentiment scores assigned by the algorithm. The markings given to the entries bear the same semantics as those used in the previous experiment. Table 5.14 tabulates the results of the manual evaluation.

Eval.	Retrospective	Prospective
✓	13	3
?	5	17
×	2	0

Table 5.14: The tallied results for the manual evaluation of the word dependencies displayed in Table 5.15

The results in Table 5.15 show that I could find explanations that justify the algorithm's sentiment classifications for more retrospective patterns than prospective patterns. According to the discussion described above, this seem to suggest that the algorithm is better at mining retrospective language patterns than mining prospective language patterns.

Table 5.17 compares the sentiment and temporality scores of the language patterns produced by models trained over the ordered dataset with those produced by models

dep	gov	rel	Freq.	S. | R.	Eval.	Temp.
share	fall	nsubj	2154.40	-0.64	✓	-1.00
share	rise	nsubj	1889.14	0.66	✓	-0.80
cent	rise	dobj	1800.00	0.56	✓	-3.21
cent	fall	dobj	1445.74	-0.30	✓	-3.12
earnings	report	dobj	1327.77	0.07	✓	-1.60
sale	rise	nsubj	1291.46	-0.10	×	-1.14
share	be	nsubj	1236.94	-0.08	?	-0.52
analyst	expect	nsubj	1223.37	-0.03	?	-0.58
revenue	rise	nsubj	1163.91	0.18	✓	-1.06
sale	fall	nsubj	930.69	-0.25	✓	-0.59
profit	report	dobj	923.20	0.33	✓	-1.18
result	report	dobj	851.34	0.23	?	-1.40
company	report	nsubj	833.06	-0.09	?	-1.29
cent	gain	dobj	798.91	0.42	✓	-4.27
loss	report	dobj	703.23	0.23	×	-0.57
profit	rise	nsubj	681.43	0.10	✓	-1.46
earnings	rise	nsubj	664.37	0.19	✓	-1.94
revenue	fall	nsubj	659.20	-0.31	✓	-0.88
share	buy	dobj	636.60	0.32	?	-0.43
question	raise	dobj	619.29	-0.33	✓	-0.39

(a) 20 most frequent retrospective word dependencies

dep	gov	rel	Freq.	S. | P.	Eval.	Temp.
comment	decline	xcomp	4404.11	-0.01	?	0.61
cost	cut	dobj	1069.69	0.11	✓	0.22
spokesman	decline	nsubj	1042.80	0.01	?	0.46
plan	announce	dobj	939.37	-0.02	?	0.30
company	be	nsubj	936.09	-0.15	?	0.20
advantage	take	dobj	883.89	0.01	?	0.20
pay	agree	xcomp	883.06	-0.13	✓	1.09
technology	use	dobj	855.91	0.27	?	1.26
role	play	dobj	798.23	0.14	?	0.39
decision	make	dobj	795.91	-0.05	?	0.52
step	take	dobj	783.26	-0.02	?	0.38
question	be	nsubj	771.66	0.13	?	0.11
place	take	dobj	725.49	-0.14	?	0.29
money	make	dobj	719.23	0.36	✓	0.10
company	plan	nsubj	697.60	-0.22	?	0.46
sense	make	dobj	695.06	-0.07	?	0.50
company	make	nsubj	694.37	0.11	?	0.46
service	offer	dobj	665.43	0.05	?	1.02
easier	make	xcomp	624.46	0.01	?	0.81
spokeswoman	decline	nsubj	607.17	0.11	?	0.61

(b) 20 most frequent prospective word dependencies

Table 5.15: Most frequent temporality-salient word dependencies extracted using the ordered dataset

The average frequencies (i.e. *Freq.*) are computed in the same way as in previous tables (i.e. average of the aggregated frequencies from each company). The *Sent.* column contains the adapted sentiment score for temporal sentiment. The *S.* | *R.* and the *S.* | *P.* columns show the *retrospective sentiment score* and the *prospective sentiment score* respectively.

trained over the shuffled dataset. The manual evaluation of the results follows the same strategy developed above: for each entry in the table, I evaluated the conditional sentiment scores assigned to it by the algorithm given their respective temporality classifications. The results are tabulated in Table 5.16. These results showed that the algorithm was able to generate more reliable 'opinions' about the sentiment connotations of the language patterns conditioned on their respective temporality classification when trained with the ordered dataset than when trained with the shuffled dataset.

Both results from Table 5.15 and Table 5.15 seem to suggest that the extended model is able to identify retrospectivity-laden sentiment patterns better than their prospective counterparts. One possible explanation to this asymmetry is that the dominating narrative in business and financial news conveyed over traditional media is retrospective — in other words, the algorithm's inferior performance in extracting prospective sentiment patterns compared to retrospective ones may be due less to the incapacity of the algorithm to detect them, but more to the fact that prospective signals in news are generally weak when delivered through traditional media.

Eval.	Count
ordered	11
?	9
shuffled	0

Table 5.16: The tallied results for the manual evaluation of the word dependencies displayed in Table 5.15

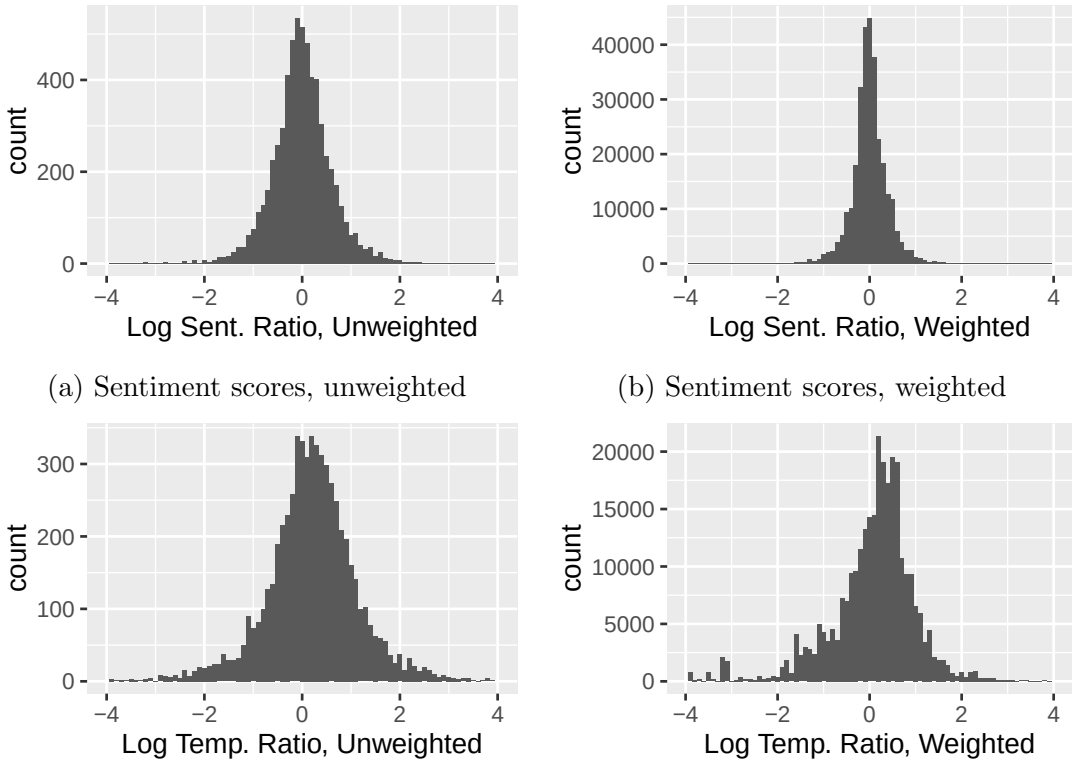
The results from this experiment also seem to suggest that the magnitudes of the temporality scores are overall greater than the magnitudes of the sentiment scores. When weighted by the frequencies of the associated language patterns, the distribution of the temporality scores exhibits a negative skew compared to a non-weighted distribution (Figure 5.6c versus 5.6d), which suggests that retrospective usage of language, as identified by the learning algorithm developed in this thesis is more prominent than prospective usage of language in business news published by formal media. The histogram for the sentiment scores remains largely symmetric after being weighted by the frequencies (Figure 5.6a versus 5.6b), which implies the uses of positive and negative language patterns were about equally common in the text.

5.5.2.3 Summary

In this experiment, I used the EM algorithm to learn word dependency distributions for the temporality-sentiment model developed in Section 3.4.3 and manually evaluated parts of the resultant distributions. The general set-up of the experiment is similar to that used

dep	gov	rel	Sent. (Ordered)	Sent. (Shuffled)	Eval.	Temp. (Ordered)	Temp. (Shuffled)	Freq.
share	fall	nsubj	-0.64	0.32	ordered	-1.00	-0.09	4307.94
cent	rise	dobj	0.56	-0.27	ordered	-3.21	-0.12	3600.00
share	be	nsubj	-0.08	0.20	?	-0.52	-0.03	2473.89
revenue	rise	nsubj	0.18	-0.11	ordered	-1.06	-0.29	2328.86
profit	report	dobj	0.33	-0.06	ordered	-1.18	-0.04	1847.20
company	report	nsubj	-0.09	0.18	?	-1.29	-0.12	1666.11
cent	gain	dobj	0.42	-0.29	ordered	-4.27	-0.07	1597.83
role	play	dobj	0.14	-0.11	?	0.39	0.19	1596.46
step	take	dobj	-0.02	0.07	?	0.38	0.33	1566.51
company	plan	nsubj	-0.22	0.09	?	0.46	0.06	1396.80
business	do	dobj	0.09	-0.09	ordered	1.28	0.24	1100.69
business	sell	dobj	-0.01	0.25	ordered	0.56	0.56	1070.23
lot	be	nsubj	0.15	-0.14	?	0.45	0.30	1054.97
reason	be	nsubj	-0.25	0.07	?	0.36	0.15	1043.31
store	open	dobj	0.43	-0.09	ordered	1.16	0.70	950.11
way	find	dobj	0.14	-0.12	ordered	0.79	0.06	861.37
company	leave	dobj	0.04	-0.13	?	0.36	0.15	859.66
earnings	fall	nsubj	-0.31	0.21	ordered	-1.54	-0.46	858.80
job	do	dobj	-0.15	0.25	?	0.32	0.07	809.60
concern	raise	dobj	-0.06	0.40	ordered	0.18	0.31	795.03

Table 5.17: 20 most frequent word dependencies whose temporality were classified differently by models trained with ordered versus shuffled setting



(a) Sentiment scores, unweighted

(b) Sentiment scores, weighted

(c) Temporality scores, unweighted

(d) Temporality scores, weighted

Figure 5.6: Histograms depicting the distributions of sentiment and temporality scores, unweighted versus weighted by frequencies

in Experiment 1; the difference being that instead of classifying language patterns into dichotomous positive-versus-negative categories, the word dependencies were categorised into four temporal sentiment classes, namely *retrospective negative*, *retrospective positive*, *prospective negative*, and *prospective positive*. My main findings are:

1. Overall, the supervised EM algorithm was able to identify the temporal sentiment categories associated word dependencies better than random guessing.

2. Retrospective language usage is in general more prevalent than prospective language usage in business and financial news published on formal media.

3. There was no significant difference between the correlational performances achieved by a temporality-enabled sentiment model that captures the sentiment and temporality all at once and that achieved by a sentiment model trained with contemporaneous sentiment and returns.

5.5.3 Experiment 3: Sentiment Modelled by Unigrams

In the sentiment analysis literature, most studies on the interrelations between news sentiment and the market movements had focused on approximating news sentiment using

the raw or relative frequencies of certain unigrams (i.e. single words) that are deemed to convey sentiment. It has been hypothesised in this thesis that sentiment, as a form of meaning, might be better captured by lexical units that are more specific than single words, e.g. bigrams, trigrams, word dependencies, etc. To test whether this hypothesis holds for this case study, I conducted empirical comparisons between the effectiveness of the learning algorithm when using word dependency as features and that of the algorithm when using different sets of unigrams as features. More specifically, I compared the correlational performances achieved by models trained using word dependencies as features against those achieved by models trained using the following lexical items as features: (i) all unigrams; (ii) unigrams containing just verbs and adjectives.

The correlational performances achieved by models trained with word dependencies had already been explored in Experiment 2; the correlational performances for the models trained with unigrams were obtained in a similar way as in Experiment 2, except that unigrams were used as linguistic features in lieu of word dependencies. For the experiment configurations that used unigrams as features, each entry in the vocabulary is represented as a tuple in the form of '*lemma part-of-speech*'; for example, 'fall VB' represents the verb 'fall' (i.e. not the season 'autumn'). Similar to the previous two experiments, the inflections that could potentially signify the tenses of the words were erased; however, the tense information could sometimes also be captured in the part-of-speech tags — the Stanford CoreNLP package produces different POS tags for verbs of different forms, e.g. the verb 'fell' will show up as 'fall VBD', indicating it is the past tense form of 'fall'. I therefore mapped all verb tag variants ('VBD' for past tense, 'VBG' for present participle, 'VBN' for past participle, 'VBP' for non-3rd-person singular present, and 'VBZ' for 3rd-person singular present) into a single 'VB' tag. The vocabulary was filtered further to remove words whose length are shorter than three characters to eliminate possible noises due to ill-formatted text. A low-pass filter like the one used in previous experiments was also applied to remove word entries that occurred 16 times or less[2] for each cross-validation fold.

5.5.3.1 Correlational Performances

Similar to the previous experiments, the meta-correlations between the predicted sentiment series and the future and past returns are summarised with box plots in Figure 5.8 and 5.7. A visual comparison between the plots thereof and the plot obtained when word dependencies were used as features (Figure 5.5) seems to suggest that the sentiment series predicted by the models learnt from unigrams correlate more strongly with the future and past returns than those predicted using models learnt with word dependencies, especially

[2]This threshold was determined empirically, and it could be improved in future developments of this work.

Return Type	Sentiment Class	F Statistic	p-value
Future.Return	proneg	6.305	0.004
Future.Return	propos	4.811	0.013
Past.Return	retneg	2.070	0.138
Past.Return	retpos	0.428	0.654
Past.Return	propos	0.339	0.714
Future.Return	retneg	0.130	0.878
Past.Return	proneg	0.107	0.899
Future.Return	retpos	0.040	0.961

Table 5.18: ANOVA tests that compare the average meta-correlations achieved by models trained with different lexical features.

Each row in this table summaries the result from one ANOVA test whose null hypothesis was that the means of the meta-correlations achieved by the three model designs (word dependencies, verbs and adjectives, and all unigrams) are the same when the models were used to predict the returns of a given type (shown by the 'Return Type' column) using the temporal sentiment series whose class is specified by the 'Sentiment Class' column. The F statistics used by the ANOVA tests are displayed in the 'F Statistic' column, and the associated p-values are shown in the column of the same name.

for negative sentiment.

Also observed in Figure 5.8 is the same kind of anomaly as found in Figure 5.5 where when 0-lag interday returns were used to derive the future return series that supervised the learning, the algorithm seemed to be able to find unigram distributions for retrospective/prospective sentiments that predict future/past returns better compared to the other lags. This observation suggests that the stock price movments on day t are a mixture of responses to news sentiment released previously and potential causes that may influence the news sentiment in the future.

A visual comparison between Figure 5.5, 5.7 and 5.8 suggests that the algorithm was able to achieve slightly better correlational performances with unigrams as features than with word dependencies as features. Additional statistical tests confirmed this observation. Table 5.18 summaries the results from eight *Analysis of Variances* (ANOVA) procedures, each testing the null hypothesis that the means of the meta-correlations achieved by the three designs (word dependencies, verbs and adjectives, and all unigrams) are the same when predicting the returns of a given type (past or future returns) using the temporal sentiment series specified; the rows are sorted by the p-values resulted from the tests in ascending order. The results show that the null hypothesis, i.e. meta-correlations achieved by the three designs share the same mean, can be rejected at a 0.05 critical level only when prospective sentiment series were used to predict future returns. In other words, no significant differences in the correlational performances of the three designs could be detected when capturing the relationships between the market movements and retrospective news sentiment.

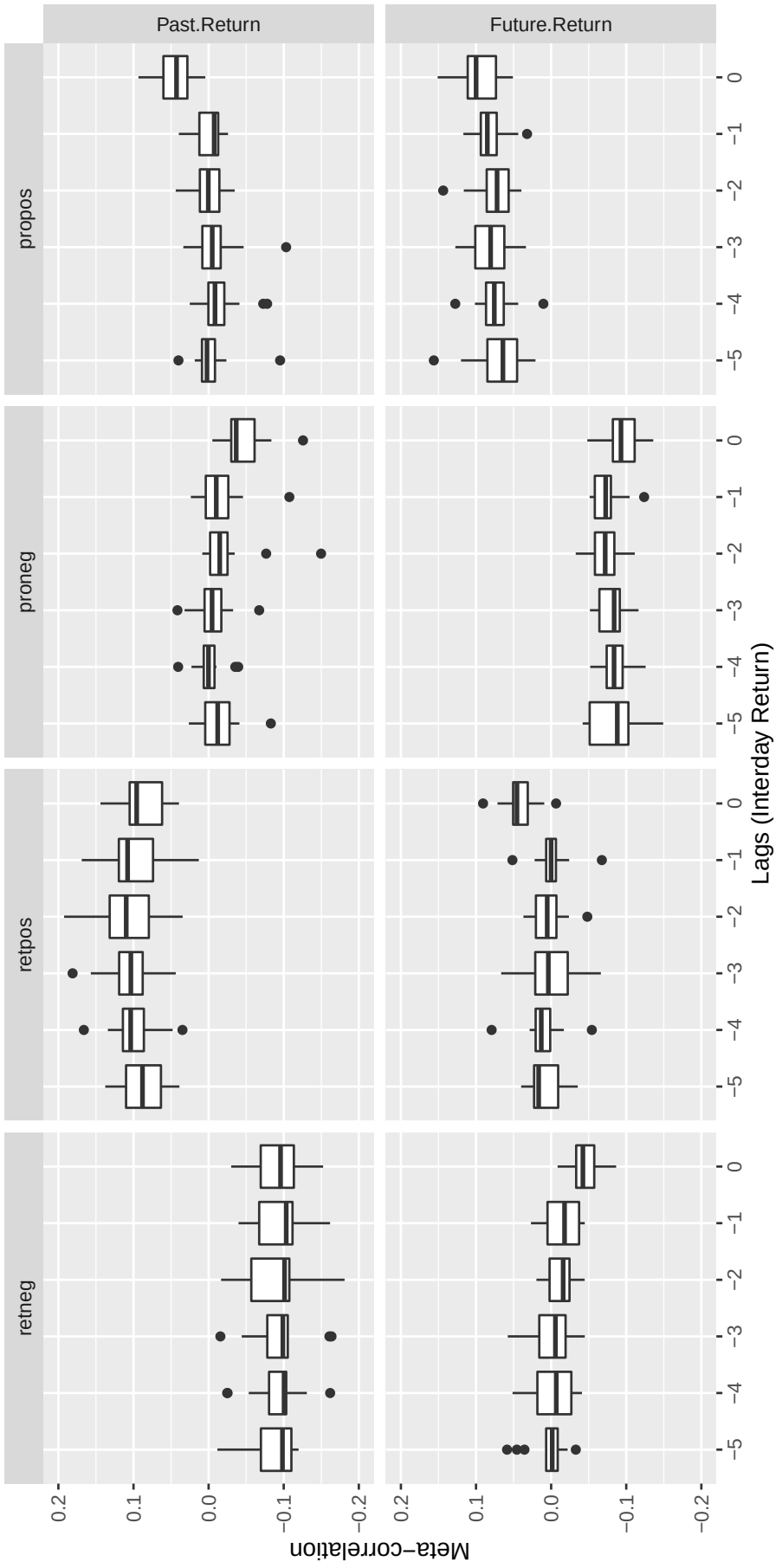


Figure 5.7: Box plots summarising the correlational performances of the temporal sentiment models trained with nouns, verbs, and adjectives (testing phase)

This diagram is comparable to Figure 5.5 produced in Experiment 2 and Figure 5.8.

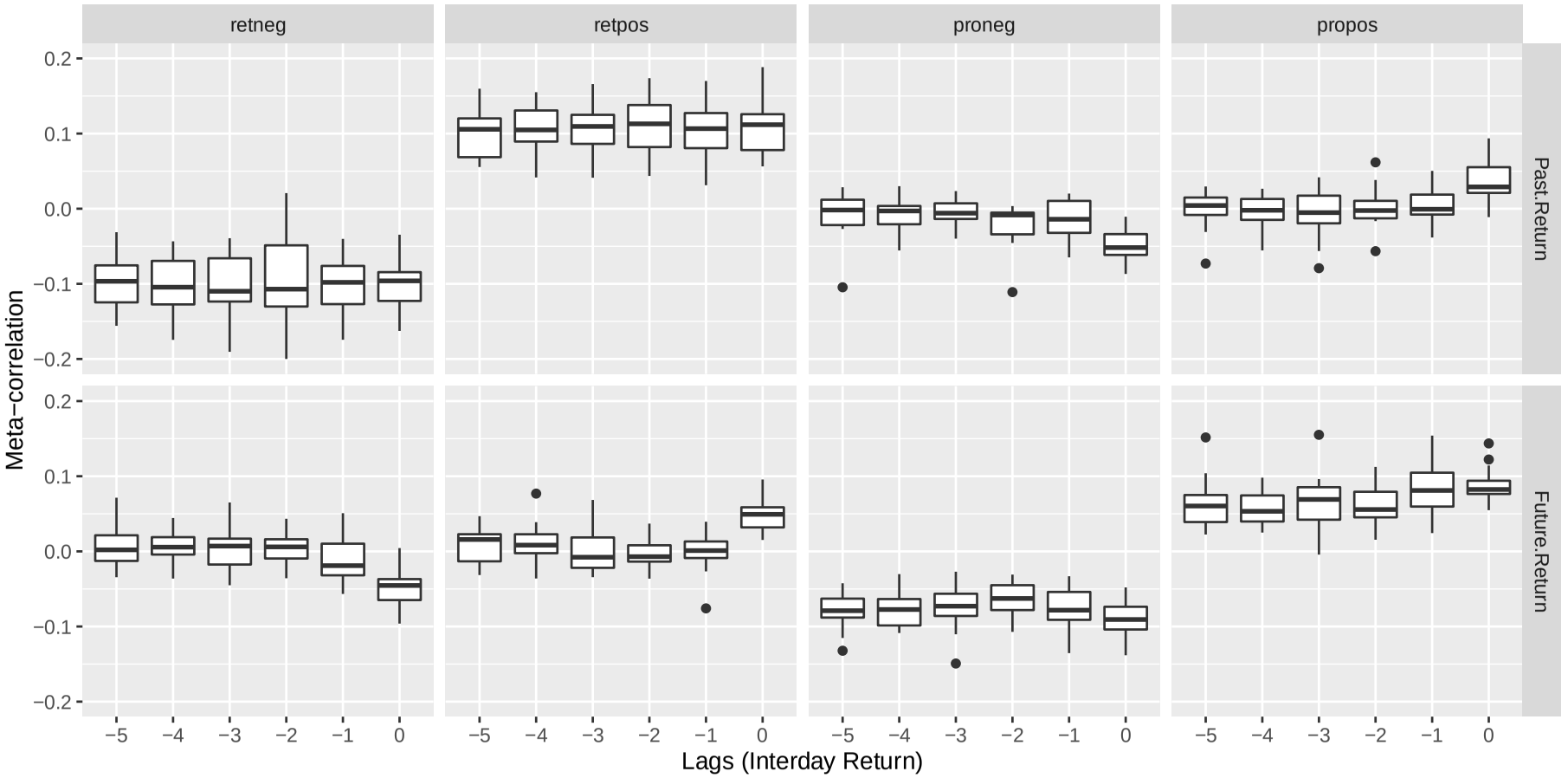


Figure 5.8: Box plots summarising the correlational performances of the temporal sentiment models trained with verbs and adjectives (testing phase)

This diagram is comparable to Figure 5.5 produced in Experiment 2 and Figure 5.7.

class	comparison	estimate	conf.low	conf.high	adj.p.value
proneg	Verbs & Adj. vs. Word Dep.	-0.03	-0.05	-0.01	0.00
proneg	All Unigrams vs. Word Dep.	-0.02	-0.04	-0.00	0.02
proneg	All Unigrams vs. Verbs & Adj.	0.00	-0.02	0.03	0.84
propos	Verbs & Adj. vs. Word Dep.	0.03	0.00	0.05	0.02
propos	All Unigrams vs. Word Dep.	0.03	0.00	0.05	0.03
propos	All Unigrams vs. Verbs & Adj.	-0.00	-0.03	0.02	0.99

Table 5.19: Results from Tukey HSD tests that compare the pairwise correlational performances between each two of the three lexical features when predicting future returns.

The 'class' column shows the temporal sentiment class of the sentiment series used to predict the future returns. The 'comparison' column indicates the null hypotheses for each pairwise test; the null hypothesis for each test is that the average meta-correlation obtained by models using the former feature (i.e. before the 'vs.') is neither greater nor lower than that obtained by models trained using the latter feature. The 'estimate' column contains the statistics of the tests. The 'conf.low' and 'conf.high' columns show the lower and upper bounds for the confidence intervals. The 'adj.p.value' column display the adjusted p-values output by the tests.

Tukey Honest Significant Differences tests were conducted to determine the uses of which features had resulted the differences observed in the ANOVA tests. The outcomes from the Tukey HSD tests (Table 5.19) indicate that (i) both models trained using unigram as features outperformed the model trained with word dependencies as features, and (ii) there are not enough evidence to reject the null hypothesis that the correlational performances of the models trained using verbs and adjectives are different from that achieved by the models trained using all unigrams as features. When interpreting the pairwise tests in Table 5.19, it should be noted that positive estimates indicate the difference between the two mean meta-correlations are positive, and negative estimates indicate the opposite. For designs involving positive sentiment series (i.e.propos) therefore, positive estimates indicate that the former design in the 'comparison' column achieved greater correlational performances than the latter design in that column; for designs involving negative sentiment series (i.e.proneg), however, the more negative a design's meta-correlation with the return series is, the better its correlational performance — and thus it is negative estimates that indicate that the former design in the 'comparison' column performed better than the latter.

Similar to before, I now move to examine the actual distributions of the words learnt by the model using the 'all unigrams' and 'verbs and adjectives' configurations.

5.5.3.2 Distribution of Language Patterns

Table 5.22 summarises the word distributions for each of the four temporal sentiment classes learnt with models trained using all the unigrams seen in the training set. The exposition of the distributions follows the same format used in the previous two experiments. It

can be seen from the table that nouns dominated the most frequent unigrams for both retrospective and prospective sentiment classes. The results of the manual evaluations of the conditional sentiment score assignments for Table 5.22 are tabulated in Table 5.20. It is noted that the nouns in the vocabulary are in general less interpretable than the verbs — among all the confirmed assignments, all but one are for verbs. Another observation is that the algorithm's conditional sentiment score assignments for prospective distributions are relatively less reliable than those made for retrospective distributions.

For the third configuration where only verbs and adjectives were used to train the models, the resultant distributions of the unigrams for the four temporal sentiment classes are summarised in Table 5.23; the results of the corresponding manual evaluation on conditional sentiment scores assignments are tabulated in Table 5.21.

The first thing to note is that the assignments of the conditional sentiment scores for verbs and adjectives in the retrospective sentiment classes are more interpretable than those in the prospective sentiment classes, as more entries were classified correctly in the retrospective word distribution than in the prospective word distribution. It is difficult to compare the evaluations in this configuration (as in Table 5.23) with that from the previous configuration (as in Table 5.22) due to the overwhelming presence of nouns.

To investigate whether including unigrams other than verbs and adjectives affect the assignments of the conditional sentiment scores, an additional Table 5.24 was produced using the 'all unigrams' configuration; this table shows only the verbs and adjectives from all the unigrams. It can be seen that there is virtual no difference between the assignments of conditional sentiment scores in Table 5.24 and in Table 5.23. Combined with the finding from the previous subsection that there is no significant difference in the correlational performances between models learnt using all unigrams as features and those learnt using only verbs and adjectives as features, it can be argued that including unigrams other than verbs and adjectives does not contribute significantly to the method's ability to capture temporal sentiment.

Another observation is that the conditional sentiment score assignments for the word dependencies in Experiment 2 are easier to interpret than those assigned to the unigrams. However, it would seem that much of word dependencies' superior interpretability is owing more to the fact that the word dependencies used to train the model were pre-selected to reflect verb-like relations (i.e. 'direct object' of a verb, 'nominal subject' of a verb, etc.) than they being more specific lexical units compared to unigrams. This is perhaps especially true for retrospective word dependencies — note how all the governor verbs in the word dependencies displayed in Table 5.15a also appeared in Table 5.23a, and how the word dependencies' conditional sentiment assignments always align with that of their governor verbs. For prospective sentiment, word dependencies do seem to be slightly more interpretable than unigrams; this may suggest that word dependencies are able to better

capture prospective sentiment than unigrams, especially considering prospective language usages are in general much less prevalent as implied by previous results.

Eval.	Retrospective	Prospective
✓	4	0
?	16	20
×	0	0

Table 5.20: The tallied results for the manual evaluation of the unigrams displayed in Table 5.22

Eval.	Retrospective	Prospective
✓	10	0
?	10	20
×	0	0

Table 5.21: The tallied results for the manual evaluation of the unigrams displayed in Table 5.23

5.5.3.3 Summary

The main goal of this experiment is to evaluate how well unigrams can be used to capture the sentiment and temporality of the language patterns used in business news compared to alternative unit of meaning word dependencies.

I first compared the correlational performances achieved by models learnt using three different lexical features: (i) word dependencies, (ii) all unigrams, (iii) verbs and adjectives. The results from a series of statistical tests suggest that the models trained using single words as features outperformed the model trained over word dependencies when predicting future returns with prospective sentiment; no significant differences in correlational performances were observed otherwise.

Within the unigram configurations, it was found that the learning method developed in this thesis were able to identify the sentiment and temporality connotations for verbs and adjectives more reliably than for words of other part-of-speeches; also, the method was able to identify retrospective sentiment connotations more accurately than prospective sentiment connotations. Combined with the finding that there is no significant difference between the correlational performances of the models trained over all the unigrams and that of the models trained over only verbs and adjectives, it is reasonable to argue that verbs and adjectives are the main bearer of retrospective sentiment in English business and financial news.

Lemma	POS	Freq.	*S.* | *R.*	Temp.	Eval.
cent	NN	56858.57	0.07	-0.61	?
share	NN	41593.20	0.09	-0.36	?
sale	NN	37331.54	-0.05	-0.13	?
after	IN	32250.80	0.07	-0.35	?
price	NN	24816.80	0.03	-0.19	?
analyst	NN	23070.17	-0.01	-0.20	?
quarter	NN	22680.94	0.03	-0.30	?
rise	VB	19964.51	0.39	-0.79	✓
stock	NN	19628.20	0.10	-0.62	?
revenue	NN	19280.00	0.06	-0.12	?
earnings	NN	19235.83	0.08	-0.67	?
week	NN	19178.60	-0.04	-0.16	?
fall	VB	18178.11	-0.30	-0.80	✓
investor	NN	18113.49	0.04	-0.21	?
profit	NN	17528.97	0.03	-0.43	?
growth	NN	14745.77	0.06	-0.19	?
report	VB	13100.43	0.11	-0.45	?
continue	VB	13049.49	0.00	-0.17	?
add	VB	12542.31	0.16	-0.16	✓
loss	NN	11879.43	-0.12	-0.34	✓

(a) 20 most frequent retrospective unigrams

Lemma	POS	Freq.	*S.* | *P.*	Temp.	Eval.
software	NN	26904.94	0.10	0.13	?
product	NN	22705.29	0.04	0.11	?
service	NN	22552.94	0.04	0.13	?
such	JJ	22338.26	0.03	0.12	?
wal-mart	NNP	21668.71	-0.01	0.13	?
technology	NN	21235.14	0.05	0.15	?
intel	NNP	20148.03	0.01	0.22	?
people	NN	19375.91	0.06	0.14	?
system	NN	18089.89	-0.01	0.15	?
industry	NN	16280.11	-0.01	0.12	?
computer	NN	16164.40	-0.02	0.14	?
accord	VB	15902.60	0.06	0.11	?
use	VB	15400.86	0.05	0.23	?
plan	NN	14695.71	0.04	0.10	?
work	VB	13363.49	0.05	0.19	?
government	NN	12815.00	0.10	0.16	?
chip	NN	11950.83	0.07	0.43	?
offer	VB	11353.17	0.07	0.13	?
under	IN	11261.20	0.01	0.11	?
china	NNP	10752.20	0.09	0.21	?

(b) 20 most frequent prospective unigrams

Table 5.22: 20 most frequent temporality-salient unigrams extracted from the ordered dataset using all unigrams as features

The 'Lemma' and 'POS' columns are self-explanatory. The part-of-speech tags appeared in this table include: (i) nouns (NN), (ii) proper nouns (NNP), (iii) verbs (VB), (iv) prepositions (IN), and (v) adjectives (JJ). Only unigrams whose absolute temporality scores are greater than 0.1 were included in the table to help suppress noises.

Lemma	POS	Freq.	$S. \mid R.$	Temp.	Eval.
rise	VB	19966.43	0.39	-0.87	✓
fall	VB	18163.29	-0.30	-0.85	✓
report	VB	13100.86	0.11	-0.50	?
continue	VB	13037.89	0.00	-0.20	?
add	VB	12541.63	0.16	-0.17	?
increase	VB	10570.54	0.12	-0.17	✓
lose	VB	8266.86	-0.17	-0.43	✓
raise	VB	8176.40	0.10	-0.31	✓
strong	JJ	8081.20	0.26	-0.36	✓
higher	JJR	8029.77	0.11	-0.50	✓
second	JJ	7777.00	0.01	-0.21	?
show	VB	7663.29	0.10	-0.14	?
compare	VB	6857.94	0.05	-0.18	?
high	JJ	6795.09	0.09	-0.18	?
gain	VB	6641.29	0.40	-0.90	✓
lower	JJR	6528.89	-0.25	-0.31	✓
third	JJ	6172.46	-0.02	-0.19	?
close	VB	5720.91	0.06	-0.25	?
boost	VB	5039.74	0.19	-0.19	✓
economic	JJ	4825.20	-0.10	-0.36	?

(a) 20 most frequent retrospective words

Lemma	POS	Freq.	$S. \mid P.$	Temp.	Eval.
make	VB	37972.17	0.03	0.10	?
such	JJ	22325.89	0.03	0.12	?
accord	VB	15892.69	0.06	0.12	?
use	VB	15394.69	0.04	0.25	?
work	VB	13364.09	0.06	0.20	?
offer	VB	11357.66	0.06	0.14	?
become	VB	11154.80	0.07	0.11	?
call	VB	10411.97	0.04	0.18	?
build	VB	9487.09	0.02	0.15	?
want	VB	9251.11	0.02	0.12	?
provide	VB	9054.34	0.04	0.16	?
need	VB	8694.89	0.01	0.18	?
large	JJ	8438.51	-0.01	0.11	?
begin	VB	8343.43	0.01	0.11	?
familiar	JJ	7956.06	0.09	0.10	?
allow	VB	7935.63	0.08	0.14	?
write	VB	7827.49	0.02	0.14	?
develop	VB	7761.91	0.04	0.28	?
know	VB	7641.26	0.11	0.14	?
former	JJ	7261.23	0.05	0.15	?

(b) 20 most frequent prospective words

Table 5.23: 20 most frequent temporality-salient unigrams extracted from the ordered dataset using only verbs and adjectives as features

The 'Lemma' and 'POS' columns are self-explanatory. The part-of-speech tags appeared in this table include: (i) verbs (VB), and (ii) adjectives (JJ). Only unigrams whose absolute temporality scores are greater than 0.1 were included in the table to help suppress noises.

Lemma	POS	Freq.	$S. \mid R.$	Temp.	Eval.
rise	VB	19964.51	0.39	-0.79	✓
fall	VB	18178.11	-0.30	-0.80	✓
report	VB	13100.43	0.11	-0.45	?
continue	VB	13049.49	0.00	-0.17	?
add	VB	12542.31	0.16	-0.16	?
increase	VB	10568.86	0.11	-0.15	✓
lose	VB	8277.31	-0.17	-0.42	✓
raise	VB	8179.03	0.10	-0.30	✓
strong	JJ	8087.17	0.26	-0.33	✓
higher	JJR	8030.66	0.11	-0.46	✓
second	JJ	7791.51	0.00	-0.19	?
show	VB	7665.14	0.12	-0.13	?
compare	VB	6865.71	0.04	-0.17	?
high	JJ	6792.77	0.09	-0.17	?
gain	VB	6645.09	0.39	-0.85	✓
lower	JJR	6525.71	-0.26	-0.30	✓
third	JJ	6165.34	-0.02	-0.17	?
close	VB	5721.97	0.07	-0.23	?
boost	VB	5038.43	0.18	-0.15	✓
economic	JJ	4833.14	-0.09	-0.33	?

(a) 20 most frequent retrospective words

Lemma	POS	Freq.	$S. \mid P.$	Temp.	Eval.
such	JJ	22338.26	0.03	0.12	?
accord	VB	15902.60	0.06	0.11	?
use	VB	15400.86	0.05	0.23	?
work	VB	13363.49	0.05	0.19	?
offer	VB	11353.17	0.07	0.13	?
call	VB	10415.86	0.04	0.16	?
build	VB	9488.66	0.01	0.14	?
want	VB	9253.23	0.02	0.11	?
provide	VB	9037.86	0.04	0.15	?
need	VB	8691.11	0.00	0.17	?
large	JJ	8444.29	-0.01	0.11	?
begin	VB	8345.26	0.01	0.11	?
familiar	JJ	7958.86	0.09	0.10	?
allow	VB	7930.97	0.08	0.13	?
write	VB	7816.71	0.02	0.14	?
develop	VB	7756.80	0.03	0.26	?
know	VB	7642.46	0.11	0.12	?
former	JJ	7265.51	0.05	0.12	?
create	VB	7106.03	0.08	0.16	?
base	VB	6833.46	0.01	0.21	?

(b) 20 most frequent prospective words

Table 5.24: 20 most frequent temporality-salient unigrams extracted from the ordered dataset using all unigrams as features (showing only verbs and adjectives)

The 'Lemma' and 'POS' columns are self-explanatory. The part-of-speech tags appeared in this table include: (i) verbs (VB), and (ii) adjectives (JJ). Only unigrams whose absolute temporality scores are greater than 0.1 were included in the table to help suppress noises.

Another observation was that, while models trained using unigrams were able to achieve superior correlational performances, models trained with word dependencies seem to have excelled at capturing the prospective temporal sentiment in the news. It was also recognised that word dependencies owe much of their sentiment and temporality interpretability to their verbal components.

5.5.4 Experiment 4: Benchmarking with Content Analysis

The previous three experiments were designed to demonstrate the effectiveness of the supervised EM algorithm under various settings. In this experiment, I compare the correlational performances of the methods developed in this thesis to that obtained from content-analysis-based methods. Such methods are typically used in combination with domain dictionaries. Two affect dictionaries were examined in this comparison: the *General Inquirer* [Stone et al., 1966] affect dictionary, as well as the more recent *Loughran and McDonald Sentiment Word Lists* [Loughran and McDonald, 2011a]. Both dictionaries have seen widespread use in previous studies on textual sentiment and market.

Several preprocessing treatments were performed on the raw GI dictionary before it was used to conduct content analysis:

1. Words in all the entries were converted into lower cases.

2. When multiple senses are defined for a word, the categories that are associated with the most commonly-used sense were used. The most commonly-used sense of a word in this case was identified as the sense that possesses the highest usage probability[3] among all the senses for the word. This served as a form of word sense disambiguation procedure.

3. Only the 'Positiv' and the 'Negativ' categories were retained. Entries that belong to neither categorises were discarded.

The above treatment leaves 3468 entries in the dictionary, of which 1559 were classified as positive terms and the other 1909 were classified as negative terms.

The original master L&M dictionary contains 85131 entries. Preprocessing procedures similar to the ones performed on the GI dictionary were applied to the L&M dictionary except for the second step — the L&M dictionary does not distinguish among different senses of the words, so the entries in the dictionary were taken as is. After the preprocessing, there are 2709 entries in the dictionary, of which 354 are marked as positive terms and the other 2355 marked as negative terms.

[3]The usage probabilities are supplied in the 'Defined' column of the raw dictionary table.

5.5.4.1 Correlational Performances

In this section, I try to evaluate how well the GI and the L&M dictionaries capture the sentiment of the words in a business news context compared to the methods developed in this thesis. The idea behind the experiment design is comparable to that of the previous experiments. The more accurate a sentiment proxy derived from the words defined in the dictionary is able to correlates with the returns using news text, the better the sentiment of the words are defined.

Unlike the approach developed in this thesis where statistical learning is used, the content-analysis-based methods do not exploit the potential associations between the language patterns and the target of prediction (i.e. returns) that exist in the data; instead they relies solely on the external knowledge it is meant to capture (i.e. expert knowledge on the sentiment connotations of the words). As such, predictions made from content analysis with sentiment dictionaries are immune to over-fitting, so the correlational performances obtained from such methods would better reflect the quality of the sentiment definitions for the dictionaries.

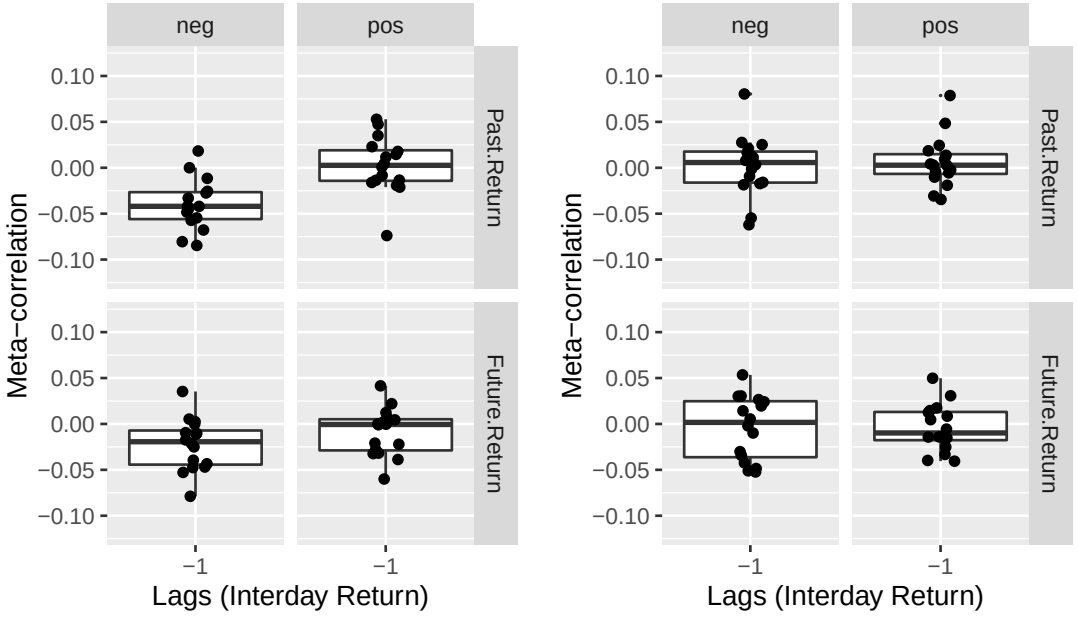
The two sentiment dictionaries were used to annotate the texts in the news articles. A word in a document was tagged as either 'positive' or 'negative', or omitted all together if the word is not defined in the dictionary; the number of words fell into each category is then counted and used as a proxy of sentiment. The correlational performance of the sentiment proxy derived using content analysis was evaluated in the same way as in the previous experiments, where a repeated random split cross-validation was conducted and the correlations for each of the companies were then summarised using meta-analysis. The set of feeds for random number generation used in this experiment is the same as the one used in previous experiments so that the splits of training and validation sets for this experiment are identical to that from the previous experiments.

The analyses of the correlational performances of the settings, similar to previous experiments, are exhibited in box plots. As a baseline, the box plots in Figure 5.9b and 5.10b both seem to suggest that in general no significant correlation can be established between the usage of sentiment-bearing words as defined in either of the two dictionaries and the prices movements of the companies' stocks when the shuffled dataset was used.

Several observations can be made based on the box plots:

1. For the ordered dataset (Figure 5.9a and 5.10a), it can be argued that usage of negative words correlates significantly with past returns for both dictionaries, while the impact of positive textual sentiment on past returns is marginal at best.

2. For both dictionaries, the negative sentiment series seem to correlate better with past returns than with future returns (Figure 5.9a and 5.10a), which conforms with the previous finding that retrospective language usage is more prominent than



(a) Ordered dataset

(b) Shuffled dataset

Figure 5.9: Summary of the correlational performances of models built with Content Analysis using the GI dictionary (testing phases)

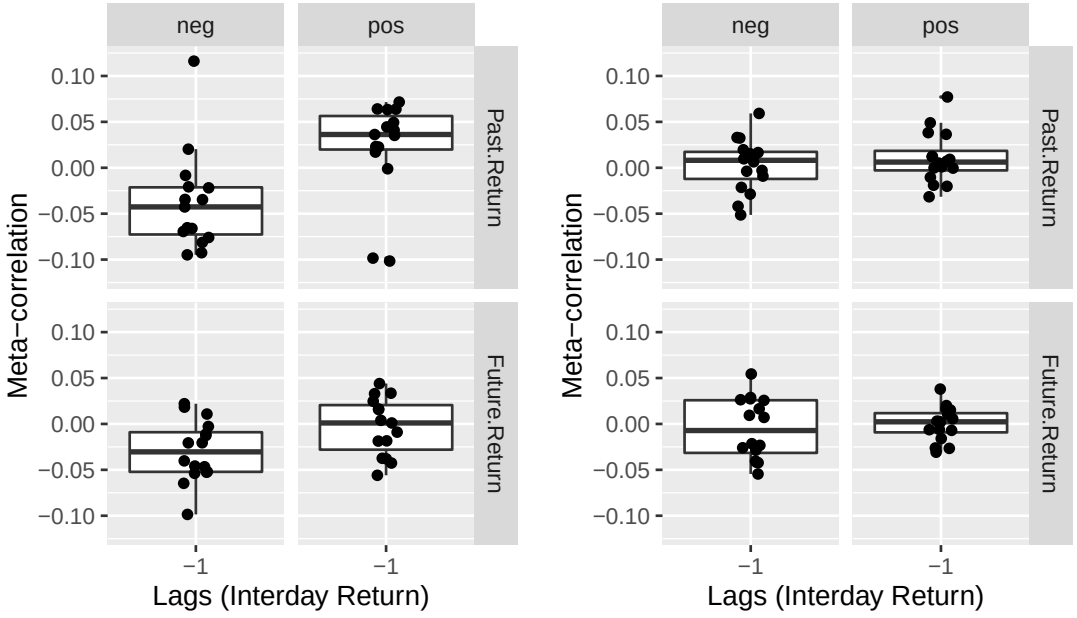
The two sets of box plots summarise the correlational performances of CA-based sentiment analysis methods when the GI dictionary was used to extract the features for learning. Figure 5.9a shows the results obtained with the ordered dataset; Figure 5.9b shows the results obtained by applying the same procedure on the shuffled dataset.

prospective usage. Further statistical tests reveal that (Table 5.25).

3. A visual comparison between the two sub-plots for the *Negative versus Past Return* configuration in Figure 5.9a and Figure 5.10a seems that sentiment proxies derived from the L&M dictionary predicts past returns more reliably than the sentiment series produced using GI's definitions, as the meta-correlations are more concentrated around -0.05. In comparison, the difference between the two sub-plots for *Positive versus Future Return* does not seem to be significant.

Additional t-tests were conducted to examine both the absolute (i.e. versus baseline) and relative correlational performances of the models trained with using words defined in both dictionaries.

Table 5.25 summarises the t-tests used to verify the absolute effectiveness of the content-analysis-based method when combined with either the GI or L&M dictionary against shuffled baselines. It can be seen from results that the use of either dictionary can produce models that outperform baseline models constructed with the shuffled dataset when predicting past returns with negative sentiment series ($p - value \approx 0.00$ for models trained with GI and $p - value \approx 0.01$ for models trained with L&M). The models trained



(a) Ordered dataset

(b) Shuffled dataset

Figure 5.10: Summary of the correlational performances of models built with Content Analysis using the L&M's sentiment word list (testing phases)

The two sets of box plots summarise the correlational performances of CA-based sentiment analysis methods when using Loughran and McDonald's sentiment word list was used to extract the features. Figure 5.10a shows the results obtained with the ordered dataset; Figure 5.10b shows the results obtained by applying the same procedure on the shuffled dataset.

with the L&M dictionary could also achieve significantly better correlational performances than baseline models when associating future returns with negative sentiment series ($p - value \approx 0.05$); the models trained with the GI dictionary did not perform as well as the one trained with the L&M dictionary on the same task, but the result in this case is quite suggestive ($p - value \approx 0.11$).

Table 5.26 shows the results from a set of t-tests that compare the average meta-correlations achieved by models trained with words defined in the GI dictionary and those achieved by models trained with the words defined in the L&M dictionary. The results show that there is insufficient evidence to reject the null hypothesis that the average meta-correlations achieved by models trained with the two dictionaries are the same. This suggests that no significant difference in the two dictionaries' ability to capture sentiment from financial news were found.

I now move to compare the correlational performances achieved by content-analysis-based methods with those achieved by the learning-based method developed in this thesis. Table 5.27 summarises the results from a series of t-tests that are designed to compare the correlational performances from models trained with word dependencies as features and those obtained by content-analysis-based methods using the words defined in GI and

Return Type	Sent. Class	t-stat.	p-value
Past.Return	pos	-0.32	0.75
Future.Return	pos	-0.56	0.58
Future.Return	neg	-1.65	0.11
Past.Return	neg	-3.78	0.00

(a) GI: ordered versus shuffled

Return Type	Sent. Class	t-stat.	p-value
Past.Return	pos	1.17	0.26
Future.Return	pos	-0.90	0.38
Future.Return	neg	-2.07	0.05
Past.Return	neg	-2.89	0.01

(b) L&M: ordered versus shuffled

Table 5.25: Testing for significant differences between the average meta-correlations achieved by content analysis methods when the ordered dataset was used and that achieved when the shuffled dataset was used.

The null hypothesis in each of the test settings (i.e. rows) is that there is no significant difference between the average meta-correlations achieved by the content-analysis-based method when used in conjunction with the specified dictionary on an ordered dataset and that achieved with the same dictionary applied to the shuffled dataset when predicting the corresponding return type (shown in the 'Return Type' column) using the specified sentiment series (shown in the 'Sentiment Class' column).

Return Type	Sent. Class	t-stat.	p-value
Future.Return	neg	0.64	0.53
Future.Return	pos	0.14	0.89
Past.Return	neg	-0.04	0.97
Past.Return	pos	-1.61	0.12

Table 5.26: Testing for significant differences between the average meta-correlations achieved by the content analysis method when used with the GI dictionary and that when the L&M dictionary was used

The null hypothesis in each of the test settings (i.e. rows) is that there is no significant difference between the average meta-correlations achieved by the content analysis method when used with the GI dictionary and that achieved with the L&M dictionary for predicting the corresponding return type (shown in the 'Return Type' column) using the specified sentiment series (shown in the 'Sentiment Class' column).

L&M as features respectively. From the results of the tests, one can see that the learning method developed in this thesis was able to outperform content-analysis-based methods combined with either dictionary in all tasks, except for when predicting past returns with negative sentiment series.

5.5.4.2 Summary

In this experiment, I explored the correlational performances of content-analysis-based methods for predicting past and future returns when used in conjunction with the GI and the L&M sentiment dictionaries; I also compared the correlational performances achieved by content-analysis-based methods with that achieved by the temporal sentiment learning method developed in this thesis. My findings are:

1. For both the GI and the L&M dictionaries, the sentiment series derived from negative word counts predicted past and future returns better than the baselines,

Return Type	Sent. Class	t-stat.	p-value
Future.Return	propos	6.60	0.00
Past.Return	retpos	6.14	0.00
Past.Return	retneg	-1.49	0.15
Future.Return	proneg	-3.18	0.00

(a) Temporal Sentiment Models versus GI

Return Type	Sent. Class	t-stat.	p-value
Future.Return	propos	5.05	0.00
Past.Return	retpos	3.12	0.00
Past.Return	retneg	-1.25	0.22
Future.Return	proneg	-2.14	0.04

(b) Temporal Sentiment Models versus L&M

Table 5.27: Testing for significant differences between the average meta-correlations achieved by temporal sentiment models and that achieved by content analysis methods used with GI and L&M dictionaries

The null hypothesis in each of the test settings (i.e. rows) is that there is no significant difference between the average meta-correlations achieved by the temporal sentiment learning method developed in this thesis and that achieved by content-analysis-based method when used with the GI or L&M dictionary for predicting the corresponding return type (shown in the 'Return Type' column) using the specified sentiment series (shown in the 'Sentiment Class' column). Note that since content-analysis-based methods produce only contemporaneous sentiment series (i.e. no distinction between 'retrospective' and 'prospective' sentiment), the positive sentiment series produced by the content-analysis-based methods served as both the 'retrospective positive' and the 'prospective positive' series during the comparisons; likewise, the negative sentiment series produced by the content-analysis-based methods served as both the 'retrospective negative' and the 'prospective negative' series during the comparisons.

whereas the correlations between positive sentiment series and the movements of the market are statistically insignificant.

2. For both the GI and the L&M dictionaries, the negative sentiment series predicted past returns better than future returns. This could be due to biases in the designs of the dictionaries (e.g. the L&M dictionary registers about 7 times more negative words than positive words); the difference may also indicate that retrospective usage of language is in general more prevalent than prospective usage of language.

3. The negative sentiment series derived from the words defined in the L&M dictionary correlated with past returns more strongly than with the sentiment series derived from the words defined in the GI dictionary. This may be due to the fact that the L&M dictionary was originally built to target texts released under business and financial contexts (i.e. analysing 10-K annual reports filed by listed companies, which summarise the companies' financial performances in the previous year), and thus paid additional attention to the accuracy of retrospective language patterns.

4. On average, the supervised EM learning algorithm developed in this thesis outperformed content-analysis-based methods when used together with either the GI or the L&M dictionary in terms of their correlational performances.

While it had been demonstrated that the algorithms and systems developed in this thesis outperformed content-analysis-based methods in predicting firm-level stock price movements, it should be recognised that content-analysis-based methods have their own

advantages. Unlike machine-learning-based methods which require large volumes of data to operate, small and domain-specific dictionaries used in content-analysis-based methods can be compiled relatively easily when expert knowledge is available. Automatic learning of sentiment-laden language patterns can complement dictionary-based content analysis methods by providing an approach to enable fast extension of specialised dictionaries using domain corpus; such dictionaries may then be used to override the general sentiment dictionaries such as the GI or improve existing manually complied specialised dictionaries.

5.6 Summary of Results

In this case study, I first explored the behaviour of the supervised EM learning algorithm under various settings. It has been shown that the algorithm was able to learn word distributions that reflect the sentiment connotations of lexical items in terms of their impacts on or responses to firm-level stock price movements (Experiment 1). It was also understood that the supervised EM algorithm is susceptible to the problem of over-fitting — the algorithm can be so aggressive in establish associations that it will find correlations where none existed. To overcome this issue, I evaluate the correlational performances of the trained models not on absolute baselines (i.e. neutral correlations), but on 'background correlations' that are achievable by the same model trained on shuffled datasets. It was shown in Experiment 1 that the method developed in this thesis was able to produce models that outperform the baselines when correlating predicted sentiment with returns. This suggests that the method developed in this thesis is effective in learning sentiment orientations of language patterns in business and financial contexts. It was also found that better correlational performances could be achieved when stock returns from the previous days were used to supervise the learning. This suggest that it is often the movements of stock prices that influence news sentiment rather than the other way around.

When the temporal aspects of sentiment were introduced into the models, it was found that the retrospective usage of language is more prevalent than prospective usage of language in business news reporting (Experiment 2). Modelling news articles as a mixture of retrospective and prospective sentiment, however, neither improved nor diminished the correlational performances of the models compared to those trained only with contemporaneous returns.

Another question raised at the beginning of the case study was whether word dependencies could capture sentiment and temporality in the news text better than unigrams. In Experiment 3, it was found that no significant differences were observed when comparing the correlational performances achieved by models trained over word dependencies and models trained with unigrams, except for cases where prospective sentiment series were used to predict future returns of the stock prices. When predicting future returns of

firm-level stocks, models trained using unigrams as features outperformed models trained over word dependencies. It was also noted that, in such cases, models trained with only verbs and adjectives performed as well as models trained with unigrams of all part-of-speech classes, which suggests verbs and adjectives are the main contributors of language sentiment in English. In additional, it was argued that much of sentiment connotations for retrospective word dependencies comes from their verbal component. Lastly, it was recognised that while word dependencies lead to inferior correlational performances when used to predict future returns, using them as features tend to produce in more interpretable language pattern distributions for prospective sentiment classes.

In the last experiment, I compared the correlational performances between the sentiment learning algorithms with content-analysis-based sentiment classification method (Experiment 4). Two popular sentiment dictionaries were tested. One is the General Inquirer affect dictionary (abbreviated as GI) and the other is the Loughran and McDonald's sentiment word list (abbreviated as L&M). Meta-analysis of correlations revealed that content analyses using either dictionary yielded was able to produce sentiment series that correlates significantly with past returns, reaffirming the findings from previous studies by Tetlock [2007]. In addition, the L&M sentiment word list was able to capture retrospective sentiment more reliably than the GI dictionary; this may be due to the fact that the L&M dictionary was specially designed to analyse retrospective materials (i.e. 10-K forms). The correlational performances obtained from content analysis methods using either two dictionary was compared with that achieved by models learnt using the supervised EM algorithm developed in this thesis. It was noted that the models learnt by the methods developed in this thesis were able to achieve superior correlational performances than that content-analysis-based methods when used with either dictionary.

Chapter 6

Conclusion and Future Work

6.1 Concluding Remark

In this thesis, I introduced a new notion called *sentiment temporality* to the traditional modelling of textual sentiment in business and financial news. Traditional methods that explore the interactions between news sentiment and market behaviours have attempted to establish a causal link between news sentiment and market behaviours. This study builds on the existing methods and hypothesise that it is possible to distinguish between two types of news sentiment: (i) the retrospective news sentiment, which refers to the textual sentiment that is associated with recounts of past events in the market; (ii) the prospective news sentiment, which refers to the textual sentiment in business and financial news that is associated with speculations and projections about the future developments of the market. The news sentiment space is therefore expanded to contain four possible values: *retrospective negative*, *retrospective positive*, *prospective negative*, and *prospective positive*.

It was argued in this thesis that the dynamics between news sentiment and the market can be better captured through the introduction of this new notion of news sentiment temporality. I theorise that retrospective news sentiment should pose limited effects on the future development of the market due to the fact that (i) it reflects information that had already been in the prices, and (ii) the market is so efficient to the extent that past prices cannot be used to predict future prices. Prospective news sentiment, on the other hand, should be largely independent from historical prices; instead, it may have a potential impact on the future movements of the market.

The main contribution of this research is a method for the automatic discovery of linguistic realisations of the different types of temporal news sentiment from raw business and financial news texts. In this method, it is assumed that (i) each of the four temporal sentiment classes is characterised by a distribution over lexical items, which represent the linguistic realisations of the corresponding temporal sentiment; (ii) an news article can be

modelled as a mixture of the four lexical item distributions; (iii) if a news article expresses primarily retrospective sentiment, it should be labelled using the market performances in its past; similarly, if a news article expresses primarily prospective sentiment, then it should be labelled using the market performances in its future. To infer the distributions of lexical items for each temporal sentiment class, the method adopts a supervised expectation maximisation algorithm, where the supervision is enforced by market data such as stock and index returns.

Another contribution from this thesis is a software system that implements the method. This system is comprised of two components. The first component is a general text analytic platform named CiCui, which uses Stanford CoreNLP toolkit to prepares raw documents with natural language processing techniques (e.g. tokenisation, part-of-speech tagging, dependency parsing, etc.) and transform a corpus into a positional inverted index. The second component, TSMiner, is a workflow implemented in the KNIME analytic platform. The TSMiner system takes the index built by CiCui and carries out the actual computations as specified by the algorithms.

Finally, a case study was conducted utilising the system to answer the research questions raised at the beginning of the thesis. The case study explored firm-level interactions between temporal news sentiment and the stock prices using a corpus containing business news articles on 16 of the S&P 500 companies were collected.

In general, it was concluded that the method developed in this thesis is indeed capable of extracting the linguistic realisations for the different types of temporal news sentiment. The extracted distributions can have many applications. It may be used to quickly gain insight into the operation of specific companies, or be used to automatically compile domain-specific sentiment dictionaries for use with content analysis.

Further analysis found there is significant evidence that it is the changes in the market that led to the changes in news sentiment. This is evidenced by the fact that the usage of language patterns associated with retrospective sentiment is more prevalent than those associated with prospective sentiment. Additional experiments were conducted to compare the effectiveness of two lexical structures, word dependencies and unigrams, in their abilities to capture temporal news sentiment. Analysis showed that the word dependencies seem to be able capture prospective sentiment better than unigrams, while models trained from unigrams were able to produce sentiment predictions for unseen documents that better correlated with returns. It was also found that the sentiment models produced by the method developed in this thesis slightly outperforms content analysis methods using either the GI or the L&M dictionary in terms of their ability to predict news sentiment for unseen news articles.

6.2 Limitations and Future Work

6.2.1 Interdependency between Sentiment

A major assumption made in this thesis regarding the relationship between business news and market behaviour is that the news articles are all considered independently in terms of their impact on the market — that is, it was assumed:

1. that news articles published on the same date are independent. More specifically, the sentiment conveyed in the text of a news article is responsible for all the price changes happened in the day it was published, disregarding the fact that other news (if any) released on the same day may also influence the market. Also, the sentiment signals predicted for each of the news articles published on the same day are aggregated in a linear fashion; in reality, it is conceivable that the marginal contribution an additional news article makes to the overall sentiment of that day will decreases (e.g. when two news articles are published on a certain story on the same day or in a short time window, the second article may impact the overall sentiment less than the first news article).

2. that the news articles published on consecutive days are independent, namely, the news sentiment today will have no effect on the news sentiment tomorrow. In reality this may not always be the case; a series articles may cover the same story for several days or even weeks. That being said, subsequent news coverage may not have the same level of influence on the market than the initial report.

One way to lift the first limitation may be to introduce an additional *daily sentiment* variable into the model. This *daily sentiment* variable would present an aggregation of the day's overall news sentiment, and it describes the *a priori* belief about how probable a news article published on that day would belong to the sentiment classes of interest. The generation of past and future returns shall be governed directly by the *daily sentiment* rather than by the sentiment class of each individual article.

The second limitation can be resolved by introducing *Markov properties* into the model. A system is said to possess Markov property if the conditional probability distribution of the future state of the system depends only on its current state. With the introduction of daily sentiment, it is possible to specify in the model such interday dependency between daily sentiments from consecutive days.

The theoretical aspects of the above two solutions may be specified using *Bayesian hierarchical modelling*, or more generally, *Bayesian networks*. The computational aspects of the solution, which involves inference over complex Bayesian networks, is typically approached through Markov Chain Monte Carlo methods.

6.2.2 Deriving EM Estimators for Stable Distributions

Another problem with the current specification of the algorithm is that the parameters for the stable distributions which govern the 'generation' of past and future returns from sentiment have to be determined manually beforehand. While this practice does reflect the 'supervised' nature of the algorithm, it can cause a number of problem. Specifically, as Table 5.6 has shown, the distributional properties of the stock returns vary significantly among different companies; using a single set of distributions to supervise the learning of sentiment for all companies will lead to reasonable solutions, but the estimated parameters (e.g. vocabulary distributions) will not be optimal.

One solution would be to manually examine the distribution properties of the return series for each company and determine the parameters for the distributions using heuristics. A better solution, however, would require the maximum likelihood estimators for the parameters be derived analytically. Literature on the derivation of maximum likelihood estimators for stable distributions is sparse [DuMouchel, 1973, Nolan, a,b]. Theoretically, it is possible to approximate the posterior distributions for the parameters within the Bayesian inference framework using Markov Chain Monte Carlo methods (e.g. through *Metropolis-Hastings* sampling, as the p.d.f. for stable distribution can be derived from its characteristic functions for specific cases), but the computational cost for large dataset may prove prohibiting.

6.2.3 Higher Data Frequency

The rate at which news announcements are absorbed into the prices is important as well. Past studies have suggested that news is absorbed in stock prices in a matter of hours [Muntermann and Guettler, 2007, Koppel and Shtrimberg, 2004]. The results from the experiments have suggested that prospective sentiment at daily level is very weak when compared to retrospective sentiment. It is hoped that by exploiting high frequency news and market data, the algorithm will be able to reveal more prospective usages of the language.

The difficulty in moving to high frequency dataset is mainly that such data is not widely available. High frequency market data (e.g. 5-minute or even bid-ask price data) can be acquired or purchased from credible sources, yet the stories published on most newspapers are often delayed for hours if not days; whatever effect the news had would have already been absorbed into the price. One possible news source for this purpose would be commercial newswires (e.g. Business Wire, Marketwire, PR Newswire, etc.). Since these agencies usually have access to first-hand information and also profit on the time-sensitivity of their news stories, it is likely that they would release news more

promptly than newspapers[1]. Social media platforms such as Twitter and Facebook can also be used as real-time news source. Certain literature has investigated the relation between the sentiment conveyed in Twitter messages and the performance of stock market [e.g. Bollen et al., 2011, Zhang et al., 2011, Oliveira et al., 2013, Qasem et al., 2015][2].

6.2.4 Back-testing

Systems that implement the methods developed in this thesis for trading purposes are likely to achieve prediction performances lesser that presented in Chapter 5 due to the nature of statistical machine learning. While the repeated random cross-validation procedure is useful for evaluation purposes, it hardly reflects what happens in real world trading scenarios. In many cases, incidents that sway the market can be unprecedented and unique, rendering statistical learning based on such events ineffective; even the occurrences from recurring events (e.g. singing contract, employee strike, merges and acquisition, etc.) may differ from each other in profound ways, making it difficult for the learning algorithm to track correlation patterns.

However, it would be interesting to see a trading strategy can be constructed exploiting the news sentiment extracted by the methods developed in this thesis — the effectiveness of the method can be further validated if such strategy is able to achieve improved performance compared to benchmarks.

[1] In fact, newspapers and other news media sometimes directly purchase first-hand news from commercial newswire services.

[2] While the statistical methods used and claims made in the study by Bollen et al. [2011] had been widely criticised and thus deemed controversial, it is still noted here for comprehensiveness.

Appendices

Appendix A

Derivation of The EM Algorithm

This appendix aims to explain the mechanism that underlies the expectation maximisation algorithm. The discussions in this chapter are mainly inspired by Minka, Neal and Hinton [1999], Bishop [2007] as well as a number of other sources. The mathematical notations used in this chapter inherits that used in Section 3.3. For the sake of convenience, some of the notations developed in Section 3.3 are reiterated here:

$\mathbf{X}$	the incomplete dataset containing only the observations
$\mathbf{Z}$	the hidden values, which governs the 'generation' of the observations
$\mathbf{X}, \mathbf{Z}$	the complete dataset with the observations and hidden values combined
$\boldsymbol{\theta}$	the parameters that govern the model of interest

A.1 Motivation

The overall goal of the algorithm is to find the $\boldsymbol{\theta}$ that maximises the likelihood function $p(\mathbf{X} \mid \boldsymbol{\theta})$ — that is, the maximum likelihood estimator for $\boldsymbol{\theta}$. The challenge here is that direct optimisation of the likelihood function $p(\mathbf{X} \mid \boldsymbol{\theta})$ with respect to the parameters may not yield closed-form solutions as was shown for the case of Gaussian mixture model. To see this, consider the example of a mixture distribution comprised of multivariate Gaussian distributions. The log-likelihood function for a sequence of identically and independently distributed observations $\mathbf{X}$ drawn from a such a mixture can be written as:

$$\ln p(\mathbf{X} \mid \boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \ln \prod_{n=1}^{N} \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}_n \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \tag{A.1}$$

$$= \sum_{n=1}^{N} \ln \left\{ \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}_n \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \right\} \tag{A.2}$$

where $\boldsymbol{\pi}_k$, $\boldsymbol{\mu}_k$ and $\boldsymbol{\Sigma}_k$ are the weight, mean vector and variance matrix for the k-th constituent Gaussian distribution respectively; N and K are the total number of observations and mixture components respectively. The probability density function for the k-th

Gaussian distribution, namely, $\mathcal{N}(\mathbf{x}_n \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$, is defined as:

$$\mathcal{N}(\mathbf{x}_n \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) = \frac{1}{(2\pi)^{D/2}} \frac{1}{|\boldsymbol{\Sigma}_k|^{1/2}} \exp\left\{-\frac{1}{2}(\mathbf{x}_k - \boldsymbol{\mu}_k)^{\mathrm{T}} \boldsymbol{\Sigma}_k^{-1}(\mathbf{x}_k - \boldsymbol{\mu}_k)\right\} \tag{A.3}$$

To obtain the maximum likelihood estimator for $\boldsymbol{\mu}_k$, first take the derivative of Equation A.2 with respect to $\boldsymbol{\mu}_k$:

$$\frac{\partial \ln p(\mathbf{X} \mid \boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma})}{\partial \boldsymbol{\mu}_k} = \frac{\partial \sum_{n=1}^{N} \ln\left\{\sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}_n \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)\right\}}{\partial \boldsymbol{\mu}_k} \tag{A.4}$$

$$= \sum_{n=1}^{N} \frac{\partial \ln\left\{\sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}_n \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)\right\}}{\partial \boldsymbol{\mu}_k} \tag{A.5}$$

$$= \sum_{n=1}^{N} \frac{1}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\mathbf{x}_n \mid \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)} \frac{\partial \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}_n \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\partial \boldsymbol{\mu}_k} \tag{A.6}$$

$$= \sum_{n=1}^{N} \frac{1}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\mathbf{x}_n \mid \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)} \frac{\partial \pi_k \mathcal{N}(\mathbf{x}_n \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\partial \boldsymbol{\mu}_k} \tag{A.7}$$

$$= \sum_{n=1}^{N} \frac{\pi_k \mathcal{N}(\mathbf{x}_n \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\mathbf{x}_n \mid \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)} \frac{\partial -\frac{1}{2}(\mathbf{x}_k - \boldsymbol{\mu}_k)^{\mathrm{T}} \boldsymbol{\Sigma}_k^{-1}(\mathbf{x}_k - \boldsymbol{\mu}_k)}{\partial \boldsymbol{\mu}_k} \tag{A.8}$$

$$= \sum_{n=1}^{N} \frac{\pi_k \mathcal{N}(\mathbf{x}_n \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\mathbf{x}_n \mid \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)} \boldsymbol{\Sigma}_k^{-1}(\mathbf{x}_n - \boldsymbol{\mu}_k) \tag{A.9}$$

The transition from Equation A.6 to equation A.7 exploited the fact that all the terms in the sum except for $\pi_k \mathcal{N}(\mathbf{x}_n \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$ become zero after the differentiation since they do not contain $\boldsymbol{\mu}_k$. This observation will be used again when deriving the maximum likelihood estimators for the temporality-enhanced model in later sections.

Set Equation A.9 to zero:

$$0 = \sum_{n=1}^{N} \frac{\pi_k \mathcal{N}(\mathbf{x}_n \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\mathbf{x}_n \mid \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)} \boldsymbol{\Sigma}_k^{-1}(\mathbf{x}_n - \boldsymbol{\mu}_k) \tag{A.10}$$

and trying to solve for $\boldsymbol{\mu}_k$ would yield:

$$\boldsymbol{\mu}_k = \frac{1}{N_k} \sum_{n=1}^{N} \frac{\pi_k \mathcal{N}(\mathbf{x}_n \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\mathbf{x}_n \mid \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)} \mathbf{x}_n \tag{A.11}$$

where

$$N_k = \sum_{n=1}^{N} \frac{\pi_k \mathcal{N}(\mathbf{x}_n \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\mathbf{x}_n \mid \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)} \tag{A.12}$$

The maximum likelihood estimators for $\boldsymbol{\Sigma}_k$ and π_k can be derived in a similar fashion:

$$\boldsymbol{\Sigma}_k = \frac{1}{N_k} \sum_{n=1}^{N} \frac{\pi_k \mathcal{N}\left(\mathbf{x}_n \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\right)}{\sum_{j=1}^{K} \pi_j \mathcal{N}\left(\mathbf{x}_n \mid \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j\right)} \left(\mathbf{x}_n - \boldsymbol{\mu}_k\right) \left(\mathbf{x}_n - \boldsymbol{\mu}_k\right)^{\mathrm{T}} \tag{A.13}$$

$$\pi_k = \frac{N_k}{N} \tag{A.14}$$

None of the expressions A.11, A.13, and A.14 do not constitute a closed-form solution to the parameters due to (i) their inter-dependency between the parameters through the $\frac{\pi_k \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}\left(\mathbf{x}_n|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j\right)}$ term which is commonly referred to in the literature as the *responsibility* mixture component k takes when generating the observation $\mathbf{x}_n$; this term will be denoted by γ_{nk} hereafter; (ii) the lack of closed-form solution to equations of form $x = e^x$.

In the case of mixture models, the complexity deriving the maximum likelihood estimator may be sometimes reduced significantly by introducing a hidden variable $\mathbf{z}$ for each observation $\mathbf{x}$. The variable governs which component distribution has actually generated the observation $\mathbf{x}$. $\mathbf{z}$ takes binary vectors of dimension K as its values; among the K elements in $\mathbf{z}$, exactly one of them equals 1 and all the other elements are 0 (i.e. $z_k \in \{0, 1\}$ and $\sum_{i=1}^{K} z_k = 1$, where z_k denotes the k-th element in $\mathbf{z}$); $\mathbf{z}$ can be therefore seen as an indicator random variable with K possible states; also, $\mathbf{z}$ is defined in such a way that:

$$p\left(z_k = 1\right) = \pi_k \tag{A.15}$$

In other words, the value of $\mathbf{x}$ is generated from the k-th component of the mixture if and only if the k-th element in $\mathbf{z}$ equals 1.

With the newly defined variable $\mathbf{x}$, it is time to revisit the maximisation problem of the log-likelihood function for mixture models. Using basic identities of probability, the log-likelihood function of a model whose parameters are denoted by $\boldsymbol{\theta}$ can be expanded as follows:

$$\ln p\left(\mathbf{X} \mid \boldsymbol{\theta}\right) = \ln \left\{ \sum_{\mathbf{Z}} p\left(\mathbf{X}, \mathbf{Z} \mid \boldsymbol{\theta}\right) \right\}$$

where $\mathbf{Z}$ represents one particular realisation of the K^N possible configurations of the hidden states for the entire dataset; K is the total number of components in the mixture, N is the total number of observations in $\mathbf{X}$. As have been shown earlier, the difficulty with the optimisation of this function arises because the summation over all the possible configurations of $\mathbf{Z}$ prevents the logarithm from being applied directly on the likelihood functions for each individual components. Now suppose that the particular configuration of the hidden variables for each observation is known, then it is possible to rid

of the summation over $\mathbf{Z}$ and work directly on the log-likelihood of the complete-data (i.e. $\{\mathbf{X}, \mathbf{Z}\}$) $\ln p(\mathbf{X}, \mathbf{Z} \mid \boldsymbol{\theta})$. Take the previously discussed Gaussian mixture as an example, the maximum likelihood estimates for a particular component's parameters can be derived by first collecting all the observations that was 'generated' by that component and then applying the usual maximum likelihood estimators to those observations only, while the prior probability distribution over the components can be derived simply by computing the relative percentages of the observations assigned to each of the components.

In practice, however, the particular configuration of the hidden variable that generated $\mathbf{x}$ is unknown, so it is not possible to work directly on the complete-data log-likelihood function. The *expectation maximisation* algorithm overcomes this problem by iteratively approximating a local maximum of the incomplete-data log-likelihood function (i.e. $\ln p(\mathbf{X} \mid \boldsymbol{\theta})$). The algorithm alternates between two steps: the E step (expectation step) and the M step (maximisation step), thus the name. In the E step, the algorithm prepares the expected complete-data log-likelihood with respect to the posterior distribution for $\mathbf{Z}$ given the data and the parameters from the previous iteration, namely, $p\left(\mathbf{Z} \mid \mathbf{X}, \boldsymbol{\theta}^{\text{old}}\right)$. More specifically, the expectation of the complete-data log-likelihood, often denoted $\mathcal{Q}\left(\boldsymbol{\theta}, \boldsymbol{\theta}^{\text{old}}\right)$, is defined by

$$\mathcal{Q}\left(\boldsymbol{\theta}, \boldsymbol{\theta}^{\text{old}}\right) = \sum_{\mathbf{Z}} p\left(\mathbf{Z} \mid \mathbf{X}, \boldsymbol{\theta}^{\text{old}}\right) \ln p(\mathbf{X}, \mathbf{Z} \mid \boldsymbol{\theta}) \tag{A.16}$$

The subsequent M step involves the maximisation of this expectation with respect to the parameters in the model:

$$\boldsymbol{\theta}^{\text{new}} = \underset{\boldsymbol{\theta}}{\operatorname{argmax}}\, \mathcal{Q}\left(\boldsymbol{\theta}, \boldsymbol{\theta}^{\text{old}}\right) \tag{A.17}$$

The updated maximum likelihood estimates $\boldsymbol{\theta}^{\text{new}}$ become the $\boldsymbol{\theta}^{\text{old}}$ for the next E step.

A.2 Principle of the EM algorithm

The basic principle of the EM algorithm is to iteratively construct a lower bound for the likelihood function $p(\mathbf{X} \mid \boldsymbol{\theta})$ and improving it until the $\boldsymbol{\theta}$ that maximises the lower bound function would also locally maximises the target likelihood function.

The lower bound of the original likelihood function $p(\mathbf{X} \mid \boldsymbol{\theta})$ is derived through the use of *Jensen's inequality*. The Jensen's inequality states that, for a real continuous *concave* function f, there is:

$$f\left(\sum_{x \in \mathbf{X}} p(x)\, g(x)\right) \geq \sum_{x \in \mathbf{X}} p(x)\, f(g(x)) \tag{A.18}$$

provided that

$$p(x) > 0 \text{ and } \sum_{x \in \mathbf{X}} p(x) = 1 \tag{A.19}$$

Note that A.19 implies that function $p(\cdot)$ defines a probability distribution over x. For the derivation of the EM algorithm, consider the case where $f(\cdot)$ corresponds to the natural logarithm function $\ln(\cdot)$ while $g(x)$ has a positive range:

$$\ln \sum_{x \in \mathbf{X}} p(x)\, g(x) \geq \sum_{x \in \mathbf{X}} p(x) \ln g(x) \tag{A.20}$$

The inequality in A.20 has an intuitive interpretation. Pulling the logarithm function out from the summation in the right hand side of the inequality:

$$\ln \sum_{x \in \mathbf{X}} p(x)\, g(x) \geq \ln \prod_{x \in \mathbf{X}} g(x)^{p(x)} \tag{A.21}$$

$$\sum_{x \in \mathbf{X}} p(x)\, g(x) \geq \prod_{x \in \mathbf{X}} g(x)^{p(x)} \tag{A.22}$$

which states that the arithmetic mean is always greater or equal to the geometric mean for a certain dataset. The key insight that relates to the EM algorithm here arises from Equation A.20. More specifically, consider the following decomposition for the log-likelihood function of the incomplete dataset:

$$\ln p(\mathbf{X} \mid \boldsymbol{\theta}) = \ln \sum_{\mathbf{Z}} p(\mathbf{X}, \mathbf{Z} \mid \boldsymbol{\theta}) \tag{A.23}$$

$$= \ln \sum_{\mathbf{Z}} q(\mathbf{Z}) \frac{p(\mathbf{X}, \mathbf{Z} \mid \boldsymbol{\theta})}{q(\mathbf{Z})} \tag{A.24}$$

where $q(\mathbf{Z})$ defines an arbitrary probability distribution over $\mathbf{Z}$. If one corresponds $q(\mathbf{Z})$ and $\frac{p(\mathbf{Z},\mathbf{X}|\boldsymbol{\theta})}{q(\mathbf{Z})}$ to $p(x)$ and $g(x)$ in A.20 respectively, then one can apply Jensen's inequality to the above expression and get:

$$\geq \sum_{\mathbf{Z}} q(\mathbf{Z}) \ln \frac{p(\mathbf{X}, \mathbf{Z} \mid \boldsymbol{\theta})}{q(\mathbf{Z})} \tag{A.25}$$

which constitutes the lower bound for the log-likelihood function.

Note that this lower bound is a function of both $\boldsymbol{\theta}$ and $q(\cdot)$ (i.e. a functional of q) — that is, the value of the lower bound varies not only depending on the value of $\boldsymbol{\theta}$, but also the choice of the distribution $q(\cdot)$ over $\mathbf{Z}$. The lower bound function can be seen as a surface on a 3-dimensional space, which is illustrated by Figure A.1.

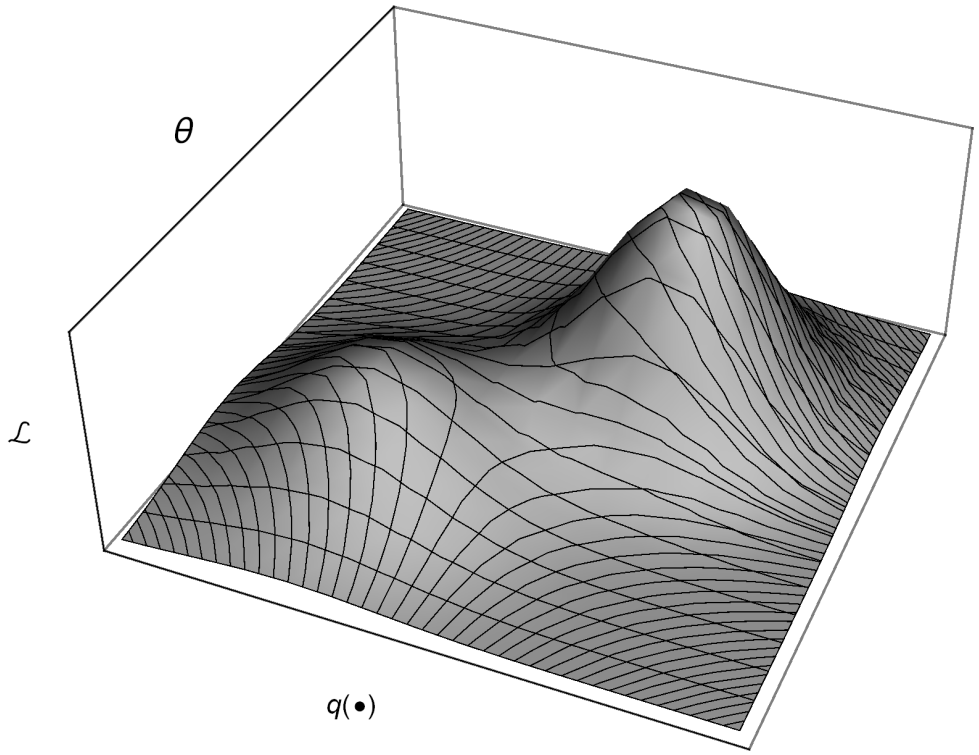


Figure A.1: A schematic visualisation of the lower bound function $\mathcal{L}(\boldsymbol{\theta}, q(\cdot))$ for the log-likelihood function of the incomplete dataset $p(\mathbf{X} \mid \boldsymbol{\theta})$

The x and y axes represent the domains for $q(\cdot)$ and $\boldsymbol{\theta}$ respectively. The paths originate from the $q(\cdot)$ axis visualise schematically the choices of $q(\cdot)$ that corresponds to $p(\mathbf{Z} \mid \mathbf{X}, \boldsymbol{\theta})$ — the paths constituting the mesh on the surface are in fact functions of $\boldsymbol{\theta}$: for different instances of $\boldsymbol{\theta}$, the $q(\cdot)$ which maximises the lower bound function varies; similarly, the $\boldsymbol{\theta}$ that maximises the lower bound also vary based on the $q(\cdot)$ chosen.

The lower bound function A.25 possesses two important properties. First is that for a given $\boldsymbol{\theta}$, the equality holds if and only if $q(\mathbf{Z}) = p(\mathbf{Z} \mid \mathbf{X}, \boldsymbol{\theta})$. This can be shown by optimising the lower bound with a Lagrange multiplier enforcing the constraint that $\sum_{\mathbf{Z}} q(\mathbf{Z}) = 1$ with respect to the $q(\cdot)$ function:

$$\frac{\partial \lambda \left(1 - \sum_{\mathbf{Z}} q(\mathbf{Z})\right) + \sum_{\mathbf{Z}} q(\mathbf{Z}) \ln \frac{p(\mathbf{X}, \mathbf{Z} \mid \boldsymbol{\theta})}{q(\mathbf{Z})}}{\partial q(\mathbf{Z})} = -\lambda + \ln p(\mathbf{X}, \mathbf{Z} \mid \boldsymbol{\theta}) - \ln q(\mathbf{Z}) - 1 \quad \text{(A.26)}$$

Setting the derivative to 0:

$$0 = -\lambda + \ln p(\mathbf{X}, \mathbf{Z} \mid \boldsymbol{\theta}) - \ln q(\mathbf{Z}) - 1 \quad \text{(A.27)}$$

Re-arrange the terms and apply the exponential function to both sides:

$$e^{\lambda+1} q(\mathbf{Z}) = p(\mathbf{X}, \mathbf{Z} \mid \boldsymbol{\theta}) \tag{A.28}$$

Sum over all possible configurations of $\mathbf{Z}$ to elinimate $q(\mathbf{Z})$:

$$e^{\lambda+1} \sum_{\mathbf{Z}} q(\mathbf{Z}) = \sum_{\mathbf{Z}} p(\mathbf{X}, \mathbf{Z} \mid \boldsymbol{\theta}) \tag{A.29}$$

Applying logarithms to both sides:

$$\lambda + 1 = \ln \sum_{\mathbf{Z}} p(\mathbf{X}, \mathbf{Z} \mid \boldsymbol{\theta}) \tag{A.30}$$

Substitue $\lambda + 1$ back into the equation:

$$0 = -\ln \sum_{\mathbf{Z}} p(\mathbf{X}, \mathbf{Z} \mid \boldsymbol{\theta}) + \ln p(\mathbf{X}, \mathbf{Z} \mid \boldsymbol{\theta}) - \ln q(\mathbf{Z}) \tag{A.31}$$

Re-arrange the terms, apply the exponential function on both sides and one reaches the solution for $q(\mathbf{Z})$ that maximises the lower bound function:

$$q(\mathbf{Z}) = \frac{p(\mathbf{X}, \mathbf{Z} \mid \boldsymbol{\theta})}{\sum_{\mathbf{Z}} p(\mathbf{X}, \mathbf{Z} \mid \boldsymbol{\theta})} \tag{A.32}$$

$$= p(\mathbf{Z} \mid \mathbf{X}, \boldsymbol{\theta}) \tag{A.33}$$

which essentially states that the maximal value of the lower bound function for a specific $\boldsymbol{\theta}$ occurs when $q(\mathbf{Z}) = p(\mathbf{Z} \mid \mathbf{X}, \boldsymbol{\theta})$; if both the (hyper-)surfaces corresponding to the lower-bound function A.25 and the likelihood function are plotted, the former shall touch the later when this maximum is achieved (i.e. the equality holds).

The second property is that, if the lower bound function has a local maximum at a particular pair of $\boldsymbol{\theta}$ and $q(\mathbf{Z})$, then the (log-)likelihood function also has a local maximum at $\boldsymbol{\theta}$. This can be shown by noting the following:

1. If the lower bound reaches a local maximum at $(\boldsymbol{\theta}, q(\mathbf{Z}))$, then it must be the case that $q(\mathbf{Z}) = p(\mathbf{Z} \mid \mathbf{X}, \boldsymbol{\theta})$ as has proved by the first property — if $q(\mathbf{Z}) \neq p(\mathbf{Z} \mid \mathbf{X}, \boldsymbol{\theta})$ then there must exist another $\boldsymbol{\theta}, q(\mathbf{Z})$ pair which maximises the lower bound.

2. When the equality holds (i.e. $q(\mathbf{Z}) = p(\mathbf{Z} \mid \mathbf{X}, \boldsymbol{\theta})$), it can be shown that the derivative of the lower bound function with respect to $\boldsymbol{\theta}$ is identical to that of the actual (log-

)likelihood function. For the log-likelihood function for the incomplete dataset:

$$\frac{\partial \ln p\left(\mathbf{X} \mid \boldsymbol{\theta}\right)}{\partial \boldsymbol{\theta}} = \frac{\partial \ln \sum_{\mathbf{Z}} p\left(\mathbf{X}, \mathbf{Z} \mid \boldsymbol{\theta}\right)}{\partial \boldsymbol{\theta}} \tag{A.34}$$

$$= \frac{\sum_{\mathbf{Z}} \frac{\partial p\left(\mathbf{X}, \mathbf{Z} \mid \boldsymbol{\theta}\right)}{\partial \boldsymbol{\theta}}}{\sum_{\mathbf{Z}} p\left(\mathbf{X}, \mathbf{Z} \mid \boldsymbol{\theta}\right)} \tag{A.35}$$

$$= \frac{\sum_{\mathbf{Z}} p\left(\mathbf{X}, \mathbf{Z} \mid \boldsymbol{\theta}\right) \frac{\partial \ln p\left(\mathbf{X}, \mathbf{Z} \mid \boldsymbol{\theta}\right)}{\partial \boldsymbol{\theta}}}{\sum_{\mathbf{Z}} p\left(\mathbf{X}, \mathbf{Z} \mid \boldsymbol{\theta}\right)}. \tag{A.36}$$

$$= \sum_{\mathbf{Z}} p\left(\mathbf{Z} \mid \mathbf{X}, \boldsymbol{\theta}\right) \frac{\partial \ln p\left(\mathbf{X}, \mathbf{Z} \mid \boldsymbol{\theta}\right)}{\partial \boldsymbol{\theta}} \tag{A.37}$$

And for the lower bound when $q\left(\mathbf{Z}\right) = p\left(\mathbf{Z} \mid \mathbf{X}, \boldsymbol{\theta}\right)$:

$$\frac{\partial \sum_{\mathbf{Z}} q\left(\mathbf{Z}\right) \ln \frac{p(\mathbf{X}, \mathbf{Z} \mid \boldsymbol{\theta})}{q(\mathbf{Z})}}{\partial \boldsymbol{\theta}} = \sum_{\mathbf{Z}} \frac{\partial \left\{q\left(\mathbf{Z}\right) \ln p\left(\mathbf{X}, \mathbf{Z} \mid \boldsymbol{\theta}\right) - q\left(\mathbf{Z}\right) \ln q\left(\mathbf{Z}\right)\right\}}{\partial \boldsymbol{\theta}} \tag{A.38}$$

$$= \sum_{\mathbf{Z}} q\left(\mathbf{Z}\right) \frac{\partial \ln p\left(\mathbf{X}, \mathbf{Z} \mid \boldsymbol{\theta}\right)}{\partial \boldsymbol{\theta}} \tag{A.39}$$

$$= \sum_{\mathbf{Z}} p\left(\mathbf{Z} \mid \mathbf{X}, \boldsymbol{\theta}\right) \frac{\partial \ln p\left(\mathbf{X}, \mathbf{Z} \mid \boldsymbol{\theta}\right)}{\partial \boldsymbol{\theta}}. \tag{A.40}$$

In other words, when the equality holds, the tangent of the lower bound function equals that of the likelihood function's. It then follows that the local maximums/minimums of both functions should occur at the same $\boldsymbol{\theta}$s provided that $q\left(\mathbf{Z}\right) = p\left(\mathbf{Z} \mid \mathbf{X}, \boldsymbol{\theta}\right)$.

3. It follows that when the lower bound reaches a local maximum at $(\boldsymbol{\theta}, p\left(\mathbf{Z} \mid \mathbf{X}, \boldsymbol{\theta}\right))$, the value of the derivative of the (log-)likelihood function with respect to $\boldsymbol{\theta}$ must be the same as that of the lower bound function — which equals zero; this implies that the original likelihood function also reaches its local maximum.

With the two properties combined, the optimisation of the (difficult-to-optimise) log-likelihood function for the incomplete dataset with respect to $\boldsymbol{\theta}$ can be re-formulated into the optimisation of the (easier-to-optimise) lower bound function with respect to $\boldsymbol{\theta}$ and $q\left(\mathbf{Z}\right)$. Traditionally, such multi-variable maximisation can be carried out by taking taking the partial derivatives for each variable, setting them to zero and solve for the roots; however, in our case, taking the first-order partial derivatives and setting them to zero result in a non-linear system, whose closed-form solutions are not available. It is therefore necessary to resort to other optimisation methods — in this case, *coordinate ascent*. Coordinate ascent (or descent, if the target function is to be minimised rather than maximised) attempts to find the local extrema of a function by iteratively updating each

of the variables with values that maximises/minimises the target function while holding the other variables' values fixed, beginning with randomly initialised variable values; every time such a new value is obtained for the variable being updated, it replaces the previous value of that variable and the procedure is repeated for the next variable in line. Formally, the algorithm is described by the following pseudo-code:

1: **procedure** CoordinateAscent(a continuous function f)
2: $t \leftarrow 1$
3: Randomly initialise $\boldsymbol{\theta}_1^t, \boldsymbol{\theta}_2^t, \ldots, \boldsymbol{\theta}_K^t$
4: **repeat**
5: **for** $i \leftarrow 1, K$ **do**
6: $\boldsymbol{\theta}_i^{t+1} \leftarrow \operatorname{argmax}_{\boldsymbol{\theta}_i} f\left(\boldsymbol{\theta}_1^{t+1}, \boldsymbol{\theta}_2^{t+1}, \ldots, \boldsymbol{\theta}_{i-1}^{t+1}, \boldsymbol{\theta}_i, \boldsymbol{\theta}_{i+1}^t, \ldots, \boldsymbol{\theta}_K^t\right)$
7: **end for**
8: $t \leftarrow t + 1$
9: **until** $\boldsymbol{\theta}_1^t, \boldsymbol{\theta}_2^t, \ldots, \boldsymbol{\theta}_K^t$ or $f\left(\boldsymbol{\theta}_1^t, \boldsymbol{\theta}_2^t, \ldots, \boldsymbol{\theta}_K^t\right)$ converges
10: **end procedure**

Specifically to the EM algorithm, there are only two variables (i.e. $K = 2$): $\boldsymbol{\theta}$ and $q(\cdot)$; applying coordinate ascent to the lower bound function gives rise to the E and M steps in each iteration of the algorithm:

E Step Maximise the lower bound function with respect to $q(\mathbf{Z})$:

$$q(\mathbf{Z})^{t+1} = \operatorname*{argmax}_{q(\mathbf{Z})} \mathcal{L}\left(\boldsymbol{\theta}^t, q(\mathbf{Z})\right) \tag{A.41}$$

$$= p\left(\mathbf{Z} \mid \mathbf{X}, \boldsymbol{\theta}^t\right) \tag{A.42}$$

M Step Maximise the lower bound function with respect to $\boldsymbol{\theta}$:

$$\boldsymbol{\theta}^{t+1} = \operatorname*{argmax}_{\boldsymbol{\theta}} \mathcal{L}\left(\boldsymbol{\theta}, q(\mathbf{Z})^{t+1}\right) \tag{A.43}$$

$$= \operatorname*{argmax}_{\theta} \sum_{\mathbf{Z}} q(\mathbf{Z})^{t+1} \ln \frac{p(\mathbf{X}, \mathbf{Z} \mid \boldsymbol{\theta})}{q(\mathbf{Z})^{t+1}} \tag{A.44}$$

It is not difficult to see that the lower bound function (i.e. $\mathcal{L}(\boldsymbol{\theta}, q(\mathbf{Z}))$) of the incomplete dataset is guaranteed to increase monotonously with every iteration of the EM algorithm; as a result, the lower bound will eventually converge at a local maximum, and thus locally maximising the (log-)likelihood function as well, as dictated by the second property described above.

In practice, the maximisation in the M step is usually carried out by taking derivatives of the lower bound function with respect to $\boldsymbol{\theta}$, setting it zero and solving for roots.

Observed that:

$$\frac{\partial \sum_{\mathbf{Z}} q\left(\mathbf{Z}\right)^{t+1} \ln \frac{p(\mathbf{X},\mathbf{Z}|\boldsymbol{\theta})}{q(\mathbf{Z})^{t+1}}}{\partial \boldsymbol{\theta}} = \sum_{\mathbf{Z}} \frac{\partial q\left(\mathbf{Z}\right)^{t+1} \ln p\left(\mathbf{X}, \mathbf{Z} \mid \boldsymbol{\theta}\right)}{\partial \boldsymbol{\theta}} - \frac{\partial q\left(\mathbf{Z}\right)^{t+1} \ln q\left(\mathbf{Z}\right)^{t+1}}{\partial \boldsymbol{\theta}} \tag{A.45}$$

$$= \frac{\partial \sum_{\mathbf{Z}} q\left(\mathbf{Z}\right)^{t+1} \ln p\left(\mathbf{X}, \mathbf{Z} \mid \boldsymbol{\theta}\right)}{\partial \boldsymbol{\theta}} \tag{A.46}$$

meaning that the M step can be simplified as:

$$\boldsymbol{\theta}^{t+1} = \underset{\boldsymbol{\theta}}{\operatorname{argmax}} \sum_{\mathbf{Z}} q\left(\mathbf{Z}\right)^{t+1} \ln p\left(\mathbf{X}, \mathbf{Z} \mid \boldsymbol{\theta}\right) \tag{A.47}$$

$$= \underset{\boldsymbol{\theta}}{\operatorname{argmax}} \sum_{\mathbf{Z}} p\left(\mathbf{Z} \mid \mathbf{X}, \boldsymbol{\theta}^{t}\right) \ln p\left(\mathbf{X}, \mathbf{Z} \mid \boldsymbol{\theta}\right). \tag{A.48}$$

The expression $\sum_{\mathbf{Z}} p\left(\mathbf{Z} \mid \mathbf{X}, \boldsymbol{\theta}^{t}\right) \ln p\left(\mathbf{X}, \mathbf{Z} \mid \boldsymbol{\theta}\right)$ is referred to as the $\mathcal{Q}\left(\boldsymbol{\theta}, \boldsymbol{\theta}^{\text{old}}\right)$ function in Chapter 3.

The EM algorithm that has been developed so far produces *maximum likelihood* estimators for models with hidden variables. It is also possible to derive *maximum a posteriori* estimators (MAP estimators for short) for the models. For MAP estimations, a prior distribution over the model's parameter $p\left(\boldsymbol{\theta}\right)$ is introduced; instead of maximising the likelihood function $p\left(\mathbf{X} \mid \boldsymbol{\theta}\right)$, the algorithm is adapted to maximise the posterior distribution with respect to the parameter $\boldsymbol{\theta}$:

$$\boldsymbol{\theta}_{\text{MAP}} = \underset{\boldsymbol{\theta}}{\operatorname{argmax}}\, p\left(\boldsymbol{\theta} \mid \mathbf{X}\right) \tag{A.49}$$

$$= \underset{\boldsymbol{\theta}}{\operatorname{argmax}} \frac{p\left(\mathbf{X} \mid \boldsymbol{\theta}\right) p\left(\boldsymbol{\theta}\right)}{\sum_{\boldsymbol{\theta}} p\left(\mathbf{X} \mid \boldsymbol{\theta}\right) p\left(\boldsymbol{\theta}\right)} \tag{A.50}$$

$$= \underset{\boldsymbol{\theta}}{\operatorname{argmax}}\, p\left(\mathbf{X} \mid \boldsymbol{\theta}\right) p\left(\boldsymbol{\theta}\right) \tag{A.51}$$

With this view, the parameter of the model (i.e. $\boldsymbol{\theta}$) can itself be considered a random variable. It can be shown that the E step and the M step should be modified for the MAP version of the algorithm accordingly:

E Step Maximise the lower bound function with respect to $q\left(\mathbf{Z}\right)$:

$$q\left(\mathbf{Z}\right)^{t+1} = \underset{q(\mathbf{Z})}{\operatorname{argmax}}\, \mathcal{L}\left(\boldsymbol{\theta}^{t}, q\left(\mathbf{Z}\right)\right) \tag{A.52}$$

$$= p\left(\mathbf{Z} \mid \mathbf{X}, \boldsymbol{\theta}^{t}\right) p\left(\boldsymbol{\theta}^{t}\right) \tag{A.53}$$

M Step Maximise the lower bound function with respect to $\boldsymbol{\theta}$:

$$\boldsymbol{\theta}^{t+1} = \underset{\boldsymbol{\theta}}{\operatorname{argmax}} \mathcal{L}\left(\boldsymbol{\theta}, q\left(\mathbf{Z}\right)^{t+1}\right) \tag{A.54}$$

$$= \underset{\theta}{\operatorname{argmax}} \sum_{\mathbf{Z}} q\left(\mathbf{Z}\right)^{t+1} \ln \frac{p\left(\mathbf{X}, \mathbf{Z} \mid \boldsymbol{\theta}\right) p\left(\boldsymbol{\theta}\right)}{q\left(\mathbf{Z}\right)^{t+1}} \tag{A.55}$$

In the MAP case, the $\mathcal{Q}\left(\boldsymbol{\theta}, \boldsymbol{\theta}^{\text{old}}\right)$ function for the M step would also contain the prior distribution for $\boldsymbol{\theta}$:

$$\mathcal{Q}\left(\boldsymbol{\theta}, \boldsymbol{\theta}^{\text{old}}\right) = \sum_{\mathbf{Z}} p\left(\mathbf{Z} \mid \mathbf{X}, \boldsymbol{\theta}^{t}\right) \left\{\ln p\left(\mathbf{X}, \mathbf{Z} \mid \boldsymbol{\theta}\right) + \ln p\left(\boldsymbol{\theta}\right)\right\}. \tag{A.56}$$

Bibliography

Ahmed Abbasi, Hsinchun Chen, Sven Thoms, and Tianjun Fu. Affect Analysis of Web Forums and Blogs Using Correlation Ensembles. *IEEE Transactions on Knowledge and Data Engineering*, 20(9):1168–1180, 2008. ISSN 1041-4347. doi: 10.1109/TKDE.2008.51.

Basant Agarwal, Soujanya Poria, Namita Mittal, Alexander Gelbukh, and Amir Hussain. Concept-Level Sentiment Analysis with Dependency-Based Semantic Parsing: A Novel Approach. *Cognitive Computation*, 7(4):487–499, 2015. ISSN 1866-9956. doi: 10.1007/s12559-014-9316-6.

Khurshid Ahmad, JingGuang Han, Elaine Hutson, Colm Kearney, and Sha Liu. Media-expressed negative tone and firm-level stock returns. *Journal of Corporate Finance*, 2015. ISSN 09291199. doi: 10.1016/j.jcorpfin.2015.12.014.

Devitt Ann and Ahmad Khurshid. Sentiment Analysis and the Use of Extrinsic Datasets in Evaluation. In Nicoletta Calzolari, Khalid Choukri, Bente Maegaard, Joseph Mariani, Jan Odijk, Stelios Piperidis, and Daniel Tapias, editors, *Proceedings of the 6th International Conference on Language Resources and Evaluation*, Marrakech, Morocco, 2008. European Language Resources Association. ISBN 2-9517408-4-0.

Werner Antweiler and Murray Z. Frank. Is All That Talk Just Noise? The Information Content of Internet Stock Message Boards. *The Journal of Finance*, 59(3):p 1259–1294, 2004. ISSN 00221082. URL `http://www.jstor.org/stable/3694736`.

Stefano Baccianella, Andrea Esuli, and Fabrizio Sebastiani. SentiWordNet 3.0: An Enhanced Lexical Resource for Sentiment Analysis and Opinion Mining. In Nicoletta Calzolari, Khalid Choukri, Bente Maegaard, Joseph Mariani, Jan Odijk, Stelios Piperidis, Mike Rosner, and Daniel Tapias, editors, *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*, Valletta, Malta, 2010. European Language Resources Association. ISBN 2-9517408-6-7.

Xue Bai. Predicting consumer sentiments from online text. *Decision Support Systems*, 50 (4):732–742, 2011. ISSN 01679236. doi: 10.1016/j.dss.2010.08.024.

Malcolm Baker and Jeffrey Wurgler. Investor Sentiment and the Cross-Section of Stock Returns. *The Journal of Finance*, 61(4):1645–1680, 2006. ISSN 00221082. doi: 10.1111/j.1540-6261.2006.00885.x.

Dmitriy Bespalov, Bing Bai, Yanjun Qi, and Ali Shokoufandeh. Sentiment classification based on supervised latent n-gram analysis. In Iadh Ounis, Ian Ruthven, and Craig Macdonald, editors, *Proceedings of the 20th ACM International Conference on Information and Knowledge Management*, page 375, 2011. doi: 10.1145/2063576.2063635.

Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer Berlin Heidelberg, New Delhi, 2007. ISBN 8132209060.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent Dirichlet Allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003. ISSN 1532-4435. URL `http://dl.acm.org/citation.cfm?id=944919.944937`.

Johan Bollen, Huina Mao, and Xiaojun Zeng. Twitter mood predicts the stock market. *Journal of Computational Science*, 2(1):1–8, 2011. ISSN 18777503. doi: 10.1016/j.jocs.2010.12.007.

Matthew Butler and Vlado Kešelj. Financial Forecasting Using Character N-Gram Analysis and Readability Scores of Annual Reports. In David Hutchison, Takeo Kanade, Josef Kittler, Jon M. Kleinberg, Friedemann Mattern, John C. Mitchell, Moni Naor, Oscar Nierstrasz, Pandu C. Rangan, Bernhard Steffen, Madhu Sudan, Demetri Terzopoulos, Doug Tygar, Moshe Y. Vardi, Gerhard Weikum, Yong Gao, and Nathalie Japkowicz, editors, *Advances in Artificial Intelligence*, volume 5549 of *Lecture Notes in Computer Science*, pages 39–51. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009. ISBN 978-3-642-01817-6. doi: 10.1007/978-3-642-01818-3{\textunderscore}7.

Erik Cambria, Bjorn Schuller, Yunqing Xia, and Catherine Havasi. New Avenues in Opinion Mining and Sentiment Analysis. *IEEE Intelligent Systems*, 28(2):15–21, 2013. ISSN 1541-1672. doi: 10.1109/MIS.2013.30.

Mark Cecchini, Haldun Aytug, Gary J. Koehler, and Praveen Pathak. Making words work: Using financial text as a predictor of financial events. *Decision Support Systems*, 50(1): 164–175, 2010. ISSN 01679236. doi: 10.1016/j.dss.2010.07.012.

Hang Cui, Vibhu Mittal, and Mayur Datar. Comparative Experiments on Sentiment Classification for Online Product Reviews. In *Proceedings of the 21th National Conference on Artificial Intelligence: 21th National Conference on Artificial Intelligence (AAAI-06); Eighteenth Innovative Applications of Artificial Intelligence Conference (IAAI-06)*, pages 1265–1270, Menlo Park, Calif, 2006. AAAI Press. ISBN 9781577352815.

Sanjiv R. Das and Mike Y. Chen. Yahoo! For Amazon: Sentiment Extraction from Small Talk on the Web. *Management Science*, 53(9):1375–1388, 2007. ISSN 00251909. URL `http://www.jstor.org/stable/20122297`.

Kushal Dave, Steve Lawrence, and David M. Pennock. Mining the Peanut Gallery: Opinion Extraction and Semantic Classification of Product Reviews. In Gusztáv Hencsey, Bebo White, Yih-Farn Robin Chen, László Kovács, and Steve Lawrence, editors, *Proceedings of the 12th International Conference on World Wide Web*, page 519, [S.l.], 2003. ACM. ISBN 1-58113-680-3. doi: 10.1145/775152.775226.

Angela K. Davis, Jeremy M. Piger, and Lisa M. Sedor. Beyond the Numbers: Measuring the Information Content of Earnings Press Release Language. *SSRN Electronic Journal*, 2011. ISSN 1556-5068. doi: 10.2139/ssrn.875399.

Luigi Di Caro and Matteo Grella. Sentiment Analysis via Dependency Parsing. *Computer Standards & Interfaces*, 35(5):442–453, 2013. ISSN 09205489. doi: 10.1016/j.csi.2012.10.005.

William H. DuMouchel. On the Asymptotic Normality of the Maximum-Likelihood Estimate when Sampling from a Stable Distribution. *The Annals of Statistics*, 1(5):948–957, 1973. ISSN 00905364. URL `http://www.jstor.org/stable/2958296`.

Joseph Engelberg. Costly Information Processing: Evidence from Earnings Announcements. *SSRN Electronic Journal*, 2008. ISSN 1556-5068. doi: 10.2139/ssrn.1107998.

Junqué de Fortuny Enric, Tom de Smedt, David Martens, and Walter Daelemans. Evaluating and understanding text-based stock price prediction models. *Information Processing & Management*, 50(2):426–441, 2014. ISSN 03064573. doi: 10.1016/j.ipm.2013.12.002.

Eugene F. Fama. The Behavior of Stock-Market Prices. *The Journal of Business*, 38:34–105, 1965a.

Eugene F. Fama. Random Walks in Stock Market Prices. *Financial Analysts Journal*, 21(5):55–59, 1965b. ISSN 0015-198X. doi: 10.2469/faj.v21.n5.55.

Eugene F. Fama. Efficient Capital Markets: A Review of Theory and Empirical Work. *The Journal of Finance*, 25(2):383, 1970. ISSN 00221082. doi: 10.2307/2325486.

Ingo Feinerer, Kurt Hornik, and David Meyer. Text Mining Infrastructure in R. *Journal of Statistical Software*, 25(5), 2008. ISSN 1548-7660. doi: 10.18637/jss.v025.i05.

Ronen Feldman, Suresh Govindaraj, Joshua Livnat, and Benjamin Segal. The Incremental Information Content of Tone Change in Management Discussion and Analysis. *SSRN Electronic Journal*, 2008. ISSN 1556-5068. doi: 10.2139/ssrn.1126962.

Nicky J. Ferguson, Dennis Philip, Herbert Y. T. Lam, and Jie Guo. Media Content and Stock Returns: The Predictive Power of Press. *SSRN Electronic Journal*, 2012. ISSN 1556-5068. doi: 10.2139/ssrn.2111352.

Kenneth R. French. Stock returns and the weekend effect. *Journal of Financial Economics*, 8(1):55–69, 1980. ISSN 0304405X. doi: 10.1016/0304-405X(80)90021-5.

Michael Gamon. Sentiment classification on customer feedback data. In *Proceedings of the 20th International Conference on Computational Linguistics*, pages 841–es, 2004. doi: 10.3115/1220355.1220476.

Diego García. Sentiment during Recessions. *The Journal of Finance*, 68(3):1267–1300, 2013. ISSN 00221082. doi: 10.1111/jofi.12027.

Andrew Gelman. *Bayesian Data Analysis*. Chapman & Hall / CRC texts in statistical science. CRC Press, Taylor & Francis Group, Boca Raton, third edition edition, 2014. ISBN 1439840954.

Michel Généreux, Thierry Poibeau, and Moshe Koppel. Sentiment Analysis Using Automatically Labelled Financial News Items. In Khurshid Ahmad, editor, *Affective Computing and Sentiment Analysis*, volume 45 of *Text, Speech, and Language Technology*, pages 101–114. Springer Netherlands, Dordrecht, 2011. ISBN 978-94-007-1756-5. doi: 10.1007/978-94-007-1757-2{\textunderscore}9.

Tomer Geva and Jacob Zahavi. Empirical evaluation of an automated intraday stock recommendation system incorporating both market data and textual news. *Decision Support Systems*, 57:212–223, 2014. ISSN 01679236. doi: 10.1016/j.dss.2013.09.013.

M. Ghiassi, J. Skinner, and D. Zimbra. Twitter brand sentiment analysis: A hybrid system using n-gram analysis and dynamic artificial neural network. *Expert Systems with Applications*, 40(16):6266–6282, 2013. ISSN 09574174. doi: 10.1016/j.eswa.2013.05.057.

Irving John Good. The Population Frequencies Of Species And The Estimation Of Population Parameters. *Biometrika*, 40(3-4):237–264, 1953. ISSN 0006-3444. doi: 10.1093/biomet/40.3-4.237.

Michael Hagenau, Michael Liebmann, and Dirk Neumann. Automated news reading: Stock price prediction based on financial news using context-capturing features. *Decision Support Systems*, 55(3):685–697, 2013. ISSN 01679236. doi: 10.1016/j.dss.2013.02.006.

Vasileios Hatzivassiloglou and Kathy McKeown. Predicting the Semantic Orientation of Adjectives. In *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics (ACL-97)*, pages 174–181, Madrid, Spain, 1997.

Elaine Henry. Are Investors Influenced by How Earnings Press Releases are Written? *SSRN Electronic Journal*, 2006. ISSN 1556-5068. doi: 10.2139/ssrn.933100.

Julian P. T. Higgins, Simon G. Thompson, and David J. Spiegelhalter. A Re-evaluation of Random-effects Meta-analysis. *Journal of the Royal Statistical Society. Series A, (Statistics in Society)*, 172(1):137–159, 2009. ISSN 0964-1998. doi: 10.1111/j.1467-985X.2008.00552.x.

Thomas Hofmann. Probabilistic latent semantic indexing. In Fredric Gey, Marti Hearst, and Richard Tong, editors, *Proceedings of the 22nd Annual International ACM SIGIR Conference*, pages 50–57, 1999. doi: 10.1145/312624.312649.

Minqing Hu and Bing Liu. Mining and Summarizing Customer Reviews. In Won Kim, Ronny Kohavi, Johannes Gehrke, and William DuMouchel, editors, *Proceedings of the 2004 ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, page 168, 2004. doi: 10.1145/1014052.1014073.

Yi Hu, Ruzhan Lu, Yuquan Chen, and Jianyong Duan. Using a Generative Model for Sentiment Analysis. *Computational Linguistics and Chinese Language Processing*, 12 (2):107–128, 2007.

Thorsten Joachims. Text categorization with Support Vector Machines: Learning with many relevant features. In Claire Nédellec and Céline Rouveirol, editors, *Machine learning*, volume 1398 of *Lecture Notes in Computer Science*, pages 137–142. Springer, Berlin and New York, 1998. ISBN 978-3-540-64417-0. doi: 10.1007/BFb0026683.

Karen Sparck Jones. A Statistical Interpretation of Term Specificity and Its Application in Retrieval. In Peter Willett, editor, *Document Retrieval Systems*, The Foundations of information science. Taylor Graham and the Institute of Information Scientists, London, 1988. ISBN 9780947568214.

Jaap Kamps, Maarten Marx, Robert J. Mokken, and Maarten de Rijke. Using WordNet to Measure Semantic Orientations of Adjectives. In *Proceedings of 4th international conference on language resources and evaluation*, volume 4, pages 1115–1118, 2004.

Hiroshi Kanayama and Tetsuya Nasukawa. Fully Automatic Lexicon Expansion for Domain-Oriented Sentiment Analysis. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 355–363, Sydney, Australia, 2006. Association of Computational Linguistics.

Colm Kearney and Sha Liu. Textual sentiment in finance: A survey of methods and models. *International Review of Financial Analysis*, 33:171–185, 2014. ISSN 10575219. doi: 10.1016/j.irfa.2014.02.006.

Alistair Kennedy and Diana Inkpen. Sentiment Classification of Movie Reviews Using Contextual Valence Shifters. *Computational Intelligence*, 22(2):110–125, 2006.

Katia Lida Kermanidis and Manolis Maragoudakis. Political sentiment analysis of tweets before and after the Greek elections of May 2012. *International Journal of Social Network Mining*, 1(3/4):298, 2013. ISSN 1757-8485. doi: 10.1504/IJSNM.2013.059090.

Ahmad Khurshid and Margaret Rogers. Corpus Linguistics and Terminology Extraction. In Sue Ellen Wright and Gerhard Budin, editors, *Handbook of Terminology Management*, pages 725–760. John Benjamins Publishing Company, Amsterdam, 2001. ISBN 978.

Soo-Min Kim and Eduard Hovy. Determining the sentiment of opinions. In *Proceedings of the 20th International Conference on Computational Linguistics*, pages 1367–es, 2004. doi: 10.3115/1220355.1220555.

Soon-Ho Kim and Dongcheol Kim. Investor sentiment from internet message postings and the predictability of stock returns. *Journal of Economic Behavior & Organization*, 2014. ISSN 01672681. doi: 10.1016/j.jebo.2014.04.015.

Dan Klein and Christopher D. Manning. Accurate Unlexicalized Parsing. In Erhard W. Hinrichs and Dan Roth, editors, *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics*, pages 423–430, 2003. doi: 10.3115/1075096.1075150.

Daphne Koller and Nir Friedman. *Probabilistic Graphical Models: Principles and techniques*. Adaptive Computation and Machine Learning. MIT Press, Cambridge, MA, 2009. ISBN 0262013193.

Moshe Koppel and Itai Shtrimberg. Good News or Bad News? Let the Market Decide. In James G. Shanahan, Janyce Wiebe, and Yan Qu, editors, *Proceedings of the AAAI Spring Symposium on Exploring Attitude and Affect in Text: Theories and Applications*, Stanford, US, 2004.

Klaus Krippendorff. *Content analysis: An introduction to its methodology*. SAGE, Los Angeles and London, 3rd ed. edition, 2013. ISBN 1412983150.

Sandra Kübler, Ryan McDonald, and Joakim Nivre. *Dependency parsing*, volume #2 of *Synthesis lectures on human language technologies*. Morgan & Claypool Publishers, [San Rafael, Calif.], 2009. ISBN 1598295977.

Feng Li. The Information Content of Forward-Looking Statements in Corporate Filings-A Naïve Bayesian Machine Learning Approach. *Journal of Accounting Research*, 48(5): 1049–1102, 2010. ISSN 00218456. doi: 10.1111/j.1475-679X.2010.00382.x.

Qing Li, TieJun Wang, Ping Li, Ling Liu, Qixu Gong, and Yuanzhu Chen. The effect of news and public mood on stock movements. *Information Sciences*, 278:826–840, 2014a. ISSN 00200255. doi: 10.1016/j.ins.2014.03.096.

Xiaodong Li, Haoran Xie, Li Chen, Jianping Wang, and Xiaotie Deng. News impact on stock price return via sentiment analysis. *Knowledge-Based Systems*, 69:14–23, 2014b. ISSN 09507051. doi: 10.1016/j.knosys.2014.04.022.

Chenghua Lin and Yulan He. Joint sentiment/topic model for sentiment analysis. In David Cheung, Il-Yeol Song, Wesley Chu, Xiaohua Hu, and Jimmy Lin, editors, *Proceedings of the 18th ACM Conference on Information and Knowledge Management*, page 375, 2009. doi: 10.1145/1645953.1646003.

Bing Liu. Sentiment Anlaysis and Subjectivity. In Nitin Indurkhya and Frederick J. Damerau, editors, *Handbook of natural language processing*, Chapman & Hall/CRC machine learning & pattern recognition series. Chapman & Hall/CRC, Boca Raton, FL, 2010. ISBN 978-1420085921.

Bing Liu. Sentiment Analysis and Opinion Mining. *Synthesis Lectures on Human Language Technologies*, 5(1):1–167, 2012. ISSN 1947-4040. doi: 10.2200/S00416ED1V01Y201204HLT016.

Tim Loughran and Bill McDonald. When is a Liability not a Liability? Textual Analysis, Dictionaries, and 10-Ks. *The Journal of Finance*, 66(1):35–65, 2011a. ISSN 00221082.

Tim Loughran and Bill McDonald. Barron's Red Flags: Do They Actually Work? *SSRN Electronic Journal*, 2011b. ISSN 1556-5068. doi: 10.2139/ssrn.1510188.

Tim Loughran and Bill McDonald. Textual Analysis in Finance and Accounting: A Survey. *SSRN Electronic Journal*, 2014. ISSN 1556-5068. doi: 10.2139/ssrn.2504147.

Christopher D. Manning and Hinrich Schütze. *Foundations of Statistical Natural Language Processing*. MIT Press, Cambridge, Mass., 1999. ISBN 9780262133609.

Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to Information Retrieval*. Cambridge University Press, New York, 2008. ISBN 0521865719.

Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. The Stanford CoreNLP Natural Language Processing Toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics System Demonstrations*, pages 55–60, 2014. URL `http://www.aclweb.org/anthology/P/P14/P14-5010`.

Shotaro Matsumoto, Hiroya Takamura, and Manabu Okumura. Sentiment Classification Using Word Sub-sequences and Dependency Sub-trees. In David Hutchison, Takeo Kanade, Josef Kittler, Jon M. Kleinberg, Friedemann Mattern, John C. Mitchell, Moni Naor, Oscar Nierstrasz, C. Pandu Rangan, Bernhard Steffen, Madhu Sudan, Demetri Terzopoulos, Dough Tygar, Moshe Y. Vardi, Gerhard Weikum, Tu Bao Ho, David Cheung, and Huan Liu, editors, *Advances in Knowledge Discovery and Data Mining*, volume 3518 of *Lecture Notes in Computer Science*, pages 301–311. Springer Berlin Heidelberg, Berlin, Heidelberg, 2005. ISBN 978-3-540-26076-9. doi: 10.1007/11430919{\textunderscore}37.

John H. McDonald. *Handbook of Biological Statistics*. Sparky House Publishing, third edition edition, 2015.

Walaa Medhat, Ahmed Hassan, and Hoda Korashy. Sentiment analysis algorithms and applications: A survey. *Ain Shams Engineering Journal*, 5(4):1093–1113, 2014. ISSN 20904479. doi: 10.1016/j.asej.2014.04.011.

Qiaozhu Mei, Xu Ling, Matthew Wondra, Hang Su, and ChengXiang Zhai. Topic Sentiment Mixture: Modeling Facets and Opinions in Weblogs. In *Proceedings of the 16th International Conference on World Wide Web*, pages 171–180, 2007. ISBN 978-1-59593-654-7.

George A. Miller. WordNet: A lexical database for English. *Communications of the ACM*, 38(11):39–41, 1995. ISSN 00010782. doi: 10.1145/219717.219748.

Thomas Minka. Expectation-Maximization as Lower Bound Maximization. URL `research.microsoft.com/en-us/um/people/minka/papers/em.html`.

Sheung Yin K. Mo, Anqi Liu, and Steve Y. Yang. News Sentiment to Market Impact and its Feedback Effect. *SSRN Electronic Journal*, 2015. ISSN 1556-5068. doi: 10.2139/ssrn.2567433.

Tony Mullen and Nigel Collier. Sentiment analysis using support vector machines with diverse information sources. In *Proceedings of the 9th Conference on Empirical Methods in Natural Language Processing*, 2004.

Jan Muntermann and Andre Guettler. Intraday stock price effects of ad hoc disclosures: The German case. *Journal of International Financial Markets, Institutions and Money*, 17(1):1–24, 2007. ISSN 10424431. doi: 10.1016/j.intfin.2005.08.003.

Kevin P. Murphy. *Machine Learning: A Probabilistic Perspective*. Adaptive Computation and Machine Learning Series. MIT Press, Cambridge, Mass., 2012. ISBN 0262304325.

J. Zvi Namenwirth and Robert Philip Weber. *Dynamics of Culture*. Allen & Unwin, Boston, 1987. ISBN 9780044970378.

Arman Khadjeh Nassirtoussi, Saeed Aghabozorgi, Ying Wah Teh, and David Chek Ling Ngo. Text mining for market prediction: A systematic review. *Expert Systems with Applications*, 41(16):7653–7670, 2014. ISSN 09574174. doi: 10.1016/j.eswa.2014.06.009.

Arman Khadjeh Nassirtoussi, Saeed Aghabozorgi, Ying Wah Teh, and David Chek Ling Ngo. Text mining of news-headlines for FOREX market prediction: A Multi-layer Dimension Reduction Algorithm with semantics and sentiment. *Expert Systems with Applications*, 42(1):306–324, 2015. ISSN 09574174. doi: 10.1016/j.eswa.2014.08.004.

Tetsuya Nasukawa and Jeonghee Yi. Sentiment Analysis: Capturing Favorability Using Natural Language Processing. In John Gennari, Bruce Porter, and Yolanda Gil, editors, *Proceedings of the 2nd International Conference on Knowledge Capture*, page 70, 2003. doi: 10.1145/945645.945658.

Radford Neal and Geoffrey E. Hinton. A View Of The Em Algorithm That Justifies Incremental, Sparse, And Other Variants. In Michael I. Jordan, editor, *Learning in Graphical Models*, Adaptive Computation and Machine Learning, pages 355–368. The MIT Press, 1999. ISBN 9780262600323.

Kamal Nigam, Andrew Kachites Mccallum, Sebastian Thrun, and Tom Mitchell. Text Classification from Labeled and Unlabeled Documents using EM. *Machine Learning*, 39 (2/3):103–134, 2000. ISSN 08856125. doi: 10.1023/A:1007692713085.

John P. Nolan. Modeling Financial Data with Stable Distributions, a.

John P. Nolan. Maximum Likelihood Estimation and Diagnostics for Stable Distributions, b. URL `http://academic2.american.edu/~jpnolan/stable/mle.pdf`.

Nuno Oliveira, Paulo Cortez, and Nelson Areal. Some experiments on modeling stock market behavior using investor sentiment analysis and posting volume from Twitter WIMS '13, Madrid, Spain, June 12-14, 2013. In David Camacho, Rajendra Akerkar, and Mar-Moreno, editors, *Proceedings of the 3rd International Conference on Web Intelligence, Mining and Semantics*, page 31. ACM, 2013. ISBN 978-1-4503-1850-1. doi: 10.1145/2479787.2479811. URL `http://doi.acm.org/10.1145/2479787.2479811`.

Charles Egerton Osgood. *The Measurement of Meaning*. University of Illinois Press, Urbana, 1957. ISBN 9780252745393.

Bo Pang and Lillian Lee. Opinion Mining and Sentiment Analysis. *Foundations and Trends in Information Retrieval*, Vol. 2(1-2):p 1–135, 2008.

Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. Thumbs Up? Sentiment Classification Using Machine Learning Techniques. In Jan Hajič and Yuji Matsumoto, editors, *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing*, pages 79–86. Association of Computational Linguistics, 2002. doi: 10.3115/1118693.1118704. URL `http://www.aclweb.org/anthology/W02-1011`.

Livia Polanyi and Annie Zaenen. Contextual Valence Shifters. In James G. Shanahan, Yan Qu, and Janyce M. Wiebe, editors, *Computing attitude and affect in text*, volume 20 of *The Information Retrieval Series*, pages 1–10. Springer, Dordrecht, 2006. ISBN 1-4020-4026-1. doi: 10.1007/1-4020-4102-0{\textunderscore}1.

Soujanya Poria, Erik Cambria, Grégoire Winterstein, and Guang-Bin Huang. Sentic patterns: Dependency-based rules for concept-level sentiment analysis. *Knowledge-Based Systems*, 69:45–63, 2014. ISSN 09507051. doi: 10.1016/j.knosys.2014.05.005.

Rudy Prabowo and Mike Thelwall. Sentiment analysis: A combined approach. *Journal of Informetrics*, 3(2):143–157, 2009. ISSN 17511577. doi: 10.1016/j.joi.2009.01.003.

Mohammed Qasem, Ruppa Thulasiram, and Parimala Thulasiraman. Twitter sentiment classification using machine learning techniques for stock markets. In Jaime Lloret Mauri, Sabu M. Thampi, Michal Wozniak, Oge Marques, Dilip Krishnaswamy, Sartaj Sahni, Christian Callegari, Hideyuki Takagi, Zoran S. Bojkovic, Vinod M, Neeli R. Prasad, Jose M. Alcaraz Calero, Joal Rodrigues, Xinyu Que, Natarajan Meghanathan, Ravi Sandhu, and Edward Au, editors, *Proceedings of 2015 International Conference on Advances in Computing, Communications and Informatics*, pages 834–840. IEEE, 2015. ISBN 978-1-4799-8790-0. doi: 10.1109/ICACCI.2015.7275714. URL `http://dx.doi.org/10.1109/ICACCI.2015.7275714`.

Sven Rill, Dirk Reinel, Jörg Scheidt, and Roberto V. Zicari. PoliTwi: Early detection of emerging political topics on twitter and the impact on concept-level sentiment analysis. *Knowledge-Based Systems*, 69:24–33, 2014. ISSN 09507051. doi: 10.1016/j.knosys.2014.05.008.

S. W. Roberts. Control Chart Tests Based on Geometric Moving Averages. *Technometrics*, 1(3):239–250, 1959. ISSN 0040-1706. doi: 10.1080/00401706.1959.10489860.

Michael Salter-Townshend and Thomas Brendan Murphy. Mixtures of biased sentiment analysers. *Advances in Data Analysis and Classification*, 8(1):85–103, 2014. ISSN 1862-5347. doi: 10.1007/s11634-013-0150-6.

G. Salton, A. Wong, and C. S. Yang. A vector space model for automatic indexing. *Communications of the ACM*, 18(11):613–620, 1975. ISSN 00010782. doi: 10.1145/361219.361220.

Robert P. Schumaker and Hsinchun Chen. Evaluating a news-aware quantitative trader: The effect of momentum and contrarian stock selection strategies. *Journal of the American Society for Information Science and Technology*, 59(2):247–255, 2008. ISSN 15322882. doi: 10.1002/asi.20739.

Robert P. Schumaker and Hsinchun Chen. A quantitative stock prediction system based on financial news. *Information Processing & Management*, 45(5):571–583, 2009a. ISSN 03064573. doi: 10.1016/j.ipm.2009.05.001.

Robert P. Schumaker and Hsinchun Chen. Textual analysis of stock market prediction using breaking financial news: The AZFinText System. *ACM Transactions on Information Systems*, 27(2):1–19, 2009b. ISSN 10468188. doi: 10.1145/1462198.1462204.

Robert P. Schumaker, Yulei Zhang, Chun-Neng Huang, and Hsinchun Chen. Evaluating sentiment in financial news articles. *Decision Support Systems*, 53(3):458–464, 2012. ISSN 01679236. doi: 10.1016/j.dss.2012.03.001.

Samira Shaikh, Kit Cho, Tomek Strzalkowski, Laurie Feldman, John Lien, Ting Liu, and George Aaron Broadwell. ANEW+: Automatic Expansion and Validation of Affective Norms of Words Lexicons in Multiple Languages. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation*, Paris, France, 2016. European Language Resources Association (ELRA). ISBN 978-2-9517408-9-1.

Robert J. Shiller. *Irrational exuberance*. Princeton University Press, Princeton, NJ, 2000. ISBN 978-0691050621.

John Sinclair. What is a Poem like a Sunset? *A Review of English Literature*, 6(2):76–91, 1965.

John Sinclair. *Corpus, Concordance, Collocation*. Describing English Language. Oxford University Press, Oxford, 1991. ISBN 9780194371445.

John Sinclair. The Lexical Item. In Edda Weigand, editor, *Contrastive Lexical Semantics*, Amsterdam studies in the theory and history of linguistic science. Series IV, Current issues in linguistic theory. J. Benjamins, Amsterdam and Philadelphia, 1998. ISBN 9027275637.

Jasmina Smailović, Miha Grčar, Nada Lavrač, and Martin Žnidaršič. Stream-based active learning for sentiment analysis in the financial domain. *Information Sciences*, 285: 181–203, 2014. ISSN 00200255. doi: 10.1016/j.ins.2014.04.034.

Lee A. Smales. Time-variation in the impact of news sentiment. *International Review of Financial Analysis*, 37:40–50, 2015. ISSN 10575219. doi: 10.1016/j.irfa.2014.11.019.

Philip J. Stone, Dexter C. Dunphy, Marshall S. Smith, and Daniel M. Ogilvie. *General Inquirer: A Computer Approach to Content Analysis*. The MIT Press, 1966. ISBN 026269011X.

Michael Stubbs. The Search for Units of Meaning: Sinclair on Empirical Semantics: Memorial Article: John Sinclair (1933-2007). *Applied Linguistics*, 30(1):115–137, 2009. ISSN 0142-6001. doi: 10.1093/applin/amn052.

Stephen Taylor. *Asset Price Dynamics, Volatility, and Prediction*. Princeton University Press, Princeton, N.J., 2007. ISBN 1400839254.

Paul C. Tetlock. Giving Content to Investor Sentiment: The Role of Media in the Stock Market. *The Journal of Finance*, 62(3):1139–1168, 2007. ISSN 00221082. doi: 10.1111/j.1540-6261.2007.01232.x.

Paul C. Tetlock, Maytal Saar-Tsechansky, and Sofus A. Macskassy. More Than Words: Quantifying Language to Measure Firms' Fundamentals. *The Journal of Finance*, 63(3): 1437–1467, 2008. ISSN 00221082. doi: 10.1111/j.1540-6261.2008.01362.x.

Peter Turney. Thumbs up or Thumbs Down? Semantic Orientation Applied to Unsupervised Classification of Reviews. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL-02)*, pages 417–424, Philadelphia, Pennsylvania, 2002.

Peter Turney and Michael L. Littman. Measuring Praise and Criticism: Inference of Semantic Orientation from Association. *ACM Transactions on Information Systems*, 21 (4):315–346, 2003. ISSN 10468188.

Cynthia Whissell, Michael Fournier, Rene Pelland, Deborah Weir, and K. MAKAREC. A Dictionary of Affect in Language: IV. Reliability, Validity, and Applications. *Perceptual and Motor Skills*, 62(3):875–888, 1986. ISSN 0031-5125. doi: 10.2466/pms.1986.62.3.875.

Theresa Wilson, Janyce Wiebe, and Paul Hoffmann. Recognizing contextual polarity in phrase-level sentiment analysis. In Raymond J. Mooney, editor, *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 347–354, 2005. doi: 10.3115/1220575.1220619.

Theresa Wilson, Janyce Wiebe, and Paul Hoffmann. Recognizing Contextual Polarity: An Exploration of Features for Phrase-Level Sentiment Analysis. *Computational Linguistics*, 35(3):399–433, 2009. ISSN 0891-2017. doi: 10.1162/coli.08-012-R1-06-90.

I. H. Witten and T. C. Bell. The Zero-frequency Problem: Estimating the Probabilities Of Novel Events In Adaptive Text Compression. *IEEE Transactions on Information Theory*, 37(4):1085–1094, 1991. ISSN 00189448. doi: 10.1109/18.87000.

Chung-Hsien Wu, Ze-Jing Chuang, and Yu-Chung Lin. Emotion recognition from text using semantic labels and separable mixture models. *ACM Transactions on Asian Language Information Processing*, 5(2):165–183, 2006. ISSN 15300226. doi: 10.1145/1165255.1165259.

B. Wuthrich, V. Cho, S. Leung, D. Permunetilleke, K. Sankaran, and J. Zhang. Daily stock market forecast from textual web data. In *Proceedings of 1998 IEEE International Conference on Systems, Man, and Cybernetics*, pages 2720–2725, 1998. doi: 10.1109/ICSMC.1998.725072.

Jeonghee Yi, Tetsuya Nasukawa, Razvan Bunescu, and Wayne Niblack. Sentiment analyzer: extracting sentiments about a given topic using natural language processing techniques. In *Proceedings of the Third IEEE International Conference on Data Mining*, pages 427–434. IEEE, 2003. doi: 10.1109/ICDM.2003.1250949.

Liang-Chih Yu, Jheng-Long Wu, Pei-Chann Chang, and Hsuan-Shou Chu. Using a contextual entropy model to expand emotion words and their intensity for the sentiment classification of stock market news. *Knowledge-Based Systems*, 41:89–97, 2013. ISSN 09507051. doi: 10.1016/j.knosys.2013.01.001.

Yuzheng Zhai, Arthur Hsu, and Saman K. Halgamuge. Combining News and Technical Indicators in Daily Stock Price Trends Prediction. In Derong Liu, Shumin Fei, Zengguang Hou, Huaguang Zhang, and Changyin Sun, editors, *Advances in Neural Networks – ISNN 2007*, volume 4493 of *Lecture Notes in Computer Science*, pages 1087–1096. Springer Berlin Heidelberg, Berlin, Heidelberg, 2007. ISBN 978-3-540-72394-3. doi: 10.1007/978-3-540-72395-0{\textunderscore}132.

Xue Zhang, Hauke Fuehres, and Peter A. Gloor. Predicting Stock Market Indicators Through Twitter "I hope it is not as bad as I fear". *Procedia - Social and Behavioral Sciences*, 26:55–62, 2011. ISSN 18770428. doi: 10.1016/j.sbspro.2011.10.562.

Yan-Yan Zhao, Bing Qin, and Ting Liu. Integrating Intra- and Inter-document Evidences for Improving Sentence Sentiment Classification. *Acta Automatica Sinica*, 36(10): 1417–1425, 2010. ISSN 18741029. doi: 10.1016/S1874-1029(09)60057-4.