Predicting when words may appear: A Connectionist Model of Sentence Processing

Simon Dennis (simon.dennis@gmail.com)
225 Psychology Building, 1835 Neil Avenue
Columbus, OH 43210 USA

Dennis Mehay (mehay@ling.ohio-state.edu)
222 Oxley Hall, 1712 Neil Avenue
Columbus, OH 43210 USA

Srikar Yekollu (srikary@gmail.com)
395 Dreese Laboratories, 2015 Neil Avenue
Columbus, OH 43210 USA

Abstract

We introduce a connectionist version of the Syntagmatic Paradigmatic model (Dennis, 2005) and train it on subcorpora drawn from the gigaword corpus. Decaying syntagmatic representations of the words to the left and right are used to estimate the paradigmatic associates of a word. The pattern of paradigmatic associates is then combined with an order independent associative representation of the surrounding words to predict the word that will appear in a given slot. The best performing version of the model produced a perplexity of 28.3 on a vocabulary of 5006 words, significantly lower than Good Turing and Kneser Ney ngram models trained under the same conditions. Furthermore, we changed parameters and lesioned components to isolate which properties of the model are critical to its performance. Online update of the weights - a kind of priming - allows the model to track short term contingencies and significantly improves performance. When we removed the paradigmatic and associative layers performance dropped, when we removed just the associative layer performance dropped and when we removed the right context from the syntagmatic and associative layers performance also dropped - suggesting that all of the hypothesized components of the model are crucial to its performance.

Keywords: syntagmatic, paradigmatic, sentence processing, perplexity, connectionist

Introduction

Dennis (2005) introduced the Syntagmatic Paradigmatic model as a general account of verbal cognition. It is based on the distinction between **syntagmatic** associations that occur between words that appear together in utterances (e.g. run fast) and **paradigmatic** associations that occur between words that appear in similar contexts, but not necessarily in the same utterances (e.g. deep and shallow, cf. Ervin-Tripp, 1970). The model has been used to explain a number of phenomena including long term grammatical dependencies and systematicity (Dennis, 2005), the extraction of statistical lexical information (syntactic, semantic and associative) from corpora (Dennis, 2003a), sentence priming (Harrington & Dennis, 2003), verbal categorization and property judgment tasks (Dennis, 2005), serial recall (Dennis, 2003b), and relational extraction and inference (Dennis, 2005, 2004).

Previous instantiations of the model have used string edit theory (SET) as a way of capturing the notion of paradigmatic association. While SET has been a fruitful mechanism with which to investigate the properties of the general theoretical framework, it has proven inadequate in some respects. In particular, it does not scale well to large corpora, primarily because it proposes the sentence as the main unit of analysis. In this paper, we will present a connectionist version of the SP model that operates at the word level and demonstrate that it is able to to predict when words will be used with some precision.

The Model

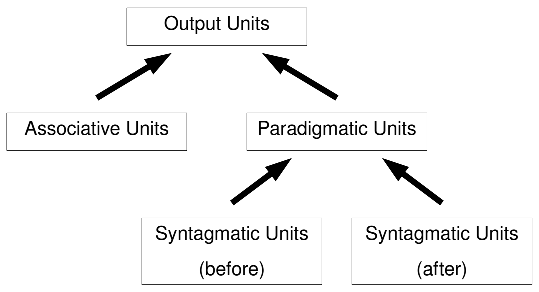

Figure 1: The syntagmatic paradigmatic model

Our connectionist implementation of the SP model has five main sets of units arranged into a two layer architecture. The initial syntagmatic inputs consist of two sets of units each of which contain a unit for each word in the vocabulary of the system. The syntagmatic units (before) represent the words that appear before a given word within the sentence, while the syntagmatic units (after) represent those words that appear after the word. To capture the temporal order of the words, the activations of these units decay in an exponential fashion (cf. ordinal models of serial order such as the primacy model, Page & Norris, 1998). If a word is repeated then its new activation is simply added to any existing activation as follows:

$$s_{t,w_i,before} = \sum_{i<t} \lambda e^{-\lambda(t-i)} \quad (1)$$

$$s_{t,w_i,after} = \sum_{t<i} \lambda e^{-\lambda(i-t)} \quad (2)$$

where w_i is the word that appears in the ith timestep, t is the current timestep, and λ is the rate of the exponential decay. This form of representation is efficient, only requiring 2V units where V is the size of the vocabulary, generalizes naturally as the length of dependencies increases and as Ding, Dennis, and Mehay (2009) argue provides implicit constraints that may explain the relative ease of processing of different kinds of embeddings.

To produce the activations of the paradigmatic units a multinomial logistic regression model is used. The syntagmatic representations are multiplied by a weight matrix and the softmax nonlinearity is applied as follows:

$$p_{t,w} = \frac{e^{W_{tw}s_t}}{\sum_j e^{W_{tj}s_t}} \quad (3)$$

Weights are then updated with a simple gradient descent rule:

$$\begin{aligned} W_{t+1,j} &= W_{t,j} + \gamma(1 - p_{t,j})s_t, j = w_t \\ &= W_{t,j} - \gamma p_{t,j} s_t, j \neq w_t \quad (4) \end{aligned}$$

where γ is the learning rate and w_t is the word that appears in the corpus at time t.

The paradigmatic layer estimates the probability that a word could have fit into the slot defined by the surrounding words, that is, it is an estimate of the paradigmatic associates of the current word.

The associative layer codes adjacent words in an order independent fashion and allows the model to capture relational regularities (e.g. the active to passive transform). The associative inputs are coded as follows:

$$a_{t,w_i} = \sum_{t \neq i} \lambda e^{-\lambda|i-t|} \quad (5)$$

Finally, the output units are again calculated using the multinomial logistic regression model (see Equation 3) and the same gradient descent training rule is applied (see Equation 4). Although the architecture has two layers these are trained independently. There is no backpropagation of error. Furthermore, in all of the simulations reported in this paper a single iteration through the training set was performed.

A Simple Example

To understand the logic of the model, consider the following simple example. Suppose we trained the model (λ = 0.1, γ = 0.1, 2000 iterations) on the following corpus:

```
john loves mary
mike loves sue
bert loves ellen
gary loves bob
adam loves eve
todd loves sarah
barak loves michelle
who loves sue ? mike
who loves ellen ? bert
who loves bob ? gary
who loves eve ? adam
who loves sarah ? todd
who loves michelle ? barak
```

and then present it with the question "who loves mary ? xxx". Any model that relies exclusively on sequential information will be unable to correctly predict that "john" should fill the "xxx" slot, because "john" only appears at the start of a sentence. With the SP model, however, we get the following paradigmatic patterns for each of the words in the sentence:

```
who:   who 92 loves 03 john 01
loves: loves 93 who 02
mary:  loves 13 who 13 michelle 10 sarah 10
       eve 10 bob 10 ellen 10 sue 10 ? 07
       mary 01 barak 01 todd 01 adam 01
?:     ? 37 michelle 07 sarah 07 eve 07
       bob 07 ellen 07 sue 07 barak 03
       todd 03 adam 03 gary 03 bert 03
xxx:   barak 09 todd 09 adam 09 gary 09
       bert 09 mike 09 ? 08 michelle 05
       sarah 05 eve 05 bob 05 ellen 05
```

Note that in the "xxx" slot we get a strong representation of the {barak, todd, adam, gary, bert, mike} pattern. This pattern represents the lover role and a similar paradigmatic pattern occurs in the "john" slot when the model is processing the sentence "john loves mary":

```
john:  who 29 john 19 loves 06 barak 05
       todd 05 adam 05 gary 05 bert 05 mike 05
loves: loves 42 john 10 mary 10
mary:  mary 17 ? 11 michelle 08 sarah 08
       eve 08 bob 08 ellen 08 sue 08
```

As a consequence, paradigmatic associations form between the {barak, todd, adam, gary, bert, mike} pattern and "john". The paradigmatic mechanism in itself, however, would not suffice to predict "john", as "barak", "todd", "adam", "gary", "bert" and "mike" are also associated with the lover pattern. Only "john" has an associative connection to "mary", however, and the additional support afforded by this connection favors "john". At the output layer, we get the following patterns:

```
who:   who 97 john 03
loves: loves 100
mary:  michelle 11 sarah 11 eve 11 bob 11
       ellen 11 sue 11 barak 05 todd 05
       adam 05 gary 05 bert 05 mike 05
?:     ? 92 john 04
xxx:   john 28 barak 06 todd 06 adam 06
```

```
gary 06 bert 06 mike 06 sue 05
ellen 05 bob 05 eve 05 sarah 05
```

The model approximates a propositional representation in an associative form that does not rely on the creation of independent propositional units of representation. Dennis (2004) shows that these representations can be used to answer simple questions about tennis matches. Taking naturally occurring text from the Association of Tennis Professionals website, the model was able to complete questions of the form "Who won the match between Sampras and Agassi? xxx". Particularly interesting was the fact that the model took advantage of the systematic occurrence of tokens through the corpus as a consequence of the causal relationships between events. It implements a kind of "inference by coincidence" to determine results even when they are not explicitly stated (Dennis, 2004).

While such results may be of theoretical interest, the question remains as to whether the model can capture naturally occurring data. In the following sections, we test the model and investigate which components are crucial to its performance.

Results

In the current paper, we focus on the models ability to predict which word will appear in a given sentential context. We will use perplexity to quantify the performance of the model. Perplexity is an information theoretic measure of the degree of uncertainty of a given choice. Formally, it is two to the power of the cross entropy of the model with respect to the empirical distribution of the data, which can be estimated as follows:

$$P = 2^{-\frac{1}{N}\sum_{k=0}^{N} log_2 p(w_k)} \tag{6}$$

where $p(w_k)$ is the probability according to the model of the kth word and N is the size of the sample. To provide an intuition, a perplexity of x is the degree of uncertainty that exists when faced with x equally probable and independent alternatives.

In typical uses of the perplexity measure one is interested in calculating the mean information content of an entire corpus. That is, one is interested in the probability of the sequence $w_1, w_2, ...w_n$. In that case, it is common to decompose this probability using the chain rule. When one does this only left or right context can be used, but not both. In our case, we are interested in the information content of individual word choices given both left and right context. The same formula applies, but the perplexity values that we report are not directly comparable to per word corpus perplexity values that appear in the literature.

Comparison with Ngram Models

Driven by the need to create language models for speech recognition and other tasks, much of the work on predicting words occurs in computational linguistics. Within this field ngram models have been shown to produce robust performance and so we will compare the SP model against ngram models using two commonly used and successful smoothing techniques - Good Turing and Kneser Ney (Chen & Goodman, 1998). Our ngram modelling results were generated using the SRILM toolkit (Stolcke, 2002).

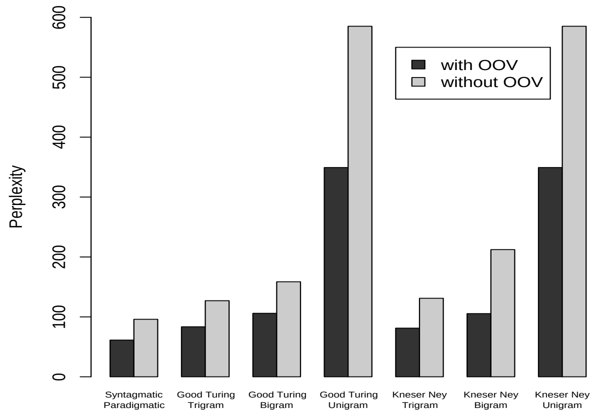

Figure 2: Performance of the SP model as compared against unigram, bigram and trigram models using both Good Turing and Kneser Ney smoothing.

Unless otherwise noted, the training set consisted of 100,000 sentences of the English gigaword corpus sections NYT200001, NYT200002 and NYT200003 (2.4 million words, Graff, Kong, Chen, & Maeda, 2007) and the test set of 1000 sentences (23611 words). In our first set of simulations, we trained the model using a learning rate γ of 1 and a decay rate λ of 1. The paradigmatic layer was restricted to the 200 most frequent words and all weights were restricted to lie between 10 and -10. The vocabulary of the model (and hence the size of the two syntagmatic banks of units, the associative bank and the output bank) was restricted to the 5006 most frequent tokens. Any tokens that occurred that were not within the 5006 most frequent, were represented as an out of vocabulary (OOV) item (xxx). Perplexity can be affected by the inclusion or exclusion of these tokens, so we report performance in both conditions. When referring to perplexity values in text, we will adopt the convention of reporting the perplexity with OOV tokens followed by the perplexity without OOV items in brackets.

Figure 2 shows the perplexity of the SP model as compared against unigram, bigram and trigram; Good Turing and Kneser Ney models. The SP model performs significantly better in all cases.

The size of the training corpus has a significant affect on performance. Figure 3 shows the performance of the model and the trigram Good Turing and Kneser Ney models when the training set contained 500,000 sentences (12 mil-

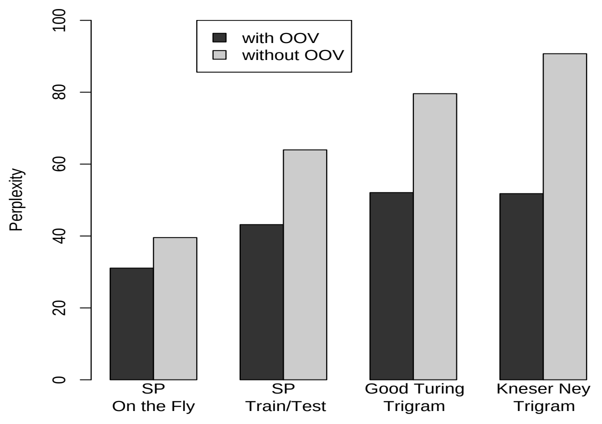

Figure 3: Performance of the models given a 500,000 sentence (12 million word) training corpus.

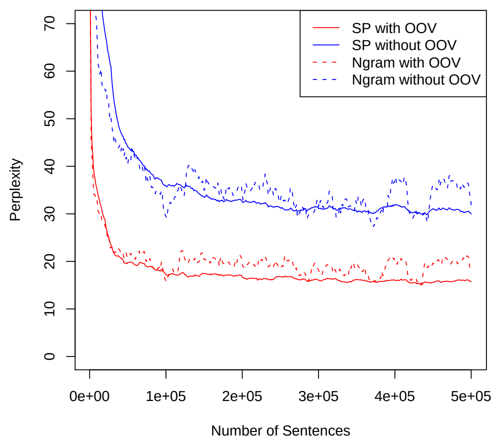

Figure 4: Performance of the SP model compared against that of the Ngram Features model.

lion words, on the fly values will be explained in the next section). The best performance of the SP model was a perplexity of 28.31 (35.61) with a training set of 27 million words.

To the best of our knowledge this is state of the art for this task, and suggests that the model is capturing important regularities to which ngram models are insensitive. However, it may be argued that left context only ngram models are unfairly compromised in this setting because they do not have access to the right context. Typically, these models would be used to assess the probability of a string of words. An inconsistent right context would then manifest in poor prediction of subsequent words.

To provide a fairer test, we constructed a multinomial logistic regression model that employed bigram and trigram features generated from the left and right context. So, for instance, when the model was exposed to the sequence w_1, w_2, w_3, w_4, w_5 and we were trying to predict w_3 then we would use the features w_2-, $-w_4$, w_1w_2-, $-w_4w_5$ and w_2-w_4.

The number of features generated by the ngram model was very large. Consequently, we restricted the vocabulary to 1000 tokens and trained on the 500,000 sentence corpus used above. Under these conditions, the ngram model had a total of 457549 features and was approximately 285 times as large as the SP model. Nevertheless, the SP model performed better although the difference was not large. The SP model produced a perplexity of 15.75 (29.93), while the ngram model produced a perplexity of 17.75 (31.91).

Figure 4 shows the performance of the SP and ngram models as a function of training examples. The performance of the ngram model improves quickly, but is eventually matched and then surpassed by the SP model. Also, it is quite noticeable that the variance of the ngram model is significantly higher than that of the SP model. To understand the relative performance of the models, we examined the tokens that were strongly predicted by one model, but not the other. Table shows the most frequent tokens that were well predicted by the SP model and by the ngram model, respectively. The SP model tends to do a better job of predicting high frequency tokens, while the ngram model is doing better on low frequency tokens.

The Importance of Priming

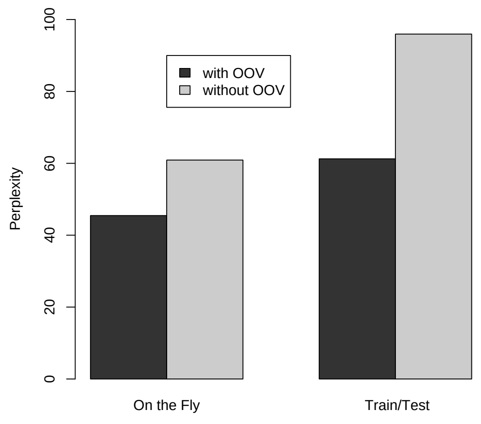

Figure 5: Performance measured as training progressed versus in train and test mode. The superior performance of online training suggests that priming plays an important role.

As training progresses through the corpus, the weights capture long term regularities, but also track short term contingencies that one might think of as a form of priming. To

SP Model		Ngram Model	
Count	Token	Count	Token
308	,	33	)
166	the	24	bush
142	.	19	“
74	of	17	.
41	to	16	a
31	”	14	angeles
29	a	11	:
28	said	11	,
12	years	10	spokesman
12	n’t	9	use
10	’s	9	and
9	new	9	(
9	i	8	secretary
8	who	8	p.m.
8	in	7	year
6	more	7	very
5	his	7	texas

Table 1: Tokens well predicted by one of the models and not the other. Counts are derived by sorting the tokens by the differences between the probability predicted by the SP model and Ngram models. The tokens corresponding to the top 1000 of these differences were then tabulated.

determine the potential significance of priming in the model, perplexity was calculated as training progressed. Note that because training consisted of a single iteration through the training set, each prediction was of an unseen word. Figure 5 shows the results on the final 50,000 tokens of the training set compared against the train/test results from the previous section in which weights were fixed during testing. Clearly, priming of this kind significantly improves the performance of the model.

The trade off between retaining long term regularities while tracking short term contingencies can be seen in the performance of the model as the learning rate is manipulated. With high learning rates recent information is emphasized, while with lower learning rates long term information predominates. Figure 6 shows that there appears to be an optimal tradeoff at around $\gamma = 4$ for this corpus. Because the train/test methodology can only take advantage of long term regularities, performance gets worse as the learning rate increases.

Size of the Paradigmatic Layer

In the simple example outlined above, all words in the vocabulary appeared in the paradigmatic representation. In order to make the model more computationally efficient, we investigated the impact of reducing the size of this layer significantly. Instead of taking the entire vocabulary of 5006 words, we kept just the most frequent words. If a word did not appear in this set no learning on the first layer of weights was conducted. Note the number of output units remained the same and the second layer of weights was trained on every

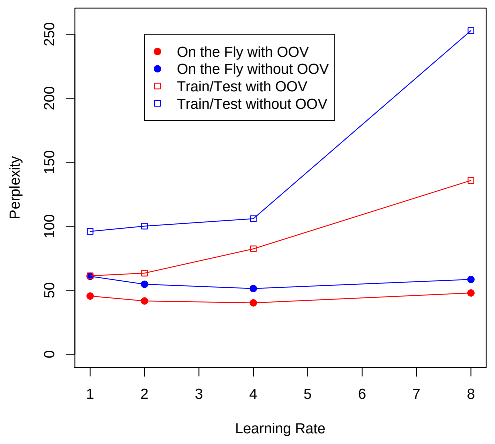

Figure 6: Performance as the learning rate is manipulated.

iteration. Performance does not change substantially, but it appears that between 100 and 200 paradigmatic units is optimal. We suspect that these most frequent words fulfil a role somewhat like parts of speech.

Adding a Second Bank of Syntagmatic Units

Ding et al. (2009) demonstrate that the syntagmatic (before) input representations as defined above have a number of interesting properties with respect to the kinds of embedding they can support. In particular, they found that with a single bank of units a single level of center embedding can be correctly predicted, but a single level of cross serial embedding cannot - that is the patterns are not linearly separable. The single bank model is not able to account for two levels of embedding of either the center or cross serial kinds. With two banks of units employing different decay rates, however, cross serial and center embedding of one and two levels is possible. Based on these results, they argued that a two bank model best approximates human capabilities.

Figure 7 shows the results as the decay rate of a second bank of syntagmatic units varies from 0 (no second bank) to 0.1. The ability of the model to predict which word can fill a given slot is affected very little by the inclusion of the additional bank.

Lesioning the Model

While the paradigmatic and associative banks of units play a central role in the logic of the model, to what extent do they really aid in prediction? To determine this, we created a version of the model which used the syntagmatic banks (before and after) to predict words directly. Perplexity rises substantially, from 45.46 (60.90) to 70.69 (99.46). Next, we lesioned just the associative layer, leaving the paradigmatic layer intact. Again performance was significantly impacted rising to 90.60 (139.23). Finally, we lesioned the right context both

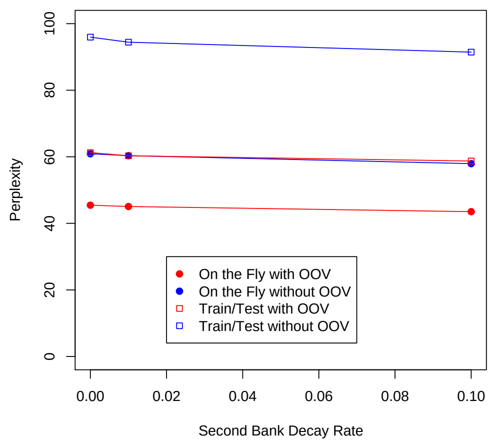

Figure 7: Performance when two banks of syntagmatic units with different decay rates are used. In each case, the decay rate of the first bank was 1.0.

at the syntagmatic layer and the associative layer. Perplexity rose to 107.58 (160.67), demonstrating that the right context makes a significant contribution to the ability of the model to predict correctly.

Discussion and Conclusions

The SP model was initially developed as framework for understanding verbal cognition (Dennis, 2004, 2005). By creating a connectionist implementation, we have been able to scale the model to large corpora, 27 million words of naturally occurring text from the gigaword corpus, with a reasonably large vocabulary of 5006 words. With the model trained on this corpora we were able to achieve a perplexity of 28.3. This result is a significant improvement on Good Turing and Kneser Ney ngram models on the same task and to the best of our knowledge represents the state of the art. When an ngram features model was trained using left and right context, the advantage for the SP model was retained, although it was not as large.

Furthermore, by observing as we changed parameters and lesioned components, we have been able to isolate which properties of the model are critical to its performance. The first lesson is that priming is important. Online update of the weights - a kind of priming - allows the model to track short term contingencies. When we allowed weights to update as we calculated the perplexity performance improved significantly. Secondly, it appears that all of the component layers make a substantial contribution to performance. When we removed the top layer performance dropped. When we removed the associative layer performance dropped and when we removed the right context from the syntagmatic layer and associative layer performance also dropped. By contrast adding a second bank of syntagmatic inputs with a different decay rate had little impact as did changing the size of the paradigmatic layer - at least over the range that we manipulated it.

We believe that these results provide prima facie evidence for the model and suggest that there may be significant advantage in applied domains to considering the cognitive constraints on the sentence processing mechanism.

Acknowledgments

We would like to acknowledge the support of Defense Research and Development Canada contract number W7711-067985. We would also like to thank Chris Brew and Eric Fosler-Lussier for their feedback on this work.

References

Chen, S. F., & Goodman, J. (1998). *An empirical study of smoothing techniques for language modeling* (Tech. Rep. No. TR-10-98). Cambridge, MA: Harvard.

Dennis, S. (2003a). An alignment-based account of serial recall. In R. Alterman & D. Kirsh (Eds.), *Twenty fifth Conference of the Cognitive Science Society* (Vol. 25). Boston, MA: Lawrence Erlbaum Associates.

Dennis, S. (2003b). A comparison of statistical models for the extraction of lexical information from text corpora. In R. Alterman & D. Kirsh (Eds.), *Twenty fifth Conference of the Cognitive Science Society* (Vol. 25). Boston, MA: Lawrence Erlbaum Associates.

Dennis, S. (2004). An unsupervised method for the extraction of propositional information from text. *Proceedings of the National Academy of Sciences*, *101*, 5206-5213.

Dennis, S. (2005). A memory-based theory of verbal cognition. *Cognitive Science*, *29*(2), 145-193.

Ding, L., Dennis, S., & Mehay, D. (2009). A single layer network model of sentential recursive patterns. In *Proceedings of the Cognitive Science Society.*

Ervin-Tripp, S. M. (1970). Substitution, context and association. In L. Postman & G. Keppel (Eds.), *Norms of word association* (p. 383-467). New York: Academic Press.

Graff, D., Kong, J., Chen, K., & Maeda, K. (2007). *English gigaword third edition.* Linguistic Data Consortium.

Harrington, M., & Dennis, S. (2003). Structural priming in sentence comprehension. In R. Alterman & D. Kirsh (Eds.), *Twenty fifth Conference of the Cognitive Science Society* (Vol. 25). Boston, MA: Lawrence Erlbaum Associates.

Page, M., & Norris, D. (1998). The primacy model: A new model of immediate serial recall. *Psychological Review*, *105*, 761-781.

Stolcke, A. (2002, September). SRILM - an Extensible Language Modeling Toolkit. In *Proceedings of the International Conference on Spoken Language Processing.* Denver, Colorado.