Latent Ambiguity in Latent Semantic Analysis?

Martin Emms and Alfredo Maldonado-Guerra
School of Computer Science and Statistics, Trinity College, Dublin, Ireland
{*mtemms,maldonaa*}*@tcd.ie*

Keywords: LSA; Dimensionality

Abstract: Latent Semantic Analyis (LSA) consists in the use of SVD-based dimensionality-reduction to reduce the high dimensionality of vector representations of documents, where the dimensions of the vectors correspond simply to word counts in the documents. We show that that there are two contending, inequivalent, formulations of LSA. The distinction between the two is not generally noted and while some work adheres to one formulation, other work adheres to the other formulation. We show that on both a tiny contrived data-set and also on a more substantial word-sense discovery data-set that the empirical outcomes achieved with LSA vary according to which formulation is chosen.

1 Introduction

Latent Semantic Analyis (LSA) is a widely used dimensionality-reduction technique. Section 2 recalls the matrix properities upon which LSA is based and then section 3 gives details of two different dimensionality-lowering transformations which may be based on those properites, which we will term the R_1 and R_2 representations, and we argue that there is ambiguity in the literature as to which representation is intended. Section 4 then shows empirical outcomes which vary with the adopted formulation.

2 Singular Value Decomposition

Latent Semantic Analysis (LSA) is based theoretically and algorithmically on Singular Value Decomposition (SVD) properties of matrices. The first concerns the existence of a particular decomposition, a property expressible as the following theorem[1]

Theorem 1 (SVD)**.** *if $m \times n$ matrix $\mathbf{A}$ has rank r, then it can be factorised as $\mathbf{A} = \mathbf{USV}'$ where:*

1. $\mathbf{U}$ *has the eigen-vectors of $\mathbf{A} \times \mathbf{A}'$ for its first r columns, in descending eigen-value order; these columns are orthonormal*
2. $\mathbf{S}$ *has zeroes everywhere, except its diagonal which has the square roots of the r distinct eigen-values of* $\mathbf{U}$*, in descending order, then 0*
3. $\mathbf{V}$ *has the eigen-vectors of $\mathbf{A}' \times \mathbf{A}$ for its first columns, in descending eigen-value order; these column are orthonormal*

Without loss of generality once can assume the dimensions of the matrices are:

$$\mathbf{U} : m \times r,\ \mathbf{S} : r \times r,\ \mathbf{V} : n \times r$$

The second essential fact is that the SVD can be used to derive optimum[2] low-rank approximations of the orginal $\mathbf{A}$, by truncating the SVD of $\mathbf{A}$ to use just the first k columns of $\mathbf{U}$ and $\mathbf{V}$ as follows (see again (Manning et al., 2008))

Theorem 2 (Low rank approximation)**.** *If $\mathbf{U} \times \mathbf{S} \times \mathbf{V}'$ is the SVD of A, then $\hat{\mathbf{A}} = \mathbf{U}_k \times \mathbf{S}_k \times \mathbf{V}'_k$ is a optimum rank-k approx of $\mathbf{A}$ where*

1. $\mathbf{S}_k$ *is diagonal with top-most k values from* $\mathbf{S}$
2. $\mathbf{U}_k$ *is just first k columns of* $\mathbf{U}$
3. $\mathbf{V}_k$ *is just first k columns of* $\mathbf{V}$

$\mathbf{U}_k \times \mathbf{S}_k \times \mathbf{V}'_k$ can be termed the 'rank k reduced SVD of $\mathbf{A}$'.

The HCI/Graph Example Figure 1 shows a 12×9 term-by-document matrix, $\mathbf{A}$ (ie. rows of $\mathbf{A}$ express terms via their document occurrence, columns of $\mathbf{A}$ express documents via their term occurrence). This term-by-document matrix is used in a number of articles by the originators of LSA. See (Deerwester et al.,

[1] This follows closely Theorem 18.3 of (Manning et al., 2008)

[2] Optimality being defined as minimising the sum of squares of corresponding matrix positions.

$\mathbf{A}=$

	c1	c2	c3	c4	c5	m1	m2	m3	m4
human	1	0	0	1	0	0	0	0	0
interface	1	0	1	0	0	0	0	0	0
computer	1	1	0	0	0	0	0	0	0
user	0	1	1	0	1	0	0	0	0
system	0	1	1	2	0	0	0	0	0
respones	0	1	0	0	1	0	0	0	0
time	0	1	0	0	1	0	0	0	0
EPS	0	0	1	1	0	0	0	0	0
survey	0	1	0	0	0	0	0	0	1
trees	0	0	0	0	0	1	1	1	0
graph	0	0	0	0	0	0	1	1	1
minor	0	0	0	0	0	0	0	1	1

$\mathbf{U}_k=$

$$\begin{bmatrix} 0.22 & -0.11 \\ 0.20 & -0.07 \\ 0.24 & 0.04 \\ 0.40 & 0.06 \\ 0.64 & -0.17 \\ 0.27 & 0.11 \\ 0.27 & 0.11 \\ 0.30 & -0.14 \\ 0.21 & 0.27 \\ 0.01 & 0.49 \\ 0.04 & 0.62 \\ 0.03 & 0.45 \end{bmatrix}$$

$\mathbf{S}_k=$

$$\begin{bmatrix} 3.34 & 0 \\ 0 & 2.54 \end{bmatrix}$$

$\mathbf{V}_k=$

$$\begin{bmatrix} 0.20 & -0.06 \\ 0.61 & 0.17 \\ 0.46 & -0.13 \\ 0.54 & -0.23 \\ 0.28 & 0.11 \\ 0.00 & 0.19 \\ 0.01 & 0.44 \\ 0.02 & 0.62 \\ 0.08 & 0.53 \end{bmatrix}$$

Figure 1: A term-by-document matrix $\mathbf{A}$, and the components matrices of its rank 2 reduced SVD $\mathbf{U}_k\mathbf{S}_k\mathbf{V}_k'$

1990; Landauer et al., 1998). It is based on an artificial data set concerning two sets of article titles, one about HCI (titles c1–c5), the other about graph theory (titles m1–m4). The columns count occurrences of 12 chosen terms. This $\mathbf{A}$ has rank 9, and has a SVD decomposition into $\mathbf{U}\times\mathbf{S}\times\mathbf{V}'$, where $\mathbf{U}$ is 12×9, and $\mathbf{V}$ is 9×9 . See p406 of (Deerwester et al., 1990).

Multiplying $\mathbf{U}$, $\mathbf{S}$ and $\mathbf{V}'$ gives back *exactly* $\mathbf{A}$. To the right in Figure 1 the component matrices $\mathbf{U}_k$, $\mathbf{S}_k$, $\mathbf{V}_k$ of its rank 2 reduced SVD are given, whereby $\hat{\mathbf{A}}=\mathbf{U}_k\mathbf{S}_k\mathbf{V}_k'$ (see also p406 of (Deerwester et al., 1990)).

3 Contending formulations of LSA

LSA concerns using the SVD to make lower dimension versions of the columns of $\mathbf{A}$ (or vectors like these ie. m dimensional 'document' vectors).

Where $\mathbf{d}$ is an m dimensional vector (such as a column of $\mathbf{A}$), we contend that the literature has basically *two* contenders for its SVD-based reduced dimensionality version, contenders we shall term $R_1(\mathbf{d})$ and $R_2(\mathbf{d})$.

Definition 1 (R_1 and R_2 document projections)**.** *If $\mathbf{A}$ is $m\times n$, and $\mathbf{U}_k\mathbf{S}_k\mathbf{V}_k'$ is its rank k reduced SVD, and $\mathbf{d}$ is an m dimensional vector, then k-dimensional versions $R_1(\mathbf{d})$ and $R_2(\mathbf{d})$ are defined by*

$$R_1(\mathbf{d}) = \mathbf{d}\times\mathbf{U}_k \quad (1)$$

$$R_2(\mathbf{d}) = \mathbf{d}\times\mathbf{U}_k\times\mathbf{S}_k^{-1} = R_1(\mathbf{d})\times\mathbf{S}_k^{-1} \quad (2)$$

and if $\mathbf{d}$ is i^{th} column of $\mathbf{A}$ and $\mathbf{V}_k^i$ is i^{th} row of $\mathbf{V}_k$ (ie. $[\mathbf{V}(i,1)\ldots\mathbf{V}(i,k)]$) the above defintions are equivalent to

$$R_1(\mathbf{d}) = \mathbf{V}_k^i\times\mathbf{S}_k \quad (3)$$

$$R_2(\mathbf{d}) = \mathbf{V}_k^i \quad (4)$$

That the alternative formulations in (3) and (4) are equivalent to the formulations in (1) and (2), for the case where $\mathbf{d}$ is a column of $\mathbf{A}$, is not immediately apparent. You can show the equivalence of (3) and (1), that is, $\mathbf{d}\times\mathbf{U}_k=\mathbf{V}_k^i\times\mathbf{S}$ when $\mathbf{d}$ is the i^{th} column of $\mathbf{A}$ starting from the defining SVD equation $\mathbf{A}=\mathbf{USV}'$ as follows:

$$\begin{array}{lrcl} & \mathbf{A}' & = (\mathbf{USV}')' & = \mathbf{VSU}' \\ \text{hence} & \mathbf{A}'\mathbf{U} & = \mathbf{VSU}'\mathbf{U} & = \mathbf{VS} \\ \text{hence} & \mathbf{d}\mathbf{U}_k & = \mathbf{V}_k\mathbf{S}_k & \end{array}$$

The equivalence of (4) and (2), that is, $\mathbf{d}\times\mathbf{U}_k\times\mathbf{S}^{-1}=\mathbf{V}_k^i$ when $\mathbf{d}$ is the i^{th} column of $\mathbf{A}$, follows from the equivalence of (3) and (1) by post-multiplication by $\mathbf{S}^{-1}$

Where $\mathbf{A}$ is a $m\times n$ matrix, the matrix $\mathbf{V}_k$ of its reduced SVD is a $n\times k$ matrix. For the example shown in Figure 1, $\mathbf{V}_k$ has exactly as many *rows* (9) as there were *column* vectors representing documents in the original term-by-document matrix $\mathbf{A}$. Therein lies the possibility to identify these rows of $\mathbf{V}_k$ as the reduced representation of the columns of $\mathbf{A}$. The fact that (2) is equivalent to (4) leads to the naturally accompanying assumption that (2) – $\mathbf{d}\times\mathbf{U}_k\times\mathbf{S}^{-1}$ – is the formula for projecting an arbitrary document vector $\mathbf{d}$.

On the other hand, where $\mathbf{A}$ is a $m\times n$ matrix, the matrix $\mathbf{U}_k$ of its reduced SVD is a $m\times k$ matrix, so its *columns* are of exactly the size for it to be possible to take dot products with an m dimensional document vector, as expressed in (1). For the example shown in Figure 1 the columns of the matrix $\mathbf{U}_k$ of $\mathbf{A}$'s reduced SVD are of size 12, the same as that of document vectors. Additionally the columns of $\mathbf{U}_k$ are orthogonal to each other and of unit length and thus the R_1 formulation is simply the projection onto a new set of orthogonal axes defined by the columns of $\mathbf{U}_k$.

Ultimately the relationship between the R_1 and R_2 formulations is a simple one of scaling: $R_2(\mathbf{d})=$

$R_1(\mathbf{d}) \times \mathbf{S}^{-1}$. However, since the entries on the diagonal of $\mathbf{S}$ are not equal, such a scaling changes the essential geometry. In particular, the nearest neighours, or the set within a certain cosine range of a given vector $\mathbf{d}$, is *not* generally preserved under a scaling. For example, given a scaling which transforms x and y according to $x' = \frac{x}{2}$, $y' = \frac{y}{8}$, the table below gives the coordinates of 3 points before and after the scaling:

a	$(0,8)$	a'	$(0,1)$
b	$(4,8)$	b'	$(2,1)$
c	$(4,0)$	c'	$(2,0)$

and before the scaling b has nearest neighbour a, whilst afterwards b' has nearest neighbour c', on both the euclidean distance and cosine measures. Machine learning methods for adapting distance measures are often predicated on precisely this fact. In view of this, the R_1 formulation of LSA, as expressed by (1) and (3) is genuinely different to the R_2 formulation, as expressed by (2) and (4) and one should expect R_1 and R_2 to give diverging outcomes when deployed within a system. We contend that this has been overlooked. To this end we will consider the work of a number of authors, arguing that some are adhering to the R_1 formulation and some to the R_2 formulation.

The R_2 formulation of LSA is one presented in many, fairly widely cited, publications, for example (Rosario, 2000; Gong and Liu, 2001; Zelikovitz and Hirsh, 2001), the relevant parts of which are below briefly noted.

In the notation of (Rosario, 2000), the reduced rank SVD of the $t \times d$, term-by-document matrix is $T_{t\times k}S_{k\times k}(D_{d\times k})^T$, with T and D used in place of $\mathbf{U}_k$ and $\mathbf{V}_k$. This is described (p3) as providing a representation in an alternative space whereby

> *the matrices T and D represent terms and documents in this new space*

and additionally the repesentation of a query is given (p4) as $q^T T_{t\times k} S^{-1}_{k\times k}$. Thus for pre-existing documents and novel queries, this matches, modulo notational switches, the R_2 formulations of (4) and (2).

In the notation of (Zelikovitz and Hirsh, 2001), the SVD of a $t \times d$ term-by-document matrix is TSD^T. The representation of a query, based on this SVD is given as

> *a query is represented in the same new small space that the document collection is represented in. This is done by multiplying the transpose of the term vector of the query with matrices T and S^{-1}*

Again modulo notational switches, this is the R_2 formulation of (2).

In the notation of (Gong and Liu, 2001), the SVD of an $m \times n$ term-by-sentence matrix is $\mathbf{U}\Sigma\mathbf{V}^T$, and the SVD is described as defining a mapping which (p21)

> *projects each column vector i in matrix* $\mathbf{A}$ *... to column vector* $\Phi_i = [v_{i1}v_{i2}\dots v_{ir}]^T$ *of matrix* $\mathbf{V}^T$

thus the i-th column of $\mathbf{A}$ is represented by the i-th row of $\mathbf{V}$, which is the R_2 formulation given in (4).

On the other hand, the R_1 formulation of LSA is also presented in many, fairly widely cited, publications, for example (Bartell et al., 1992; Papadimitriou et al., 2000; Kontostathis and Pottenger, 2006), the relevant parts of which are below briefly noted.

In the notation of (Bartell et al., 1992) the reduced rank SVD of a term-by-document matrix is $\mathbf{U}_k\mathbf{L}_k\mathbf{A}_k^T$, and their definitions of document and query representations are (p162)

> *row i of* $\mathbf{A}_k\mathbf{L}_k$ *gives the representation of document i in k-space. ...Let the query be encoded as a row vector* $\mathbf{q}$ *in* $\mathcal{R}^t$*. Then the query in k-space would be* $\mathbf{q}\mathbf{U}_k$

These coincide, modulo notational differences, with the R_1 formulations of (3) and (1).

In the notation of (Papadimitriou et al., 2000) the reduced rank SVD of a term-by-document matrix is $U_kD_kV_k^T$. Then concerning document representation they have (p220)

> *The rows of V_kD_k above are then used to represent the documents. In other words, the column vectors of A (documents) are projected to the k-dimensional space spanned by the column vectors of U_k*

which coincides, modulo notation, with the R_1 formulations in (3) and (1).

In the notation of (Kontostathis and Pottenger, 2006), the reduced rank SVD of a term-by-document matrix is $T_kS_k(D_k)^T$, with T_k and D_k used in place of $\mathbf{U}_k$ and $\mathbf{V}_k$. Their definition of query representation and document representation is (p3)

> *Queries are represented in the reduced space by $T_k^T q$. ...Queries are compared to the reduced document vectors, scaled by the singular values ($S_kD_k^T$)*

These column vector formulations would be a row vector formulation qT_k and D_kS_k, which, modulo notational differences are the R_1 formulations of (1) and (3).

On the basis of these works, there would appear to be an R_1-vs-R_2 ambiguity in the formulation of LSA, possibly a fairly wide-spread one. Let us now return to the HCI/Graph example from (Deerwester

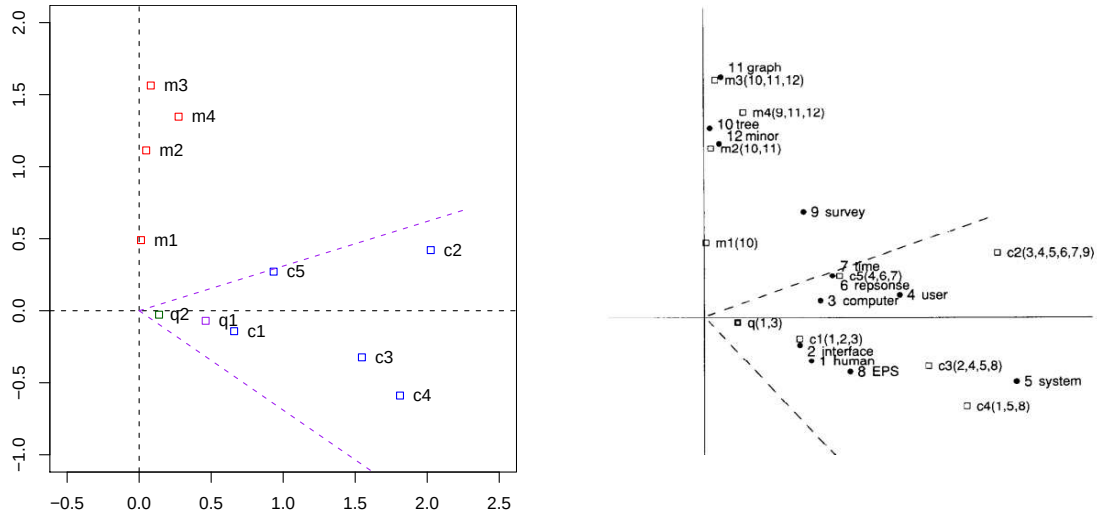

Figure 2: *Left*: shows the R_1 representation of the $c1$–$c5$ and $m1$–$m4$ documents from Figure 1 and as $q1$ and $q2$, the R_1 and R_2 representations of the query from the text; also shows cosine 0.9 cone around $q1$ *Right* screen shot from (Deerwester et al., 1990), also showing $c1$–$c5$, $m1$–$m4$ and the query

et al., 1990). We shall see that there is ambiguity as to whether it is the R_1 or R_2 representation that is intended by the text of (Deerwester et al., 1990).

Recall that Figure 1 showed the basic term-by-document matrix for this example, and the component matrices of its rank-2 reduced SVD. The two dimensional nature of the reduced representations allows for simple plotting. The left part of Figure 2 plots the 9 documents using the R_1 projection, based on the rank-2 reduced SVD shown in Figure 1. The positions of the documents are indicated by boxes labelled 'c' and 'm'.

To the right in Figure 2 is a reproduction of the figure on p397 of (Deerwester et al., 1990). Their plot shows (amongst other things) *a* reduced representation of the documents, as boxes labeled c1-c5 and m1-m4. Whether their plot is intended to depict the documents in the R_1 or R_2 representation is moot: the axes in the original plot are not labeled. We have endeavoured to scale the two plots in such a way that the document vectors are identically placed in the two pictures.

In (Deerwester et al., 1990), they consider the query 'human computer interaction'. Given the terms chosen for the document vectors, the unreduced vector $\mathbf{q}$ is $[1,0,1,0,0,0,0,0,0,0,0,0]$. Applying the R_1 definition (1), we have $R_1(\mathbf{q}) = [0.46, -0.07]$ and applying the R_2 definition (2), we $R_2(\mathbf{q}) = [0.14, -0.03]$. We have plotted these alternative reductions of $\mathbf{q}$ also in the left part of Figure 2, where they are are shown as $q1$ and $q2$.

In the plot reproduced from (Deerwester et al., 1990) a reduced image of the same query vector was depicted. Considering their placement of the representation of the query relative to the document representations, and comparing it to our own placment of its R_1 and R_2 representation relative to the R_1 representations of the documents, it seems the only interpretation that can be put on the plot from (Deerwester et al., 1990) is that it shows the *documents* in the R_1 projection, but the *query* in the *the* R_2 *projection*. Note that because the R_2 representation is simply a scaling of the R_1 representation, with a different scaling of each dimension, the relative position of the document and query points in the plot from (Deerwester et al., 1990) is not consistent with all points being shown in the R_2 representation. To emphasize this, Figure 3 gives the plot of documents and query in the R_2 representation, again in such a way that the documents are positioned identically to the plot from (Deerwester et al., 1990) and one can see that the query representations are differently placed.

This seeming equivocation between the R_1 and R_2 projection occurs in the text of (Deerwester et al., 1990) also. In their notation the SVD of the term-by-document matrix is TSD', thus using T and D in place of our $\mathbf{U}$ and $\mathbf{V}$. Concerning document representation, there is (p398)

> *'the rows of the reduced matrices of singular vectors are taken as coordinates of points representing the documents and terms in a k dimensional space'*

As we noted above, identifying the *rows* of $\mathbf{V}_k$ as the reduced representations of documents means adopt-

ing the R_2 representation (see (4)). Concerning a query, if its unreduced representation as a column vector is X_q, they give its reduced representation as $X_q' T S^{-1}$, which again, modulo notation, is the R_2 formulation (see (2))

On the other hand p399 has (recall their 'D' is $\mathbf{V}_k$ in our notation)

> *so one can consider rows of a DS matrix as coordinates for documents, and take dot products in this space . . . note that the DS space is just a stretched version of the D space*

As we noted above, in equation (3), this amounts to adopting the R_1 representation for documents.

4 Contrasting outcomes

Setting aside these expository details, it is more important to know whether system outcomes may change according to which representation, R_1 or R_2, is adopted. The LSA dimensionality reduction technique has been deployed in quite a variety of contexts and in each one might investigate the effect of whether R_1 or R_2 is adopted. In this section we consider two such contexts.

The first context is the original one presented in (Deerwester et al., 1990): the issue is which documents should count as similar to a given query under the two representations. Returning again to the HCI/Graph example, in our R_1 depiction of the documents and query that is the left-hand plot of Figure 2, we have also shown a cone which encloses the points that have a cosine value of 0.9 or higher to $R_1(\mathbf{q})$. Figure 3 shows the documents and the query $\mathbf{q}$ instead in the R_2 projection, and shows the corresponding cone around $R_2(\mathbf{q})$.

On the R_1 projection, the representations of c1–c5 are all included in the cone around the query. In (Deerwester et al., 1990), this inclusion of all the HCI document representations (c1–c5) within cosine 0.9 of the given query is also noted, notwithstanding the above-noted R_1-vs-R_2 ambiguities concerning their plot of the data. As Figure 3 shows, on the R_2 projection (of queries and documents), the representations of c5 and c2 are *not* included. Note that the visual similarity of Figure 3 and the left part of Figure 2 is a bit misleading, as the values on the axes in the R_2 representation in Figure 3 are considerably smaller than those on the axes in the R_1 representation, (by a factor of 0.29 for the first dimension, and 0.39 for the second).

Another context in which LSA dimensionality reduction has been used is in *word clustering*. The aim

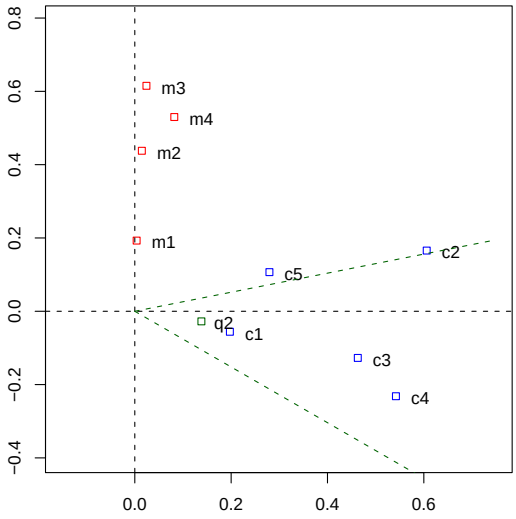

Figure 3: The R_2 representation of the $c1$–$c5$ and $m1$–$m4$ documents from Figure 1 and as $q2$ the R_2 representation of the query from the text; also shows cosine 0.9 cone around $q2$

is to cluster occurrences of an ambiguous word into coherent clusters, clusters each of which reflect a distinct sense of the word. To this end each occurrence of an ambiguous term at a position p is represented by its so-called *first-order context vector*, $\mathbf{C^1}(p)$, a vector which for a given uni-gram vocabulary Σ_f records for each unigram its frequency in the window between $p-10$ and $p+10$.

We conducted an experiment making use of the so-called HILS dataset, which consists of manually sense annotated occurrences of the four words *hard-interest-line-serve*. Thus for each word there is a sub-corpus consisting of its occurrences, and for each word, a 60% subset was taken and clustered by the *k*-means algorithm, where *k* is set to the number of attested senses of the given word. The clustering is evaluated using the remaining 40% test-set: these items are first assigned to their nearest cluster centres and then for each possible sense-to-cluster mapping, a precision score on the test set is determined, with the maximum of these reported as the final score.

All so-called non-stop unigrams constitute the features of the context vectors. making the context vectors *high dimensional*: around 10^4, and before clustering SVD-based dimensionality reduction was applied. Each of the occurrences of an ambiguous word is thus treated as a miniature 20 word document to give a term-by-'document' matrix, the dimensions of which were of the order of $10^4 \times 10^3$. Then from this, the reduced rank SVD was calculated for various percentages of the original dimension size, between 1% and 14%. To give an idea of absolute numbers,

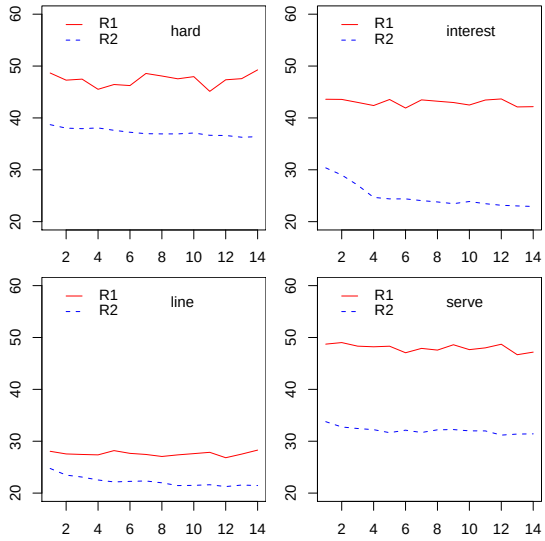

Figure 4: Unsupervised clustering results using R_1 and R_2 representations. Vertical axis is accuracy, horizontal axis is % reduction of dimensions.

for the various words the 10% reduction level corresponds to a dimensionality of 856(*hard*), 494(*interest*), 1297(*line*) and 1304(*serve*). From these reduced SVDs, the thereby defined R_1 and R_2 versions of the context vectors were then used. Figure 4 gives the results (the 60-40 split was randomly made, and repeated 4 times, with the figure summarising the outcomes over these splits).

This confirms the indications from the tiny 2-dimensional HCI/Graph example, namely that the outcomes under the R_1 and R_2 representations are not identical. In this word clustering context, at each level of reduction, the outcomes with the R_1 and R_2 representations are clearly different. In fact there is a persistent pattern of the R_1 representation giving consistently better outcomes than the R_2 representation.

5 Conclusion

We have shown that there is a discrepancy amongst researchers concerning the precise dimensionality reduction technique to which they give the name 'LSA'. The R_1 representation is defined by equations (1) and (3) whilst the R_2 representation is defined by (2) and (4), and these alternatives give a different geometry to the space of reduced representations, manifesting itself in different nearest-neighbour sets. We showed that, unsurprisingly, this can lead to different system outcomes according to which representation, R_1 or R_2, is adopted in a given system.

We have not argued for one of these representations over the other one. Whilst Theorem 2 establishes that $\hat{\mathbf{A}} = \mathbf{U}_k \times \mathbf{S}_k \times \mathbf{V}_k'$ is the optimum rank-k approximation of $\mathbf{A}$ in the sense of minimising the sum of squared differences between corresponding matrix positions, there is a good deal of conceptual clear water between this and consequent 'optimality' of a particular SVD-based reduction of document vectors in a particular system. This is testified to by the range of attempts there have been to give a theoretical justification for an observed system 'optimality' of a given deployed SVD-based reduction. Therefore the R_1 and R_2 alternatives are as theoretically motivated (or unmotivated) as each other, at least at first glance, and there is some merit in putting both to the test empirically. What is beyond doubt, though, is that these R_1 and R_2 altneratives are genuinely different and will not always give the same empirical outcomes.

Acknowledgements

This research is supported by the Science Foundation Ireland (Grant 07/CE/I1142) as part of the Centre for Next Generation Localisation (www.cngl.ie) at Trinity College Dublin.

REFERENCES

Bartell, B. T., Cottrell, G. W., and Belew, R. K. (1992). Latent semantic indexing is an optimal special case of multidimensional scaling. In *Proceedings of the Fifteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 161–167. ACM Press.

Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., and Harshman, R. (1990). Indexing by latent semantic analysis. *JOURNAL OF THE AMERICAN SOCIETY FOR INFORMATION SCIENCE*, 41(6):391–407.

Gong, Y. and Liu, X. (2001). Generic text summarization using relevance measure and latent semantic analysis. In *SIGIR*, pages 19–25.

Kontostathis, A. and Pottenger, W. M. (2006). A framework for understanding latent semantic indexing (lsi) performance. *INFORMATION PROCESSING AND MANAGEMENT*, 42(1):56–73.

Landauer, T., Foltz, P., and Laham, D. (1998). An introduction to latent semantic analysis. *Discourse Processes*, 25(1):259–284.

Manning, C. D., Raghavan, P., and Schütze, H. (2008). *Introduction to Information Retrieval*. Cambridge University Press.

Papadimitriou, C. H., Raghavan, P., Tamaki, H., and Vempala, S. (2000). Latent semantic indexing: A probabilistic analysis. *J. Comput. Syst. Sci.*, 61(2):217–235.

Rosario, B. (2000). Latent semantic indexing: An overview. Technical report, Berkeley University. available at http://people.ischool.berkeley.edu/~rosario/projects/LSI.pdf.

Zelikovitz, S. and Hirsh, H. (2001). Using lsi for text classification in the presence of background text. In *Proceedings of CIKM-01, 10TH ACM International Conference on information and knowledge management*, pages 113–118. ACM Press.