Revisiting the Case for Explicit Syntactic Information in Language Models

Ariya Rastrow, Sanjeev Khudanpur, Mark Dredze
Human Language Technology Center of Excellence,
Center for Language and Speech Processing, Johns Hopkins University
Baltimore, MD USA
{ariya,khudanpur,mdredze}@jhu.edu

Abstract

Statistical language models used in deployed systems for speech recognition, machine translation and other human language technologies are almost exclusively n-gram models. They are regarded as *linguistically naïve*, but estimating them from any amount of text, large or small, is straightforward. Furthermore, they have doggedly matched or outperformed numerous competing proposals for *syntactically well-motivated* models. This unusual resilience of n-grams, as well as their weaknesses, are examined here. It is demonstrated that n-grams are good word-predictors, even linguistically speaking, in a large majority of word-positions, and it is suggested that to improve over n-grams, one must explore syntax-aware (or other) language models that focus on positions where n-grams are weak.

1 Introduction

Language models (LM) are crucial components in tasks that require the generation of coherent natural language text, such as automatic speech recognition (ASR) and machine translation (MT). Most language models rely on simple n-gram statistics and a wide range of smoothing and backoff techniques (Chen and Goodman, 1998). State-of-the-art ASR systems use $(n-1)$-gram equivalence classification for the language model (which result in an n-gram language model).

While simple and efficient, it is widely believed that limiting the context to only the $(n-1)$ most recent words ignores the structure of language, and several statistical frameworks have been proposed to incorporate the "syntactic structure of language back into language modeling." Yet despite considerable effort on including longer-dependency features, such as syntax (Chelba and Jelinek, 2000; Khudanpur and Wu, 2000; Collins et al., 2005; Emami and Jelinek, 2005; Kuo et al., 2009; Filimonov and Harper, 2009), n-gram language models remain the dominant technique in automatic speech recognition and machine translation (MT) systems.

While intuition suggests syntax is important, the continued dominance of n-gram models could indicate otherwise. While no one would dispute that syntax informs word choice, perhaps sufficient information aggregated across a large corpus is available in the local context for n-gram models to perform well even without syntax. To clearly demonstrate the utility of syntactic information and the deficiency of n-gram models, we empirically show that n-gram LMs lose significant predictive power in positions where the syntactic relation spans beyond the n-gram context. This clearly shows a performance gap in n-gram LMs that could be bridged by syntax.

As a candidate syntactic LM we consider the Structured Language Model (SLM) (Chelba and Jelinek, 2000), one of the first successful attempts to build a statistical language model based on syntactic information. The SLM assigns a joint probability $P(W,T)$ to every word sequence W and every possible binary parse tree T, where T's terminals are words W with part-of-speech (POS) tags, and its internal nodes comprise non-terminal labels and lexical "heads" of phrases. Other approaches include using the exposed headwords in a maximum-entropy based LM (Khudanpur and Wu, 2000), us-

NAACL-HLT 2012 Workshop: Will We Ever Really Replace the N-gram Model? On the Future of Language Modeling for HLT, pages 50–58,
Montréal, Canada, June 8, 2012.

ing exposed headwords from full-sentence parse tree in a neural network based LM (Kuo et al., 2009), and the use of syntactic features in discriminative training (Rastrow et al., 2011). We show that the long-dependencies modeled by SLM, significantly improves the predictive power of the LM, specially in positions where the syntactic relation is beyond the reach of regular n-gram models.

2 Weaknesses of n-gram LMs

Consider the following sentence, which demonstrates why the $(n-1)$-gram equivalence classification of history in n-gram language models may be insufficient:

`<s> i asked the vice president for his endorsement </s>`

In an n-gram LM, the word `for` would be modeled based on a 3-gram or 4-gram history, such as `<vice president>` or `<the vice president>`. Given the *syntactic* relation between the preposition `for` and the verb `asked` (which together make a compound verb), the strongest evidence in the history (and hence the best classification of the history) for word `for` should be `<asked president>`, which is beyond the 4-gram LM. Clearly, the *syntactic relation* between a word position and the corresponding words in the history spans beyond the limited $(n-1)$-gram equivalence classification of the history.

This is but one of many examples used for motivating *syntactic features* (Chelba and Jelinek, 2000; Kuo et al., 2009) in language modeling. However, it is legitimate to ask if this deficiency could be overcome through sufficient data, that is, accurate statistics could somehow be gathered for the n-grams even without including syntactic information. We empirically show that $(n-1)$-gram equivalence classification of history is not adequate to predict these cases. Specifically, n-gram LMs lose predictive power in the positions where the *headword* relation, exposed by the syntactic structure, goes beyond $(n-1)$ previous words (in the history.)

We postulate the following three hypotheses:

Hypothesis 1 *There is a substantial difference in the predictive power of n-gram LMs at positions within a sentence where syntactic dependencies reach further back than the n-gram context versus positions where syntactic dependencies are local.*

Hypothesis 2 *This difference does not diminish by increasing training data by an order of magnitude.*

Hypothesis 3 *LMs that specifically target positions with syntactically distant dependencies will complement or improve over n-gram LMs for these positions.*

In the following section (Section 3), we present a set of experiments to support the hypotheses 1 and 2. Section 4 introduces a SLM which uses dependency structures followed by experiments in Section 5.

3 Experimental Evidence

In this section, we explain our experimental evidence for supporting the hypotheses stated above. First, Section 3.1 presents our experimental design where we use a statistical constituent parser to identify two types of word positions in a test data, namely positions where the headword syntactic relation spans beyond recent words in the history and positions where the headword syntactic relation is within the n-gram window. The performance of an n-gram LM is measured on both types of positions to show substantial difference in the predictive power of the LM in those positions. Section 3.3 describes the results and analysis of our experiments which supports our hypotheses.

Throughout the rest of the paper, we refer to a position where the headword syntactic relation reaches further back than the n-gram context as a *syntactically-distant* position and other type of positions is referred to as a *syntactically-local* position.

3.1 Design

Our experimental design is based on the idea of comparing the performance of n-gram LMs for *syntactically-distant* vs. *syntactically-local* . To this end, we first parse each sentence in the test set using a constituent parser, as illustrated by the example in Figure 1. For each word w_i in each sentence, we then check if the "syntactic heads" of the preceding constituents in the parse of $w_1, w_2, \cdots, w_{i-1}$ are within an $(n-1)$ window of w_i. In this manner, we split the test data into two disjoint sets, $\mathcal{M}$ and $\mathcal{N}$,

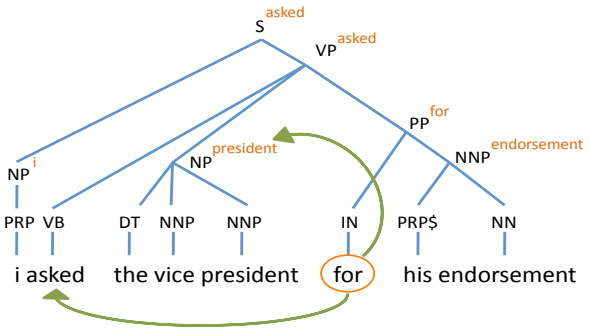


Figure 1: Example of a *syntactically distant* position in a sentence: the exposed headwords preceding `for` are $h.\mathrm{w}_{-2}$ =`asked` and $h.\mathrm{w}_{-1}$ = `president`, while the two preceding words are w_{i-2} = `vice` and w_{i-1} = `president`.

as follows,

$$\mathcal{M} = \{j | \text{positions s.t } h.\mathrm{w}_{-1}, h.\mathrm{w}_{-2} = w_{j-1}, w_{j-2}\}$$
$$\mathcal{N} = \{j | \text{positions s.t } h.\mathrm{w}_{-1}, h.\mathrm{w}_{-2} \neq w_{j-1}, w_{j-2}\}$$

Here, h_{-1} and h_{-2} correspond, respectively, to the two previous *exposed headwords* at position i, based on the syntactic structure. Therefore, $\mathcal{M}$ corresponds to the positions in the test data for which two previous *exposed heads* match exactly the two previous words. Whereas, $\mathcal{N}$ corresponds to the position where at least on of the *exposed heads* is further back in the history than the two previous words, possibly both.

To extract the exposed headwords at each position, we use a constituent parser to obtain the syntactic structure of a sentence followed by *headword percolation* procedure to get the headwords of corresponding syntactic phrases in the parse tree. The following method, described in (Kuo et al., 2009), is then used to extract exposed headwords from the history of position i from the full-sentence parse trees:

1. Start at the leaf corresponding to the word position (w_i) and the leaf corresponding to the previous context word (w_{i-1}).
2. From each leaf, go up the tree until the two paths meet at the lowest common ancestor (LCA).
3. Cut the link between the LCA and the child that is along the path from the context word w_{i-1}. The head word of the the LCA child, the one that is cut, is chosen as previous exposed headword $h.\mathrm{w}_{-1}$.

These steps may be illustrated using the parse tree shown in Figure 1. Let us show the procedure for our example from Section 2. Figure 1 shows the corresponding parse tree of our example. Considering word position w_i=`for` and w_{i-1}=`president` and applying the above procedure, the LCA is the node $\mathrm{VP}^{\texttt{asked}}$. Now, by cutting the link from $\mathrm{VP}^{\texttt{asked}}$ to $\mathrm{NP}^{\texttt{president}}$ the word `president` is obtained as the first exposed headword ($h.\mathrm{w}_{-1}$).

After the first previous exposed headword has been extracted, the second exposed headword also can be obtained using the same procedure, with the constraint that the node corresponding the second headword is different from the first (Kuo et al., 2009). More precisely,

1. set $k = 2$
2. Apply the above headword extraction method between w_i and w_{i-k}.
3. if the extracted headword has previously been chosen, set $k = k + 1$ and go to step (2).
4. Otherwise, return the headword as $h.\mathrm{w}_{-2}$.

Continuing with the example of Figure 1, after `president` is chosen as $h.\mathrm{w}_{-1}$, `asked` is chosen as $h.\mathrm{w}_{-2}$ of position `for` by applying the procedure above. Therefore, in this example the position corresponding to word `for` belongs to the set $\mathcal{N}$ as the two extracted exposed headwords (`asked,president`) are different from the two previous context words (`vice,president`).

After identifying sets $\mathcal{N}$ and $\mathcal{M}$ in our test data, we measure *perplexity* of n-gram LMs on $\mathcal{N}$, $\mathcal{M}$ and $\mathcal{N} \cup \mathcal{M}$ separately. That is,

$$\mathrm{PPL}_{\mathcal{N}\cup\mathcal{M}} = \exp\left[-\frac{\sum_{i\in\mathcal{N}\cup\mathcal{M}} \log p(w_i|W_{i-n+1}^{i-1})}{|\mathcal{N}\cup\mathcal{M}|}\right]$$
$$\mathrm{PPL}_{\mathcal{N}} = \exp\left[-\sum_{i\in\mathcal{N}} \frac{\log p(w_i|W_{i-n+1}^{i-1})}{|\mathcal{N}|}\right]$$
$$\mathrm{PPL}_{\mathcal{M}} = \exp\left[-\sum_{i\in\mathcal{M}} \frac{\log p(w_i|W_{i-n+1}^{i-1})}{|\mathcal{M}|}\right],$$

where $p(w_i|w_{i-1}w_{i-2}\cdots w_{i-n+1})$ is the conditional probability calculated by an n-gram LM at position i and $|.|$ is the size (in number of words) of the corresponding portion of the test.

In addition, to show the performance of n-gram LMs as a function of training data size, we train different n-gram LMs on 10%,20%,$\cdots$,100% of a large corpus of text and report the PPL numbers using each trained LM with different training data size. For all sizes less than 100%, we select 10 random subset of the training corpus of the required size, and report the average perplexity of 10 n-gram models. This will enable us to observe the improvement of the n-gram LMs on as we increase the training data size. The idea is to test the hypothesis that not only is there significant gap between predictive power of the n-gram LMs on sets $\mathcal{N}$ and $\mathcal{M}$, but also that this difference does not diminish by adding more training data. In other words, we want to show that the problem is not due to lack of robust *estimation* of the model parameters but due to the fact that the included features in the model (n-grams) are not *informative* enough for the positions $\mathcal{N}$.

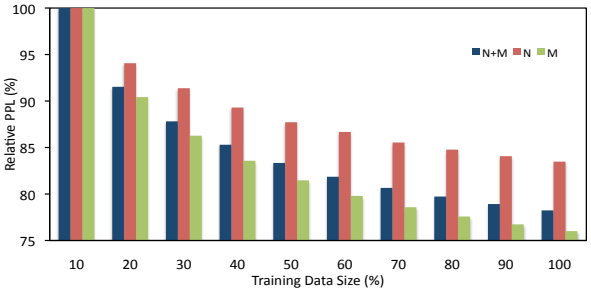
3.2 Setup

The n-gram LMs are built on $400M$ words from various Broadcast News (BN) data sources including (Chen et al., 2006): 1996 CSR Hub4 Language Model data, EARS BN03 closed captions, GALE Phase 2 Distillation GNG Evaluation Supplemental Multilingual data, Hub4 acoustic model training scripts (corresponding to the 300 Hrs), TDT4 closed captions, TDT4 newswire, GALE Broadcast Conversations, and GALE Broadcast News. All the LMs are trained using modified Kneser-Ney smoothing. To build the LMs, we sample from each source and build a source specific LM on the sampled data. The final LMs are then built by interpolating those LMs. Also, we do not apply any pruning to the trained LMs, a step that is often necessary for speech recognition but not so for perplexity measurement. The test set consists of the NIST rt04 evaluation data set, dev04f evaluation set, and rt03 evaluation set. The test data includes about $70K$ words.

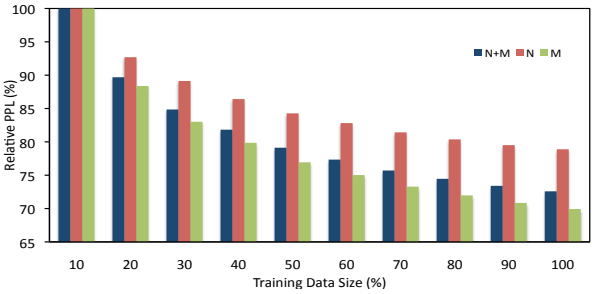
We use the parser of (Huang and Harper, 2009), which achieves state-of-the-art performance on broadcast news data, to identify the word poisons that belong to $\mathcal{N}$ and $\mathcal{M}$, as was described in Section 3.1. The parser is trained on the Broadcast News treebank from Ontonotes (Weischedel et al., 2008) and the WSJ Penn Treebank (Marcus et al., 1993) along with self-training on 1996 Hub4 CSR (Garofolo et al., 1996) utterances.

3.3 Analysis

We found that $\frac{|\mathcal{N}|}{|\mathcal{N}\cup\mathcal{M}|} \approx 0.25$ in our test data. In other words, two previous exposed headwords go beyond 2-gram history for about 25% of the test data.



(a)



(b)

Figure 2: Reduction in perplexity with increasing training data size on the entire test set $\mathcal{N}+\mathcal{M}$, on its *syntactically local* subset $\mathcal{M}$, and the *syntactically distant* subset $\mathcal{N}$. The figure shows relative perplexity instead of absolute perplexity — 100% being the perplexity for the smallest training set size — so that (a) 3-gram and (b) 4-gram LMs may be directly compared.

We train 3-gram and 4-gram LMs on 10%,20%,$\cdots$,100% of the BN training data, where each 10% increase corresponds to about $40M$ words of training text data. Figure 2 shows reduction in perplexity with increasing training data size on the entire test set $\mathcal{N}+\mathcal{M}$, on its *syntactically local* subset $\mathcal{M}$, and the *syntactically distant* subset $\mathcal{N}$. The figure basically shows relative perplexity instead of absolute perplexity — 100% being the

Position in Test Set	Training Data Size			
	40M words		400M words	
	3-gram	4-gram	3-gram	4-gram
$\mathcal{M}$	166	153	126	107
$\mathcal{N}$	228	217	191	171
$\mathcal{N}+\mathcal{M}$	183	170	143	123
$\frac{PPL_{\mathcal{N}}}{PPL_{\mathcal{M}}}$	138%	142%	151%	161%

Table 1: Perplexity of 3-gram and 4-gram LMs on *syntactically local* ($\mathcal{M}$) and *syntactically distant* ($\mathcal{N}$) positions in the test set for different training data sizes, showing the sustained higher perplexity in distant v/s local positions.

perplexity for the smallest training set size — so the rate of improvement for 3-grams and 4-gram LMs can be compared. As can be seen from Figure 2, there is a substantial gap between the improvement rate of *perplexity* in syntactically distant positions compared to that in syntactically local positions (with 400M woods of training data, this gap is about 10% for both 3-gram and 4-gram LMs). In other words, increasing the training data size has much more effect on improving the predictive power of the model for the positions included in $\mathcal{M}$. Also, by comparing Figure 2(a) to 2(b) one can observe that the gap is not overcome by increasing the context length (using 4-gram features).

Also, to better illustrate the performance of the n-gram LMs for different portions of our test data, we report the absolute values of PPL results in Table 1. It can be seen that there exits a significant difference between *perplexity* of sets $\mathcal{N}$ and $\mathcal{M}$ and that the difference gets larger as we increase the training data size.

4 Dependency Language Models

To overcome the lack of predictive power of n-gram LMs in *syntactically-distant* positions, we use the SLM framework to build a long-span LM. Our hope is to show not only that long range syntactic dependencies improve over n-gram features, but also that the improvement is largely due to better predictive power in the syntactically distant positions $\mathcal{N}$.

Syntactic information may be encoded in terms of headwords and headtags of phrases, which may be extracted from a syntactic analysis of a sentence (Chelba and Jelinek, 2000; Kuo et al., 2009), such as a *dependency structure*. A dependency in a sentence holds between a *dependent* (or *modifier*) word and a *head* (or *governor*) word: the dependent *depends* on the head. These relations are encoded in a *dependency tree* (Figure 3), a directed graph where each edge (arc) encodes a head-dependent relation.

The specific parser used to obtain the syntactic structure is not important to our investigation. What is crucial, however, is that the parser proceeds left-to-right, and only hypothesized structures based on $w_1, \ldots, w_{i-1}$ are used by the SLM to predict w_i.

Similarly, the specific features used by the parser are also not important: more noteworthy is that the SLM uses $(h.\mathrm{w}_{-3}, h.\mathrm{w}_{-2}, h.\mathrm{w}_{-1})$ and their POS tags to predict w_i. The question is whether this yields lower perplexity than predicting w_i from $(w_{i-3}, w_{i-2}, w_{i-1})$.

For the sake of completeness, we next describe the parser and SLM in some detail, but either may be skipped without loss of continuity.

The Parser: We use the shift-reduce incremental dependency parser of (Sagae and Tsujii, 2007), which constructs a tree from a transition sequence governed by a maximum-entropy classifier. Shift-reduce parsing places input words into a queue Q and partially built structures are organized by a stack S. Shift and reduce actions consume the queue and build the output parse on the stack. The classifier g assigns probabilities to each action, and the probability of a state $p_g(\pi)$ can be computed as the product of the probabilities of a sequence of actions that resulted in the state. The parser therefore provides (multiple) syntactic analyses of the history $w_1, \ldots, w_{i-1}$ at each word position w_i.

The Dependency Language Model: Parser states at position w_i, called *history-states*, are denoted $\Pi_{-i} = \{\pi^0_{-i}, \pi^1_{-i} \cdots, \pi^{K_i}_{-i}\}$, where K_i is the total number of such states. Given Π_{-i}, the probability assignment for w_i is given by

$$p(w_i|W_{-i}) = \sum_{j=1}^{|\Pi_{-i}|} p\left(w_i|f(\pi^j_{-i})\right) p_g(\pi^j_{-i}|W_{-i}) \quad (1)$$

where, W_{-i} is the word history $w_1, \ldots, w_{i-1}$ for w_i, π^j_{-i} is the j^{th} history-state of position i, $p_g(\pi^j_{-i}|W_{-i})$ is the probability assigned to π^j_{-i} by

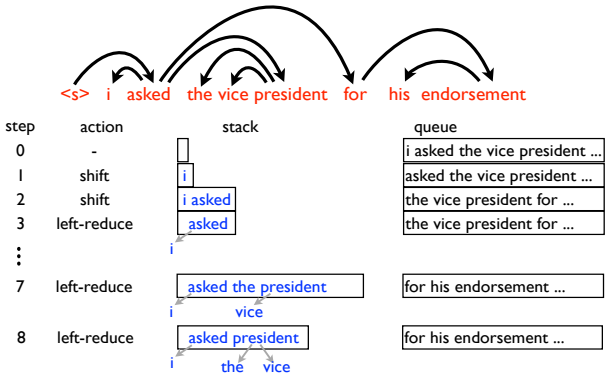


Figure 3: Actions of a shift-reduce parser to produce the dependency structure (up to the word `president`) shown above.

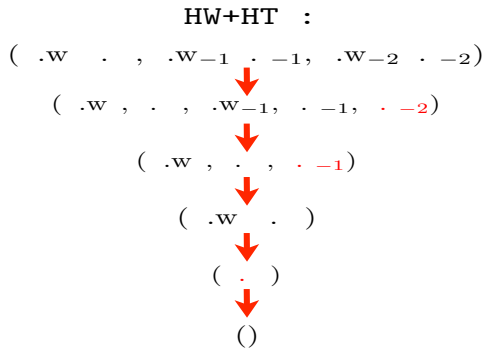


Figure 4: The hierarchal scheme of fine-to-coarse contexts used for Jelinek-Mercer smoothing in the SLM.

the parser, and $f(\pi^j_{-i})$ denotes an equivalence classification of the parser history-state, capturing features from π^j_{-i} that are useful for predicting w_i.

We restrict $f(\pi)$ to be based on only the heads of the partial trees $\{s_0\, s_1\, \cdots\}$ in the stack. For example, in Figure 3, one possible parser state for predicting the word `for` is the entire stack shown after step 8, but we restrict $f(\cdot)$ to depend only on the headwords `asked/VB` and `president/NNP`.

Given a choice of $f(\cdot)$, the parameters of the model $p(w_i|f(\pi^j_{-i}))$ are estimated to maximize the log-likelihood of the training data $\mathcal{T}$ using the Baum-Welch algorithm (Baum, 1972), and the resulting estimate is denoted $p_{\rm ML}(w_i|f(\pi^j_{-i}))$.

The estimate $p_{\rm ML}(w|f(\cdot))$ must be smoothed to handle unseen events, which we do using the method of Jelinek and Mercer (1980). We use a fine-to-coarse hierarchy of features of the history-state as illustrated in Figure 4. With

$$f_M(\pi_{-i}) \;\rightarrow\; f_{M-1}(\pi_{-i}) \;\rightarrow\; \ldots \;\rightarrow\; f_1(\pi_{-i})$$

denoting the set of M increasingly coarser equivalence classifications of the history-state π_{-i}, we linearly interpolate the higher order estimates $p_{\rm ML}\left(w|f_m(\pi_{-i})\right)$ with lower order estimates $p_{\rm ML}\left(w|f_{m-1}(\pi_{-i})\right)$ as

$$\begin{aligned} & p_{\rm JM}(w_i|f_m(\pi_{-i})) \\ &= \lambda_{f_m} p_{\rm ML}(w_i|f_m(\pi_{-i})) \\ &\quad +(1-\lambda_{f_m}) p_{\rm JM}(w_i|f_{m-1}(\pi_{-i})), \end{aligned}$$

for $1 \le m \le M$, where the 0-th order model $p_{\rm JM}(w_i|f_0(\pi_{-i}))$ is a uniform distribution.

The coefficients $\lambda_{f_m(\pi_{-i})}$ are estimated on a held-out set using the bucketing algorithm suggested by Bahl (1983), which ties $\lambda_{f_m(\pi_{-i})}$'s based on the count of $f_m(\pi_{-i})$'s in the training data. We use the expected count of the features from the last iteration of EM training, since the π_{-i} are latent states.

We perform the bucketing algorithm for each level $f_1, f_2, \cdots, f_M$ of equivalence classification separately, and estimate the bucketed $\lambda_{c(f_m)}$ using the Baum-Welch algorithm (Baum, 1972) to maximize the likelihood of held out data, where the word probability assignment in Eq. 1 is replaced with:

$$p(w_i|W_{-i}) = \sum_{j=1}^{|\Pi_i|} p_{\rm JM}\Big(w_i|f_M(\pi^j_{-i})\Big)\, p_g(\pi^j_{-i}|W_{-i}).$$

The hierarchy shown in Figure 4 is used[1] for obtaining a smooth estimate $p_{\rm JM}(\cdot|\cdot)$ at each level.

5 SLM Experiments

We train a dependency SLM for two different tasks, namely *Broadcast News* (`BN`) and *Wall Street Journal* (`WSJ`). Unlike Section 3.2, where we swept through multiple training sets of multiple sizes,

[1]The original SLM hierarchical interpolation scheme is aggressive in that it drops both the tag and headword from the history. However, in many cases the headword's tag alone is sufficient, suggesting a more gradual interpolation. Keeping the headtag adds more specific information and at the same time is less sparse. A similar idea is found, e.g., in the back-off hierarchical class n-gram language model (Zitouni, 2007) where instead of backing off from the n-gram right to the $(n-1)$-gram a more gradual backoff — by considering a hierarchy of fine-to-coarse classes for the last word in the history— is used.

training the SLM is computationally intensive. Yet, useful insights may be gained from the $40M$ word case. So we choose the source of text *most suitable* for each task, and proceed as follows.

5.1 Setup

The following summarizes the setup for each task:

- BN **setup** : EARS BN03 corpus, which has about $42M$ words serves as our training text. We also use rt04 ($45K$ words) as our evaluation data. Finally, to interpolate our structured language models with the baseline 4-gram model, we use rt03+dev04f (about $40K$ words) data sets to serve as our development set. The vocabulary we use in BN experiments has about $84K$ words.
- WSJ **setup** : The training text consists of about $37M$ words. We use eval92+eval93 ($10K$ words) as our evaluation set and dev93 ($9K$ words) serves as our development set for interpolating SLMs with the baseline 4-gram model.

In both cases, we sample about $20K$ sentences from the training text (we exclude them from training data) to serve as our heldout data for applying the bucketing algorithm and estimating λ's. To apply the dependency parser, all the data sets are first converted to Treebank-style tokenization and POS-tagged using the tagger of (Tsuruoka et al., 2011)[2]. Both the POS-tagger and the shift-reduce dependency parser are trained on the Broadcast News treebank from Ontonotes (Weischedel et al., 2008) and the WSJ Penn Treebank (after converting them to dependency trees) which consists of about $1.2M$ tokens. Finally, we train a modified kneser-ney 4-gram LM on the tokenized training text to serve as our baseline LM, for both experiments.

5.2 Results and Analysis

Table 2 shows the perplexity results for BN and WSJ experiments, respectively. It is evident that the 4-gram baseline for BN is stronger than the $40M$ case of Table 1. Yet, the interpolated SLM significantly improves over the 4-gram LM, as it does for WSJ.

Language Model	Dev	Eval
BN		
Kneser-Ney 4-gram	165	158
SLM	168	159
KN+SLM Interpolation	**147**	**142**
WSJ		
Kneser-Ney 4-gram	144	121
SLM	149	125
KN+SLM Interpolation	**132**	**110**

Table 2: Test set perplexities for different LMs on the BN and WSJ tasks.

Also, to show that, in fact, the syntactic dependencies modeled through the SLM parameterization is enhancing predictive power of the LM in the problematic regions, i.e. *syntactically-distant* positions, we calculate the following (log) probability ratio for each position in the test data,

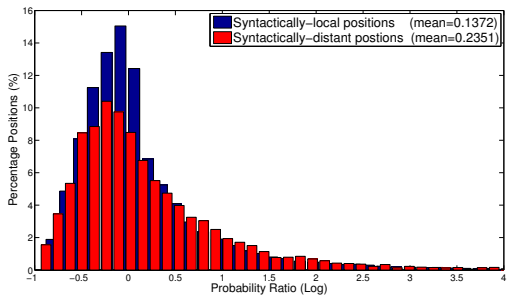
$$\log \frac{p_{\text{KN+SLM}}(w_i|W_{-i})}{p_{\text{KN}}(w_i|W_{-i})}, \qquad (2)$$

where $p_{\text{KN+SLM}}$ is the word probability assignment of the interpolated SLM at each position, and $p_{\text{KN}}(w_i)$ is the probability assigned by the baseline 4-gram model. The quantity above measures the improvement (or degradation) gained as a result of using the SLM parameterization[3].

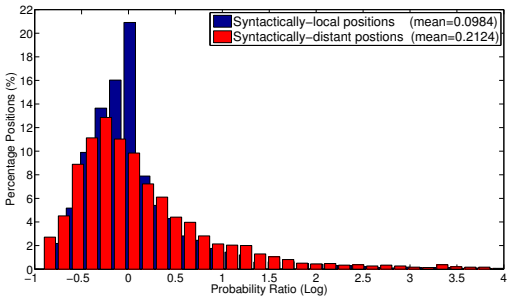
Figures 5(a) and 5(b) illustrate the histogram of the above probability ratio for all the word positions in evaluation data of BN and WSJ tasks, respectively. In these figures the histograms for *syntactically-distant* and *syntactically-local* are shown separately to measure the effect of the SLM for either of the position types. It can be observed in the figures that for both tasks the percentage of positions with $\log \frac{p_{\text{KN+SLM}}(w_i|W_{-i})}{p_{\text{KN}}(w_i|W_{-i})}$ around zero is much higher for *syntactically-local* (blue bars) than the *syntactically-distant* (red bars). To confirm this, we calculate the average $\log \frac{p_{\text{KN+SLM}}(w_i|W_{-i})}{p_{\text{KN}}(w_i|W_{-i})}$ —this is the average log-likelihood improvement, which is directly

[2]To make sure we have a proper LM, the POS-tagger and dependency parser only use features from history to tag a word position and produce the dependency structure. All lookahead features used in (Tsuruoka et al., 2011) and (Sagae and Tsujii, 2007) are excluded.

[3]If $\log \frac{p_{\text{KN+SLM}}(w_i|W_{-i})}{p_{\text{KN}}(w_i|W_{-i})}$ is greater than zero, then the SLM has a better predictive power for word position w_i. This is a meaningful comparison due to the fact that the probability assignment using both SLM and n-gram is a proper probability (which sums to one over all words at each position).



(a) BN



(b) WSJ

Figure 5: Probability ratio histogram of SLM to 4-gram model for (a) BN task (b) WSJ task.

related to perplexity improvement— for each position type in the figures.

Table 3, reports the perplexity performance of each LM (baseline 4-gram, SLM and interpolated SLM) on different positions of the evaluation data for BN and WSJ tasks. As it can be observed from this table, the use of long-span dependencies in the SLM partially fills the gap between the performance of the baseline 4-gram LM on *syntactically-distant* positions $\mathcal{N}$ versus *syntactically-local* positions $\mathcal{M}$. In addition, it can be seen that the SLM by itself fills the gap substantially, however, due to its underlying parameterization which is based on Jelinek-Mercer smoothing it has a worse performance on regular *syntactically-local* positions (which account for the majority of the positions) compared to the Kneser-Ney smoothed LM[4]. Therefore, to improve the overall performance, the interpolated SLM takes advantage of both the better modeling performance of Kneser-Ney for *syntactically-local* positions and

[4]This is merely due to the superior modeling power and better smoothing of the Kneser-Ney LM (Chen and Goodman, 1998).

Test Set	4-gram	SLM	4-gram + SLM
Position	BN		
$\mathcal{M}$	146	152	132
$\mathcal{N}$	201	182	171
$\mathcal{N}+\mathcal{M}$	158	159	142
$\frac{PPL_{\mathcal{N}}}{PPL_{\mathcal{M}}}$	138%	120%	129%
	WSJ		
$\mathcal{M}$	114	120	105
$\mathcal{N}$	152	141	131
$\mathcal{N}+\mathcal{M}$	121	125	110
$\frac{PPL_{\mathcal{N}}}{PPL_{\mathcal{M}}}$	133%	117%	125%

Table 3: Perplexity on the BN and WSJ evaluation sets for the 4-gram LM, SLM and their interpolation. The SLM has lower perplexity than the 4-gram in *syntactically distant* positions $\mathcal{N}$, and has a smaller discrepancy $\frac{PPL_{\mathcal{N}}}{PPL_{\mathcal{M}}}$ between preplexity on the distant and local predictions, complementing the 4-gram model.

the better features included in the SLM for improving predictive power on *syntactically-distant* positions.

6 Conclusion

The results of Table 1 and Figure 2 suggest that predicting the next word is about 50% more difficult when its syntactic dependence on the history reaches beyond n-gram range. They also suggest that this difficulty does not diminish with increasing training data size. If anything, the relative difficulty of word positions with nonlocal dependencies relative to those with local dependencies appears to *increase* with increasing training data and n-gram order. Finally, it appears that language models that exploit long-distance syntactic dependencies explicitly at positions where the n-gram is least effective are beneficial as complementary models.

Tables 2 and 3 demonstrates that a particular, recently-proposed SLM with such properties improves a 4-gram LM trained on a large corpus.

Acknowledgments

Thanks to Kenji Sagae for sharing his shift-reduce dependency parser and the anonymous reviewers for helpful comments.

References

LR Bahl. 1983. A maximum likelihood approach to continuous speech recognition. *IEEE Transactions on Pattern Analysis and Machine Inteligence (PAMI)*, 5(2):179–190.

L. E. Baum. 1972. An equality and associated maximization technique in statistical estimation for probabilistic functions of Markov processes. *Inequalities*, 3:1–8.

C. Chelba and F. Jelinek. 2000. Structured language modeling. *Computer Speech and Language*, 14(4):283–332.

SF Chen and J Goodman. 1998. An empirical study of smoothing techniques for language modeling. Technical report, Computer Science Group, Harvard University.

S. Chen, B. Kingsbury, L. Mangu, D. Povey, G. Saon, H. Soltau, and G. Zweig. 2006. Advances in speech transcription at IBM under the DARPA EARS program. *IEEE Transactions on Audio, Speech and Language Processing*, pages 1596–1608.

M Collins, B Roark, and M Saraclar. 2005. Discriminative syntactic language modeling for speech recognition. In *ACL*.

Ahmad Emami and Frederick Jelinek. 2005. A Neural Syntactic Language Model. *Machine learning*, 60:195–227.

Denis Filimonov and Mary Harper. 2009. A joint language model with fine-grain syntactic tags. In *EMNLP*.

John Garofolo, Jonathan Fiscus, William Fisher, and David Pallett, 1996. *CSR-IV HUB4*. Linguistic Data Consortium, Philadelphia.

Zhongqiang Huang and Mary Harper. 2009. Self-Training PCFG grammars with latent annotations across languages. In *EMNLP*.

Frederick Jelinek and Robert L. Mercer. 1980. Interpolated estimation of Markov source parameters from sparse data. In *Proceedings of the Workshop on Pattern Recognition in Practice*, pages 381–397.

S. Khudanpur and J. Wu. 2000. Maximum entropy techniques for exploiting syntactic, semantic and collocational dependencies in language modeling. *Computer Speech and Language*, pages 355–372.

H. K. J. Kuo, L. Mangu, A. Emami, I. Zitouni, and L. Young-Suk. 2009. Syntactic features for Arabic speech recognition. In *Proc. ASRU*.

M.P. Marcus, M.A. Marcinkiewicz, and B. Santorini. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):330.

Ariya Rastrow, Mark Dredze, and Sanjeev Khudanpur. 2011. Efficient discrimnative training of long-span language models. In *IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*.

K. Sagae and J. Tsujii. 2007. Dependency parsing and domain adaptation with LR models and parser ensembles. In *Proc. EMNLP-CoNLL*, volume 7, pages 1044–1050.

Yoshimasa Tsuruoka, Yusuke Miyao, and Jun'ichi Kazama. 2011. Learning with Lookahead : Can History-Based Models Rival Globally Optimized Models ? In *Proc. CoNLL*, number June, pages 238–246.

Ralph Weischedel, Sameer Pradhan, Lance Ramshaw, Martha Palmer, Nianwen Xue, Mitchell Marcus, Ann Taylor, Craig Greenberg, Eduard Hovy, Robert Belvin, and Ann Houston, 2008. *OntoNotes Release 2.0*. Linguistic Data Consortium, Philadelphia.

Imed Zitouni. 2007. Backoff hierarchical class n-gram language models: effectiveness to model unseen events in speech recognition. *Computer Speech & Language*, 21(1):88–104.