Abstract

We present a new method of introducing domain knowledge into an n-gram language model. It is based on a combination of language models for individual word domains. Each word model is built from an individual corpus which is formed by extracting those subsets of the entire training corpus which contain that significant word. When testing, significant words are extracted from a cache and their models are combined with a global language model. Different methods of combining the models are described; one simple method based on combining frequencies rather than probabilities gives promising results and provides a relatively simple method of introducing domain information into an n-gram language model. A 20% reduction in language model perplexity over the standard 3-gram approach is obtained which is similar to results obtained with other more complex domain models. The model also requires a small cache compared with other models requiring a cache.

1. Introduction

A popular type of statistical language model is one which dynamically modifies conditional probabilities depending on the recent word history. For example the cached-based natural language model [1] incorporates a cache component into the model, which increases the probability of a word if it is found in a cache. Trigger based models trigger associated words to each content word in a cache giving each word a higher probability [2].

This paper extends these ideas by triggering a new language model for each content word in the cache and combining them with a standard global n-gram language model to produce an improved model.

A training corpus for each significant word is formed from the amalgamation of the text fragments taken from the global training corpus in which that word appears. A significant word is any word that significantly contributes to the content of the text. We define this as any word which is not a stop word, i.e. not articles, prepositions and some of the most frequently used words in the language such as “will” and common adverbs and adjectives such as “now”, “very”, “some”, etc. The individual language models are then obtained from these corpora. They should be more precise than both the global language model and other domain models. They should also outperform previous cache methods for small caches.

A vital feature of this model is the method of combining the global language model with the individual word language models. This is explored later.

2. Dynamic Language Model Based on Word Language Models

Our dynamic language model first builds a language model for the global corpus. Frequencies of words and phrases are derived from the corpus and the conditional probability of a word given a sequence of preceding words is estimated. The individual conditional probabilities are approximated by the maximum likelihood:

\[
P_{ML}(w_i | w_{i-1}^{l-1}) = \frac{f(w_i | w_{i-1}^{l-1})}{f(w_{i-1}^{l-1})} = \frac{f(w_1 \cdots w_{i-1} w_i)}{f(w_1 \cdots w_{i-1})}
\]

where \( f(X) \) is the frequency of the phrase \( X \) in the text.

In equation (1), there are often unknown sequences of words i.e. phrases which are not in the dictionary; there are also words with low frequency. The maximum likelihood probabilities are then zero or unreliable. In order to improve the prediction of these unseen or unlikely events, and hence the language model, a number of techniques have been explored, for example, the Turing-Good estimate [3], the Katz back-off method [4], deleted interpolation [5] or the weighted average (WA) n-gram model [6]. Although any of these or others could be used in our model we use the WA n-gram technique which combines n-gram phrase distributions of several orders using a series of weighting functions. It has been shown to exhibit similar predictive powers to other n-gram techniques whilst being simple and having the advantages for this project that it both facilitates easy model extension and that it is easily combined with other models, as we shall see.

1 The “ease of extension” applies to the fact that additional training data can be incorporated into an existing WA model without the need to re-estimate smoothing parameters.
The weighted average probability of a word given the preceding words is

\[ P_{w_i}(w|w_m^n) = \frac{\mu_0 P_{ML}(w) + \sum_{i=1}^{m} \mu_i P_{ML}(w|w_{m+i-1})}{\sum_{i=0}^{m} \mu_i} \]  

(2)

where the weighted functions are:

\[ \mu_0 = \ln(N) \]

\[ \mu_i = \ln(f(w^m_{m+i-1})) \cdot 2^i \]  

(3)

where \( N \) is the number of tokens in the corpus and \( f(w^m_{m+i-1}) \) is the frequency of the sentence \( w_{m+i-1} \ldots w_m \) in the text.

The language model defined by equation (2) and (4) is called here the global language model when trained on the global corpus.

Following the creation of the global model comes the creation of a language model for each significant word, which is formed in the same manner as the global language model. The word training corpus is the amalgamation of the text fragments taken from the global training corpus in which the significant word appears. These text fragments can be any length or they can be the sentences or paragraphs containing the word. We used paragraph length fragments (although we have not yet determined that this is the best choice). Additionally some restrictions on the number of content words contributing to the model were imposed (from 5 to 40).

### 3. Combining the Models

We need to combine the probabilities obtained from each word language model and from the global language model, in order to obtain a combined probability for a word given a sequence of words. One simple way to doing this is an arithmetic combination of the global language model and the word language models in a linear interpolated model as follows:

\[ P(w|w^n_1) = \lambda_G P_{Global}(w|w^n_1) + \sum_{i=1}^{m} \lambda_i P_i(w|w^n_1) \]  

(5)

where \( \lambda_G + \sum_{i=1}^{m} \lambda_i = 1 \)  

(6)

and \( P_i(w|w^n_1) \) is the conditional probability in the word language model for the significant word \( w_i \), \( \lambda_i \) is the correspondent weight and \( m \) is the maximum number of word models that we are including.

Ideally the \( \lambda_i \) parameters would be optimised using a held-out training corpus; however this is not practical as we do not know which combination of words \( w_i \) will arise in the cache. So a simpler approach is needed.

### 3.1. Linear Interpolation

A simple way of choosing the \( \lambda \) values is to give the same weight to all the word language models but a different weight to the global language model, and put a restriction on the number of word language models to be included. This weighted model is defined as:

\[ P(w|w^n_1) = \alpha \cdot P_{Global}(w|w^n_1) + \frac{(1-\alpha)}{m} \sum_{i=1}^{m} P_i(w|w^n_1) \]  

(7)

and \( \alpha \) and \( m \) are parameters which are chosen to optimise the model.

### 3.2. Exponential Decay

Furthermore, a method was used based on an exponential decay of the word model probabilities with distance since a word appearing several words previously will generally be less relevant than more recent words. Given a sequence of words, for example, “We had happy times in America…”

<table>
<thead>
<tr>
<th>We</th>
<th>Had</th>
<th>Happy</th>
<th>Times</th>
<th>In</th>
<th>America</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>4</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

where 5, 4, 3, 2, 1 represent the distance of the word from the word America. The words Happy and Times are significant words for which we have an individual word language models. The exponential decay model for the word \( w \), where in this case \( w \) represents the significant word America, is as follows:

\[ P(w|w^n_1) = \frac{P_{Global}(w|w^n_1) + P_{Happy}(w|w^n_1) \cdot \exp(-3/d)}{1 + \exp(-3/d) + \exp(-2/d)} \]  

(8)

where \( P_{Global}(w|w^n_1) \) is the conditional probability of the word \( w \) following a phrase \( w_1 \ldots w_k \) in the global language model \( P_{Happy}(w|w^n_1) \) is the conditional probability of the word \( w \) following a phrase \( w_1 \ldots w_n \) in the word language model for the significant word Happy. The same definition applies for the word model Times. \( d \) is the exponential decay distance with \( d = 5, 10, 15 \), etc. The decaying factor \( \exp(-l/d) \) introduces a cut off:
if \( l \geq d \) \( \Rightarrow \) replace \( \exp(-l/d) \) by 0

where \( l \) is the distance from the significant word to the target word and \( d \) is the decay distance. Presently the combination methods outlined above have been experimentally explored. However, they offer a reasonably simplistic means of combining the individual and global language models with modest performance gains, as we shall see. So more sophisticated models are likely to offer improved performance gains.

### 3.3. Union Model

Another method of combining models is the Probabilistic-Union model [8], which has been effective for noisy speech. This model is based on the logical concept of a disjunction of conjunctions which is implemented as a sum of products. The form of the union model which we found most effective [9] can be expressed as:

\[
P(w_i | w_R) = \alpha P_G(w_i | w_R) + (1 - \alpha) K \sum_{i > j} P_i P_j
\]

where \( P_i = P(w_i | w_R) \), \( K \) is a normalizing constant and \( \alpha \) is a parameter. The summation for \( i > j \) is a logical sum of probabilities, i.e., if \( Q_1 \) and \( Q_2 \) are probabilities, their logical sum is:

\[
Q_{1 \text{and} 2} = Q_1 + Q_2 - Q_1 Q_2
\]

The difficulty with the method in equation (9) is that it requires a great deal of processing to normalize.

### 3.4. Combined Frequency Model

Instead of combining probabilities to obtain a dynamic language model, it is also possible to combine frequencies before calculating probabilities, i.e. a revised maximum likelihood, replacing equation (1), is:

\[
P_{ML}(w_i | w^{(-1)}_1) = \frac{\lambda_G \hat{f}_G(w_i^1) + \sum_{i=1}^{m} \hat{\lambda}_i \hat{f}_i(w_i^1)}{\lambda_G \hat{f}_G(w^{(-1)}_1) + \sum_{i=1}^{m} \hat{\lambda}_i \hat{f}_i(w^{(-1)}_1)}
\]

This can then be combined using the WA model in equation (2). This simple method is automatically normalised and it is easy to implement and fast to execute. The choice of \( \lambda \) is still critical but cannot be optimized before hand for the same reason that the \( \lambda \)'s in equation (1) cannot be optimized: we do not know beforehand which combination of words will occur in the cache. One solution is the use of empirical formulae for these \( \lambda \) values or to optimize each separately (work in progress).

### 4. Results

The methods described above were compared in an experiment using the Wall Street Journal (version WSJ0\(^3\)) which contains about 38 million words, and a dictionary of approximately 65,000 words. We select one quarter of the articles in the global training corpus as our training corpus. To test the new language model we use a subset of the test file given by WSJ0, selected at random. The training corpus that we are using contains 172,796 paragraphs, 376,589 sentences, 9,526,187 tokens. The test file contains 150 paragraphs, 486 sentences, 8824 tokens and 1908 words types. Further experiments with bigger training corpora and test files are planned.

All probabilities are based on dynamic trigram models and are compared with a basic trigram model.

The model based on linear interpolation offers improved results, up to 10% when \( \alpha=0.6 \) in equation (7) with a combination of a maximum of 10 word models. Better results were found when the word models were weighted depending on their distance from the current word, that is, for the exponential decay model in equation (8) where \( d=7 \) and the number of word models is selected by the exponential cut off (Table 1). For this model improvements of over 17% have been found.

<table>
<thead>
<tr>
<th>( \text{Cut} )</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>4d</td>
<td>15.53%</td>
<td>16.31%</td>
<td>16.46%</td>
<td>16.44%</td>
</tr>
<tr>
<td>5d</td>
<td>15.90%</td>
<td>16.42%</td>
<td>16.52%</td>
<td>16.43%</td>
</tr>
<tr>
<td>6d</td>
<td>15.92%</td>
<td>16.45%</td>
<td>16.53%</td>
<td>16.41%</td>
</tr>
<tr>
<td>7d</td>
<td>16.02%</td>
<td>16.46%</td>
<td>16.51%</td>
<td>16.40%</td>
</tr>
<tr>
<td>8d</td>
<td>16.02%</td>
<td>16.46%</td>
<td>16.51%</td>
<td>16.39%</td>
</tr>
<tr>
<td>9d</td>
<td>15.97%</td>
<td>16.45%</td>
<td>16.51%</td>
<td>16.39%</td>
</tr>
</tbody>
</table>

For the probabilistic-union model in equation (9), the best result obtained so far is an improvement of 20% when we have a maximum of 6 word models and the value of alpha is 0.6 (Table 2).

<table>
<thead>
<tr>
<th>( \alpha )</th>
<th>0.3</th>
<th>0.4</th>
<th>0.5</th>
<th>0.6</th>
<th>0.7</th>
<th>0.8</th>
<th>0.9</th>
</tr>
</thead>
<tbody>
<tr>
<td>%</td>
<td>15</td>
<td>18</td>
<td>20</td>
<td>20</td>
<td>18</td>
<td>16</td>
<td>11</td>
</tr>
</tbody>
</table>

---
\(^3\) CSR-I(WSJ0) Sennheiser, published by LDC, ISBN:1-58563-007-1
The results in Table 2 took a great deal of processing to obtain a 20% improvement. However with an order of magnitude less processing the same 20% improvement was obtained for the simple combined frequency model with weights $\lambda_k = \frac{T_G}{T_k}$, where $T_k$ is the number of tokens in the language model for word $k$. In the table below (Table 3) results obtained using the combined frequency model and different weights are shown. This table also shows that the model requires a very small cache (only 5 content words) compared to previous cache based models.

<table>
<thead>
<tr>
<th>Number Word Models</th>
<th>Weights</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$\text{Ln}T_{G}/\text{Ln}T_k$</td>
</tr>
<tr>
<td>5</td>
<td>7.60%</td>
</tr>
<tr>
<td>10</td>
<td>9.97%</td>
</tr>
<tr>
<td>20</td>
<td>11.43%</td>
</tr>
<tr>
<td>30</td>
<td>11.67%</td>
</tr>
<tr>
<td>40</td>
<td>11.68%</td>
</tr>
</tbody>
</table>

This implies that the best results are obtained when the most precise word models have the highest weights.

5. Conclusions

In this paper we have introduced the concept of individual word language models to improve language model performance. Individual word language models permit an accurate capture of the domains in which significant words occur and hence improve the language model performance. Even though the results are preliminary, they indicate that individual word models offer a promising and simple means of introducing domain information into a n-gram language model. The improvement in perplexity so far (20%) in Table 4 is similar to that obtained in much more computationally intensive methods based on clustering [10][11].

<table>
<thead>
<tr>
<th>Method</th>
<th>Equation</th>
<th>Improvement</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linear Interpolation</td>
<td>Eq. (7)</td>
<td>10%</td>
</tr>
<tr>
<td>Exponential Decay</td>
<td>Eq. (8)</td>
<td>17%</td>
</tr>
<tr>
<td>Union Model</td>
<td>Eq. (9)</td>
<td>20%</td>
</tr>
<tr>
<td>Combined Frequency Model</td>
<td>Eq. (11)</td>
<td>20%</td>
</tr>
</tbody>
</table>

Note that the Combined Frequency model is much simpler and faster than the Union model.

6. References


