Multidimensional Backoff for Factored Language Models Using Word Vectors

Anna Currey

April 1, 2015

Contents

1 Introduction

In this paper, I present an adaptation of factored language models for use with semantic word vector representations. Using words and progressively less fine-grained vector-based word clusters as factors means that there is a natural backoff direction in the factor dimension that can be exploited to improve estimations. This backoff in the dimension of the factors is combined with backoff in the dimension of the history to yield the multidimensional backoff described in this paper. This new backoff framework applied to factored language models allows us to better utilize semantic word representations to improve language models.

1.1 Factored Language Models

The first component on which I base my model is factored language models (FLMs), which were introduced in [4]. FLMs are an extension of traditional n-grams in which the words in the history are decomposed into features, such as word class, stem, or affixes. This means that instead of having to drop an entire word from the history when backing off, as in n-grams, we can drop some features from a word and still consider other features of the word in FLMs. This allows us to look at a longer history while lessening the data sparsity issue that necessitates backoff.

Clearly, FLMs can be advantageous for highly inflected languages, where words can easily be factored into stem, root, morphological class, and other features ([4]). In addition, FLMs can help combat the problem of the high vocabulary growth rate in such languages by considering a smaller number of sub-word units ([15]). In fact, the experiments on Arabic in [4] were much more successful than those on English. FLMs have also proven to be relatively flexible, and have been combined with morpheme-based models ([2], [6], [15]) and neural networks, both through interpolation ([1]) and by using factors in the feature stream ([17]).

Since the factors in FLMs can be of different types, there is not necessarily any natural way to back off among them ([4]). In particular, it is not even possible to decide that we should drop the oldest elements in the history first, since it might be advantageous to take, for example, the morphological classes of the entire history over all the word factors in a shorter history. To address this issue, [4] also introduced generalized parallel backoff (GPB). This allows the backoff path(s) to be chosen at runtime based on some heuristic function. In addition to using GPB, a model structure can be automatically learned using a genetic algorithm, as described in [5].

Although different backoff methods have been proposed, no method has been found that consistently finds the best backoff path, suggesting that there is room for improvement in this area. Another disadvantage of factored language models is that an additional training phase or additional tools (such as a lemmatizer or a stemmer) are needed in order to find the factors. For morphological factors, the same tools can be used on the test data to find factors even for unseen words, but for data-driven factors, this is not possible. This means that when the factors are all data-driven, unseen words in the history cannot be considered; this could be part of the reason why FLMs tend to perform better on morphologically rich languages. My framework aims to improve upon this by assigning default factors to unseen words in the history, allowing the unseen words to be considered.

1.2 Word Vectors

Recently, methodology has been introduced for representing words semantically as vectors, most notably through the GLoVe ([13]) and word2vec ([11]) tools. These word vectors can be trained using the local context of a word (for example, through the continuous bag of words or skip-gram models in [11]), global matrix

factorization ([13]), or both. The vectors then give us a notion of similarity between words ([11]).

Once the vector representations for words are obtained, they can easily be clustered into word classes; word2vec, for example, uses k-means clustering over the vectors ([10]). Multiple sets of clusters of different sizes can then be created on the training data and used in a language model.

These semantic representations have proven to be useful aids to a variety of semantic tasks, including dependency parsing ([3]) and machine translation ([12], [18]). However, to the best of my knowledge, word vectors have not yet been applied to the task of non-neural network-based language modeling. This paper, therefore, gives a framework for doing so.

1.3 Multidimensional Backoff

The main advantage of multidimensional backoff is that it gives us a concrete way of improving a factored language model by using word clusters gained from semantic word vectors. This will reduce the severity of the data sparsity problem, as we will be able to replace words in the history with their semantic clusters. This methodology is also applicable to all types of languages, rather than just morphologically complex languages where words can easily be split into components.

Like standard FLMs, this framework considers the words in the history as well as their factors. In this case, the word itself is taken as a factor, and the remaining factors are obtained through clustering of word vectors. We can have multiple factors by breaking the vocabulary up into larger and larger clusters. For example, we might first obtain 1,000 clusters from the word vectors and then 100 clusters. In this case, we would have three factors: the word itself, which of the 1,000 clusters the word belongs to, and which of the 100 clusters the word belongs to.

The key insight in this model is that, unlike in the general case for FLMs, there is a notion of backoff order among the factors. Since the word is more informative than its fine-grained cluster, which in turn is more informative than the word's coarse-grained cluster, it would be natural to consider the word first before backing off to the fine-grained cluster if necessary. Note that there would be no reason to consider the word and its factors together, since the factors are a function of the word itself. The coarse-grained clusters can also be forced to be a function of the fine-grained clusters (so that each fine-grained cluster is a subset of exactly one coarse-grained cluster), and indeed this may make more intuitive sense (since we consider only one factor per word at a time); however, this is not required for the model described below to work. Even if the larger clusters are not a function of the smaller clusters, we can still say that the factor containing fewer words contains more specific and discriminating information, so it is still natural to back off to that factor first.

Words that appeared in the training data less than k times for some threshold k can all be clustered together, along with any previously unseen words in the

test data. This means that we are still able to consider an unseen word in a history (by considering its clusters).

Combining the intuitive backoff direction in the factor dimension with the traditional backoff in the history dimension (drop the oldest word first) allows us to naturally back off from factored n-grams in two different dimensions. The rest of this paper gives a formalization for how this would work.

Before diving into the formalization, I will give a simple illustration. Given the sequence *a blue dog* in the test data, we would like to estimate $P(dog|a\ blue)$. If this sequence did not appear in the training data, in traditional n-gram language models we would estimate its probability by considering only the bigram *blue dog*. However, using multidimensional backoff means we might assign *a* to a cluster containing articles, so in backing off we could consider $P(dog|\text{ARTICLE}\ blue)$. In addition, the word vector tool might place *blue* in a cluster containing all color words, so in backing off we could also consider $P(dog|a\ \text{COLOR})$. Clearly this gives us more information with which to estimate the sequence's probability. Below, I will go into more detail about how these additional backoff probabilities would be utilized.

2 Backing off through Factors

In this section, I will describe a framework that can be used to back off through factors (instead of simply through the history). Then, in section 3, I will give an example of backing off through both factors and history. Finally, in section 4, I will generalize this to n-grams with m clusters (for arbitrary m and n).

2.1 Overview

In order to use multidimensional backoff, we need a set of clusters that contain an increasing number words; the word itself can be the first factor. The intuition used in this model is that the words give the most information, while the clusters that contain the most words give the least information. Therefore, we have a natural backoff path among factors: backing off to progressively less fine-grained clusters.

Below, I give the model for factored bigrams with multidimensional backoff among three factors (the word itself and two clusters). This means that only the previous word and its factors were considered in the history when predicting the current word. To use the example given in section 1.3, the first non-word factor could consist of one of the 1,000 clusters; the second non-word factor could consist of one of the 100 clusters.

I will use the following notation to indicate the backoff from word to its less specific factors. For word w_i, I define the following:

- w_i^0 refers to the word itself (no backoff)
- w_i^1 refers to the word's most informative cluster (backoff of one step)
- w_i^2 refers to the word's least informative cluster (backoff of two steps)

2.2 Formulation of the Probability

In order to make use of the backoff through factors, I present a modified version of Katz's backoff model described in [8] adapted to the factored language model with multidimensional backoff. Similar to the standard n-gram version, I define the probably of a word w_i given a factor j of the previous word w_{i-1} as follows:

$$P_{BO}(w_i|w_{i-1}^j) = \begin{cases} d(w_{i-1}^j, w_i)\frac{C(w_{i-1}^j, w_i)}{C(w_{i-1}^j)} & , C(w_{i-1}^j, w_i) > 0 \\ \alpha(w_{i-1}^{j+1}, w_{i-1}^j) P_{BO}(w_i|w_{i-1}^{j+1}) & , else \end{cases} \quad (1)$$

where $C(w_{i-1}^j, w_i)$ is the standard count function.

The discounting factor $d(w_{i-1}^j, w_i)$ can be found using a slight adaptation of any discounting algorithm, including absolute discounting, Witten-Bell discounting, modified Kneser-Ney smoothing, or Good-Turing discounting. In this paper, I will use the equation for $d(w_{i-1}^j, w_i)$ found using Good-Turing discounting as an example of the modifications needed. This is very similar to that used in standard Good-Turing discounting:

$$d(w_{i-1}^j, w_i) = \frac{C(w_{i-1}^j, w_i) + 1}{C(w_{i-1}^j, w_i)} \cdot \frac{|\left\{w_{k-1}^j, w_k : C(w_{k-1}^j, w_k) = C(w_{i-1}^j, w_i) + 1\right\}|}{|\left\{w_{k-1}^j, w_k : C(w_{k-1}^j, w_k) = C(w_{i-1}^j, w_i)\right\}|} \quad (2)$$

In practice, we would use a smoothed version of this discounting factor such as the one described in [7] in order to ensure that the discount is between 0 and 1.

Note that discounting leaves some non-zero probability probability mass for unseen words; in the case of the discount given in equation 2, this amounts to:

$$P_{BO}(< unk >) = \frac{|\{w_i : C(w_i) = 1\}|}{|\{w_i : C(w_i) > 0\}|} \quad (3)$$

The backoff weight $\alpha(w_{i-1}^{j+1}, w_{i-1}^j)$ is defined in order to ensure that probabilities sum to 1 over all words w_i. I will define this weight here and then derive it in section 2.3. In order to define this weight, I first define an auxiliary factor $\beta(w_{i-1}^{j+1}, w_{i-1}^j)$:

$$\beta(w_{i-1}^{j+1}, w_{i-1}^j) = 1 - \sum_{\{w_k : C(w_{i-1}^j, w_k) > 0\}} d(w_{i-1}^j, w_k) \cdot \frac{C(w_{i-1}^j, w_k)}{C(w_{i-1}^j)} \quad (4)$$

Then $\alpha(w_{i-1}^{j+1}, w_{i-1}^j)$ is defined as follows:

$$\alpha(w_{i-1}^{j+1}, w_{i-1}^j) = \frac{\beta(w_{i-1}^{j+1}, w_{i-1}^j)}{\sum_{\{w_k : C(w_{i-1}^j, w_k) = 0\}} P_{BO}(w_k|w_{i-1}^{j+1})} \quad (5)$$

For computational purposes, since there are usually fewer trigrams that have occurred than those that have not occurred ([9]), this can be rewritten as:

$$\alpha(w_{i-1}^{j+1}, w_{i-1}^{j}) = \frac{\beta(w_{i-1}^{j+1}, w_{i-1}^{j})}{1 - \sum_{\{w_k : C(w_{i-1}^{j}, w_k) > 0\}} P_{BO}(w_k | w_{i-1}^{j+1})} \tag{6}$$

since $P_{BO}(w_k|w_{i-1}^{j+1})$ represents a probability distribution (which I will show in section 2.3).

2.3 Justification

In this section, I give a justification of why $P_{BO}(w_i|w_{i-1}^{j})$ is a valid probability distribution. In order to do so, I need to show the following:

1. $\sum_{w_k} P(w_k|w_{i-1}^{j}) = 1$
2. $P(w_k|w_{i-1}^{j}) \geq 0$ for all w_k

The backoff factor $\alpha(w_{i-1}^{j+1}, w_{i-1}^{j})$ was in fact derived in order to make the following equation hold for all w_{i-1}^{j}:

$$\sum_{w_k} P(w_k|w_{i-1}^{j}) = 1 \tag{7}$$

Here, I will simply show the derivation. Expanding the left-hand side of equation 7 gives us:

$$\begin{aligned}\sum_{w_k} P(w_k|w_{i-1}^{j}) = &\sum_{\{w_k : C(w_{i-1}^{j}, w_k) > 0\}} d(w_{i-1}^{j}, w_k) \cdot \frac{C(w_{i-1}^{j}, w_k)}{C(w_{i-1}^{j})} \\ &+ \sum_{\{w_k : C(w_{i-1}^{j}, w_k) = 0\}} \alpha(w_{i-1}^{j+1}, w_{i-1}^{j}) \cdot P_{BO}(w_k|w_{i-1}^{j+1})\end{aligned} \tag{8}$$

by equation 1.

Combining equation 7 and equation 8 and using the fact that $\alpha(w_{i-1}^{j+1}, w_{i-1}^{j})$ does not depend on w_k, we can simplify to:

$$\begin{aligned}\alpha(w_{i-1}^{j+1}, w_{i-1}^{j}) \cdot \sum_{\{w_k : C(w_{i-1}^{j}, w_k) = 0\}} P_{BO}(w_k|w_{i-1}^{j+1}) = \\ 1 - \sum_{\{w_k : C(w_{i-1}^{j}, w_k) > 0\}} d(w_{i-1}^{j}, w_k) \cdot \frac{C(w_{i-1}^{j}, w_k)}{C(w_{i-1}^{j})}\end{aligned} \tag{9}$$

I claim that the left-hand side of equation 9 is never zero, since it must at minimum contain the probability of unseen w_k given in equation 3. Therefore,

we can isolate $\alpha(w_{i-1}^{j+1}, w_{i-1}^{j})$, which yields an identical equation to that given in 5.

I will prove that $P(w_k|w_{i-1}^{j}) \geq 0$ for all w_k by reverse induction on the number of backoff steps j (which is necessarily a finite set).

For the base case, take unseen sequences. This is by definition the maximum amount of backoff. By equation 3, $P(w_k|w_{i-1}^{j})$ clearly must be defined and non-negative for all unseen w_k. (Note that I am implicitly assuming throughout that the training set is non-empty.)

For the inductive step, assume that $P(w_k|w_{i-1}^{l+1}) \geq 0$ for some $l+1$ number of backoff steps and consider $P(w_k|w_{i-1}^{l})$:

$$P_{BO}(w_k|w_{i-1}^{l}) = \begin{cases} d(w_{i-1}^{l}, w_k)\frac{C(w_{i-1}^{l}, w_k)}{C(w_{i-1}^{l})} & , C(w_{i-1}^{l}, w_k) > 0 \\ \alpha(w_{i-1}^{l+1}, w_{i-1}^{l}) P_{BO}(w_k|w_{i-1}^{l+1}) & , else \end{cases} \tag{10}$$

First, take the first line in the right-hand side. As mentioned above, $d(w_{i-1}^{l}, w_i)$ is explicitly defined and smoothed to ensure that it is between 0 and 1 for all sequences in the training data. In addition, the count function $C(x)$ is non-negative by definition. If $C(w_{i-1}^{l}, w_k) > 0$, then $C(w_{i-1}^{l}) > 0$, so it follows that $P_{BO}(w_k|w_{i-1}^{l})$ is indeed defined and non-negative for all w_{i-1}^{l}, w_k such that $C(w_{i-1}^{l}, w_k) > 0$.

Now consider the second case. I need to prove that the backoff probability $\alpha(w_{i-1}^{l+1}, w_{i-1}^{l}) P_{BO}(w_k|w_{i-1}^{l+1}) \geq 0$ for all w_{i-1}^{l}, w_k such that $C(w_{i-1}^{l}, w_k) = 0$. By the inductive hypothesis, $P_{BO}(w_k|w_{i-1}^{l+1})$ is non-negative (and defined), so it will suffice to show that $\alpha(w_{i-1}^{l+1}, w_{i-1}^{l})$ is non-negative (my derivation above showed that it is always defined). From the definition of the backoff factor in equation 5, the denominator of $\alpha(w_{i-1}^{l+1}, w_{i-1}^{l})$ must be non-negative, again by the inductive hypothesis. Therefore, the only remaining component is to prove that $\beta(w_{i-1}^{l+1}, w_{i-1}^{l})$ defined in equation 4 is non-negative, which is equivalent to proving the following inequality:

$$\sum_{\{w_k : C(w_{i-1}^{l}, w_k) > 0\}} d(w_{i-1}^{l}, w_k) \cdot \frac{C(w_{i-1}^{l}, w_k)}{C(w_{i-1}^{l})} \leq 1 \tag{11}$$

Recall that $C(x)$ is non-negative for all x and that we explicitly set $d(w_{i-1}^{l}, w_k)$ to be between 0 and 1 for all word sequences. Then the following inequality follows:

$$\sum_{\{w_k : C(w_{i-1}^{l}, w_k) > 0\}} d(w_{i-1}^{l}, w_k) \cdot \frac{C(w_{i-1}^{l}, w_k)}{C(w_{i-1}^{l})} \leq \sum_{\{w_k : C(w_{i-1}^{l}, w_k) > 0\}} \frac{C(w_{i-1}^{l}, w_k)}{C(w_{i-1}^{l})} \tag{12}$$

since taking $d = 1$ for all cases would maximize the summation. Since $C(w_{i-1}^{l})$ does not depend on w_k and is non-negative, we can simplify; it now suffices to prove that

$$\sum_{\{w_k : C(w_{i-1}^l, w_k) > 0\}} C(w_{i-1}^l, w_k) \leq C(w_{i-1}^l) \tag{13}$$

Clearly, this inequality must hold, since each time we have an occurrence of a sequence (w_{i-1}^l, w_k), we necessarily have an occurrence of w_{i-1}^l. Therefore, if $P(w_k|w_{i-1}^{l+1}) \geq 0$ for all w_{i-1}^{l+1} and w_k, then $P(w_k|w_{i-1}^l) \geq 0$ for all w_{i-1}^l and w_k. Hence, by induction, the formula defined in equation 1 does indeed give nonzero values for all w_k, so equation 1 defines a valid probability distribution.

3 Example: Trigram with Two Factors

Now that we have seen how backoff in the factor dimension can work, in this section I will work through an example of backoff in multiple dimensions using trigrams with two factors (the word and a word class). In addition to backing off in the factor dimension as described in section 2, we can back off in the direction of the history, as in standard n-gram language models. This allows for more flexibility, but also means that there are multiple n-grams to consider at any given number of backoff steps. In this model, I propose linearly interpolating the n-grams when this is the case.

I will continue using the notation introduced in section 2, where w_i^0 refers to word i itself (no backoff) and w_i^1 refers to the word's cluster (one backoff step in the factor dimension). Therefore, if we are trying to predict the probability of w_i given the two previous words w_{i-2} and w_{i-1}, we have the following backoff options for the words in the history:

- 0 backoff steps: $P(w_i|w_{i-2}^0, w_{i-1}^0)$
- 1 backoff step: $P(w_i|w_{i-2}^1, w_{i-1}^0)$, $P(w_i|w_{i-2}^0, w_{i-1}^1)$
- 2 backoff steps: $P(w_i|w_{i-1}^0)$, $P(w_i|w_{i-2}^1, w_{i-1}^1)$
- 3 backoff steps: $P(w_i|w_{i-1}^1)$
- 4 backoff steps: $P(w_i)$ (ignore history)

Note that I left out skip-grams of the form $P(w_i|w_{i-2}^j)$ from the above list of backoff steps. I did this because I wanted to include only the natural backoff order (word to less informative factor, n-gram to shorter history), but the formulas given in this section could easily be adapted to include such skip-grams.

For the given words w_{i-2} and w_{i-1} in the history, I define:

$$[w_{i-2}, w_{i-1}]_j^k \tag{14}$$

to be the k^{th} backoff option for j backoff steps. For example, given the schema above for trigrams with two factors, we might take $[w_{i-2}, w_{i-1}]_1^2$ to be (w_{i-2}^0, w_{i-1}^1).

Then the probability for a trigram with two factors using multidimensional backoff can be defined analogously to equation 1 as follows:

$$P_{BO}(w_i|[w_{i-2}, w_{i-1}]_j) =$$
$$\begin{cases} \sum_k \lambda_j^k d([w_{i-2}, w_{i-1}]_j^k, w_i) \frac{C([w_{i-2}, w_{i-1}]_j^k, w_i)}{C([w_{i-2}, w_{i-1}]_j^k)} & , \exists k : C([w_{i-2}, w_{i-1}]_j^k, w_i) > 0 \\ \alpha([w_{i-2}, w_{i-1}]_{j+1}, [w_{i-2}, w_{i-1}]_j) P_{BO}(w_i|[w_{i-2}, w_{i-1}]_{j+1}) & , else \end{cases} \tag{15}$$

where $\sum_k \lambda_j^k = 1$ for all backoff steps j.

The discounting factor $d([w_{i-2}, w_{i-1}]_j^k, w_i)$ is defined analogously to the definition in equation 2. Note that each backoff option has a different discount (instead of the discount depending on the backoff step, as it did previously). The backoff factor $\alpha([w_{i-2}, w_{i-1}]_{j+1} j, [w_{i-2}, w_{i-1}]_j)$ is also defined similarly to the definition in equation 5 in order to ensure that $\sum_k P_{BO}(w_k|[w_{i-2}, w_{i-1}]_j) = 1$.

4 General Case for Multidimensional Backoff

Now that we have an idea of how multidimensional backoff can work in practice, in this section I will give a generalization of multidimensional backoff for an arbitrary history length and number of factors. This will be a straightforward adaptation of what was done in sections 2 and 3.

For a history of length $n-1$ and words with $m-1$ additional factors, using the notation described in the previous sections, I define the probability of a word w_i given its history as:

$$P_{BO}(w_i|[w_{i-n+1}...w_{i-1}]_j) =$$
$$\begin{cases} \sum_k \lambda_j^k d([w_{i-n+1}...w_{i-1}]_j^k, w_i) \frac{C([w_{i-n+1}...w_{i-1}]_j^k, w_i)}{C([w_{i-n+1}...w_{i-1}]_j^k)} & , \exists k : C([w_{i-n+1}...w_{i-1}]_j^k, w_i) > 0 \\ \alpha([w_{i-n+1}...w_{i-1}]_{j+1} j, [w_{i-n+1}...w_{i-1}]_j) P_{BO}(w_i|[w_{i-n+1}...w_{i-1}]_{j+1}) & , else \end{cases} \tag{16}$$

As before, I define the probability of an unknown word as:

$$P_{BO}(< unk >) = \frac{|\{w_i : C(w_i) = 1\}|}{|\{w_i : C(w_i) > 0\}|} \tag{17}$$

If using Good-Turing discounting, II can define each discounting factor $d([w_{i-n+1}...w_{i-1}]_j^k, w_i)$ similarly to what has already been described in equation 2:

$$d([w_{i-n+1}...w_{i-1}]_j^k, w_i) = \frac{C([w_{i-n+1}...w_{i-1}]_j^k, w_i) + 1}{C([w_{i-n+1}...w_{i-1}]_j^k, w_i)} \times \frac{|\left\{[w_{l-n+1}...w_{l-1}]_j^k, w_l : C([w_{l-n+1}...w_{l-1}]_j^k, w_l) = C([w_{i-n+1}...w_{i-1}]_j^k, w_i) + 1\right\}|}{|\left\{[w_{l-n+1}...w_{l-1}]_j^k, w_l : C([w_{l-n+1}...w_{l-1}]_j^k, w_l) = C([w_{i-n+1}...w_{i-1}]_j^k, w_i)\right\}|} \quad (18)$$

Similarly, I define the backoff factor $\alpha([w_{i-n+1}...w_{i-1}]_{j+1}, [w_{i-n+1}...w_{i-1}]_j)$ as follows. Start by defining an auxiliary factor $\beta([w_{i-n+1}...w_{i-1}]_{j+1}, [w_{i-n+1}...w_{i-1}]_j)$:

$$\beta([w_{i-n+1}...w_{i-1}]_{j+1}, [w_{i-n+1}...w_{i-1}]_j) = 1 - \sum_{\left\{w_l : \exists k, C([w_{i-n+1}...w_{i-1}]_j^k, w_l) > 0\right\}} \sum_k \lambda_j^k d([w_{i-n+1}...w_{i-1}]_j^k, w_l) \frac{C([w_{i-n+1}...w_{i-1}]_j^k, w_l)}{C([w_{i-n+1}...w_{i-1}]_j^k)} \quad (19)$$

Then the backoff factor is:

$$\alpha([w_{i-n+1}...w_{i-1}]_{j+1}, [w_{i-n+1}...w_{i-1}]_j) = \frac{\beta([w_{i-n+1}...w_{i-1}]_{j+1}, [w_{i-n+1}...w_{i-1}]_j)}{\sum_{\left\{w_l : \forall k, C([w_{i-n+1}...w_{i-1}]_j^k, w_l) = 0\right\}} P_{BO}(w_l | [w_{i-n+1}...w_{i-1}]_{j+1})} \quad (20)$$

We can easily follow the same reasoning given in section 2.3 to derive this backoff factor. Since it was explicitly derived to fulfill

$$\sum_{w_l} P_{BO}(w_l | [w_{i-n+1}...w_{i-1}]_j) = 1 \quad (21)$$

we know that this property of probabilities holds.

As for showing that for all w_l, $P_{BO}(w_l | [w_{i-n+1}...w_{i-1}]_j)$ is non-negative, again this is an easy extension to the proof given in section 2.3.

5 Comparison to Existing Models

5.1 Factored Language Models

Although this model is a version of factored language models, there are some key differences between this model and FLMs with GPB as introduced in [4]. The principal difference is that this method adds back the notion of a backoff order that we had in traditional n-grams; in GPB, no logical backoff order is

assumed and all orders can be considered. Although what is proposed in this paper is just a first attempt at best combining the backoff order among factors with the backoff in the history, this model can easily be tweaked based on data to improve results. The natural backoff order also means that multiple factors of the same word are not considered at the same time, unlike in traditional FLMs; this makes sense because the more informative factors more or less predict the less informative ones. Multidimensional backoff also is potentially less computationally expensive than GPB; with GPB, the backoff functions do not necessarily represent a probability distribution, which means that we cannot reduce the calculation of the backoff factor $\alpha(w_i^{j+1}, w_i^j)$ as was done in equation 6. Also, not having to search for the best backoff function at runtime could be a computational advantage.

Another main difference is that this method relies on data-driven factors that are progressively less informative, whereas the FLMs in [4] were only successful when using morphological factors. No additional tool is needed to find the factors of the test data; the same factors that were found using word vectors on the training data can be used. In contrast, [4] relied on testing data that already contained factors. This has an interesting effect on the influence of unseen words. Traditional FLMs with GPB (or with an automatically learned model structure as in [5]) greatly benefit highly inflected languages, since the factors of an unseen word might still be included in the test data. However, for test data that is not pre-tagged, FLMs with GPB give no real advantage over traditional n-grams when an unseen word is in the history. In multidimensional backoff, on the other hand, all words that were seen less than a given threshold are assigned to the same factor. This means that there is no great advantage for highly inflected languages compared to sparsely inflected languages, but also that we can still consider unseen words in the history in test data that did not already include factors. This might be helpful because it could be that especially rare words tend to appear in similar contexts. Hence, multidimensional backoff is expected to be almost equally applicable to analytic and synthetic languages. It should be noted, though, that the fast vocabulary growth rate of synthetic languages is a feature that multidimensional backoff deals with particularly well, due to its treatment of unseen words.

5.2 Other Models

This model is an extension of traditional n-grams, and as such, it shares a number of properties with n-gram models. For example, it uses a history of limited length to predict the next word, and many of the smoothing methods used in n-gram models can be adapted for use with multidimensional backoff. However, using FLMs with multidimensional backoff allows us to consider different factors than just the word itself, so the language model is based on more information. In addition, multidimensional backoff adds the ability to consider unseen words in the history, which means that it works well for languages with large vocabularies and for models based on sparse training data.

A simplified version of this model would consist of linearly interpolating

class-based language models, with each word vector-based factor being a class. While this would also make use of word vectors in language modeling, it is more limited; for example, we would not be able to include n-grams that depend on a mix of factors. However, it would be a good baseline against which to test FLMs with multidimensional backoff.

6 Conclusion

This paper presented a novel framework for FLMs using word vector resources by combining linear interpolation with backoff and by backing off in multiple dimensions (among words in the history and among the factors). This framework is distinct from traditional FLMs with GPB in ways that may make it more broadly applicable and give it the potential to improve language models for languages that are not morphologically rich. Although I implemented a small prototype of multidimensional backoff, I have yet to test the framework on large data sets with adequate discounting measures, so I am not yet able to compare results between this framework and FLMs with GPB.

The most logical next step would be to fully implement multidimensional backoff for a factored language model in order to see whether it results in reduced perplexity over n-grams, class-based language models, and factored language models with generalized parallel backoff. This would also give some insight into the best method for calculating the discounting and interpolation factors. It would also be interesting to do a backoff-level analysis of the model, similar to what is described in [1]. Another option would be to compare different word vector programs, including the syntactic adaptation of word2vec described in [16], or different cluster sizes to see how their results differ. Finally, I would like to compare the results of applying this model to different languages to see whether it performs significantly better on morphologically rich languages (as the factored language models with generalized parallel backoff were found to in [4]). I plan on implementing at least a basic bigram with three factors (as described in section 2) and a trigram with two factors (as described in section 3) and testing them on various languages in the coming weeks, and if the results are promising, I hope to continue this line of work next semester.

References

[1] Adel, Heike, Ngoc Thang Vu, and Tanja Schultz. "Combination of Recurrent Neural Networks and Factored Language Models for Code-Switching Language Modeling." *ACL (2)*. 2013.

[2] Alumae, T. "Sentence-Adapted Factored Language Model for Transcribing Estonian Speech." *ICASSP 2006 Proceedings*. Vol. 1. IEEE, 2006.

[3] Bansal, Mohit, Kevin Gimpel, and Karen Livescu. "Tailoring Continuous Word Representations for Dependency Parsing." *Proceedings of the Annual Meeting of the Association for Computational Linguistics*. 2014.

[4] Bilmes, Jeff A., and Katrin Kirchhoff. "Factored Language Models and Generalized Parallel Backoff." *Proceedings of HLT-NAACL 2003 - short papers*. Association for Computational Linguistics, 2003.

[5] Duh, Kevin, and Katrin Kirchhoff. "Automatic Learning of Language Model Structure." *Proceedings of the 20th International Conference on Computational Linguistics*. Association for Computational Linguistics, 2004.

[6] El-Desoky, Amr, Ralf Schlüter, and Hermann Ney. "A Hybrid Morphologically Decomposed Factored Language Models for Arabic LVCSR." *Proceedings of HLT-NAACL 2010*. Association for Computational Linguistics, 2010.

[7] Gale, William, and Geoffrey Sampson. "Good-Turing Smoothing Without Tears." *Journal of Quantitative Linguistics* 2.3. 1995: 217-237.

[8] Katz, Slava. "Estimation of Probabilities from Sparse Data for the Language Model Component of a Speech Recognizer." *Acoustics, Speech and Signal Processing, IEEE Transactions on* 35.3. 1987: 400-401.

[9] Kirchhoff, Katrin, Jeff Bilmes, and Kevin Duh. "Factored Language Models Tutorial." Dept. of Electrical Engineering, University of Washington, Technical Report. 2007.

[10] Mikolov, Tomas. "word2vec - Tool for Computing Continuous Distributed Representations of Words." Web. 02 Apr. 2015. https://code.google.com/p/word2vec/.

[11] Mikolov, Tomas, Kai Chen, Greg Corrado, and Jeffrey Dean. "Efficient Estimation of Word Representations in Vector Space." *ICLR Workshop*. 2013.

[12] Mikolov, Tomas, Quoc V. Le, and Ilya Sutskever. "Exploiting Similarities among Languages for Machine Translation." *arXiv preprint arXiv:1309.4168*. 2013.

[13] Pennington, Jeffrey, Richard Socher, and Christopher D. Manning. "GLoVe: Global Vectors for Word Representation." *Proceedings of the Empiricial Methods in Natural Language Processing (EMNLP 2014)* 12. 2014.

[14] Shi, Tianze, and Zhiyuan Liu. "Linking GloVe with word2vec." *arXiv preprint arXiv:1411.5595*. 2014.

[15] Tachbelie, Martha Yifiru, Solomon Teferra Abate, and Wolfgang Menzel. "Morpheme-Based Language Modeling for Amharic Speech Recognition." *The 4th Language and Technology Conference*. 2009.

[16] Ling, Wang, Chris Dyer, Alan Black, and Isabel Trancoso. "Two/Too Simple Adaptations of word2vec for Syntax Problems." *Proceedings of the North American Chapter of the Association for Computational Linguistics (NAACL)*. 2015.

[17] Wu, Youzheng, et al. "Factored Language Model Based on Recurrent Neural Network." *Proceedings of COLING 2012: Technical Papers*. 2012.

[18] Zhang, Jiajun, et al. "Bilingually-Constrained Phrase Embeddings for Machine Translation." *Proceedings of the 52th Annual Meeting on Association for Computational Linguistics*. Association for Computational Linguistics, 2014.