USE OF NON-NEGATIVE MATRIX FACTORIZATION FOR LANGUAGE MODEL ADAPTATION IN A LECTURE TRANSCRIPTION TASK

Miroslav Novak

IBM T.J. Watson Research Center
P. O. Box 218, Yorktown Heights, NY 10598, USA
miroslav@us.ibm.com

Richard Mammone

Dept. of ECE, Rutgers University
Piscataway, NJ 08834
mammone@caip.rutgers.edu

ABSTRACT

This paper introduces the Non-negative Matrix Factorization for Language Model adaptation. This approach is an alternative to Latent Semantic Analysis based Language Modeling using Singular Value Decomposition (SVD) with several benefits. A new method, which does not require an explicit document segmentation of the training corpus is presented as well. This method resulted in a perplexity reduction of 16% on a database of biology lecture transcriptions.

1. INTRODUCTION

In language modeling one has to model the probability of occurrence of a predicted word given its history $P(w_n|H)$. N-gram based Language Models have been used successfully in Large Vocabulary Automatic Speech Recognition Systems. In this model, the word history consists of the $N-1$ immediately preceding words. Particularly, tri-gram language models ($P(w_n|w_{n-1},w_{n-2})$) offer a good compromise between modeling power and complexity. A major weakness of these models is the inability to model word dependencies beyond the span of the n-grams. As such, n-gram models have limited semantic modeling ability. Alternate models have been proposed with the aim of incorporating long term dependencies into the modeling process. Methods such as word trigger models [4], high-order n-grams, cache models and etc. have been used in combination with the standard n-gram models [5].

One such method, a Latent Semantic Analysis based model has been proposed [2]. A word-document occurrence matrix $A_{N\times K}$ is formed (N = size of the vocabulary, K = number of documents), using a training corpus explicitly segmented into a collection of documents. A Singular Value Decomposition $A = USV^T$ is performed to obtain a low dimensional linear space $\mathcal{S}$, which is more convenient to perform tasks such as word and document clustering, using an appropriate metric. This framework can be used to construct a Language Model, and the metric can be used to measure the closeness of predicted words to the words of history H, expressed as a vector d_H (such vector would be a column of the matrix A, if it were included in the training corpus). Methods for construction of the actual conditional distributions is also presented [2].

This process can be alternatively viewed as the projection of a history vector onto the vector space $\mathcal{S}$:

$$d'_H = UU^T d_H \tag{1}$$

This projected history vector reflects a word distribution consistent with the training corpus and thus can be used for near future predictions. The linear projection operator UU^T involves a rotation in R^N, thus it is possible that some elements of the projected vector will have a negative values. Since these represent word counts, such negative values are difficult to interpret.

An objection has been raised to the use of SVD in LSA in [1]. Its property as the best approximation for a given rank is related to the assumption of normality of the data samples. Clearly, the normality assumption is not valid for word counts, Poisson or other non-negative distributions [1] have been suggested as more appropriate alternatives.

2. NON-NEGATIVE FACTORIZATION

As an alternative to SVD, use of non-negative matrix factorization is proposed. This method has been suggested for use in several fields, including information retrieval systems [3] or statistical translation [7]. It has a form of:

$$A_{N\times K} \approx W_{N\times r} H_{r\times K} \qquad \begin{array}{l} r < min(N,K) \\ A_{i,j} \geq 0 \\ W_{i,j} \geq 0 \\ H_{i,j} \geq 0 \end{array} \tag{2}$$

The dimension r is typically much lower then either dimension of the matrix A. A numerical method has been presented to find a solution for a given dimension r. The update formulas satisfy the non-negative constraint and can be derived using an assumption that the word counts follow Poisson distributions [3]:

$$\begin{aligned} W_{ia} &= W_{ia} \sum_m \frac{A_{im}}{(WH)_{im}} H_{am} \\ W_{ia} &= \frac{W_{ia}}{\sum_j W_{ja}} \\ H_{am} &= H_{am} \sum_i W_{ia} \frac{A_{im}}{(WH)_{im}}. \end{aligned} \tag{3}$$

Iteration of these update rules converge to a local maximum of the objective function:

$$F = \sum_i \sum_j [A_{ij} log(WH)_{ij} - (WH)_{ij}]. \tag{4}$$

This objective function can be derived from a maximum likelihood formulation of the problem - finding a parametric model P(x,$(WH)_{ij}$) which maximizes the likelihood of observing A_{ij}.

In addition to the non-negativity, another property of this factorization is that the columns of W tend to represent clusters of locally related elements; in our case they represent groups of associated words. This is in contrast with the SVD approach, where columns of U are orthogonal. This property suggests that the columns of W can be interpreted as conditional word probability distributions, since they satisfy the conditions of a probability distribution by the definition. Thus the matrix W describes a hidden document space $\mathcal{D} = \{d_j\}$ by providing conditional distributions $W = \mathbf{P}(w_i|d_j)$. The task is to find a matrix W, given the word-document count matrix A. The second term of the factorization, matrix H, reflects the properties of the explicit segmentation of the training corpus into individual documents. This information is not of interest in the context of Language Modeling. In fact, in the SVD case, we could use factorization of the following form:

$$AA^T = US^2U^T, \tag{5}$$

which leads to the identical values as far as the U matrix is concerned. The matrix AA^T is basically a word-word co-occurrence matrix and does not explicitly shows the document segmentation. A similar concept could be used in NMF approach. We shall show an alternative way to construct a matrix with same properties as AA^T, so that an explicit document segmentation is not needed.

Let us consider a matrix M, elements of which are :

$$\begin{aligned} m_{ij} &= E[C_iC_j] \\ &= \sum_u \sum_v uvP(C_i = u, C_j = v). \end{aligned} \tag{6}$$

The count C_i is the occurrence count of word w_i in a document instance and its expected value is considered across the whole document collection. Let us assume that the joint probability $P(C_i = u, C_j = v)$ can be modeled by a mixture of conditional distributions. Let us further assume that within each mixture component, the words occur independently of each other. This assumption is reasonable for relatively short documents. Then we can write:

$$\begin{aligned} E[C_iC_j] &\approx \sum_u \sum_v uv \sum_l P(u, v|d_l)P(d_l) \\ &\approx \sum_u \sum_v uv \sum_l P(u|d_l)P(v|d_l)P(d_l) \\ &= \sum_l \sum_u uP(u|d_l) \sum_v vP(v|d_l)P(d_l) \\ &= \sum_l E[C_i|d_l]E[C_j|d_l]P(d_l). \end{aligned} \tag{7}$$

By denoting $q_{il} = E[C_i|d_l]$, we can rewrite this result in a matrix form:

$$M = QPQ^T, \tag{8}$$

where P is a diagonal matrix, $p_{ll} = P(d_l)$.

When normalized column-wise, the matrix Q can be considered as an estimate of the conditional word distributions:

$$\begin{aligned} Q_n &= QK^{-1} = \hat{\mathbf{P}}(w_i|d_j) \\ q_{i,j} &= \frac{E[C_i|d_j]}{\sum_k E[C_k|d_j]} \end{aligned} \tag{9}$$

so we can rewrite (8) as:

$$M = Q_nKPK^TQ_n^T, \tag{10}$$

where both K and P are diagonal matrices.

This result could be directly related to the NMF algorithm as:

$$W = Q_n \qquad H = K^2PQ_n^T = ZQ_n^T, \tag{11}$$

or it is possible to modify the update formulas to reflect the symmetry of (8):

$$\begin{aligned} Q_{ia} &= Q_{ia}z_a \sum_m \frac{A_{im}}{(QZQ^T)_{im}} Q_{ma} \\ z_a &= \sum_j Q_{ja} \\ Q_{ia} &= \frac{Q_{ia}}{z_a}, \end{aligned} \tag{12}$$

where z_a is an element of the diagonal matrix Z. These new update formulas lead to somewhat more efficient implementation, particularly in terms of memory requirements.

In the experiments presented further, a rectangular sliding window was used to estimate the matrix M. Details of this method will be presented in section 4.

3. LANGUAGE MODEL CONSTRUCTION

Given a set of conditional distributions $\mathbf{P}(w_i|d_j)$, word probability distributions conditioned on observed history $P(w|H)$ can be constructed in the form of a weighted mixture, where the weights depend on the history.

The probability can be expressed as:

$$P(w_n|H) = \sum_j P(w_n|d_j)P(d_j|H) \tag{13}$$

The assumption made here is that the probability of the predicted word does not directly depend on the history,
$P(w_n|d_j, H) \approx P(w_n|d_j)$, which is consistent with the earlier assumption that within one document d_j words are generated independently.

The first factor of (13) is already available. To determine the second factor, one choice would be to compute:

$$\begin{aligned} P(H|d_j) &= \prod_i P(w_{Hi}|d_j), \\ P(d_j|H) &= \frac{P(H|d_j)}{\sum_l P(H|d_l)}, \end{aligned} \tag{14}$$

where w_{Hi} is one word of the history H. This approach is not practical, because it assumes that all words in the history are being generated by a single document. If the probability $P(w_{Hi}|d_j)$ is zero for any word in the history, the resulting probability $P(d_j|H)$ becomes zero as well.

An alternative approach assumes that the history can be generated by any of the hidden documents, independently one word at a time. We can then express the contribution of a particular word in the history to the conditional weight of each hidden document:

$$\mu_{ij} = \frac{P(w_{Hi}|d_j)P(d_j)}{\sum_l P(w_{Hi}|d_l)P(d_l)}. \tag{15}$$

Then we can find the probability $P(d_j|H)$ by normalizing these weights:

$$P(d_j|H) = \frac{\sum_i \mu_{ij}}{\sum_l \sum_i \mu_{il}} = \frac{\sum_i \mu_{ij}}{|H|} \tag{16}$$

where $|H|$ is size of the history.

4. EXPERIMENT DESCRIPTION

Experiments were conducted in the context of the Aristotle project [6] developed at CAIP. In this project, we used the IBM ViaVoice speech recognition system for automated transcription of recorded lectures. The recognizer's vocabulary was extended to cover the specific subject (Biology 101), but the performance was not satisfactory. Further improvement was achieved by speaker and language model adaptation. We built a topic language model for biology, using the course textbook as a training corpus (total 600k words). The final error rate was around 20%. Analysis of the errors led us to the conclusion that further improvement could be achieved if a language model with semantic modeling capability was used. The nature of the speech used in the lecture presentations is more spontaneous than in read speech (looser syntax), at the same time it exhibits distinct content word patterns in contexts beyond the reach of trigrams.

In the first case (explicit segmentation), we have applied the NMF technique on a manually segmented training corpus, obtained from the biology course textbook. Since the chapter and section boundaries were clearly marked, we have used them as natural document boundaries. The whole textbook was segmented into 1500 documents. A manually created stop list of 70 word was applied to filter out function words. In the second case (implicit segmentation), we have used a rectangular window of 21 words, moving through the training corpus one word at a time. For each instance, we updated the occurrence counts of word pairs consisting of the word in the center of the window together with all the words included within the window. In the former case we used the original update formulas (3), in the later case we used the symmetric version (12).

Performance of the model was evaluated by measuring the perplexity on a test set, consisting of the transcriptions of several lectures. Since the described model does not incorporate any syntactical rules, the perplexity can be expected to be quite high (of a same order as of a unigram model). In addition, it does not model most of the function words (which were removed by the stop list), so a direct perplexity measurement would be very unfavorable. For this reasons, we have measured perplexity on a combined model:

$$P(w|H) = \lambda P_{ngram}(w|H) + (1-\lambda) P_{NMF}(w|H). \tag{17}$$

The following table shows perplexity improvement on both training and test data sets.

Perplexity	n-gram	explicit	implicit
training data	127.8	107.8	103.8
test data	288.8	249.8	240.6

A comparison is made here between the baseline system (trigram model, adapted to the biology topic) and the two methods described earlier: using explicit document segmentation and the implicit method (rectangular window count collection). For both models, there was an improvement in perplexity. It can also be seen that there is a substantial mismatch between the training corpus and the transcribed lectures. We attribute the higher test set perplexity to the more spontaneous nature of the lectures.

For both cases, the dimension of factorization was chosen to be 500. We believe that the choice of this dimension is not very critical. As opposed to the SVD method, an increment of dimensionality does not simply add new columns to the Q matrix, but affects all columns. As can be seen in figure (1), the perplexity monotonically decreases with increasing dimension. A logarithmic dependence $Perp \approx a \log(r)$ can be seen. The history includes $|H|$ immediate predecessors to the predicted word with equal weight. The effect of the history size $|H|$ on the perplexity can be seen in figure (4). The best result was achieved with a history size slightly larger than the size of the window used to collect the co-occurrence counts during the training.

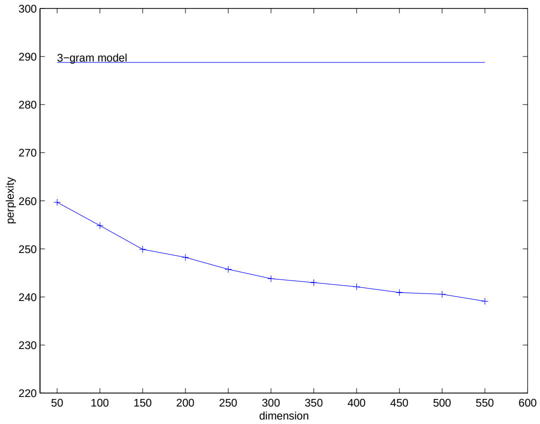

Fig. 1. Perplexity versus factor dimension

Table 1 shows three columns of the matrix Q, the top words with highest conditional probability.

male	oxygen	common
female	carbon	evolutionary
males	dioxide	ancestor
females	respiration	characters
chromosome	cells	more
x	hemoglobin	derived
sex	aerobic	shared
mating	blood	ancestral
mate	concentration	group
many	alcohol	example
species	fermentation	character

Table 1. Examples of the hidden document distributions

The word distributions tend to be very sparse, as can be seen in figure (2). In this figure, the distributions corresponding to several columns of the Q matrix are shown, sorted by the probability values. As can be seen, most of the probability mass is assigned to few hundred words with a sharp decline afterwards.

We have observed that with increasing dimensionality, similar clusters of words tend to reappear. So a dimensionality increase

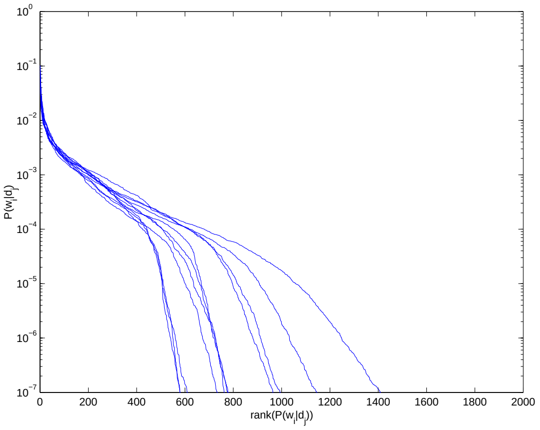

Fig. 2. Examples of Conditional word distributions

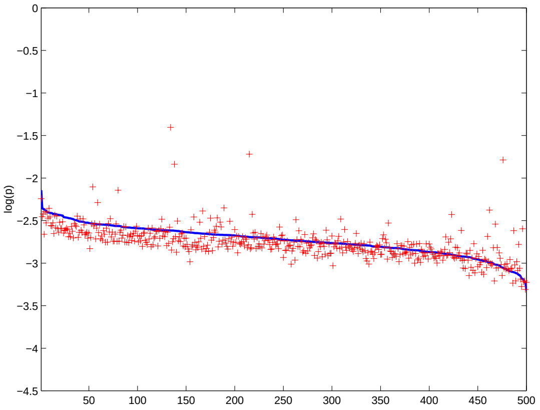

Fig. 3. Hidden document priors $P(d_j)$.

has a refining effect with more-or-less equal weight on all distributions. This can be observed in figure (3) , which shows the document priors $P(d_j)$ (solid line). In the same figure, we also show the direct estimates of the priors from the training data in a form of:

$$\hat{P}(d_j) = \sum_{h \epsilon H} P(d_j|h)P(h), \tag{18}$$

where we consider all history samples equally likely and $P(d_j|h)$ is obtained using (16). It can be seen that these estimates are fairly close to the values of $P(d_j)$ obtained in the factorization process.

5. SUMMARY

We have presented use of Non-negative matrix factorization for Language Model adaptation based on Latent Semantic Analysis framework. A novel approach, which does not require an explicit document segmentation of the training corpus is presented. Based on the obtained perplexity improvements, this method produces results comparable to the method which requires explicit segmentation of the training corpus into documents.

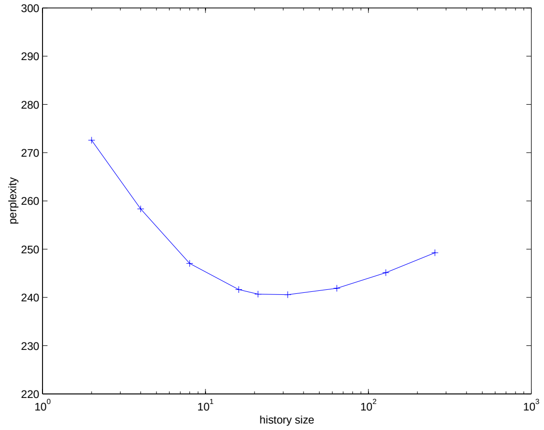

Fig. 4. Perplexity versus history size

6. REFERENCES

[1] C. D. Manning, H. Schuetze, ”Foundations of Statistical Natural Language Processing”, MIT Press , 1999.

[2] J.R. Bellegarda, “A Multispan Language Modelling Framework for Large Vocabulary Speech Recognition”, IEEE Transactions on Speech and Audio Processing, September 1998, vol. 6, num. 5, pp 456 - 467.

[3] D.D. Lee, H. S. Seung, Learning the parts of objects by non-negative matrix factorization, Nature , October 1999, vol. 401, pp 1451 - 1454.

[4] R. Lau, R. Rosenfeld, S. Roukos, “Trigger-based Language Models Using Maximum Likelihood Estimation of Exponetial Distributions”, Proceedings ICASSP 93, Mineapolis, April 93

[5] J. T. Goodman, “Putting It All Together: Language Model Combination”, Proceedings ICASSP 2000, Istambul, May 2000

[6] G. Faulkner, S. Gopal, A. Ittycheriah, R. Mammone, A. Medl, M. Novak, “The Aristotle Project: A Distributed Learning System, Proceedings of Ed-Media2000”, World Conference on Educational Multimedia, Hypermedia, and Telecommunications, pp. 292-297, Montreal, June 2000

[7] J. S. McCarley, “Statistical Machine Translation as Non-negative Matrix Factorization”, IBM Technical Report, to be published, 2001.