Experiments with Latent Semantic Analysis for Word Tagging

ASHRAF ANWAR
Computer Science Department
Gulf University for Science and Technology
PO Box 7207, Hawalli, 32093
KUWAIT

Abstract: - In this paper I show the possible use of Latent Semantic Analysis (LSA) as an aid for word tagging and ambiguity resolution for words in test sentences. The idea is to use large corpora of training sentences, previously tagged by a human expert; for building an LSA “engine” that is used to aid in tagging future test sentences. Various training and testing phases were done. The results show that LSA seems to do somewhat fair; compared to other statistical word tagging and ambiguity resolution methods. However, there is one main drawback for this approach; the accuracy of word tagging depends on the training corpus.

Key-Words: - Ambiguity Resolution, Latent Semantic Analysis (LSA), Statistical Methods, Word Tagging.

1 Introduction
In this work, I examine the use of Latent Semantic Analysis (LSA) to aid in word tagging and ambiguity resolution. For training and test purposes, I use a fixed number of word tags since LSA is not good at associating or relating things that have never been seen before. The training sentences are of a limited length. Apparently, no common words were excluded from the LSA keywords candidacy [3, 7]. In this work, test sentences have a limit on their length and use the same set of tags used in training. I began training and testing with sentences having same number of words. Then I use multiple lengths to check if the LSA engine can generalize up to a certain threshold of sentence length tolerance.

The aim of my work is to compare and contrast the performance of this LSA corpus-based technique with other known tagging and ambiguity resolution techniques including Viterbi [1, 12].

2 How it works
The problem at hand is to determine the word tags of an input test sentence. The main idea is to match the test sentence against a training corpus previously supplied to LSA engine. A sentence in the training corpus that gets highest match with the test sentence is used to determine the tags of the test sentence. So, essentially, instead of calculating the conditional probabilities and taking the one with maximum likelihood out of the candidates as in Viterbi algorithm [1], I rely on LSA to yield the closest tagging pattern. Depending on the similarity and context of words in the corpora, LSA is tested for ability to give a good suggestion of how the words in the input test sentence should be tagged. For example, one can argue that when we hear or see the sentence "The bell rang", we can immediately say that "The" is an article because of its position in the sentence as well as its preceding of a noun. Such an immediate judgment is based to a certain degree [5] on previously heard and seen sentences, i.e. training. Such training can be argued to be somewhat similar to and depicted in the workings of LSA.

3 LSA workings
LSA is based on building a large semantic association frequency matrix (SAFM) based on keywords coexistence and mutual context [4, 6, 7]. In a typical LSA usage, the training input documents are processed for keywords candidacy, and keywords are chosen. However, in this work, I do not exclude any common words from keyword candidacy; to allow for proper tagging of all parts of sentences. Then, a matrix is formed with its rows standing for the input documents and its columns standing for the keywords. The cells of the matrix contain the frequency or how many times a certain keyword (column-identified) occurred in a specific document (row-identified).
A note to be made is that, very infrequent or rare words are not considered in a typical LSA usage. In the general case, one way to achieve this is to consider only words appearing in at least two documents. But again for our case here, these infrequent words are indeed considered in training the LSA engine and building the semantic association frequency matrix (SAFM).

Some weighting of cell entries, then, takes place based on information theory techniques [6]. The document by keyword matrix is then processed using Singular Value Decomposition (SVD). SVD is used to construct an abstract semantic space of some dimension; equal to number of factors which is determined during training [6, 7, 10].

Each original keyword and document is represented as a vector in that space; column-wise and row-wise respectively. Important to note is that, LSA representation of a document is the average of the vectors of the words it contains independent of their order. The dimension reduction step accomplished during SVD is a form of induction that can extract a great deal of added information from mutual constraints among a large number of terms in different contexts.

The linear decomposition of the original term-document matrix $X$, by SVD is defined by:

$$ X = W S C' $$

Where $W$: Orthonormal-column matrix for keyword  
$S$: Diagonal Matrix of Singular Values  
$C'$: Orthonormal-column matrix for documents

If there are enough dimensions, we can reconstruct the original data $X$, but not the original documents. The minimization of dimensionality is a form of induction since it requires the simultaneous accommodation of all the data.

The relation between the dimension of the LSA space and the accuracy of the achieved understanding is non-monotonic. The hypothesis is that natural dimensionality is based on neural processing architecture and statistical properties of the input corpus [6, 7].

Embedding the data observations in spaces with too small dimensions causes unnatural distortions. Spaces with greater than optimal dimensions do not exploit mutual constraints in the data in the way that humans typically do [11].

### 4 Related Work

Viterbi algorithm [1], is based on calculating the conditional probability of a specific word tag given the tag of the immediate preceding word. Using this Markovian assumption, processing time can be greatly reduced.

To find the most likely tagging sequence, we sweep forward through the words one at a time finding the most likely sequence for each ending category; based on same sentence previously encountered tags. This process is repeated until the tags for all the words in the sentence have been accounted for.

### 5 LSA Training

I used training sentences with lengths varying from 3 to 5 words for sake of simplicity. I also restricted the variation of sentence length in this phase to be able to judge how well LSA can do initially in such a new domain. The lexicon used is of very limited size in these initial experiments. To be noted is that I am not aware of any similar work using LSA, and indeed I strongly doubt that there is one. Once performance in such a limited domain is thoroughly tested, evaluation of more general and longer lengths; even arbitrary ones, may be considered.

All training and testing sentences were labeled by a human expert. The lexical categories or word tags used were the eight tags illustrated in table 1. The Count column represents how many times each Tag appeared in the training corpora.

<table>
<thead>
<tr>
<th>Tag</th>
<th>Symbol</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Noun</td>
<td>N</td>
<td>272</td>
</tr>
<tr>
<td>2 Verb</td>
<td>V</td>
<td>272</td>
</tr>
<tr>
<td>3 Adjective</td>
<td>ADJ</td>
<td>118</td>
</tr>
<tr>
<td>4 Adverb</td>
<td>ADV</td>
<td>64</td>
</tr>
<tr>
<td>5 Article</td>
<td>ART</td>
<td>248</td>
</tr>
<tr>
<td>6 Pronoun</td>
<td>PRO</td>
<td>138</td>
</tr>
<tr>
<td>7 Auxiliary</td>
<td>AUX</td>
<td>48</td>
</tr>
<tr>
<td>8 Preposition</td>
<td>PREP</td>
<td>60</td>
</tr>
</tbody>
</table>

Table 1: Tags used in training corpora and their counts.
The number of documents or sentences used for training was 272. No common words were excluded from keyword candidacy due to the nature of the task; because common words could represent tagged words. The number of factors used in this limited size task was 70. There were 70 different keywords in the limited training corpora.

Some examples of the training sentences follow:

i. A cat ran in lazy way
ii. He did it with success.
iii. She can bank the plane.
iv. She went to bed.
v. The mighty lion ran fast.
vi. They went to the bank.
vii. You ate a fruit.
viii. You ate an apple.

6 LSA Testing

Test sentences were formed to test LSA for word tagging. In particular; simple alterations of a training keyword or a substitution of one word from the training lexicon with another unseen new word were used in testing. In both cases a very high match was obtained and the tagging is almost perfect. However when changing the tagging pattern of the test sentence to a totally new one which was never seen before, the matches are typically non significant. This has to do in part with the way LSA functions in terms of decompressing a document into keywords without regard to their relative ordering [6, 10].

Table 2 shows some test sentences and their matches from the LSA training corpora along with the cosine between the two vectors; the one for the test sentence and the matched sentence. Table 2 is sorted descendingly according to the cosine value for the match.

Some observations on the test results obtained so far are noted here as follows:

i. Poor matches (small cosine values) occur when a new tagging pattern is encountered; for example rows 13 through 15.

ii. A threshold for relevance and consideration can be statistically determined. For the current tests, it is taken to be 0.5. In other words rows 13 and above are insignificant and do not constitute a match at all.

iii. The match is pretty high when the testing tagging pattern has been seen before and is indeed recognized; for example rows 1 through 5.

iv. Despite new variations of word positions and tags; LSA is able to generalize to these circumstances to some extent, and still yield the correct tagging pattern with good percentage; for example row 7.

v. Test results should become more accurate when training occurs over bigger training corpora.

vi. Training over all possible words and tagging combinations is inherently combinatoric.

<table>
<thead>
<tr>
<th>Test Sentence</th>
<th>Best Match</th>
<th>Cosine</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>It can bank the plane</td>
<td>0.990923</td>
</tr>
<tr>
<td>2</td>
<td>It ate an apple</td>
<td>0.984057</td>
</tr>
<tr>
<td>3</td>
<td>I ate a banana</td>
<td>0.917659</td>
</tr>
<tr>
<td>4</td>
<td>He did it elegantly</td>
<td>0.902231</td>
</tr>
<tr>
<td>5</td>
<td>He did it brightly</td>
<td>0.900012</td>
</tr>
<tr>
<td>6</td>
<td>The wild cat ran fast</td>
<td>0.895903</td>
</tr>
<tr>
<td>7</td>
<td>The lazy cat jumped</td>
<td>0.895146</td>
</tr>
<tr>
<td>8</td>
<td>The mighty train went fast</td>
<td>0.806077</td>
</tr>
<tr>
<td>9</td>
<td>The mighty tiger came</td>
<td>0.789160</td>
</tr>
<tr>
<td>10</td>
<td>Sue ran to bed</td>
<td>0.719160</td>
</tr>
<tr>
<td>11</td>
<td>She ran to college</td>
<td>0.671320</td>
</tr>
<tr>
<td>12</td>
<td>The mighty elephant fell</td>
<td>0.599151</td>
</tr>
<tr>
<td>13</td>
<td>A new student came</td>
<td>0.378721</td>
</tr>
<tr>
<td>14</td>
<td>A boring movie came</td>
<td>0.248703</td>
</tr>
<tr>
<td>15</td>
<td>Success in life depends on hard work</td>
<td>0.225678</td>
</tr>
</tbody>
</table>
7 Conclusion
LSA captures the contextual neighboring of words which may be used for ambiguity resolution. If still successful in more sophisticated cases, this technique might serve as a new corpus-based ambiguity resolution technique. It can be used instead of or with conjunction with any word tagging module. However, it appears somewhat unlikely that LSA alone can generalize to the extent that total random combinations, as in everyday discourse, of word tags can be recognized.

A closer tagging pattern is very likely to be retrieved, but to get the exact tagging pattern further processing will be needed. The positive side is that when the tested tagging pattern is totally unfamiliar, the highest match is typically poor and insignificant but not misleading. Adding new tagging patterns for recognition should prove useful in enhancing the performance.

8 Future Research
One possible alteration is the focusing on keywords which affect the preceding or following words like the preposition to. Putting more weight for such keywords may well be of help. Another direction is to make some post-processing of the LSA output, i.e. just take it as a start point for further tagging action.

References