LEXICAL DISAMBIGUATION IN MACHINE TRANSLATION WITH LATENT SEMANTIC ANALYSIS

An Honors Thesis

Presented to

The Faculty of the Department of Computer Science

Washington and Lee University

In Partial Fulfillment Of the Requirements for

Honors in Computer Science

---

by

Elizabeth E. Davis

May 2006

To my mother, for teaching me how to write.

Contents

ACKNOWLEDGMENTS

My deepest gratitude to my advisor Simon Levy for all his help and advice, for all his time, and for putting up with me even while he was in Scotland.

ABSTRACT

This paper examines a possible solution to the problem of disambiguating polysemous nouns in machine translation. Latent Semantic Analysis (LSA), a statistical method of finding and representing word sense, is used to differentiate between the different meanings of ambiguous words according to the given context. A collection of training texts are sorted according to polysemous word and meaning. A word-by-text matrix is created from this data and transformed by the LSA method, creating vectors for each text defining it in terms of the (non-polysemous) words that appear in it. These representations of textual meanings are compared to the context of an ambiguous word to determine the most similar meaning. The viability of this LSA model is compared with a simple Bayesian probability model.

LEXICAL DISAMBIGUATION IN MACHINE TRANSLATION WITH LATENT SEMANTIC ANALYSIS

Chapter 1

Introduction

While researchers have made great progress in the field of machine translation, no existing general-purpose translation program can yet rival a human translator. One of the greatest obstructions to the development of a professional-quality machine translator is the task of correctly interpreting an ambiguous word. Such words with multiple possible meanings are called *polysemous*. While a human translator usually finds it simple to differentiate among possible translations for a word in a familiar language, machine translation systems such as those freely available online are notoriously faulty in this area of disambiguation. A classic example dates from the 1960s: given the idiom *The spirit is willing, but the flesh is weak*, an early English-Russian translator transformed the sentence to Russian and back again, whereupon it emerged as *The wine is good, but the meat is spoiled.*

Solutions for this flaw in machine translation depend on analysis of the context surrounding the ambiguous word. Given an isolated word such as *bow*, a human has no better chance than a computer of guessing whether the word is intended to signify a weapon or a knot of ribbons. It is only when some form of context is provided that a human can understand the idea behind the passage, and thus choose the correct meaning of the ambiguous word. Even then, if the vocabulary of the passage and its possible relationships to a target word are unfamiliar, the human reader can deduce nothing about the target's meaning.

The common approach to solving the disambiguation problem is to give the program

processing a passage a simulated understanding of the passage's meaning, as represented by the context of the polysemous word. While complete comprehension on the human level is currently impossible for the computer, a practical imitation thereof can be gained through statistical language processing, as the program represents "meaning" in terms of words related to the topic. Theoretically, if a translation program has a body of knowledge about words and inter-word relationships comparable to a human's, and if it can use this information to represent the historical likelihood of a polysemous word in a particular context conveying a particular meaning, then the program can choose the correct meaning of such a word if provided a familiar context.

Such a program should be able to choose between the two possible meanings of *bat* in the sentences *The bat flew from the cave* and *The batter swung the bat*, if it has been trained on a selection of texts involving baseball and chiropterology. This simulation of human disambiguation capabilities is often achieved through creation of a matrix counting which words appear in which training texts. The two meanings of the word *bat* could then be represented by the appropriate column vectors, respectively containing collections of baseball- and flying-nocturnal-mammal-related words.

In such a design, a vast quantity of training material is required for the representation to be effective, and the knowledge database may grow prohibitively large. In addition, a simple word-by-text matrix does not fully represent all the connections between words which the human mind is capable of drawing. While this design counts all words that appear near each polysemous word, the matrix offers no immediate information on the second generation of relationships: all the words near which the "context words" tend to appear. Humans can draw such connections readily, even to the nth degree.

A human who has read an article on baseball which contained the word *bat* but not the word *umpire*, and another article (perhaps on the woes of the refereeing life) which contained references to an umpire in connection with baseball, but never mentioned the word *bat*, would probably, if asked to translate the sentence *The umpire had a bat*, assume that both words had something to do with baseball, and that therefore the polysemous

word referred to the baseball bat rather than the animal. If a computer had made a simple word-by-text co-occurrence matrix of both articles, it would see no such association between *bat* and *umpire*. Possibly programs using such a co-occurrence matrix could implement a function examining the contexts of all words appearing in the context to some degree, but this would be a time-consuming and highly complicated method. Through matrix manipulation, Latent Semantic Analysis offers a solution to this problem - and may offer further benefits as well.

Latent Semantic Analysis (LSA, also known as Latent Semantic Indexing, or LSI) is a well-developed technique for representing word and passage meanings as vectors in a high-dimensional "semantic" space. Through application of linear algebra methods singular value decomposition and dimensional reduction, a co-occurrence matrix is transformed to better reflect the "latent," or hidden, similarities between words and documents. The technique can be used to determine the most likely meaning of a polysemous word from some given context by comparing a vector constructed from that context with document vectors. Vectors representing similar passage meanings should be near each other, as LSA is said by some of its creators to "closely approximate human judgments of meaning similarity between words." [7]

Most studies to date have focused on LSA's applications in searching and document retrieval. In this field, LSA has been shown to offer a marked improvement over other methods. [2] Cross-language information retrieval search results in languages differing from the query has also received attention, [11] as has LSA's use in language modeling. [5] LSA has also been tried with human vocabulary synonym and word-sorting tests, in the course of research on how well LSA models human conceptual knowledge, and scored not far below group norms. [7] On the practical side, LSA has been used in a commercial product called the "Intelligent Essay Assessor," which evaluates students' knowledge and writing skills. [8]

However, at least one study has addressed LSA's potential in machine translation, specifically in dealing with polysemy in Korean-English translation. [6] This study did not use the general context of an ambiguous word, but rather considered a single argument word in a

specific grammatical relationship, such as *subject-verb*, between the argument and the target polysemous word. The correct meaning of the target was drawn from a dictionary storing examples of argument words. If the given argument did not appear in the dictionary, the correct translation class was that of the example word most similar to the argument. The project used an LSA model to determine this similarity by finding the example word whose vector representation was closest to the argument word's, under the theory that words of similar meaning are "close" in the semantic space. Thus this LSA model relied on vector representations of individual argument and example words rather than on representations more closely associated to the meaning of the polysemous words themselves.

In contrast, the LSA model in this paper considers the entire context of an ambiguous word. The model represents each document on which it was trained as a separate vector according to the words contained in that text. Each such vector is tagged according to the particular meaning of the polysemous word contained in it. Thus, to disambiguate a target word, the vector representation of the word's whole context is compared to all the vectors of training documents for that word, and the closest is chosen as most similar. Such a design removes the necessity for a dictionary of grammatical relationships, for part-of-speech parsing of the context, and for any other exterior input. Theoretically, this contained disambiguation module could be attached to a machine translator for use in English-to-any-language.

Chapter 2

The LSA Model

To demonstrate its disambiguation capabilities, this LSA model deals only with polysemous nouns, leaving ambiguities between parts of speech for later research. Construction of the model requires no thesaurus, dictionary, or set of rules as input: only English text segments each related to a particular ambiguous noun, tagged according to the meaning of this noun. For instance, an article on baseball and an article on vampire bats could be training texts respectively identified by tags "bat-1" and "bat-2," where the numerals 1 and 2 are somewhere connected to the meanings "baseball" and "animal." Such specification of separate meanings and classification of texts are themselves challenging tasks, and will be addressed later.

Given such a collection of labeled texts, the content words are extracted and formed into a matrix in which each row stands for a unique word and each column for a text, as shown in Table 2.1, such that if the word i occurs in document j, cell $a_{i,j}$ will have value 1 (or some value indicating the number of times i appears in the text) and otherwise 0. An alternate approach would be for each column to stand for a unique word-meaning, with summed wordcounts from all associated training texts. However, since the following matrix manipulations work better on square or nearly-square matrices, and the number of training texts is likely to be greater than the number of word-meanings and thus closer to the number of words contained in those texts, each text is represented in its own column.

So each training document is represented as a vector in an m-dimensional space, where m is the total number of unique words across all the documents. An individual word-meaning such as "bat-1," with t training texts, can be viewed as the sum or the average of the t associated vectors.

A point of interest here is the exclusion of common words conveying relatively little information about the polysemous nouns. This will be discussed in more detail later: in the example below, only the italicized words are considered "interesting."

Training Text Examples (Table 2.1):

bat-1A: The *baseball flew* past both the bat and the *catcher* for another *strike.*

bat-1B: The *player hit* the *catcher* with his bat after the third *strike.*

ball-1A: In *baseball*, the *umpire* sometimes gets *hit* with the ball.

bat-2A: The bat *flew* out of the *cave* on *wings* as *black* as *night* itself.

bat-2B: Bats *hunt* from their *cave* on *wings* of *terror.*

ball-2A: They *danced* to the *music* all *night* at the ball.

The unique capabilities of an LSA model lie in the transformation of this simple matrix (A) with singular-value decomposition (SVD) and a successive dimensional reduction to better capture relationships between words and texts. The SVD for an $m \times n$ matrix A with $m > n$ rewrites the matrix as

$$A = UDV^T \tag{2.1}$$

where U and V are orthogonal matrices with columns of unit length (that is, $U^TU = I_m$, $V^TV = I_n$) and D is an $m \times n$ diagonal matrix. [3] D holds the unique singular values of A, which are the positive square roots of the nonzero eigenvalues for A^TA. It has been proven that $d_{11} > d_{22} > ... > d_{rr} > 0$, where r is the rank of A. If $r < n$, the remaining diagonal entries of D are 0. [10]

	Bat-1A	**Bat-1B**	**Ball-1A**	**Bat-2A**	**Bat-2B**	**Ball-2A**
Baseball	1	0	1	0	0	0
Flew	1	0	0	1	0	0
Catcher	1	1	0	0	0	0
Strike	1	1	0	0	0	0
Player	0	1	0	0	0	0
Hit	0	1	1	0	0	0
Umpire	0	0	1	0	0	0
Cave	0	0	0	1	1	0
Wings	0	0	0	1	1	0
Black	0	0	0	1	0	0
Night	0	0	0	1	0	1
Hunt	0	0	0	0	1	0
Terror	0	0	0	0	1	0
Danced	0	0	0	0	0	1
Music	0	0	0	0	0	1

Table 2.1: Example of a simple co-occurrence matrix on six training documents covering four word-meaning pairs.

"The singular values of a matrix A are precisely the lengths of the semi-axes of the hyperellipsoid E defined by $E = \{y \mid y = Ax, \quad \| x \| = 1\}$." [3] In other words, the ith singular value can be said to represent the amount of variation in A along the ith axis of the m-dimensional space.

Thus if $A = UDV^T$ is reconstituted using only the first k singular values and the first k columns of U and V, where $k < n$ the result $\hat{A}$ is the optimal projection of A into a k-dimensional space, [9] which is the closest approximation of rank k to A in the least-squares sense. This truncation, or dimensional reduction, preserves information from the axes with the most significance.

$$\hat{A} = U_{mxk} D_{kxk} V^T_{nxk} \tag{2.2}$$

Theoretically, $\hat{A}$ could be represented as a $k \times n$ matrix if different axes were chosen, but by keeping the original axes, it has the same rows and columns as A — a necessity for use in the disambiguation process.

One way to think of the effect of this reduction to k dimensions is to consider the

inevitable error or "noise" in the original word-document matrix. Because the training space is limited, there will necessarily be words that never occur in contexts where they should - no occurrence of the word umpire in the training texts for *bat*, for instance - and there will be words which by chance appear disproportionately often or infrequently in certain contexts as compared to their distribution in the "semantic space" of the whole written language. LSA can be viewed as a way to filter out this noise.[10] Assuming the error is evenly distributed in all directions, the smallest singular values must correspond to the dimensions in which the error-to-information ratio is largest. By reconstituting the decomposed matrix with only the k largest singular values, the error is reduced while the dimensions containing the most information about A are retained.

The resulting structure estimates what the word-document relationships should be. [1] The cell values in the approximation $\hat{A}$ depend upon the whole of the original matrix A, since the elements from which they were calculated had been computed to describe the original rows and columns as vectors of derived orthogonal factor values. Although some information has been removed, the underlying pattern remains, so that the values of the new matrix reflect all the original cell values to some extent. Rather than simply representing word-text occurrence, a cell $a_{i,j}$ can be thought of as an estimation of the average occurrence of word i in all documents related to the other words associated with document j. [7] Co-occurring words have been mapped onto the same dimensions, causing similar documents to show similar values for words in their shared "meaning space."

The matrix illustrated in Table 2.1 has been reduced into two dimensions. The result is shown in Table 2.2. Observe that while *catcher* retains a value close to 0 for the columns not associated with baseball, it now has a comparable positive value in all three of the first columns, despite the fact that it appears only in the first and second documents. Likewise, *umpire*, appearing only in the training text for Ball-1, is now positively associated with the Bat-1 texts. The reconstruction reflects the degree of variance in the original data: since the first and second texts share words with the third, the vector representations in the first three columns are similar. Similarly, the negative value in $a_{1,6}$ (*baseball*, Ball-2A) reflects

the lack of word overlap between the training texts for the second meaning of *ball* and the baseball-related documents.

	Bat-1A	Bat-1B	Ball-1A	Bat-2A	Bat-2B	Ball-2A
Baseball	0.6687	0.6668	0.3750	0.0967	-0.0433	-0.0041
Flew	0.5426	0.4032	0.2228	0.7279	0.4520	0.1724
Catcher	0.8532	0.8573	0.4823	0.0926	-0.0783	-0.0136
Strike	0.8532	0.8573	0.4823	0.0926	-0.0783	-0.0136
Player	0.4209	0.4364	0.2459	-0.0177	-0.0862	-0.0239
Hit	0.6574	0.6823	0.3845	-0.0313	-0.1373	-0.0382
Umpire	0.2364	0.2459	0.1386	-0.0136	-0.0511	-0.0144
Cave	0.1182	-0.1039	-0.0647	1.0617	0.7758	0.2820
Wings	0.1182	-0.1039	-0.0647	1.0617	0.7758	0.2820
Black	0.1103	-0.0177	-0.0136	0.6176	0.4441	0.1621
Night	0.1206	-0.0416	-0.0280	0.7797	0.5639	0.2055
Hunt	0.0079	-0.0862	-0.0511	0.4441	0.3317	0.1198
Terror	0.0079	-0.0862	-0.0511	0.4441	0.3317	0.1198
Danced	0.0103	-0.0239	-0.0144	0.1621	0.1198	0.0434
Music	0.0103	-0.0239	-0.0144	0.1621	0.1198	0.0434

Table 2.2: The reconstituted matrix $\hat{A}$ after application of SVD and reduction to two dimensions.

Then to calculate the most probable meaning of an instance of the word *bat*, the given context is transformed into a vector in the same semantic space: for each row in the matrix, a value is assigned to the corresponding component of the context vector Q indicating whether the word connected to that row appears in the context. The meaning associated to the nearest text-vector is chosen via cosine computation between Q and all the column vectors V where the column is tagged with the polysemous word in question:

$$cos(\Theta(Q,V)) = (Q^T V)/(\| Q \| \cdot \| V \|) = (Q^T V)/(\sqrt{Q^T Q} \cdot \sqrt{V^T V}) \tag{2.3}$$

The cosine computations for the sentence *The bat flew from its cave* are shown in Table 2.3. As the smallest calculated angle is that between Q and a vector belonging to the second meaning of *bat* (column 4: text bat-2A), the second meaning is correctly chosen. Note that although neither the word *flew* nor the word *cave* appeared in training document ball-2A (column 6), the LSA method nonetheless calculated a relatively small angle for this

vector, due to the overlap of the word *night*.

Text	Column	Θ
bat-1A	1	74.13°
bat-1B	2	82.78°
ball-1A	3	83.22°
bat-2A	4	52.13°
bat-2B	5	54.32°
ball-2A	6	53.55°

Table 2.3: Angles calculated between the context vector Q and the column vectors of $\hat{A}$, where 0 indicates identical vectors and 90 orthogonal or completely dissimilar vectors.

An alternative method of meaning selection is to take the average of all the computed angles for each meaning, choosing the smallest average rather than the overall minimum.

Note also that before the linear decomposition is carried out, log-entropy weighting can be applied to the matrix, modifying values to indicate their frequency. Words appearing across many documents are less strongly associated with meaning and thus likely to be of less significance in telling polysemous words apart. [9] This has been found in the past to significantly improve performance of other LSA applications such as information retrieval. [7] Weighting has not been implemented here, for the sake of simplicity, readability, and computational time constraints. However, as such a transformation is said to emphasize meaning-bearing words, [7] presumably it would only have a positive effect on this LSA model.

The simple example given above demonstrates the LSA model's ability to disambiguate polysemous nouns, but in this case at least, the same result could easily have been achieved using the original co-occurrence matrix with a naïve Bayesian probability calculation, an approach common in the field of word sense disambiguation. [9]

Chapter 3

The Bayesian Model

For ease of use with the Bayesian probability calculation, the co-occurrence matrix originally constructed from the training texts (Table 2.1) is condensed so that each column represents the entire collection of word counts from all texts associated with a particular meaning. An $m \times n$ matrix becomes $m \times w$ where w is the total number of meanings over all the polysemous words, so that cell $a_{i,j}$ now holds the number of times word i appeared across all the training texts for meaning j. To disambiguate, a probability is calculated for each column mapped to a potential meaning of the target noun, as to how likely the given context words are to appear in that column.

Bayes' Rule is used for this probability calculation, the particular implementation based on code from a spam-identification program. [4] The algorithm works as follows.

For every word in the context which appears in the relevant part of the matrix, let its adjusted frequency f_m be its value in the column corresponding to meaning m, divided by the number of training texts *ntxts* used for that meaning. If this frequency is greater than 1, as might occur if the word is used more than once in every document, it is adjusted downward to 1. Then for meaning m, the probability $p_{m,i}$ that the ith word occurs in connection with that meaning is its corresponding frequency divided by its spread across all k meanings. If this is at 0% or 100%, it is adjusted respectively to 1% or 99% in consideration of the limited certainty of the knowledge base. That a word has not yet

appeared in texts of meaning m is no guarantee that it can never do so.

$$p_{m,i} = \min(.99, \max(0.01, f_{m,i}/(f_{1,i} + f_{2,i} + ... + f_{k,i}))) \tag{3.1}$$

Words which do not appear in the matrix may be assigned a probability of 50%, or left out entirely. Then the overall probability P_m that all i words in the context appear near meaning m - based on their past distribution among the training texts - is

$$P_m = \Pi p_{m,i}/(\Pi p_{1,i} + \Pi p_{2,i} + ... + \Pi p_{k,i}) \tag{3.2}$$

For the test sentence *The bat flew from its cave*, word probabilities for *flew* and *cave* are respectively at 0.5 and 0.99 for meaning 2 (the nocturnal flying mammal). The probability P_2 is $(0.5 \times 0.99)/(0.5 \times 0.01 + 0.5 \times 0.99) = 0.99$, quite correctly.

Variations on this calculation, such as using only some number of "interesting" probabilities - the two furthest from 0.5, for instance - may improve results. For this project, since the LSA model examines the whole context, the Bayesian model also takes all included words into consideration.

In general, given context words of which one or more have appeared in the relevant training texts, the probability calculation can reliably choose the correct meaning. However, the Bayesian method has limitations. Only words which appeared in the training texts of the polysemous word in question will be useful in the disambiguation process. All the other words in the matrix are ignored, which does not accurately simulate the human ability to make connections among words. Thus the co-occurrence matrix and Bayesian calculation has more limited disambiguation potential than does the LSA model.

Chapter 4

Implementation and Brief Comparison

The Matlab programs written to create the co-occurrence and LSA matrices work with simple text files: a flexible approach using separate files for each training text, and a master document relating each file to its particular meaning. From this master document is drawn the list of polysemous words and their meanings which is stored in parallel with the matrix so that each column of the matrix corresponds to the word and meaning related to the text of that column. Likewise, a list of the context words is stored in the same order as the rows representing those words. The number of training texts used for each word-meaning is also stored for use in the Bayesian probability calculation once the columns for each word-meaning have been merged. Thus the disambiguation program can locate the columns corresponding to the target polysemous word, and the rows corresponding to the words of the context, as is required by the algorithm.

On a restricted, deliberately-constructed set of training data (2-5 polysemous words, 20-100 context words), the Bayesian and LSA methods perform much as expected. Both can successfully disambiguate obvious sentences, such as *The bat flew from the cave*, although with such limited training data there is a chance that the LSA method may err in drawing non-existent connections. In Table 2.2, note the strong correlation between *music* and

"Bat-2," a potential source of error. Unsurprisingly, neither method can effectively handle a confusing sentence such as *The umpire carried the bat into the cave.*

LSA performs significantly better than the Bayesian method on disambiguation of test sentences chosen for context not contained in the texts associated to the polysemous noun in question. On the training set of Table 2.1, the LSA method can successfully disambiguate *bat* in *The umpire caught the bat*, while the Bayesian method, examining only overlap with words appearing in the "Bat-1" and "Bat-2" training texts, cannot determine a meaning at all. On sentences whose context belied their meanings (such as *The bat flew over the baseball diamond*), both methods performed equally poorly — unsurprisingly, as a human translator might also have difficulty with such a sentence.

As far as the construction of the matrices for the Bayesian and LSA methods goes, the former is superior in terms of scaling up the amount of training data. Not only does it require less computational time and less space in memory (all cell values are positive integers, and a sparse matrix can be used), but additional training texts can be easily added, incrementing values in the appropriate column. Adding documents to the LSA matrix is more difficult, as every cell value depends upon the entirety of the original co-occurrence matrix. However, there is an alternative to re-computing the entire matrix: the vector of a new document can be folded into the space by multiplication with one of the SVD matrices: for a matrix $M = UDV^T$ projected into a k-dimensional space and a new text vector t, t's representation in the lower-dimensional space is $U_n \times k^T t$. [9] A drawback is that this requires storage of the reduced matrix U.

In this project, at least, the LSA model is also limited by the efficiency of the computation program and the power of the computer manipulating the matrix. For large test sets (over 10000 words), the computer's memory could not handle the particular non-optimized implementation of the SVD and dimensional reduction.

Chapter 5

Applications in Translation

The construction of the LSA and Bayesian models requires only English texts, but further information must be supplied for actual use in a machine translator. A possible advantage of these models is that they could be used for any foreign language. The meanings "bat-1" and "bat-2", denoting the general concept of the object thrown in baseball and the flying mammal, can be assigned the appropriate translations in French, Italian, or any language that actually includes such nouns. All that is required is a mapping from the meaning tags used to classify the training texts to the appropriate translation.

The only point of the process in which the specific language to be translated into matters is the assignment of different meanings to the polysemous nouns. This can be based on the non-English language: *bat* is divided into two meanings because French contains the separate translations *batte* and *chauve-souris* for it. If instead meanings are assigned to the English words according to all different definitions in an English dictionary, for instance, then the appropriate translations in any language can be mapped to the meanings. If the target language does not actually differentiate between some of the meanings (French uses the word *grue* for both the flying crane and the construction crane), no harm has been done. The question of assigning such meanings and matching translations to them is addressed in a later section.

Assuming that a mapping has been made between the words and meanings represented

in the model and their translations in the target language, this disambiguator can be incorporated into a machine translator. Whenever a polysemous word occurs in the text to be translated, its most likely meaning (and thus translation) can be calculated by the LSA or Bayesian model.

This project connects the LSA model to a free online translator to demonstrate its translation capabilities, using French as the target language. The program takes a user-input sentence containing a polysemous word included in the model, and acquires a French translation through GoogleTranslate (GT), which supplies only one meaning for most polysemous English words: regardless of context, "bat" is always *batte*, the baseball bat. A "repair" function calculates the meaning of the polysemous word using the English context and the LSA model, and replaces the French translation of that word if it does not match the most probable meaning.

As currently implemented, this demo has a number of flaws which could probably be removed by incorporating the disambiguation module more fully into the translator. The replacement of an incorrect French translation relies on a list of GT's default French translations for the polysemous words and their plurals, which must be generated and attached to the disambiguator. Although this works, it is an awkward addition to the process.

Far more important is the matter of necessary alterations to the French translation if the target word must be replaced. As French nouns are gendered, and articles, adjectives, and some verb forms must agree with the gender of the noun they modify, the simple substitution of a differently-gendered correct translation for the default will result in a grammatically erroneous sentence. Without access to the inner workings of the translator, where the parts of speech and grammatical relationships are presumably defined, the disambiguator cannot fully correct for such changes. However, since gender indicators are included in the model's mapping of meanings to translations, correction of contiguous articles could easily be added to the module as it stands.

The current version of the demo can handle plurals of polysemous words, but has not been implemented for possessive forms. Nor does the current version handle multiple poly-

semous words in one target sentence: it chooses only the first word appearing in the model's set of ambiguous words. This opens the door for erroneous identifications - where the word *bats* is in fact a verb, for instance. Again, a fuller meshing with the translator could easily solve this problem.

Chapter 6

Automation of Data Acquisition

An attempt to automate the acquisition of training data and of polysemous words and meanings led to less-than-satisfactory results and underlined some of the deepest problems with machine translation: the complications of language itself, and the difficulty of using texts made for humans in computations.

A list of polysemous words and their meanings, to be covered by the model, was extracted from a machine-readable dictionary (MRD) from Project Gutenberg. This required analysis of the dictionary entries to find nouns with more than one definition. A sample entry (for *bat*) is provided below. Since part-of-speech tags were provided, the nouns could be distinguished from verbs and adjectives of the same form. Aside from all technical difficulties of processing the text, the great problem lay in the specification of "meanings" in the dictionary:

> **Bat**, *n.* [OE. *batte, botte*, AS. *batt*; perhaps fr. the Celtic; cf. Ir. *bat, bata*, stick, staff; but cf. also F. *batte* a beater (thing), wooden sword, *battre* to beat.]
> 1. A large stick; a club; specifically, a piece of wood with one end thicker or broader than the other, used in playing baseball, cricket, etc.
> 2. (Mining) Shale or bituminous shale. *Kirwan.*
> 3. A sheet of cotton used for filling quilts or comfortables; batting.
> 4. A part of a brick with one whole end.

> **Bat**, *n.* [Corrupt. from OE. *back, backe, balke*; cf. Dan. *aften-bakke* (evening), Sw. *natt-backa* (night), Icel. *ler-blaka* (leather), Icel. *blaka* to flutter.] (*Zoöl.*) One of the Cheiroptera, an order of flying mammals, in which the wings are formed by a membrane stretched between the elongated fingers, legs, and tail. The common bats are small and insectivorous. See Cheiroptera and Vampire.

On first glance, entries are laid out so that nouns of radically different meanings have separate entries denoted by the repeated headword, making identification and selection of polysemous nouns a relatively simple matter. The different meanings of each headword-entry could be represented by assigning a number in order of appearance: here, bat-1 and bat-2, with the appropriate dictionary entry attached to define the meaning in a human-comprehensible way.

However, separate headwords contain sub-definitions if other meanings spring from the original source (here, from the Old English *batte* and the Danish *aften-bakke*). Whatever their common linguistic origin, these sub-definitions are often radically different, necessitating their inclusion in the list of separate meanings, so that *bat* has five rather than two. For many polysemous nouns, this is problematic, as some are obsolete, others colloquial, and many simply obscure. In addition, some are virtually indistinguishable in what we might consider general meaning. For the word *ball*, two of the definitions provided are

1. Any round or roundish body or mass; a sphere or globe; as, a ball of twine; a ball of snow.

2. A spherical body of any substance or size used to play with, as by throwing, knocking, kicking, etc.

Thus the list of words and meanings is faulty, compounding the errors in successive automation procedures. This ambiguity in the definitions of polysemous words reflects the challenge of differentiating between shades of meaning in a way significant to the translation process.

Once this list has been produced, a possible method of acquiring training texts would be to search a text corpus for occurrences of the polysemous word, and then to assign it a meaning based on overlap between its context and the words in the dictionary definitions

attached to the different meanings. The most obvious problem with this approach is that the limited and selective nature of the definitions makes correct identification rare and unlikely. Augmenting the dictionary definitions with corresponding encyclopedia entries would vastly increase accuracy and chance of success, but automated use of encyclopedias was beyond the scope of the current project.

The other time-consuming process which seems a prime candidate for automation is the mapping from English meanings to the corresponding translation in a foreign language. Again, MRDs in non-English languages and bilingual MRDs are not readily available for research purposes. The challenge in meshing entries in such a dictionary with a list generated as described above is an entire research project unto itself.

In the MRD approach, a bilingual dictionary is necessary to find the possible translations of the target polysemous noun. Few such dictionaries are sufficiently exhaustive to cover all the meaning-definitions generated from the English MRD, especially colloquial or obscure meanings. Then, a mapping of the different possible translations to the appropriate meanings must somehow be found. Consider the entry for *bat* from the bilingual *Concise Oxford-Hachette French Dictionary*:

> **BAT** noun Sport batte f ; table tennis raquette f de tennis de table ; Zoölogy chauve-souris f

Leaving aside the matter of the *raquette*, assigning *batte* and *chauve-souris* to the appropriate meanings is itself a difficult task to automate. The provided tagwords ("Sport" and "Zoölogy") are clear enough to human readers, but since neither words fully appears in the corresponding definitions in the English dictionary, they are here useless. A larger context might be acquired from a less concise bilingual dictionary, or by finding the corresponding entry in a French dictionary and translating the definition appearing there into English, yet in neither of these cases would there be any guarantee of overlap with the correct English definition. In the latter method, the probability of overlap is high, but the translation itself would be prone to error.

The order in which the definitions appear is another possible approach to matching, especially if English and English-French dictionaries arranged in similar manner could be procured. However, unless the dictionaries were fully aligned, errors would still occur. Here, while the "baseball" definition does appear before the "zoölogy" one in both dictionaries, they are separated in the bilingual and English entries by one and three irrelevant definitions respectively.

Further complicating the process is the case where multiple meanings are matched to the same translation. For instance, the English word *crane* is translated *grue* in French whether signifying the bird or the machine. For a case such as *crane* where every possible English meaning is translated by the same French word, assignment is simple, but there are unfortunate cases where two or more French translations must be matched to a greater number of English meanings. This prevents a one-to-one mapping between the meanings, unless the French dictionary nonetheless divides the entry into multiple definitions. The opposite problem — where one English meaning, such as Bat-1 above, must be assigned to several French words, such as *batte* and *raquette* from the entry above — is still more inconvenient.

The most common solution to these problems is to abandon altogether the attempt to extract the English meanings from an MRD and instead to train a model on parallel corpora. A bilingual MRD might still be used to find French words translating an English polysemous word. A set of aligned texts (such as the Hansard) in French and English could be processed, searching for these French words. Whenever one is found, the corresponding English text around that position would be considered an instance of the appropriate English meaning. As a means of specifying meanings for English polysemous words as well as acquiring training texts for them, this limits flexibility by assigning meanings based solely on translations of a specific language. In addition, it requires aligned corpora in the two languages, meaning that the results are only as good as the available corpora and the translation between the parallel texts.

In the near future, solutions to the matching problem may be able to make use of

a new resource: "WordNet", a Princeton University project which sorts English words into "synonym sets" each representing a lexical concept. This promises to offer potential representations for polysemous words according to which synonymy sets they appear in, when completed. Projects to create and link together WordNet projects for other languages are also underway, but none are yet fully available.

As for MRDs, if a LSA or co-occurrence model has already been constructed for the polysemous English nouns on a representative range of training texts, correct mapping may be achieved by calculating meaning similarities on the contexts, although there remains an unacceptable potential for error given the likely brevity of such context. However, working solely from MRDs without access to an extensive knowledge base, such a mapping seems impossible unless the dictionaries are fully aligned.

Chapter 7

Extended Tests

Due to the difficulties of automation and to the absence of an acceptable bilingual MRD, in the final model of this project, the automated acquisition of polysemous words, meanings, and translations was not implemented.

Instead, a collection of training materials with all the necessary information for the disambiguator was created largely by hand for an arbitrarily-chosen set of twenty polysemous words with fifty meanings between them, for use in tests of larger size. French translations and other relevant information, such as plural forms and indication of gender, were also manually assigned. This information, and all the training texts, were combined into a single XML document, the different parts tagged to indicate function:

```
<poly>
  <noun plural="bats">bat</noun>
  <meaning>
    <trans plural="battes" gender="f">batte</trans>
    <ex>The umpire saw the baseball bat in the dugout.</ex>
    <ex>The batter swung his bat.</ex>
  </meaning>
  <meaning>
    <trans plural="chauves-souris" gender="f">chauve-souris</trans>
    <ex>The bat flew from the cave at night.</ex>
    <ex>The bat resembles a winged mouse.</ex>
  </meaning>
</poly>
```

This format allows easy decomposition into text files of the form needed for the matrix-construction and the disambiguation/translation programs, via the Java XML parser. Any number of training texts can be added to the lexicon, and the number of texts per meaning to be used in the model can also be specified.

The training data for the extended model was drawn from websites turned up in Google searches for the polysemous words, from places such as Wikipedia entries for the appropriate definitions of the words, or random, unprofessional webpages. The separate training blocks were drawn from text within ten words of an occurrence of the polysemous noun in question, without any human supervision.

This automatically-gathered collection of training texts is highly faulty. Apart from the unhelpful or unrepresentative nature of some of the sources used, processing webpages produces much questionable data. Although HTML tags and their contents were screened out, the files nonetheless contained ads, links, captions, special characters, formatting information, the occasional misspelling, and a host of proper nouns. This had less of a negative impact on the Bayesian method than on the LSA model. In the latter, similarities between texts are of vital importance, and an over-abundant occurrence of the word *internet*, for instance, causes the method to draw connections which do not actually exist. This was compensated for, in some degree, by screening out a set of irrelevant words such as *click*.

A series of comparisons was carried out using varying numbers of training texts. Although these sets remained relatively small due to computational constraints, varying from 700 to 7000 words, both disambiguation methods performed well on short, simple sentences containing polysemous nouns and between one and ten "interesting" context words that is, words which had not been screened out as too common. As expected, the LSA method could sometimes disambiguate sentences in which the Bayesian method recognized none of the words (meaning that none of the words had appeared in training documents related to that polysemous word). However, LSA not infrequently erred on sentences which the Bayesian method had interpreted correctly. In some cases this was due to the faulty nature of the training data: LSA was representing as similar training texts which had little in

common other than webpage-vocabulary.

A more drastic improvement in LSA's performance came when all words occurring only once in the test set were cut out of the training data. With such a small training space, one can expect a large number of words to occur only once: since the LSA method benefits little from the addition of such rare words, excluding them significantly improves the representation, as well as the speed and size of the model. A point of interest is that the LSA model performed similarly to the corresponding Bayesian model even when the unique words were left in the latter. There is no logical reason to remove these words from the co-occurrence matrix, nor does their removal cause any significant improvement in size and speed. However, for the sake of accurate comparison, the Bayesian model was built on the same reduced training set as the LSA.

LSA results varied dramatically according to the number of dimensions used in creating the model. Several different models were constructed for each training set size with varying dimensions, and the results of the best used in the experiment. In general, LSA seemed to work best with a number of dimensions at about 5% of the number of rows/context words. In addition, a limited number of tests suggested a slight improvement in LSA's accuracy when similarity was calculated using the average angle difference between context and matrix vectors, rather than the absolute minimum. All results here discussed were acquired with the average.

For all training spaces, the LSA and Bayesian methods both performed well, usually with between 70% and 90% accuracy. Both showed marked improvement as the size of the training space increased. Looking strictly at the number of successful disambiguations, LSA consistently performs better than the Bayesian method, as illustrated in Figure 7.1.

Considering only the test sentences which the Bayesian method has some hope of disambiguating, those containing overlap with the columns/training texts of the relevant polysemous word, the average success percentage is high for both methods: between 85% and 95% for all training-set sizes (Figure 7.2). While there is always overlap between the LSA and Bayesian errors, the sets of incorrect identifications are by no means identical. The

Bayesian method has on average a 89% success rate, with no consistent improvement or decay in accuracy. The LSA model has an average performance better by only 1.5%.

Thus we see that the margin in LSA's favor in Figure 7.1 is primarily due to the test sentences in which the Bayesian method fails to recognize any context words, which are interpreted as errors for the Bayesian method. LSA usually chooses the correct sense of the polysemous word for between 40% and 70% of such sentences, leading to a substantial increase in accuracy.

A point of interest in the results lies in the difficulty both the Bayesian and LSA methods have in distinguishing between relatively close shades of meaning. While both are fairly adept at differentiating between homophones, words alike in spelling but of different origin (*bat* and *bank*, for instance), errors seem more common when the meanings of the polysemous word are similar and thus have significant contextual overlap. For example, the word *age* can have two translations where its second meaning is that of *era*, as in the "modern age" or "past ages." With the smaller sets of training data, both the Bayesian and LSA methods had difficulty with this disambiguation. As differentiating between closely related meanings is one of the most challenging aspects of word sense disambiguation in natural language processing[6], further tests to see if LSA performs better than the Bayesian method on such words might be useful.

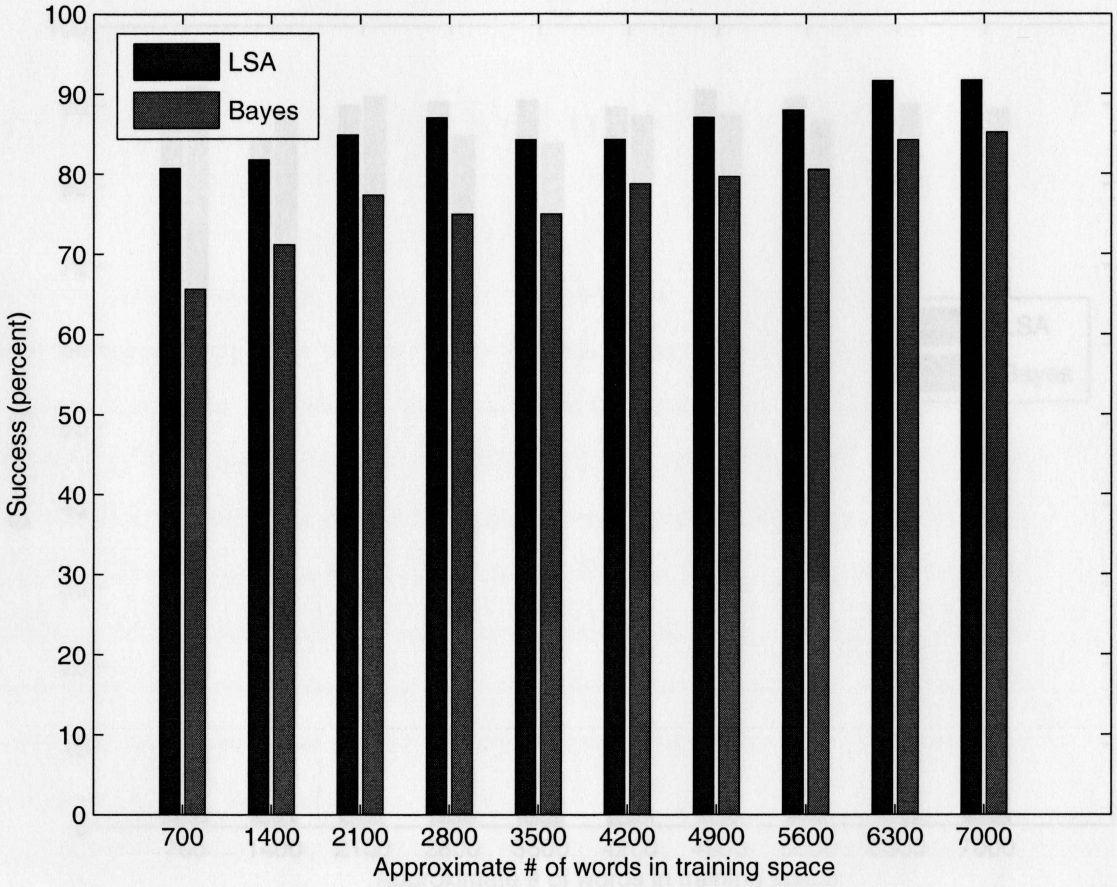

Figure 7.1: Accurate disambiguations for the two methods as a percentage of attempts. The labels along the horizontal axis roughly correspond to increases of 700 words in the size of the matrix.

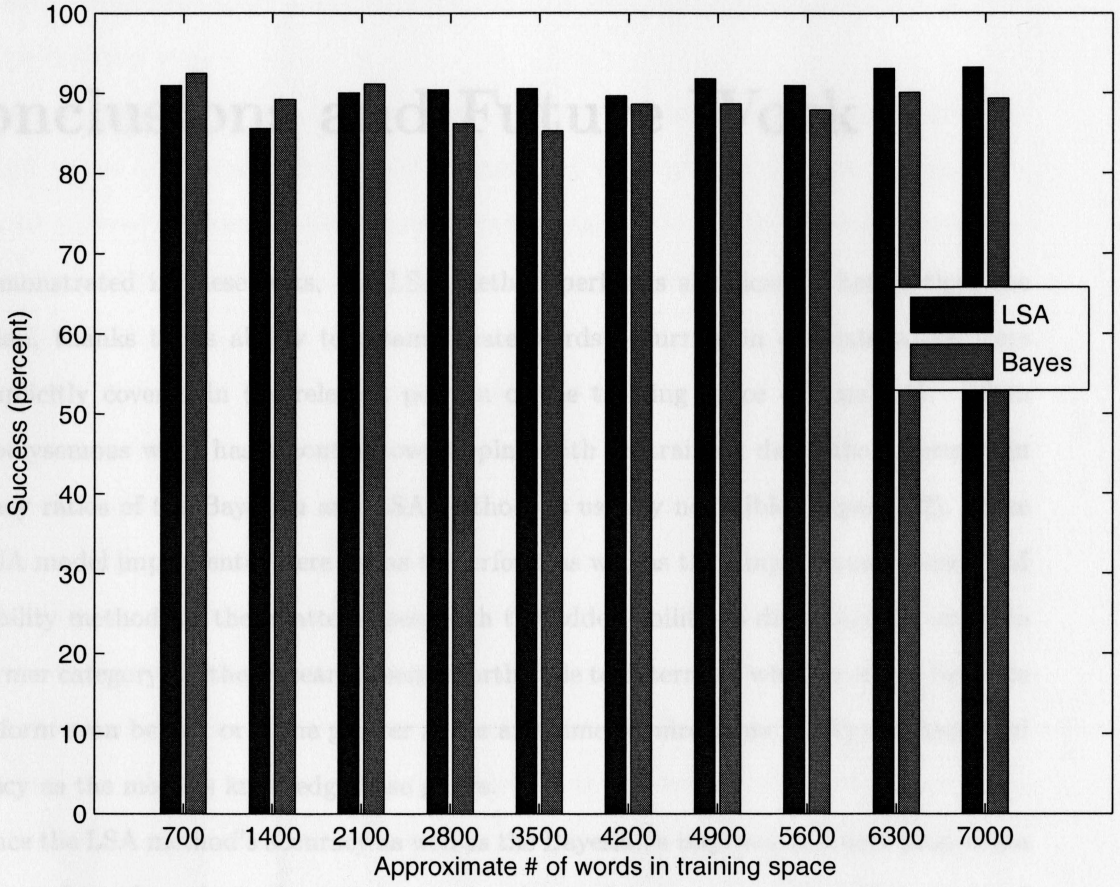

Figure 7.2: Adjusted chart of successful disambiguations, counting only those which overlapped the training data for the polysemous word in question.

Chapter 8

Conclusions and Future Work

As demonstrated in these tests, the LSA method performs significantly better than the Bayesian, thanks to its ability to disambiguate words occurring in contexts which were not explicitly covered in the relevant portion of the training space (Figure 7.1). When each polysemous word has a context overlapping with its training data, the difference in accuracy ratios of the Bayesian and LSA methods is usually negligible (Figure 7.2). Since the LSA model implemented here seems to perform as well as the simple co-occurrence and probability method for these latter cases, with the added ability to disambiguate words in the former category, further research seems worthwhile to determine whether it can be made to perform even better, or if the greater space and time requirements nullify the improved accuracy as the model's knowledge base grows.

Since the LSA method's accuracy as well as the Bayesian's improves in direct proportion to the number of words in the training space, the model seems to scale well in terms of successful disambiguation. It is likely that accuracy of the two models will converge as the size of the training space increases toward infinity, but LSA has significantly superior results for limited training spaces. This suggests that an LSA disambiguator requires a smaller training space than the Bayesian model. However, the LSA model grows large and slow much faster than the Bayesian, and thus would require a great deal of optimization to be of practical use in a machine translator.

Although no absolute quantitative conclusions can be drawn as to the relative accuracy of the LSA and Bayesian methods, given the small number of tests run, the small size of the training sets, and the relatively rough version of LSA implemented here, nevertheless, the results offer proof that LSA can indeed be used in disambiguation for machine translation. It offers a marked improvement over the simple Bayesian method, which works through exact matches between context words, while the LSA method represents and analyzes relationships between words.

As for the idea of using the entire context (less common words such as articles and conjunctions) of an ambiguous word to find its meaning, we may note that the results of the extended-set tests are comparable in success percentage to the results of the grammatical-relationship LSA model, [6] without the extra work necessitated by creation of a grammatical-relationship dictionary.

The next step would be to enhance the model with larger sets of polysemous words and of training data from a better corpus — news articles, for instance. The number of dimensions used in construction of the LSA matrix could be optimized, and more extensive and comprehensive tests performed. There is also the question of whether the model can be expanded to work with verbs, adjectives, and other parts of speech.

Probably the most important issue is that of automating acquisition of training texts to build a model sufficiently representative of word frequency distributions in the real world. Use of parallel, aligned corpora is one potential approach; analysis of MRDs followed by matching with encyclopedia entries is another. In the latter method, further texts could be classified as to meaning according to their similarities with the small contextual spaces already built.

If aligned or context-heavy bilingual dictionaries are available, mappings between translations and word-meanings could also be automated. The disambiguator module described here is separate from these mappings themselves, returning meanings to the translator interface, which then has only to choose the appropriate meaning from the table of whatever language is implemented. The same LSA model can thus be used to disambiguate trans-

lations in a wide range of languages, in association with machine translators of various types.

This LSA model has shown that a certain type of human knowledge - contextual association - can be reproduced for use in word sense disambiguation and machine translation. Yet other kinds of real-world knowledge, closer to actual human understanding, remain beyond the reach of methods such as LSA. This is clearly demonstrated by the well-known ambiguity example *The box is in the pen.*

Here, only the word *box* is considered "interesting" in the disambiguation process. Common, low-information words such as *the*, or *and* or *is*, often called "stop words", are assumed to convey little to no information about the polysemous nouns in whose context they appear. Indeed, the inclusion of stop words may badly skew the LSA and co-occurrence matrices, especially if the training data is drawn from real-world texts. Through sheer chance, the texts for one meaning may contain more occurrences of *the*, where in fact both meanings are nouns equally likely to be assigned the definite article. In the LSA model, all the words for the first meaning could then be incorrectly considered "similar" to context elements of some other word-meaning which had a heavy *the* count.

The natural solution to this seems to be to exclude stop words from the co-occurrence matrix altogether. Lists of "Top 100" words are readily available, and could easily be used as criteria for exclusion. The model in this paper excludes a moderate number of words: articles, prepositions, conjunctions, certain common adverbs, and omnipresent verb forms such as *was*, *had*, and *used.*

Yet in a case such as *The box is in the pen*, the meaning hinges on the innocuous word *in.* The disambiguation models addressed here will run into difficulties with this sentence that a human would find absurdly simple to understand. Only *box* is used in the disambiguation process - but is *box* any more likely to be associated with the fenced enclosure than with the writing implement? What if the sentence were *The papers are in the pen*? A human reading such a sentence uses a sense of relative size in assigning meaning to *pen.* A box and a stack of papers are both clearly too large to fit inside a writing utensil, whereas things of

any size are commonly put inside fenced pens - whose whole function, after all, is to hold things. Any statistical model like those addressed here necessarily lacks a human's ability to visualize as a scene the concept being described, and thus to assess it for plausibility. Yet a rule-based solution — assigning a "size" field to every noun in the translation dictionary and comparing sizes whenever prepositions such as *in* are utilized — is clearly impractical.

Since the word *in* seems more likely to occur in connection with the enclosure than with the writing utensil, one might think including the stop words in the co-occurrence matrix could give LSA a chance to disambiguate these occurrences of *pen*. Similarly, if the sentence read *The box is on the pen*, which implies the opposite meaning, a model including *on* might have a better chance of correctly interpreting the word than one excluding it — which would interpret the in/on sentences in exactly the same way.

Quite apart from the dangers inherent in consideration of stop words, which could damage the LSA model's usefulness even if a weighting scheme is applied, other limitations of the LSA/co-occurrence model negate this approach. The methods addressed here consider words without reference to order, function, or grammatical classification. For sentences in which meaning is dependent upon the configuration of the words, LSA can evaluate them semantically only, ignorant of any syntactic undercurrents. As currently designed, the LSA model is quite incable of differentiating between *The box is in the pen* and *The pen is in the box*, viewing them as identical sets of disconnected words.

Despite all its strengths, some ambiguities fall beyond LSA's scope. Its ability to simulate real-world knowledge, though broad, is nevertheless strictly limited. If researchers of other approaches to deal with such problems do not succeed, the *box is in the pen* problem may well remain an example of something at which humans are simply better than computers.

Bibliography

[1] S. Deerwester, S. T. Dumais, T. K. Landauer, G. W. Furnas, and R. A. Harshman. Indexing by latent semantic analysis. *Journal of the Society for Information Science*, 41(6):391–407, 1990.

[2] S.T. Dumais. Latent semantic indexing (lsi) and trec-2. In *The Second Text Retrieval Conference (TREC2)*, D. Harman, editor, number 500-215 in National Institute of Standards and Technology Special Publication, pages 105–116, 1994.

[3] Gene H. Golub and C.F. Van Loan. *Matrix Computations.* Johns Hopkins University Press, Baltimore, 1983.

[4] Paul Graham. *A Plan for Spam*, chapter 8. O'Reilly, 2004.

[5] Woosung Kim and Sanjeev Khudanpur. Lexical triggers and latent semantic analysis for cross-lingual language model adaptation. *ACM Transactions on Asian Language Information Processing*, 3(2):94–112, 2004.

[6] Y-Seop Kim, Jeong-Ho Chang, and Byoung-Tak Zhang. A comparative evaluation of data-driven models in translation selection of machine translation. In *Proceedings of the 19th International Conference on Computational Linguistics*, pages 1–7, 2002.

[7] T.K. Landauer, P.W. Foltz, and D. Laham. Introduction to latent semantic analysis. *Discourse Processes*, 25:259–284, 1998.

[8] T.K. Landauer, D. Laham, and P.W. Foltz. The intelligent essay assessor. *IEEE Transactions on Intelligent Systems*, 15(5):27–31, 2000.

[9] Christopher D. Manning and Heinrich Schütze. *Foundations of Statistical Natural Language Processing.* MIT Press, 1999.

[10] Carl D. Meyer. *Matrix Analysis and Applied Linear Algebra.* SIAM, Philadelphia, 2000.

[11] Bob Rehder, Michael L. Littman, Susan Dumais, and T.K. Landauer. Automatic 3-language cross-language information retrieval with latent semantic indexing. In *Proceedings of The Sixth Text Retrieval Conference (TREC-6)*, Gaithersburg, MD, November 1998. National Institute of Standards Technology.