A NOVEL METHOD FOR LANGUAGE MODEL SMOOTHING

CHEN Langzhou, HUANG Taiyi, XU Bo
Institute of Automation, CAS, Beijing, 100080

Abstract: This paper presented a new method for language model smoothing. In statistical language model, sparse data is a serious problem. Traditional smoothing method for n-gram model such as backoff, interpolation, uses the value of low level model to estimate the n-gram values that did not occur in training corpus. In the method presented in this paper, the unseen event was estimated using the same level model. We find the similar word of current word using singular value decomposition(SVD) of the word relation matrix, and replace the current word using the similar words we have found. The unseen event is estimated according to this new word pairs. The experiments showed that the new method can reduce about 6% perplexity compared with traditional model.

1. Introduction

Data sparse problem is a very important problem in statistic language model. We can not observe all the events in natural language even if very large corpus is used. In order to solve this problem, we must adopt the smoothing method. The traditional methods[1][2] for n-gram model smoothing used the strategy that discount the probability mass of the seen events that estimated by maximum likelihood method, and redistribute this discounted mass to the unseen event according to the less specific estimation such as (n-1)-gram. But this kind of less specific estimation must lead to much loss of information. Reference[3] presented a more precise model that the redistribution of discounted mass is not based on the (n-1)-gram, but class-based n-gram. No matter (n-1)-gram backoff or class based backoff, the unseen event is estimated using a less level model. In this work, we try to avoid this less specific estimation and estimate the probability of unseen event more accurately.

In the method presented in this paper, we did not estimate the unseen event using less specific distribution, we try to find a seen event which is closest to the unseen event and using the estimation of this seen event to replace the unseen one. For example, using our method to estimate a bigram model, the probability of an unseen word pair $w_1 w_2$ can be expressed as:

$$p(w_1 w_2) = p(w_S w_2) \qquad (1)$$

where $w_S = \arg\max_w Sim(w_1, w)$ and $Sim(w_1, w)$ is a similarity function to express the similarity of two words. The similarity between the words can be defined as similarity between two words' context. In n-gram model, the context is the word which following the current word. So the similarity of two words is expressed as the degree of these two words' following words overlap. But because of the plenty of synonymy, the simple word match can not get good performance. In order to get credible expression of context, We calculated the word vectors using a truncated singular value decomposition(SVD) of the word relation matrix. After get the similar words of current word, we modified the equation of backoff model and interpolation model according to the new smoothing method.

Section 2 is a introduction of word relation matrix and SVD method. Section 3 is the new smoothing algorithm based on similar context. In section 4, we present some experiment result of our new algorithm.

2. The Similarity of Two Words

In order to scale the similarity of two words, at first, a matrix A of word relation has been presented. A is a square. The elements of this matrix are the frequencies in which a word followed by other words, i.e. the element a_{ji} is the frequency the word w_i following the word w_j. As mentioned above, the similarity of two word is defined as the similarity between two words' context. In n-gram model, the context is the history word. Because every column of word relation matrix represents the history words of corresponding word in vocabulary, we can compare different columns in word relation matrix to get the similarity of two words. But this method is not accurate. There are plenty of synonymy in natural language and the same word can express different means. It will lead to many error that simply comparing the context to evaluate the similarity of two word. In order to avoid this kind of error, we introduced a more robust method to calculate the word similarity, i.e. get the similarity information through the

truncated SVD of relation matrix A.The SVD of A is expressed as:

$$A = U\Sigma V^T \qquad (2)$$

where $\Sigma = diag(\lambda_1\lambda_2\cdots\lambda_n)$ and $U^TU = V^TV = I_n$.

If $rank(A) = R$, then matrix A can be restored as

$$A = \sum_{i=1}^{R} u_i\lambda_i v_i^T$$

If we sort the singular value of A as

$$\lambda_1 > \lambda_2 > \cdots > \lambda_k > \cdots$$

then the best approximation to A is

$$A_k = \sum_{i=1}^{k} u_i\lambda_i v_i^T$$

SVD method had been used in Latent semantic indexing(LSI). In LSI, a term by document matrix was constructed, then using SVD to get its document vectors, and compute the nearest document vector to the query vector as the result.

In our application, we do truncated SVD to word relation matrix A. Then, only reserve the r biggest singular values $\lambda_1\lambda_2\cdots\lambda_r$ and the singular vectors associated with them, whereby the context of each word w_i is represented as a right singular vector v_i in r-space. These context vectors contain compressed and more reliable information of words' context. The closer two context vectors are, the more similar two words are. In figure 1, three word words' context vectors are showed, we can find that the context vectors of w_1 and w_2 are much closer than w_3, it means word w_1 and w_2 are very similar.

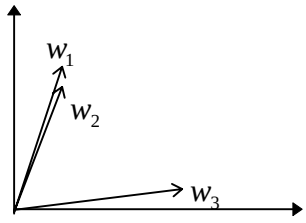

Figure 1

After getting these word context vectors, the similarity between words can be represented accurately. We defined the similarity function of two words as the cosine angle between their word context vectors, i.e.

$$Sim(w_1, w_2) = \frac{v_1 \bullet v_2}{|v_1||v_2|} \qquad (3)$$

We introducing the truncated SVD to matrix A can avoid the direct comparing between the two words' context. After this processing, every word and it's context is represented as a vector in r space and the comparison of two words is changed as comparison of two vector. This processing can filter the noise brought by the direct words comparing and get the substantial structure of word context.

Some most similar word of word "教授" is showed in table 1.

Table 1

Similar word	value of similar function
专家	0.92
学者	0.87
先生	0.86
局长	0.79

From table 1 we can find that the word "专家","学者" are very close to "教授" in semantic, and "先生" is a little difficult, "局长" is more difficult.

To every word w_i in vocabulary, we can get a set of it similar words $\Theta(w_i)$ according to similarity function. And we define the probability $P(w_j/\Theta(w_i))$ as

$$P(w_j/\Theta(w_i)) = \frac{\sum_{w_k\in\Theta(w_i)} Count(w_k w_j)}{\sum Count(\Theta(w_i))} \qquad (4)$$

$Count(*)$ is the times that event '*' occur in corpus. Eq(4) is a more approximate value of the probability of unseen event than the low level model.

3. Smoothing Method

Now, we can present our smoothing method. The smoothing method often has two ways, one is backoff, the other is interpolation.

The traditional backoff method based on Turing's formula, as Equation(5)

$$\tilde{p}(x/y)=\begin{cases} d_c(yx)*\dfrac{Count(yx)}{Count(y)}, \\ if(Count(yx)>0) \\ \dfrac{1-\sum_{x:Count(yx)>0}\tilde{p}(x/y)}{\sum_{x:Count(yx)=0}p(x)}*p(x), \\ if(Count(yx)=0) \end{cases} \quad (5)$$

where $d_c(yx)$ is the discount factor determined by Turing's formula.

In the method presented in this paper, the backoff method has changed as eq(6). The traditional backoff processing has two cases. If the event is seen in corpus, then the probability of this event is estimated as the maximum likelihood estimation weighted with a discount factor, the probability of unseen events, it backoff to less specific model, i.e. the unseen events distribute the discounted mass of probability according to the less specific distribution. In the method presented in this paper, the backoff processing have three cases. If the event is seen in corpus, it's same as the traditional model. When the event is unseen, two case must be considered: replacing the current word with it similar word that calculated using Eq(3), if the co-occurrence of the history word and the similar word is seen in the corpus, then the probability will be estimated according to the distribution of Eq(4), the most of unseen event can be calculated in this case, if co-occurrence of the history word and the similar word is still unseen, the probability will be estimated by less specific distribution.

$$\tilde{p}(x/y)=\begin{cases} d_c(yx)*\dfrac{Count(yx)}{Count(y)}, Count(yx)>0 \\ d_c(\Theta(y)x)*\dfrac{1-\sum_{x:Count(yx)>0}\tilde{p}(x/y)}{\sum_{x:Count(yx)=0\&\&Count(\Theta(y),x)>0}p(x/\Theta(y))} \\ *p(x/\Theta(y)), \\ Count(yx)=0\&\&Count(\Theta(y),x)>0 \\ \dfrac{1-\sum_{x:Count(yx)>0}\tilde{p}(x/y)-\sum_{x:Count(yx)=0\&\&Count(\Theta(y),x)>0}\tilde{p}(x/y)}{\sum_{x:Count(yx)=0\&\&Count(\Theta(y),x)=0}p(x)} \\ *p(x), \\ Count(yx)=0\&\&Count(\Theta(y),x)>0 \end{cases} \quad (6)$$

Traditional interpolation method using discounting model can be represented as:

$$p(x/y)=\frac{Count(yx)-d(y,x)}{Count(y)}+Q_y[d]*p(x) \quad (7)$$

where $d(y,x)$ is a discounting function and

$$Q_y[d]=\frac{\sum_x d(y,x)}{Count(y)} \quad (8)$$

It is interpolation of n-gram model and its less special model.

In the method presented here, we interpolate 1. the probability of current word pair, 2. the probability of co-occurrence of history word and similar word, 3. the probability estimated by less specific model. The interpolation model is changed as

$$p(x/y)=\frac{Count(yx)-d(y,x)}{Count(y)}+ Q_y[d]*\frac{\sum_{w:w\in\Theta(y)}Count(wx)-d'(y,x)}{\sum_{w:w\in\Theta(y)}Count(w)}+Q'_y[d]*\frac{Count(x)}{N} \quad (9)$$

where $Q'_y[d] = \frac{\sum_x d'(y,x)}{Count(y)}$ (10)

4. Experiment

We have do some experiments about this new method. We used the corpus about 5M from "People Daily" to find the similarity between among the words. In order to compress the storage space, we reduced the size of vocabulary to 6,000 words. We use other 30K corpus as test text.

The experiment result of backoff model is showed in table2

Table 2

	traditional backoff	new backoff
perplexity	87	93

In the experiment of interpolation, we make the discounting function $d(y,x)$ as a constant d ,figure 2 is the result of interpolation method, the perplexity varied along with the constant d

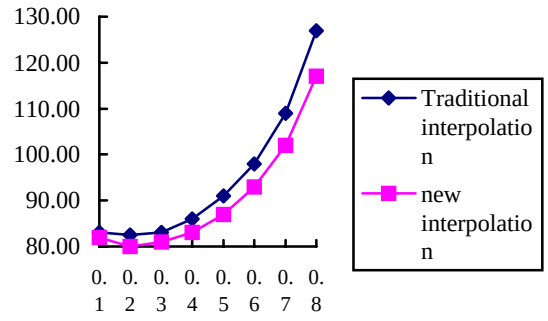

Figure 2

The result showed that using the method presented in this paper, the perplexity fell about 6% comparing with the traditional backoff model. and fell about 5.7% comparing with the traditional interpolation model.

The next experiment is to test the relation between number of word context vector dimensions and the performance of language model. We modified the number of word context vector dimensions n from 5 to 200, the result is showed in figure 3.

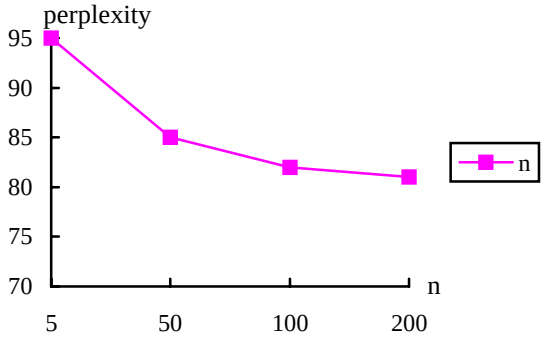

Figure 3

From figure 3 we can find that when n is less than 100 the perplexity fell evidently, when n is bigger than 100, the performance of system has not much improvement.

REFERENCE

[1] Slava M.Katz, Estimation of Probabilities from Sparse Data for the Language Model Component of a Speech Recognition. IEEE Trans on ASSP, Vol35, No 3. P400-P401

[2] N.Ney et al, Probabilistic Dependences in Stochastic Language Modeling, Computer Speech and Language.

[3] Iohn W.Miller Fil Alleva, Evaluation of a Language Model using a Clustered Model Backoff, ICSLP96

[4] Michael W.Berry et al, Using Linear Algebra for Intelligent Information Retrieval. SIAM Review, Vol.37, No 4, pp573-595

[5] M.W.Berry et al, SVDPACKC: Version 1.0 User's Guide, Tech.Report CS-93-194, University of Tennessee, Oct 1993

[6] F.Jelinek, "Self Organized Language Modeling for Speech Recognition", in Readings in Speech Recognition, Morgan Kuafmann,1989.