# Automatic Character Assignation

Gerard Lynch & Carl Vogel
Computational Linguistics Group
Department of Computer Science and Statistics
Trinity College
Dublin 2,Ireland
{gplynch,vogel}@tcd.ie

**Abstract**

This article outlines a simple method for parsing an ASCII-format dramatic work from the Project Gutenberg Corpus into separate characters. The motivation for the program is a upcoming study in computational stylistics and characterization in drama. Various previous approaches involving interactive media are examined and the parser is evaluated by comparing the output to data annotated by hand and parsed automatically by the Opensourceshakespeare.org project parser. An acceptable level of accuracy is achieved, and it is identified how to improve accuracy to extremely high levels.

## 1 Introduction

The need for a program to parse drama files into constituent characters was born out of ongoing research on stylistics and character strengths in written drama. A script that parsed a play and separated characters was needed, as carrying this out by hand would be an arduous task, at great risk of introducing human error. Although there are XML annotated drama files available for some playwrights, the largest repository of drama files, Project Gutenberg, are for the most part in the ASCII text format, and do not all confirm to a common formatting style. A Java program was developed, named *PlayParser* which takes ASCII play files from Project Gutenberg as input and extracts a group of individual character files. It is important to note that wherever parsing is mentioned in the text, it refers to extracting the utterances from different characters and not any form of sentence-level grammatical parsing. This paper proceeds by situating the work with respect to its background motivations for parsing plays, and related work in information extraction from semi-structured texts. The operation of the parser is detailed, and the result is evaluated with respect to gold-standard annotations of ten Shakespeare plays, from the Opensourceshakespeare.org project.

# 2 Contextualization

## 2.1 Computational Stylistics

A number of points relevant to this paper can be usefully illustrated by recourse to the following excerpt from the screen play of *Dead Poet's Society*, [20]

```
                         KEATING
               Gentlemen, open your text to page
               twenty-one of the introduction. Mr.
               Perry, will you read the opening
               paragraph of the preface, entitled
               "Understanding Poetry"?

                         NEIL
               Understanding Poetry, by Dr. J. Evans
               Pritchard, Ph.D. To fully understand
               poetry, we must first be fluent with
               its meter, rhyme, and figures of speech.
               Then ask two questions: One, how artfully
               has the objective of the poem been
               rendered, and two, how important is that
               objective. Question one rates the poem's
               perfection, question two rates its
               importance. And once these questions have
               been answered, determining a poem's
               greatness becomes a relatively simple
               matter.

Keating gets up from his desk and prepares to draw on the chalk board.

                         NEIL
               If the poem's score for perfection is
               plotted along the horizontal of a graph,
               and its importance is plotted on the
               vertical, then calculating the total
               area of the poem yields the measure of
               its greatness.

Keating draws a corresponding graph on the board and the students
dutifully copy it down.

                         NEIL
               A sonnet by Byron may score high on the
               vertical, but only average on the
               horizontal. A Shakespearean sonnet, on
               the other hand, would score high both
               horizontally and vertically, yielding a
               massive total area, thereby revealing the
               poem to be truly great. As you proceed
               through the poetry in this book, practice
               this rating method. As your ability to
               evaluate poems in this matter grows, so
               will - so will your enjoyment and
               understanding of poetry.

Neil sets the book down and takes off his glasses. The student sitting
across from him is discretely trying to eat. Keating turns away from
the chalkboard with a smile.

                         KEATING
               Excrement. That's what I think of Mr. J.
               Evans Pritchard. We're not laying pipe,
               we're talking about poetry.

Cameron looks down at the graph he copied into his notes and quickly
scribbles it out.
```

The first point is that quantitative approaches to text stylistics may well not be "excrement". Suppose one wants to examine the spoken contributions of each character within a play to the overall play. One might want to apply authorship attribution techniques, or methods for corpus classification to as-

sess whether on the basis of the text alone one can track the development of a character in a play. This could be viewed as a sort of sentiment analysis, applied to literary purposes. The second point that this excerpt illustrates from a typographical perspective is the sort of information that would want to extract from a script is nontrivial to receive: one must decide if stage instructions interrupting a speech are in fact a continuation of the same character's speech, or constitutes a new speech (presumably by the same character), or even a long name for the next character to speak (with inconsistent capitalization conventions indicating speaking turn owners); one must accept that not all of the text intended to be spoken by a character is directly the speech of the character (in the example here, the first contribution by Keating uses the speech of the character to provide a stage instruction that lets the audience know that Neil's speech is indirect); one must account for variability in the conventions that offset indication of the character speaking from the speech and from accompanying stage instructions and background narrative. These issues specific to plays are detailed further in §2.2.2.

While applying computational stylistics to character analysis is novel, computational stylistics is not at all new. [19] provides an overview of such work beginning with the middle of the 19th Century. Much of the work is in authorship attribution in a context in which some work is available with unquestioned provenance for an author, but also where attributions are doubtful. This sort of work is in contrast to that of [5] in which subjective analysis of textual features and information external to the texts are applied to a range of attribution problems. An open problem even within the attribution task is noted by [25], typically attribution research makes the idealizing assumption that (for example) the complete works of Shakespeare were written all at the same instant. An exception is in the work of [27] which attempts to leverage information about distributions of syllable stress and pauses indicated by line breaks in the Shakespeare plays over time of composition to date works whose temporal origin is less certain. Statistical methods have been applied to comparisons of political speeches [9] and political party manifestos [24]. However, this form of sentiment analysis, which has currency in the political science literature ([13, 14]), attempts to discern *content*. Certainly, [8] demonstrates that an algorithmic approach to textual analysis can also support fine-grained *stylistic* characterization of, for example, cohesion.

The approach we have in mind to support with the research described in this paper is rather more like that of [26], which performs a sort of cluster analysis on the poetry of Brendan Kennelly in order to identify poems with one or more of the narrative voices that Kennelly appeals to in the composition of his works: "the woman", "the child", "Ozzie", "the chorus", etc. Here we want to consider the textual contributions of characters in plays and identify, among other things, how homogeneous the contributions of each character are. Using authorship attribution tools, if all of the contributions of a character cluster together (without attracting the contributions of other characters), then that character is textually very strong. It is intriguing to know whether, across a playwright's canon, their characters are strong. Further, it is interesting to

know whether the character is stronger than the author—this is the situation if the textual contributions of the character are easier to predict as contributions of the character than they are to be spotted as the product of the author, using only textual internal features. This would bolster any claim about the author's ability to construct strong characters, as opposed to constructing characters that are all alter egos of the author. The point is perhaps nuanced, but the claim is that it is in general more difficult to write characters that are strong in the above sense than to write characters whose authorship can be guessed.

In order to perform this sort of analysis, it is necessary to extract from the plays the relevant information associated with each character, and assigning it appropriately. This gives rise to the problem of parsing a play.

## 2.2 Information Extraction from Semi-Structured Text

The task of parsing a play is a problem in the area of parsing of semi-structured documents. Much of the literature on document parsing presupposes that the the text begins with XML mark-up (e.g. [12]), and set the task of identifying larger discourse structure in the text. In our case, the relevant information is within unannotated text; nonetheless, the text has implicit information packaging in the formatting. The problem exists in other tasks in extracting information from text.

### *2.2.1 The problem in general*

More challenging problems in this field include dictionary parsing and the parsing of business cards. [17] describes methods for parsing machine readable copies of dictionaries using a grammar-based parser written in Prolog. This is a complicated procedure due to the fact that a dictionary entry may contain different fields including parts of speech markers, explanation, examples of usage, related words and phrases and various other categories including dialectical and cultural information.

[4] describes work done on using a similar Definite Clause Grammar based system for parsing business cards.This article describes how although the information on a business card is usually fairly uniform, fields such as name, address, professional degrees, organization, email address and website occur in nearly all instances, there is no fixed order as to how they should appear, which means any parser must be very flexible in order to capture all of the information contained on the card. [11] describes using OCR data to parse track names from CD covers for an imagined scenario in which the custodian of a large CD library equipped with a mobile phone with a camera can communicate with a library-based server to identify whether a CD found at a car-boot sale contains tracks not already in the library. This task also requires a good deal of flexibility in order to tackle the various different fonts and styles of layout that are encountered.

These are all instances of the general problem of information extraction from raw data into templates [6]. [18] discusses how to generalize across do-

mains to enable re-use of information extraction methods. However, this sort of information has the harder problem of linguistic processing as the relevant information is embedded in the sentences of the document. They appeal to shallow linguistic processing to make progress. In the context of filling templates from business cards, or dictionaries, or CD covers or plays, one has to engage in meta-linguistic processing to understand document structure, rather than content. Yet, essentially finite state methods still apply [1]. Thus, the risks pointed out by [7] do not apply directly: named entity recognition is an issue, though, as discussed in the second step of our method (see §3). Ours is a specialization of the information extraction task orthogonal to scanning documents for particular sorts of information [23].

#### *2.2.2 The problem of parsing plays*

In contrast, parsing a play is a relatively straightforward process. The majority of plays written since the Renaissance have contained the same structure, a piece of writing, divided into a number of acts, each act divided into a number of scenes, each scene containing a dialogue(or monologue) between at least one character, with the utterance of each character clearly marked at the beginning by a string identifying the character and normally some stage instructions which inform the reader when and where the scene takes place, what characters are present and when characters enter and leave.

The constancy of conventions in structuring a play as a document is slightly surprising if one reflects on other changes of form that occur in the period that stretches even just from iambic pentameter to free verse to flat prose. However, the conventions of recording interactive dialogue in writing has shown little innovation since antiquity.

Assume $D$ is the *dramatis personae* or the set of characters in the play, $S$ is the set of all stage directions in the play, $L$ is the sequence of all utterances.[1] It is possible to model a play as a structure as in (1), the semantic structure that corresponds to a template for a play.

(1) $P = \langle D, S, L, \lhd, \leq \rangle$

A binary relation ($\lhd$) maps characters to their lines (2).

(2) $\lhd \subseteq D \times L$

Finally, $\leq$ is a partial order on $\lhd$, which thus models the order of spoken lines, and allows for overlapping turns. Any information extraction device tailored to plays must aspire to approximating the structure $P$ that corresponds to the play. In general it suffices to approximate $\leq$ by simply keeping track of all of the lines uttered by each character in the order the lines are uttered, without separating them as individual turns.

[1] It is a sequence rather than a set because the same sentence may be uttered more than once, and each instance is a line, by union rules.

The difficulty lies in the different formats that are present in the Gutenberg corpus. Some random examples from the Gutenberg corpus illustrate this. from Henrik Ibsen's *The Feast at Solhoug*

```
ERIK.

  [Rising at the table.]  In one word, now, what answer have you to
make to my wooing on Knut Gesling's behalf?

BENGT.

  [Glancing uneasily towards his wife.]  Well, I--to me it seems-- [As
she remains silent.]  H'm, Margit, let us first hear your thought in
the matter.
```

from *The Blue Bird : A Fairy Play in Six Acts* by Maurice Maeterlinck

```
TYLTYL Of course; there's no one to stop us.... Do you hear the
music?... Let us get up....

(_The two_ CHILDREN _get up, run to one of the windows, climb on to
the stool and throw back the shutters. A bright light fills the room.
The_ CHILDREN _look out greedily_.)

TYLTYL We can see everything!...
```

from *The Jew of Malta* by Christopher Marlowe

```
MERCHANT. I go.

BARABAS. So, then, there's somewhat come.-- Sirrah, which of my ships
art thou master of?

MERCHANT. Of the Speranza, sir.
```

from *Volpone, the Fox* by Ben Jonson

```
VOLP: I thank you, signior Voltore; Where is the plate? mine eyes are
bad.

VOLT [PUTTING IT INTO HIS HANDS.]: I'm sorry, To see you still thus
weak.

MOS [ASIDE.]: That he's not weaker.
```

There are other elements that are present in some plays, for example *dramatis personae* provided in a block at the outset. A lot of transcriptions also contain historical information about the play, when and where the play was first staged, and the actors that played the various roles. However, this information is not always provided and it would be unwise to rely on a *dramatis personae* as the ultimate authority on which characters are in a drama.

### 2.3 Available tools for interactive drama

There are several resources already available for interactively dealing with drama files on the Internet. [10] is a website implementing fully searchable versions of all the major works of Shakespeare. The website is built around a database coupled with an interactive interface written in PHP. One of the options available is to search for a particular character in any Shakespeare play and display all of the utterances of the character, including line numbers and corresponding act and scene markers. A concordancer and statistics based on word counts and frequencies are also available. In addition to being available for use via the internet, the source code for this system is freely available, which allows for modification. The drawbacks of this system for the task at hand is the fact that the plays must be annotated in a specific format to be read by the parser, and as the forthcoming study mentioned in §2.1 deals with a number of different playwrights, this would be necessary for each piece of work. If successful *PlayParser* will abet reliable automatic annotation of all plays that adhere to the conventions mentioned in §2.2.2, even with variability within those conventions.

[22] describes an prototype interactive Flash-based system called *Watching the Script* for displaying interactive scripts. It displays a number of views including the basic play text, and creates small character avatars on a stylized stage for students of directing to control. This system doesn't perform any pre-parsing of characters, simply displaying them as encountered and also takes plays in a special input format, though the possibility of incorporating non-formatted drama files is mentioned. Success in our endeavor would provide theirs with usefully structured input.

*Scenario* [16] is a commercially available interactive stage directors tool for all of Shakespeare's major works. It features drag and drop graphically represented characters and props, sound effects and allows creation of different frames. It does not appear to feature any concordancing elements or searching tools and is restricted to the plays of William Shakespeare.

## 3 Specification & Realization

The *PlayParser* is a two pass parser. The algorithm and input/output assumptions are as follows.

**Input**

Raw files of plays, except for manual preprocessed removal of headers and footers like glossaries, from the Project Gutenberg archive are supplied as input.

**Output**

The output is a set of files, one file for each character in each input play, and an index. Each file consists of all of the lines in the play spoken by the character. The index is a list of files and unique character identifiers.

**Algorithm**

1. The first pass reads through the play and compiles a list of character specifiers
2. The list is then presented to the user and the user is expected to remove any invalid entries. The number of occurrences of the character is displayed beside the name, to aid disambiguation, certain names may look like characters but in fact not be.
3. The parser then reads through the play again using the list of character specifiers as definitive points for termination and divides the file into the utterances of the different characters

Thus, *PlayParser* can be described as function mapping a text $T$ to a structure $P'$ that approximates (1) as (4).

(3) $PP(T) \mapsto P'$

(4) $P' = \langle D, \emptyset, L \cup S, \lhd, \leq \rangle$

In particular, in its current form, stage instructions are not eliminated, but are identified with the character whose speech they apply to.

There are two main advantages over the system described in [10] The first is the fact that the *PlayParser* does not rely on uniquely formatted input and is relatively flexible. The other is in the way the list of characters is created. [10] requires that the user enter the list of characters manually for the play that will be extracted. Although most of the Shakespeare plays contain a *dramatis personae* this is not always the case for other playwrights, and even when it is provided, it is not always possible to deduce some of the lesser characters' names from it. A solution to this would be of course to skim through the play manually looking for all unique character names, but this could take some time. It was decided to use algorithms trained on a base set of plays to determine whether a line contained a character specifier, check whether this character specifier is contained in the list of characters, if this is not the case, add it to the list and then move on. The list is then presented to the user who can look over the list of character specifiers and remove anything that is not a character specifier.

If one were to push the system away from manual intervention altogether, it would be in this second step that one would do so. One could imagine heuristics for guessing whether two strings refer to the same character. This is a rich open problem, as big as the puzzles of naming and definite reference in the philosophy of language and the problems of named entity recognition and anaphor resolution in computational linguistics.

The time complexity of the method is tractable (essentially finite-state, with a look-up operation), measured in terms of $n$, the number of tokens of text in a play's file. The worst case is defined by the situation in which the play consists of $n$ characters who have one turn each of silence.

1. $n*log(n)$: This is the amount of time to read each line and decide whether the speaker's character has been encountered before.

2. $n$: Each character name is shown to the user once.

3. $n * log(n)$: This is the amount of time to process each line and decide whether it is a turn.

Thus, the complexity is dominated by $n * log(n)$.

Although the current prototype of the parser is flexible in the sense that it does not require a specific file format to parse, it does have some limitations. The dataset that it was developed on consists of the Project Gutenberg ASCII versions of plays by the following playwrights: William Shakespeare, Ben Jonson, W.B Yeats, George Bernard Shaw and Oscar Wilde. The formatting of all plays by each playwright was not necessarily consistent and was often undertaken by a number of different transcribers. The character specifier formatting fell into two main categories. The first example is taken from the Gutenberg file of *A Woman of No Importance* by Oscar Wilde

```
LADY CAROLINE.  I believe this is the first English country house you
have stayed at, Miss Worsley?

HESTER.  Yes, Lady Caroline.

LADY CAROLINE.  You have no country houses, I am told, in America?

HESTER.  We have not many.
```

In this case, the formatting for character name is given by the full character name in capitals. This particular formatting style is relatively easy to parse, as the character name is distinguishable from any proper noun occurring in the text. The second most popular formatting style is the following.

```
Ham. Whither wilt thou lead me? Speak! I'll go no further.
Ghost. Mark me.
 Ham. I will.
```

This short extract from the Project Gutenberg file of Shakespeare's *Hamlet* illustrates constructions that must be overcome by the parser. Here, the characters utterances are marked by an abbreviated form of their name written in conventional English writing style with a capital letter at the beginning. The names are not always abbreviated, as in the case of the Ghost character in this extract. In the current prototype of the parser, it looks for strings terminated by a full stop at the beginning of a sentence to indicate a character marker. The parser could read the two second sentences as unique characters in their own right, and add them to the list of characters. These would then have to be removed by the user.

There are issues that are beyond the reach of the program, such as keeping a character consistent when the character's name is changed. Many plays have characters who for one reason or another, begin with one name and then later take on a different name, sometimes when we learn the name of the character or when his or her status changes, for example, being crowned king or queen in some Shakespearean drama. For example, in Shaw's *Pygmalion*, the character of "Flower Girl" subsequently becomes named "Eliza". Moreover, annotations of characters speaking lines in plays within plays are also complex. Inconsistency on the part of the human transcriber is also not handled by the program, for example, if the transcriber suddenly decides to abbreviate a character name halfway through a play, the program will create two separate files for each version of the name it occurs if in the second pass, human intervention has not normalized the spelling to that of an established orthography for the character. Improvements that take some of these issues into account are a source for future work.

# 4 Evaluation

The parser was evaluated by comparing its output to the gold standard data from Opensourceshakespeare.org. Ten plays were chosen for the comparison. [2] claim that 90% precision is the threshold of accuracy necessary for information to be acceptable in "live" applications. It has been noted [6] that by the end of the 1990s, precision in the Message Understanding Conferences was around 70%. However, our task involves not text understanding, but document structure recognition. A wide range of evaluation statistics beyond precision and recall are available [3]. The problem of parsing a play is different from information retrieval, certainly. There precision and recall correspond to having correctly retrieved relevant documents and all relevant documents, respectively. It is in fact normal to distinguish information retrieval and information extraction. In the context of document understanding, as opposed to template filling from interpretation of sentences the documents contain, recall is nearly equivalent to precision because all segments of the document get classified in terms of the relevant document definition. Thus, like [15], we compute evaluation statistics more directly. A simple value for parsing accuracy was used, which is calculated as in (5).

(5) $Accuracy = \frac{CorrectlyAssignedLines}{TotalNumberofLines}$

Correctly parsed lines were lines that were assigned correctly to a character as marked in the source text. Ambiguities in the original text markup were not considered as errors, for example if the original text gave a number of character X's lines to character Y, this was ignored. Examples of errors include, skipping a number of lines because of the lack of correct formatting for the character specifier, the creation of a new character that does not exist in the text, or the inclusion of stage directions in the text of a character. Table 1 gives a simple but immediate view of the accuracy of the parser.

| Results | | |
|---|---|---|
| Playname | Average accuracy | Average accuracy less stage directions |
| A Comedy Of Errors | 89 | 100 |
| Loves Labour Lost | 91.8 | 99.4 |
| A Midsummer Night's Dream | 81 | 100 |
| Macbeth | 80.1 | 99.7 |
| Much Ado About Nothing | 88.9 | 98.3 |
| Romeo And Juliet | 82 | 100 |
| The Taming Of The Shrew | 91 | 100 |
| The Tempest | 90 | 99.9 |
| Twelfth Night | 85.1 | 97.3 |
| Coriolanus | 88.7 | 99.5 |
| Average accuracy | 86.76 | 99.94 |

Table 1: Benchmarking Results

The two columns indicate accuracy under two different standards of success. The first column gives the method a penalty for each line of stage instructions that is mistakenly attributed as a part of a character's speech. This is clearly in the neighborhood of acceptability suggested by [2]. More interestingly, the second column displays the accuracy under the idealization of stage instructions having been hand tagged. The average resulting accuracy of 99.94 compellingly suggests that the next important open question to address in this area is connected to the binary decision about whether a line of text is a stage instruction or not.

# 5 Speculation

From the evaluation done on the parser, it is clear that a more flexible algorithm is needed that can deal with issues like missing full stops after character specifiers and indented character specifiers. The program should be able to deal with misspelt character specifiers and at the very least report to the user where possible parse errors may have occurred. Another extension, as discussed above, would be to implement a system to automatically detect stage directions. There are many transcribed works where stage directions and scene descriptions are given no particular marking and are not bracketed in any clear way. One such indicator could be to look for certain verb inflection and large concentrations of character names. For example, while much of the content of a play is written in second person as characters speak to each other, or third person past, stage instructions tend to be in third person present. However, they range from very terse, simple Shakespearean instructions (e.g. "exeunt") to rather involved descriptions (e.g. those of Shaw's plays). A large enumeration of character names would tend to signal a description of the action that

is set to happen rather than, forming a constituent of a character's speaking part. The algorithm that detects character specifiers could be revised to incorporate regular expressions which might improve the accuracy by detecting slight differences in character specifiers that would otherwise be passed over.

# 6 Rumination

This paper has detailed a prototype parser for separating a play into characters, and assigning the text intended to be spoken by each character correctly to the character. It performed relatively well as compared to hand annotated input but although flexible, is still in need of streamlining in order to be fully autonomous. The character data will be used in a study on character strengths in different styles of drama. Although this was the initial task of the parser, it could also be a useful tool for parsing speech transcripts, minutes of meetings, screenplays or chatroom and instant messenger dialogues with some minor adjustments. It works without XML annotation of documents but could put to use in generating textual markup appropriate to the structure of plays. And, [21], "The play's the thing."

# References


[1] D. Appelt, J. Hobbs, J. Bear, and D. I. M. Tyson. Fastus: a finite-state processor for information extraction from real-world text. In *Proceedings of the 13 International Joint Conference on Artificial Intelligence*, pages 1172–1178, 1993.

[2] J. Cowie and W. Lehnert. Information extraction. *Communications of the ACM*, 39(1):80–90, 1996.

[3] T. Fawcett. Roc graphs: Notes and practical considerations for researchers. Technical report, HP Laboratories, Page Mill Road, Palo Alto CA, 1994.

[4] R. Ferguson. Parsing business cards with an extended logic grammar. Technical report, CCRIT, 1988.

[5] D. Foster. *Author Unknown. On the trail of Anonymous.* Macmillan: London, Basingstoke and Oxford, 2001.

[6] R. Gaizauskas and Y. Wilks. Information extraction: Beyond document retrieval. *Journal of Documentation*, 54(1):70–105, 1998.

[7] R. Grishman. Information extraction: Techniques and challenges. In M. Pazienza, editor, *Information Extraction - a Multidisciplinary Approach to an Emerging Information Technology*, Lecture Notes in Artificial Intelligence, pages 10–27. Springer-Verlag: Berlin, 1997.

[8] M. Hoey. *Patterns of Lexis in Text.* Oxford University Press, 1991.

[9] L. Hogan. A corpus linguistic analysis of american, british and irish political speeches. Master's thesis, Centre for Language and Communication Studies, Trinity College, University of Dublin, 2005.

[10] E. M. Johnson. http://www.opensourceshakespeare.org, 2004. last verified 1st May 2007.

[11] P. Kilkenny. Information retrieval from cd covers using ocr text. Department of Computer Science, Trinity College, University of Dublin. Bachelor in Computer Science. Final Project Dissertation., 2006.

[12] H. Langer, H. Lüngen, and P. S. Bayerl. Text type structure and logical document structure. 2004. Proceedings of the ACL-Workshop on Discourse Annotation, Barcelona.

[13] M. Laver, editor. *Estimating the Policy Position of Political Actors*. Routledge, 2001.

[14] M. Laver, K. Benoit, and J. Garry. Extracting policy positions from political texts using words as data. *American Political Science Review*, 97, 2003.

[15] J. Makhoul, F. Kubala, R. Schwartz, and R. Weischedel. Performance measures for information extraction. In *Proc. of the DARPA Broadcast News Workshop*, pages 249–252, 1999. Virginia, USA.

[16] I. Media. http://ise.uvic.ca/Annex/ShakespeareSuite/scenario.html, 2004. last verified 1st May 2007.

[17] M. S. Neff and B. K. Boguraev. Dictionaries, dictionary grammars and dictionary entry parsing. In *Proceedings of the 27th annual meeting on Association for Computational Linguistics*, pages 91–101, Morristown, NJ, USA, 1989. Association for Computational Linguistics.

[18] G. Neumann and T. Declerck. Domain adaptive information extraction. In *Proceedings of the International Workshop on Innovative Language Technology and Chinese Information Processing (ILT & CIP '01), April*, Shanghai, 2001.

[19] M. P. Oakes. *Statistics for Corpus Linguistics*. Edinburgh Textbooks in Empirical Linguistics. Edinburgh: Edinburgh University Press, 1998.

[20] T. Schulman. http://www10.pair.com/crazydv/weir/dps/script.html. last verified August 23, 2007.

[21] W. Shakespeare. Hamlet, prince of denmark. In M. Mack, B. Knox, J. McGalliard, P. M. Pasinetti, H. Hugo, R. Wellek, and K. Douglas, editors, *World Masterpieces: Through the Renaissance*, volume 1. New York: Norton, 1973.

[22] S. Sinclair, S. Gabriele, S. Ruecker, and A. Sapp. Digital scripts on a virtual stage: the design of new online tools for drama students. In *WBE'06: Proceedings of the 5th IASTED international conference on Web-based education*, pages 155–159, Anaheim, CA, USA, 2006. ACTA Press.

[23] J. Thomas, D. Milward, C. Ouzounis, S. Pulman, and M. Carroll. Automatic extraction of protein interactions from scientific abstracts. In *Pacific Symposium on Biocomputing*, volume 5, pages 538–549, 2000.

[24] S. Van Gijsel and C. Vogel. Inducing a cline from corpora of political manifestos. In M. A. et al., editor, *Proceedings of the International Symposium on Information and Communication Technologies*, pages 304–310, 2003.

[25] C. Vogel. N-gram distributions in texts as proxy for textual fingerprints. In A. Esposito, E. Keller, M. Marinaro, and M. Bratanic, editors, *The Fundamentals of Verbal and Non-Verbal Communication and the Biometrical Issue*. IOS Press, 2007.

[26] C. Vogel and S. Brisset. Hearing voices in the poetry of brendan kennelly. In *Varieties of Voice*, 2006. 3rd international BAAHE conference. Leuven, 7-9 December 2006.Revised version to appear in *Belgian Journal of English Language & Literature*.

[27] M. R. Yardi. A statistical approach to the problem of the chronology of shakespear's plays. *Sankhya: The Indian Journal of Statistics*, 7(3):263–8, 1946.