LINGUISTIC FEATURES FOR WHOLE SENTENCE MAXIMUM ENTROPY LANGUAGE MODELS

Xiaojin Zhu *Stanley F. Chen* *Ronald Rosenfeld*
School of Computer Science
Carnegie Mellon University
{zhuxj, sfc, roni}@cs.cmu.edu

ABSTRACT

Whole sentence maximum entropy models directly model the probability of a sentence using *features* – arbitrary computable properties of the sentence. We investigate whether linguistic features that capture the underlying linguistic structure of a sentence can improve modeling. We use a shallow parser to parse sentences into linguistic constituents in two corpora; one is the original training corpus, and the other is an artificial corpus generated from an initial trigram model. We define three sets of candidate linguistic features based on these constituents, and compute the prevalence of each feature in the two data sets. We select features with significantly different frequencies. These correspond to phenomena poorly modeled by traditional trigrams, and reveal interesting linguistic deficiencies of the initial model. We found 6798 linguistic features in the Switchboard domain and achieved small improvements in perplexity and speech recognition accuracy with these features.

Keywords: maximum entropy, linguistic features

1. INTRODUCTION

Previous language models typically apply the chain rule to decompose the probability of a sentence into a product of conditional word probabilities. For example, trigram language models compute *P(w3 | w1, w2)*, where the probability of a word is conditioned on its two preceding words. Successful as these conditional models are, they lack the flexibility to incorporate global information. For instance, it is awkward for a trigram language model to express information about sentence length.

In [1], we introduced a non-conditional Whole Sentence Maximum Entropy model that directly models the probability of an entire sentence *s*: [1]

$$P_{ME}(s) = \frac{1}{Z} \cdot P_0(s) \cdot \exp(\sum_i \lambda_i f_i(s))$$

$P_0(s)$ is an initial probability distribution over sentences. We can obtain $P_0(s)$ from a traditional word trigram language model. The f_i's are *features*, or arbitrary computable properties of a sentence. The λ_i's are the parameters of the maximum entropy language model, to be estimated from training data. The value Z is a normalization factor to make $P_{ME}(s)$ a probability distribution. The model $P_{ME}(s)$ satisfies these properties: **1**. It is consistent with the training data in terms of feature expectations

$$\sum_s P_{ME}(s) f_i(s) = \sum_s \widetilde{P}(s) f_i(s), \qquad \forall f_i$$

where $\widetilde{P}(s) = (\#s / \#total\ sentences)$ is the empirical distribution of the training data. **2**. Among all of the probability distributions which satisfy 1, it is the one most similar to the initial model $P_0(s)$ in terms of Kullback-Leibler divergence. In [2], we presented efficient methods for training and applying such models, and constructed a model for Switchboard data using word *n*-grams (up to *n*=4), distance-two word *n*-grams (up to *n*=3), and class *n*-grams (up to *n*=5).

However, we believe that significant improvement in language modeling can be obtained by using *linguistic features*, namely features that capture underlying linguistic phenomena. These features typically correspond to hidden (and often global) linguistic structure, as opposed to *n*-grams, which capture localized, surface phenomena.

In this paper, we present a method of automatically finding certain linguistic features for a whole sentence maximum entropy model. Section 2 describes the linguistic features we are interested in. Section 3 details the feature selection method. In

We are grateful to Klaus Zechner for the shallow parser.

[1] In fact, it is the Minimum Discrimination Information (MDI) model relative to P_0.

Section 4, we apply the method to the Switchboard task and discuss its performance.

2. LINGUISTIC FEATURES

In order to use linguistic features, we need a linguistic module to process input sentences and output certain linguistic information. Due to the nature of our domain (Switchboard), we derive linguistic features using a shallow parser [3] that produces constituent sequences given an input utterance. The linguistic features are defined on the output constituent sequences.

2.1 The Shallow Parser

The shallow parser is designed to parse spontaneous, conversational speech in unrestricted domains. It is very robust and fast for such sentences. Due to the high irregularity of spontaneous speech, a series of preprocessing steps are carried out first. These include eliminating word repetitions, expanding contractions, and cleaning dysfluencies, etc. In addition, the parser assigns a Penn Treebank and Brown Corpus style part-of-speech tag to each word. For example, the input sentence

Okay I uh you know I think it might be correct

is processed as

I/NNP think/VBP it/PRPA might/AUX be/VB correct/JJ

Next, the parser breaks the preprocessed sentence into one or more *simplex clauses*, which are clauses that contain an inflected verbal form and a subject. This simplifies the input sentence and makes parsing more robust. In our example above, the parser generates two simplex clauses:

simplex 1: *I/NNP think/VBP*

simplex 2: *it/PRPA might/AUX be/VB correct/JJ*

Finally, with a set of handwritten grammar rules, the parser parses each simplex clause into constituents. The parsing is shallow since it doesn't generate embedded constituents; i.e., the parse tree is flat. In the example, simplex 1 has two constituents:

[_np] ([NP_head] I/NNP)
[_vb] ([VP_head] think/VBP)

and simplex 2 has three constituents:

[_np] ([NP_head] it/PRPA)
[_vb] (might/AUX [VP_head] be/VB)
[_prdadj] (correct/JJ)

The parser sometimes leaves a few function words (e.g. *to*, *of*, *in*) unparsed in the output. From a feature selection point of view, we regard each of these function words as a special constituent in itself. In this way, there are a total of 110 constituent types.

2.2 Feature Form

As mentioned above, the shallow parser breaks an input sentence into one or more simplex clauses. For each simplex clause, it generates a flat sequence of constituents. We define three types of features based solely on the types of constituents; i.e., we ignore the identities of words within the constituents:

1. Constituent Sequence features: for any constituent sequence x and simplex clause s, $f_x(s)$=1 if and only if the constituent sequence of simplex clause s exactly matches x. Otherwise $f_x(s)$=0. For instance, $f_{_np_vb}$("I think")=1, $f_{_np_vb_prdadj}$("it might be correct")=1, $f_{_np_vb}$("it might be correct")=0, and so forth.
2. Constituent Set features: for any set x of constituents, $f_x(s)$=1 if and only if the constituent set of sentence s exactly matches x. Otherwise $f_x(s)$=0. This set of features is a relaxation of Constituent Sequence features, since it doesn't require the position and number of constituents to match exactly. As an example, both $f_{\{_np,\ _vb\}}$("I laugh")=1 and $f_{\{_np,\ _vb\}}$("I see a bird")=1, although the constituent sequence of "I laugh" is "_np _vb" while that of "I see a bird" is "_np _vb _np".
3. Constituent Trigram features: for any ordered constituent triplet *(c1, c2, c3)*, $f_{(c1,c2,c3)}(s)$=1 if and only if sentence s contains that contiguous sequence at least once. Otherwise $f_{(c1,c2,c3)}(s)$=0. This set of features resembles traditional class trigram features.

3. FEATURE SELECTION

We follow the procedure in [2] to find salient features for the whole sentence maximum entropy model. We have a training corpus and an initial distribution $P_0(s)$. We generate an artificial corpus, roughly the same size as the training corpus, by sampling from $P_0(s)$. We run both corpora through the shallow parser and investigate the behavior of each candidate feature. If the number of times a feature is 'on' (equals 1) in the training corpus differs significantly from that in the artificial corpus, the feature is considered important and will be incorporated into the model. The rationale is that the difference is due to the deficiency of the initial model $P_0(s)$, and adding such a feature will bring the model closer to reality.

3.1 Significance Test

We assume that our features occur independently, and are therefore binomially distributed. More precisely, we have two independent sets of Bernoulli trials. One is the set of n simplex clauses of the training corpus. The other is the set of m simplex clauses of the artificial corpus. Let x be the number of times a feature occurs in the training corpus trials and y that in the artificial corpus trials. Let P_x and P_y be the true occurrence probabilities associated with each set of trials. We test the hypothesis

$$H_0 : P_x = P_y \qquad \text{versus} \qquad H_\alpha : P_x \neq P_y$$

at the significance level α. Approximating the Generalized Likelihood Ratio Test, we reject H_0 at confidence level α if

$$\left| \frac{x/n - y/m}{\sqrt{\left(\frac{x+y}{n+m}\right)\cdot\left(1-\frac{x+y}{n+m}\right)\cdot(n+m)/(nm)}} \right| \geq z_{\alpha/2}$$

(see [4]). We include into the whole sentence maximum entropy model those features whose H_0 is rejected.

3.2 Estimating Perplexity Reduction

After multiple features are added to the initial model and trained, the test set perplexity reduction can be estimated as follows. Let

$$R(s) = \exp(\sum_i \lambda_i f_i(s))$$

be the unnormalized modification made to the initial model for sentence s. By the normalization constraint, we have

$$\sum_{all\ s} P_{ME}(s) = \frac{1}{Z}\sum_{all\ s} P_0(s)\cdot R(s) = \frac{1}{Z} E_0\left[R(s)\right] = 1$$

The expectation is over the initial model P_0. Thus the normalization factor is $Z = E_0\left[R(s)\right]$. We can estimate it from a large text T_0 drawn from P_0:

$$\hat{Z} = \underset{s\in T_0}{\text{arithmetic mean}}\left[R(s)\right]$$

The estimation is very accurate since the number of sentences in T_0 is large. Let T_e be the test set on which we want to compute perplexity. By definition

$$PP(T_e) = P(T_e)^{-\frac{1}{\#W_e}}$$

where $\#W_e$ is the number of words in T_e. It can be shown that the perplexity reduction ratio is

$$\frac{PP_{ME}(T_e)}{PP_0(T_e)} = \left(\frac{Z}{\sqrt[\#S_e]{\prod_{s\in T_e} R(s)}} \right)^{\frac{\#S_e}{\#W_e}}$$

where $\#S_e$ is the number of sentences in T_e. Substituting the estimation of Z in, the estimate of perplexity reduction is

$$\frac{PP_{ME}(T_e)}{PP_0(T_e)} \approx \left(\frac{\underset{s\in T_0}{\text{arithmetic mean}}\left[R(s)\right]}{\underset{s\in T_e}{\text{geometric mean}}\left[R(s)\right]} \right)^{\frac{\#S_e}{\#W_e}}$$

where $\#S_e/\#W_e$ is the average number of words per sentence in the test set. Interestingly, if T_e=T_0, i.e. if the test set is also sampled from the prior distribution, it follows from the law of inequality of averages that the new perplexity will always be higher. This, however, is appropriate because any correction to the prior probability distribution will assign a lower likelihood (and hence higher perplexity) to the prior-generated data.

4. EXPERIMENT RESULT

Our training corpus is Switchboard (SWB) conversational telephone speech with nearly 187,000 sentences and 2,895,000 words. An artificial corpus of similar size was generated from a trigram model. We investigate the three sets of features discussed in section 2.2 with the test in section 3.1.

4.1 Features Discovered

1. There are 186,903 candidate Constituent Sequence features that occur at least once in the two corpora. Of those, 1,935 show a significant difference between the two corpora at a 95% confidence level (two-tailed). The feature *f_np_vb_np_pp()* has the most significant standard score 21.9 in the test, with x=2968 occurrences in the SWB corpus and y=1548 in the artificial corpus. More interesting is the feature *f_conj_np_aux_prdadj*() with z-score 4.3, and x=0, y=19. This means there are 19 simplex clauses in the artificial corpus with the constituent sequence "_conj _np _aux _prdadj", but none in the SWB corpus. One may suspect that this is where the initial trigram model 'makes up' some nonsense sentence. Looking at the 19 simplex clauses confirms our suspicion:

"so I have never really interesting"

"and they might practical"
"that we have good"
"that you could last"
"but I would sure"
"and you can convenient" ...

Similarly, the feature $f_{_wh_np_vb_np_vb_in}()$ has standard score -4.0, x=16 and y=0. This stands for a perfectly legal simplex clause form that has never been generated in the artificial corpus. Here are the simplex clauses in SWB:

"what area do you work in"
"what area do you live in"
"what home do you live in"
"what exercise do you get involved in"...

2. As expected, Constituent Set features are more general than Constituent Sequence features and thus there are fewer of them. A total of 61,741 candidate Constituent Set features occur in either corpus, while 1310 show a significant difference. The one with the most significant z-score, 27.8, is $f_{\{_conj,\ _np,\ _pp,\ _vb\}}()$ with x=10420 and y=6971. Like Constituent Sequence features, there are some Constituent Set features that occur only in the artificial corpus. For example, $f_{\{_adv,\ _conj,\ a\}}()$ has a z-score of 4.0 with x=0 and y=16:

"or a totally"
"and a properly"
"if a whatsoever" ...

There are also features that only occur in the SWB corpus, such as $f_{\{_np,\ _pp,\ _vb,\ _wh,\ from\}}()$ with z-score 3.8, x=14 and y=0.

3. 36,448 candidate Constituent Trigram features appear in the corpora. 3535 are significant. The feature $f_{(_np,\ _adv,\ some)}()$ with z-score 4.9, x=0 and y=25 is another good example of how the initial trigram model generates bad sentences:

"but ***he never some****"*
"we ***the gym even some****"*
*"****it really some*** *really bad"*
*"****myself sometimes some*** *on channel 8 dollars"*

4.2 Perplexity and WER

We added the 1953 Constituent Sequence features, 1310 Constituent Set features, and 3535 Constituent Trigram features to the whole sentence maximum entropy language model, and trained the parameters with the GIS algorithm [5].

The perplexity of a 90,600-word test set was calculated under the initial model P_0, and was found to be 81.37. The perplexity under the new maximum entropy model was estimated as 80.49±0.02, a relative improvement of 1%.

We tested speech recognition word error rate by N-best list rescoring. A 200-best list with 8,300 words was used. The WER was 36.53% with the initial model, and 36.38% with all the linguistic features added, a 0.4% relative improvement.

5. CONCLUSION

We presented an approach of combining linguistic and statistical methods to model natural language and showed the improvement obtained in the Switchboard domain. It demonstrates the generality of the whole sentence maximum entropy paradigm; i.e., any computable property of natural language can be included in such a model.

The improvement we achieved was very small. We hypothesize that it is mainly because most of our features are infrequent. An ideal feature should occur frequently enough, yet display a significant difference between the two corpora. "Does the sentence make sense to a human reader?" is such a feature, although it is hard to approximate it computationally. In addition, the SWB domain is also too unstructured to allow deep parsing. New types of linguistic features, more structured domains (like Broadcast News), and a more sophisticated parser may have a bigger impact.

REFERENCES

[1] R. Rosenfeld (1997), A Whole Sentence Maximum Entropy Language Model. *Proceedings of IEEE ASRU*, pp. 230–237

[2] S. Chen, R. Rosenfeld (1999), Efficient Sampling And Feature Selection in Whole Sentence Maximum Entropy Language Models. *ICASSP99*, V1. pp. 549-552

[3] K. Zechner (1997), Building Chunk Level Representations for Spontaneous Speech. *MS Project Report, Department of Philosophy, Carnegie Mellon University*

[4] R. Larsen, M. Marx (1981), An introduction to mathematical statistics and its applications. *Prentice-Hall,* N.J. pp 335

[5] J. Darroch and D. Ratcliff (1972), Generalized iterative scaling for log-linear models. *Ann. Of Math. Stat.*, 43:1470-1480