Optimization of Latent Semantic Analysis based Language Model Interpolation for Meeting Recognition

Michael Pucher[†] [‡], **Yan Huang**[*], **Özgür Çetin**[*]

[‡]Telecommunications Research Center
Vienna, Austria
pucher@ftw.at

[†]Speech and Signal Processsing Lab, TU Graz
Graz, Austria

[*]International Computer Science Institute
Berkeley, USA
yan@icsi.berkeley.edu, ocetin@icsi.berkeley.edu

Abstract

Latent Semantic Analysis (LSA) defines a semantic similarity space using a training corpus. This semantic similarity can be used for dealing with long distance dependencies, which are an inherent problem for traditional word-based n-gram models. This paper presents an analysis of interpolated LSA models that are applied to meeting recognition. For this task it is necessary to combine meeting and background models. Here we show the optimization of LSA model parameters necessary for the interpolation of multiple LSA models. The comparison of LSA and cache-based models shows furthermore that the former contain more semantic information than is contained in the repetition of words forms.

Optimizacija latentne semantične analize temelječe na interpolaciji jezikovnega modela za namene razpoznavanja sestankov

Latentna semantična analiza (LSA) definira prostor semantične podobnosti z uporabo učnega korpusa. To semantično podobnost je mogoče uporabiti pri odvisnostih dolgega dosega, ki so inherenten problem za tradicionalne, na besedah temelječe n-gramske modele. Prispevek predstavlja analizo interpoliranih modelov LSA, ki so uporabljeni za razpoznavanje sestankov. Za to nalogo je potrebno združiti modela sestankov in ozadja. Predstavljena je optimizacija parametrov modela LSA za interpolacijo med večimi modeli LSA. Primerjava modelov LSA in modelov s predpomnilnikom pokaže tudi, da prvi vsebujejo več semantičnih informacij kot ponavljanje besednih oblik.

1. Introduction

Word-based n-gram models are a popular and fairly sucessful paradigm in language modeling. With these models it is however difficult to model long distance dependencies which are present in natural language (Chelba and Jelinek, 1998).

LSA maps a corpus of documents onto a semantic vector space. Long distance dependencies are modeled by representing the context or history of a word and the word itself as a vector in this space. The similarity between these two vectors is used to predict a word given a context. Since LSA models the context as a bag of words it has to be combined with n-gram models to include word-order statistics of the short span history. Language models that combine word-based n-gram models with LSA models have been successfully applied to conversational speech recognition and to the Wall Street Journal recognition task (Bellegarda, 2000b)(Deng and Khudanpur, 2003).

We conjecture that LSA-based language models can also help to improve speech recognition of recorded meetings, because meetings have clear topics and LSA models adapt dynamically to topics. Due to the sparseness of available data for language modeling for meetings it is important to combine meeting LSA models that are trained on relatively small corpora with background LSA models which are trained on larger corpora.

LSA-based language models have several parameters influencing the length of the history or the similarity function that need to be optimized. The interpolation of multiple LSA models leads to additional paramters that regulate the impact of different models on a word and model basis.

2. LSA-based Language Models

2.1. Constructing the Semantic Space

In LSA first the training corpus is encoded as a word–document co-ocurrence matrix W (using weighted term frequency). This matrix has high dimension and is highly sparse. Let $\mathcal{V}$ be the vocabulary with $|\mathcal{V}| = M$ and $\mathcal{T}$ be a text corpus containing n documents. Let c_{ij} be the number of occurrences of word i in document j, c_i the number of occurrences of word i in the whole corpus, i.e. $c_i = \sum_{j=1}^{N} c_{ij}$, and c_j the number of words in document j. The elements of W are given by

$$[W]_{ij} = (1 - \epsilon_{w_i}) \frac{c_{ij}}{c_j} \tag{1}$$

where ϵ_{w_i} is defined as

$$\epsilon_{w_i} = -\frac{1}{\log N} \sum_{j=1}^{N} \frac{c_{ij}}{c_i} \log \frac{c_{ij}}{c_i} \ . \quad (2)$$

ϵ_w will be used as a short-hand for ϵ_{w_i}. Informative words will have a low value of ϵ_w. Then a semantic space with much lower dimension is constructed using Singular Value Decomposition (SVD) (Deerwester et al., 1990).

$$W \approx \hat{W} = U \times S \times V^T \quad (3)$$

For some order $r \ll \min(m, n)$, U is a $m \times r$ left singular matrix, S is a $r \times r$ diagonal matrix that contains r singular values, and V is a $n \times r$ right singular matrix. The vector $u_i S$ represents word w_i, and $v_j S$ represents document d_j.

2.2. LSA Probability

In this semantic space the cosine similarity between words and documents is defined as

$$K_{\text{sim}}(w_i, d_j) \triangleq \frac{u_i S v_j^T}{||u_i S^{\frac{1}{2}}|| \cdot ||v_j S^{\frac{1}{2}}||} \ . \quad (4)$$

Since we need a probability for the integration with the n-gram models, the similarity is converted into a probability by normalizing it. According to (Coccaro and Jurafsky, 1998), we extend the small dynamic range of the similarity function by introducing a temperature parameter γ.

We also have to define the concept of a pseudo-document $\tilde{\mathrm{d}}_{t-1}$ using the word vectors of all words preceding w_t, i.e. $w_1, \ldots, w_{t-1}$. This is needed because the model is used to compare words with documents that have not been seen so far. In the construction of the pseudo-document we also include a decay parameter $\delta < 1$ that is multiplied with the preceding pseudo-document vector and renders words closer in the history more significant.

The conditional probability of a word w_t given a pseudo-document $\tilde{\mathrm{d}}_{t-1}$ is defined as

$$P_{\text{LSA}}(w_t|\tilde{\mathrm{d}}_{t-1}) \triangleq \frac{[K_{\text{sim}}(w_t, \tilde{\mathrm{d}}_{t-1}) - K_{\min}(\tilde{\mathrm{d}}_{t-1})]^\gamma}{\sum_w [K_{\text{sim}}(w, \tilde{\mathrm{d}}_{t-1}) - K_{\min}(\tilde{\mathrm{d}}_{t-1})]^\gamma} \quad (5)$$

where $K_{\min}(\tilde{\mathrm{d}}_{t-1}) = \min_w K(w, \tilde{\mathrm{d}}_{t-1})$ to make the resulting similarities nonnegative (Deng and Khudanpur, 2003).

2.3. Combining LSA and n-gram Models

For the interpolation of the word based n-gram models and the LSA models we used the methods defined in Table 1. λ is a fixed constant interpolation weight, and $\propto$ denotes that the result is normalized by the sum over the whole vocabulary. λ_w is a word-dependent parameter defined as

$$\lambda_w \triangleq \frac{1 - \epsilon_w}{2} \ . \quad (6)$$

This definition ensures that the n-gram model gets at least half of the weight. λ_w is higher for more informative words.

We used two different methods for the interpolation of n-gram models and LSA models. The *information weighted geometric mean* and simple *linear interpolation*.

Model	Definition
n-gram (baseline)	$P_{n-\text{gram}}$
Linear interpolation (LIN)	$\lambda P_{\text{LSA}} + (1-\lambda) P_{n-\text{gram}}$
Information weighted geometric mean interpolation (INFG)	$\propto P_{\text{LSA}}^{\lambda_w} P_{n-\text{gram}}^{1-\lambda_w}$

Table 1: Interpolation methods.

The *information weighted geometric mean* interpolation represents a loglinear interpolation of normalized LSA probabilities and the standard n-gram.

2.4. Combining LSA Models

For the combination of multiple LSA models we tried two different approaches. The first approach was the linear interpolation of LSA models with optimized λ_i where $\lambda_{n+1} = 1 - (\lambda_1 + \ldots + \lambda_n)$:

$$P_{\text{lin}} \triangleq \lambda_1 P_{\text{Lsa}_1} + \ldots + \lambda_n P_{\text{Lsa}_n} + \lambda_{n+1} P_{n-\text{gram}} \quad (7)$$

Our second approach was the INFG Interpolation with optimized θ_i where $\lambda_w^{(n+1)} = 1 - (\lambda_w^{(1)} + \ldots + \lambda_w^{(n)})$:

$$P_{\text{infg}} \propto P_{\text{Lsa}_1}^{\lambda_w^{(1)}\theta_1} \ldots P_{\text{Lsa}_n}^{\lambda_w^{(n)}\theta_n} P_{n-\text{gram}}^{\lambda_w^{(n+1)}\theta_{n+1}} \quad (8)$$

The parameter θ_i have to be optimized since the $\lambda_w^{(k)}$ depend on the corpus, so that a certain corpus can get a higher weight because of a content-word-like distribution of w, although the whole data does not well fit the meeting domain. In general we saw that the λ_w values were higher for the background domain models than for the meeting models. But taking the n-gram mixtures as an example the meeting models should get a higher weight than the background models. For this reason the λ_w of the background models have to be lowered using θ.

To ensure that the n-gram model gets a certain part α of the distribution, we define $\lambda_w^{(k)}$ for word w and LSA model Lsa_k as

$$\lambda_w^{(k)} \triangleq \frac{1 - \epsilon_w^{(k)}}{\frac{n}{1-\alpha}} \quad (9)$$

where $\epsilon_w^{(k)}$ is the uninformativeness of word w in LSA model Lsa_k as defined in (2) and n is the number of LSA models. This is a generalization of definition (6). Through the generalization it is also possible to train α, the mimimum weight of the n-gram model.

For the INFG interpolation we had to optimize the model parameters θ_i, the part of the n-gram model α, and the γ exponent for each LSA model.

3. Analysis of the models

To gain a deeper understanding of our models we analyzed the effects of the model parameters and compared our models with other similar models. For this analysis we used meeting heldout data, containing four ICSI, four CMU and four NIST meetings. The perplexities and similarities were estimated using LSA and 4-gram models trained on the Fisher conversational speech data (Bulyko et al., 2003)

and the meeting data (Table 2) minus the meeting heldout data. The models were interpolated using the INFG interpolation method (Table 1).

Training Source	# of words ($\times 10^3$)
Fisher	23357
Meeting	880

Table 2: Training data sources.

3.1. Perplexity Space of Combined LSA Models

Figure 1 shows the perplexities for the meeting and the Fisher LSA model, that were interpolated with an n-gram model using linear interpolation (Definition 7) were λ_1 and λ_2 are the corresponding LSA model weights. Zeros are plotted where the interpolation is not defined, e.g. where $\lambda_1 + \lambda_2 \geq 1$, which would mean that the n-gram model gets zero weight.

This figure shows that the minimum perplexity is reached with $\lambda_1 = \lambda_2 = 0$. Furthermore we can see that the graph gets very steep with higher values of λ. This is beneficial for the gradient descent optimization since we always know where to go to reach the minimum perplexity. The minimum perplexity is however reached when we do not use the LSA model and solely rely on the n-gram.

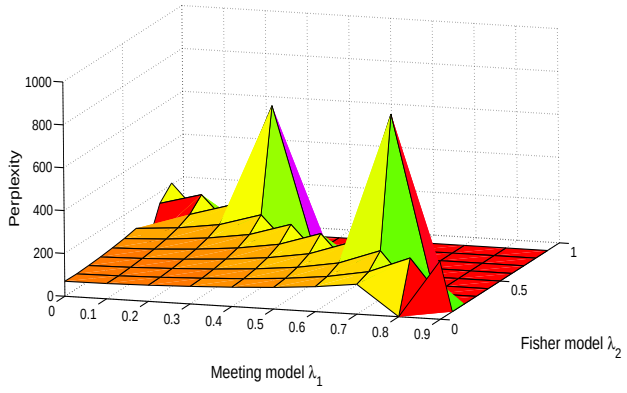

Figure 1: Perplexity space for 2 linearly interpolated LSA models.

Figure 2 shows the perplexity space of the INFG interpolation (Definition 8) for the meeting and the Fisher model that is much flatter than the linear interpolation space. We can estimate the difference in steepness by looking at the perplexity scale, which is $[67, 72]$ for the INFG interpolation compared to $[0, 1000]$ for the linearly interpolated models. Therefore the parameter optimization is harder and slower for this interpolation.

On the other hand we can achieve an improvement over the n-gram model when using this interpolation. The optimum perplexity is not reached when giving both LSA models $\theta_i = 0$, but when setting the parameter for the Fisher model to $\theta_2 = 0$ and the meeting model parameter to $\theta_1 = 1$. The θ_i's have only the function of boolean model selectors in this 2-model case. But there is still the word entropy that is varying the interpolation weight between LSA and n-gram model.

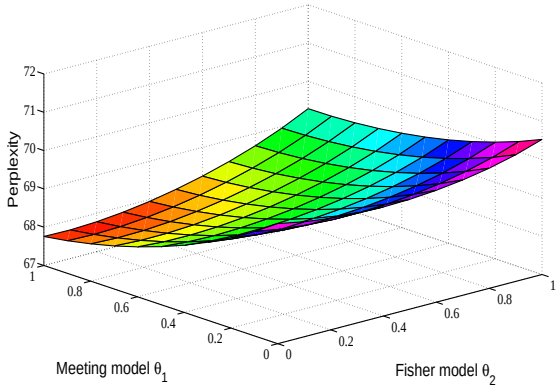

Figure 2: Perplexity space for 2 INFG interpolated LSA models.

When we conducted word-error-rate experiments with combinations of more than two LSA models (Pucher et al., 2006) we used gradient-descent optimization to optimize all the interpolation parameters together. Here we used a brute-force approach to get a picture of the whole perplexity space.

3.2. The Repetition Effect: LSA Models and Cache Models

Some improvements of LSA-based language models over n-gram models are surely due to the redundant nature of language and speech. A lot of words that pop-up in a meeting for example are likely to pop-up again in a short window of context. A word will be highly similar to a context when the word appears in the context. A cache-based language model can exploit this fact by keeping a cache of words that already have been seen, and giving them higher probability (Kuhn and De Mori, 1990). To test if the performance of LSA-based models only rests on this cache-effect we checked the word probabilities of the models.

	+ Meet	+ Fish	− Meet	− Fish
Word in hist.	60%	63%	5%	6%
Word out of hist.	8%	7%	27%	24%
	68%	70%	32%	30%

Table 3: Number of improved LSA word probabilities.

Table 3 shows the number of improved word probabilities for the meeting and the Fisher model on the heldout data. '+' means that the probability of the LSA model was higher than the n-gram model probability, '−' means that it was lower. The end-of-sentence event is not included.

For the meeting model 60% of the improvements are due to the cache-effect where the word appears in the history. This value is so high because we use the decay parameter, so that a word disappears from the pseudo-document, but it still stays in our cache for the whole meeting and increases the cache-effect. So a certain amount of this improvement is actually due to the semantic of the LSA model. This happens because the word vector is decayed

in the pseudo-document but the word stays in the cache for the whole meeting. The percentage of the class +/Word not in hist. has to be increased by this amount.

We can estimate this amount by assuming that each meeting contains ≈ 7500 ($90455/12$ meetings) words, and that the last 100 words are present in the pseudo-document. We know that 60% of the words fall under the category +/Word in hist. (≈ 4500 words). But this is only true if we assume the history to be the whole preceding meeting and not just the last 100 words. The mean length of the history for a document of length k is given by the arithmetic mean $\frac{0+1+2+...+k-1}{k} = \frac{k+1}{2}$.

In our case the mean length of the history is ≈ 3700. So we know that given a mean length of around 3700, 60% of the words fall under the former class, but given a mean length of the history around 100, some improvement also falls into the class +/Word not in hist, which must therefore be significantly higher than 7%. The same reasoning applies to the Fisher model where the performance is even better.

According two t-tests for paired samples the differences between LSA and n-gram models for the following classes are significant: +/Word in hist, −/Word in hist., −/Word not in hist. for the meeting and the Fisher model ($p < 0.05$). The difference within the class +/Word not in hist. is however not significant, but as already mentioned the true size of this class is bigger than the estimated size.

This analysis shows that LSA-based models cannot be simply replaced by cache-based models. Although the repetition effect is important for LSA models they also cover other semantic information.

3.3. The Temperature Effect: γ Exponent Optimization

The temperature parameter γ (Definition 5) is used to extend the small dynamic range of the LSA similarity (Coccaro and Jurafsky, 1998). Here we want to optimize this parameter and show how it changes the LSA similarities.

The similarities were scaled by using the minimum similarity given the history as in Definition 5. Otherwise the exponent would make negative similarities positive.

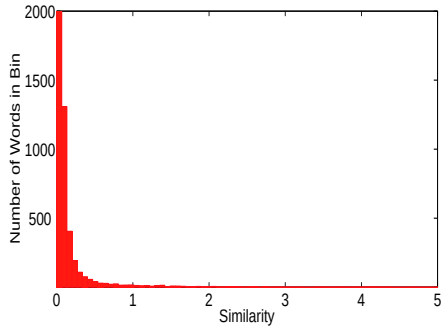

Figure 3: Similarities for $\gamma = 8$.

Figure 3 shows the similarity distribution for a γ value of 8 for the Fisher model on the heldout data. This distribution expanded the similarity range and assembles a lot of similarities around zero.

In Figure 3 all similarities < 1.0 get pruned in comparison to $\gamma = 1$. This is due to the nature of the exponentiation where all values between in $[0, 1]$ get smaller if exponentiated. To change this one can add an offset $\beta \in [0, 1]$ to the similarities to avoid pruning of similarities in the interval $[1 - \beta, 1]$. For $\beta = 1$ there is no pruning since all similarities are bigger than or equal to 1. Than the similarity distribution gets flatter. We also optimized β to find the effect of values that are smaller than 1.

For our work it is interesting to see which γ values optimize the perplexity on the heldout data. Figure 4 shows perplexities of the Fisher model on the heldout data for different values of γ and β.

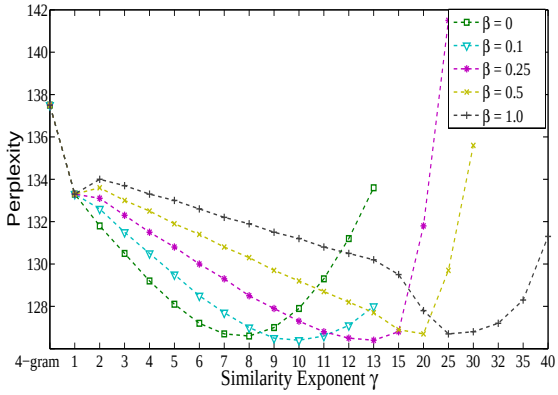

Figure 4: Perplexities for the Fisher LSA model with different γ and β values.

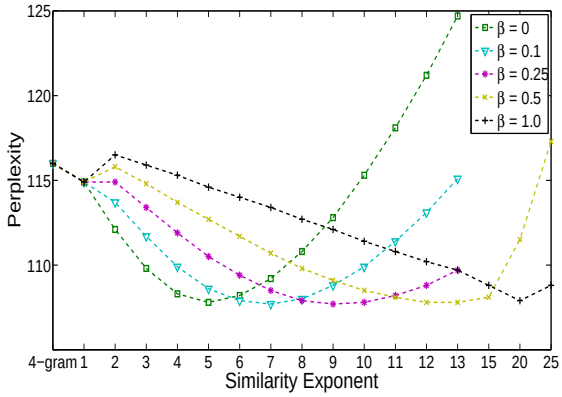

Figure 5: Perplexities for the meeting LSA model with different γ and β values.

One can see that the lowest perplexity for all β values is nearly the same while only the exponent is shifting. It can also be seen that all LSA models outperform the 4-gram model even for $\gamma = 1$. The optimal γ value for the meetings is for all β smaller than for the Fisher model (Figure 5). One generalization we can make from experiments with other models is that the optimal γ value is in general higher for bigger models, e.g. models that are trained on larger corpora. This is also reflected in the relation between the meeting model and the Fisher model which can be seen from figure 5 and 4.

With the first approach one comes up with a much smaller exponent than with the second. We conjecture that the different values of exponents found in the literature ranging from 7 (Coccaro and Jurafsky, 1998) to 20 (Deng and Khudanpur, 2003) are due to the usage of different values of β. Since we do not see a difference in perplexity we conclude that it does not matter which approach one chooses.

The temperature parameter was optimized independently from the interpolation parameters. We found that this value is stable over different test data sets.

3.4. The History Effect: δ Decay Optimization

Here we show how the decay parameter δ influences the perplexity. The perplexity of the 4-gram Fisher and meeting models are again our baselines. As a test set we use again the meeting heldout data. The idea of the decay parameter is to update the pseudo-document in a way that words that were recently seen get a higher weight than words that are in a more distant history. Finally the words that are far away from the actual word are forgotten and have no more influence on the prediction of the actual word. (Bellegarda, 2000a) finds a value around 0.98 to be optimal for the decay parameter δ.

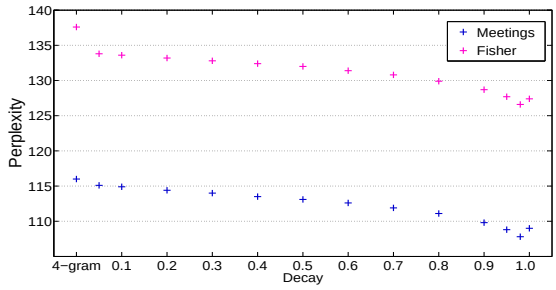

Figure 6: Perplexities for a 4-gram and LSA models with different decays δ.

Figure 6 shows that there is a constant drop in perplexity as we increase the length of the history, e.g. the value of δ. There is however not much difference for the decay value 0.98 and 1.0, which means that no words are forgotten. But even the shortest history with a decay of 0.05 has a lower perplexity than the 4-gram for the Fisher and the meeting model on the heldout data. We can conclude that it is beneficial for our models not to forget too fast but there is no big difference between forgetting very slow and never forgetting. In our experiments we used nevertheless a decay of 0.98 because it still has the best performance concerning perplexity and because it was also found to be optimal by others (Bellegarda, 2000a).

The decay parameter was optimized independently of the interpolation parameter and the temperature parameter.

4. Conclusion

We showed how to optimize the parameters for interpolated LSA-based language models and saw that simple linear interpolation did not achieve any improvements. With the INFG interpolation we achieved an improvement and a model selection.

The comparison between LSA and cache-based models showed that a large amount of the improvement is due to the repetition of words, but there is also an improvement that relies on other features of the LSA-based models. So cache-based models cannot simply replace LSA-based models.

We also presented the optimization of similarity exponent and offset and saw the relation between the offset selection and the similarity exponent.

The optimization of the decay parameter showed that it makes little difference when being close to or equal to one, but a bigger difference when the value gets close to zero.

We can conclude that the optimzation of parameters is crucial for outperforming word-based n-gram language models by interpolated LSA models.

5. References

J.R. Bellegarda. 2000a. Exploiting latent semantic information in statistical language modeling. *Proceedings of the IEEE*, 88(8):1279–1296, August.

J.R. Bellegarda. 2000b. Large vocabulary speech recognition with multispan statistical language models. *IEEE Transactions on Speech and Audio Processing*, 8(1):76–84, January.

I. Bulyko, M. Ostendorf, and A. Stolcke. 2003. Class-dependent interpolation for estimating language models from multiple text sources. Technical Report UWEETR-2003-0000, University of Washington, EE Department.

C. Chelba and F. Jelinek. 1998. Exploiting syntactic structure for language modeling. In *Proceedings of COLING-ACL*, pages 225–231, San Francisco, California.

N. Coccaro and D. Jurafsky. 1998. Towards better integration of semantic predictors in statistical language modeling. In *Proceedings of ICSLP-98*, volume 6, pages 2403–2406, Sydney.

S. Deerwester, S.T. Dumais, G.W. Furnas, and T.K. Landauer. 1990. Indexing by latent semantic analysis. *Journal of the American Society for Information Scienc*, 41(6):391–407.

Y. Deng and S. Khudanpur. 2003. Latent semantic information in maximum entropy language models for conversational speech recognition. In *Proceedings of HLT-NAACL*, pages 56–63, Edmonton.

R. Kuhn and R. De Mori. 1990. A cache-based natural language model for speech recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(6):570–583.

M. Pucher, Y. Huang, and Ö. Çetin. 2006. Combination of latent semantic analysis based language models for meeting recognition. In *Computational Intelligence 2006*, San Francisco, USA.