interspeech 2008 incorporating SST 08

Bag-Of-Word normalized n-gram models

Abhinav Sethy, Bhuvana Ramabhadran

IBM T. J. Watson Research Center
Yorktown Heights, NY
{asethy,bhuvana}@us.ibm.com

Abstract

The Bag-Of-Word (BOW) model uses a fixed length vector of word counts to represent text. Although the model disregards word sequence information, it has been shown to be successful in capturing long range word-word correlations and topic information. In contrast, n-gram models have been shown to be an effective way to capture short term dependencies by modeling text as a Markovian sequence. In this paper, we propose a probabilistic framework to combine BOW models with n-gram models. In the proposed framework, we normalize the n-gram model to build a model for word sequences *given* the corresponding bag-of-words representation. By combining the two models, the proposed approach allows us to capture the latent topic information as well as local Markovian dependencies in text. Using the proposed model, we were able to achieve a 10% reduction in perplexity and a 2% reduction in WER (relative) over a state-of-the-art baseline for transcribing broadcast news in English.

Index Terms: language modeling, bag-of-words, topic modeling

10.21437/Interspeech.2008-265

1. Introduction

A commonly used representation of text for statistical natural language processing (NLP) is the Bag-Of-Word (BOW) vector. In the BOW representation, text (such as a sentence or a document) is treated as an unordered collection of words by using a fixed length sparse vector containing word occurrence counts. The BOW representation has been shown to be an effective way to capture topic information and long range word-word correlation information [1]. By reducing text to a fixed length vector of discrete counts, the BOW representation makes it possible to extend a range of vector based modeling techniques to NLP tasks such as document classification [2, 3, 4] and information retrieval [5].

In contrast to BOW models, n-gram models focus on the use of local context information by modeling text as a Markovian sequence. N-gram models are extensively used in Automatic Speech Recognition (ASR) and Statistical Machine Translation (SMT) where they serve as priors on word sequences. Integration of word-word correlation and topic information which can be derived from a BOW model, with n-gram models is a promising research area for Language Model (LM) adaptation [6]. In Section 2, we review current research where BOW models have been successfully used for clustering n-gram model training data, smoothing word marginal probabilities for unsupervised LM adaptation and clustering for extracting topic and style relations.

In this paper, we introduce the idea of a BOW normalized n-gram model, which provides a general framework to merge probabilistic BOW models with n-gram models. Intuitively, the BOW model is expected to capture the latent topic information while the n-gram model captures the speaking style and linguistic elements from the training data that it is built on. By combining the two models, we now have a semantic language model that also effectively captures local dependencies. In our proposed approach, the n-gram model is normalized such that the total probability assigned to all the possible word sequences, which share the same BOW vector is equal to the probability mass assigned by the BOW model. In contrast to the aggregate BOW probability which is derived from any BOW model, the conditional probability of a word sequence given the BOW vector is derived from the n-gram model. In the next Section we provide an overview of BOW models with emphasis on probabilistic BOW models and their application towards LM adaptation for Automated Speech Recognition (ASR). In Section 3 we describe the proposed BOW normalized sequence model and methods to efficiently compute its parameters. In Section 4, we present our experiments on a state-of-the-art English broadcast news transcription system. We conclude with remarks on the scope of this work, analysis of experimental results and directions for future work.

2. Overview of BOW models

Latent Semantic Analysis (LSA) [1] is a commonly used paradigm in the information retrieval community which relies on a BOW representation. LSA reveals meaningful associations in the language based on word-document co-occurrences.

The LSA approach makes three claims: that semantic information can be derived from a word-document co-occurrence matrix; that dimensionality reduction is an essential part of this derivation; and that words and documents can be represented as points in Euclidean space. Probabilistic BOW models differ from LSA in the third claim. In probabilistic BOW models the semantic properties of words and documents are expressed in terms of probabilistic topics.

Probabilistic topic models [2, 4, 3] are based upon the idea that documents are mixtures of topics, where a topic is a probability distribution over words. A topic model is a generative model for documents: it specifies a simple probabilistic procedure by which documents can be generated. To make a new document, one chooses a distribution over topics. Then, for each word in that document, one chooses a topic at random according to this distribution, and draws a word from that topic. Standard statistical techniques can be used to invert this process, inferring the set of topics that were responsible for generating a collection of documents.

One of the most popular algorithms for probabilistic BOW modeling of text is Latent Dirichlet Allocation (LDA) [2]. LDA is a generative probabilistic model for collections of discrete data such as text corpora. LDA is a multi-layer hierarchical

Accepted after peer review of full paper

Bayesian model, in which each item of a collection is modeled as a finite mixture over an underlying set of topics. Each topic is, in turn, modeled as an infinite mixture over an underlying set of topic probabilities. The LDA generative model can be described with the following steps:

- For every document d:
  Sample topic distribution $\theta(d)$ from T-dimensional Dirichlet distribution $Dir(\alpha)$
- For every word w in the document
  Sample a topic z from a distribution, $\theta(d)$
  Sample a word from a distribution, $\phi(w|z,\beta)$

θ indicates the relative importance of topics for a document. $\phi(w|z,\beta)$ is a multinomial distribution of word probabilties with a Dirichlet prior conditioned on the topic. $\phi(w|z,\beta)$ indicates the relative importance of particular words in a topic, and α and β are Dirichlet priors which control the smoothing of the topic distribution and the topic-word distribution respectively. Details on inference and estimation of the LDA model can be seen in [2, 3]. Additional information on the inference of LDA model parameters for adjusting word marginal probabilities is given in [7].

LDA and associated probabilistic BOW models have been successfully used in recent research work in the context of unsupervised LM adaptation. In [7, 8] n-gram unigram probabilities are adjusted using the MAP estimate of LDA parameters from the variational lower-bound [2], based on the first pass hypothesis. This approach can be seen as an unsupervised LM adaptation technique where the LDA model smoothens the decoded word distribution mitigating the effect of decoding errors in the hypothesis. Adjusting word marginals without LDA smoothing was shown to result in higher WER. Use of word marginals for unsupervised LM adaptation can also be done in an entropy minimization framework such as [9]. Based on the same unsupervised LM adaptation idea, in [10], a correlated topic model with topic distribution derived from a Dirichlet tree is used in place of LDA to fit a better model on the BOW distribution. In [11], PLSA is used in place of LDA in a similar unsupervised framework. LDA has also been used as the clustering algorithm to partition the LM training data into topics [12, 13]. N-gram models are built separately for each topic and then interpolated with weights optimized either on a fixed heldout set or the first pass hypothesis. Other variants including restricting the BOW vector for clustering to named entities [14], separating words into topic and style clusters [15] and the use of HMM-LDA [16] models have been proposed in the recent past.

In the next Section, we describe our proposed model for merging probabilistic BOW models with N-gram models. In the proposed model, the aggregate probability assigned to all sequences which share the same BOW representation is tied to the probabilistic BOW model while the probability of words occurring in a particular sequence given the BOW vector is assigned from an n-gram model.

3. BOW-normalized n-gram model

In a BOW model, probability of a text sequence T is independent of the order of words. Thus, for any BOW model $P_b(T)$, the probability mass assigned to the sequence T with word counts w_j is related to the aggregate probability mass, $P_b(B(T))$, assigned to its BOW representation $B(T)$ as follows:

$$P_b(B(T)) = \frac{(\sum w_j)!}{\prod w_j!} P_b(T)$$

For an n-gram model $P_n(T)$, the probability assigned to text is dependent on the sequence of words. Let S be the set of all distinct permutes of the words with counts represented by the BOW vector $B(T)$. Then, the aggregate probability mass, $P_n(B(T))$ assigned to the BOW representation is

$$P_n(B(T)) = \sum_S P_n(T)$$

Also, the cardinality $|S|$ of set S is

$$|S| = \frac{(\sum w_j)!}{\prod w_j!}$$

We denote $P_{seq}(T)$ the probability of a sequence given the bag of words. We have

$$P_{seq}(T) = \frac{P_n(T)}{\sum_S P_n(T)}$$

Given $B(T)$, the sequencing model $P_{seq}(T)$ assigns a probability to each of the possible $|S|$ permutes. Thus, while $P(B(T))$ determines the distribution of a collection of words, $P_{seq}(T)$ determines the distribution of the possible sequence assignments. A uniform $P_{seq}(T)$ model, such as a unigram model, will have $|S|$ equally likely sequences and would require $ln|S|$ bits to represent the sequence. A non-uniform, sequence model which strongly prefers a particular sequence over another will require lesser bits to represent the set of possible assignments. The number of bits required for the sequence assignment and the number of bits required for representing $B(T)$ is equal to the perplexity (in bits) of the n-gram model for T.

Combining the sequence model with the BOW model gives us a composite model, $P_c(T)$, which assigns the same probability mass to the BOW representation as the BOW model P_b and the same sequence probability given the bag-of-words as the n-gram model $P_{seq}(T)$.

$$\begin{aligned} P_c(T) = &\quad P_b(B(T)) * P_{seq}(T) \\ = &\quad P_b(T) \frac{(\sum w_j)!}{\prod w_j!} \frac{P_n(T)}{\sum_S P_n(T)} \end{aligned} \tag{1}$$

3.1. Computational complexity

For a text sequence of length K the computation of $P_c(T)$ requires computation of $|S|$ probabilities to compute the sum $\sum_S P_n(T)$. In general, each computation of $P_n(T)$ requires K lookups. However for a n-gram model, the probability of a word is conditioned only on the previous words, which allows us to share computations across subsequences which share the same history. Thus, by generating the permutes in an alphabetically sorted (lexicographic) order [17], we can significantly reduce the number of lookups required. We illustrate this with an example. For example, consider the sequence *a b e d c*. The words in this sequence form the ordered set *(a b c d e)*. The subsequence *e d c* starting from postion 3, contains the words *c,d,e*, sorted in reverse alphabetical order. Thus, to generate the next in-order, lexicographic sequence we need to change the word at postion 2 to the next word in the ordered set *(a,b,c,d,e)* which is *c*. Positions 3,4,5 are then sorted in alphabetical order. This yields the sequence *a c b d e*.

For any given sequence s of these words, let $p(i)$ be the joint probability of the first $0..i$ words in the sequence. For an n-gram model, $p(i) = p(i-1) * P(w_i/w_{i-1..0})$. In Table 1 we show the first 4 permutes in lexicographic order starting with

sequence	recomputes
a b c d e	initialize p(0..4)
a b c e d	recompute p(3,4)
a b d c e	recompute p(2,3,4)
a b d e c	recompute p(3,4)

Table 1: The first 4 permutes in lexicographic order starting with *a b c d e* and indices which need to be recomputed

the sequence *a b c d e* and the indices which need to be recomputed. Without lexicographic ordering, we will need to do 5 lookup operations to compute the probabilities p(0..4). However this computation can be reduced by sharing subsequence probabilities. For example the last entry in Table 1 requires only two lookups as it shares the first 3 words with the previous sequence in the ordered list. As can be seen from the table using lexicographic ordering reduces the number of lookups required.

To further speed up the computation of the permute sum we can use a position based threshold. If a change at index i in the lexicographic permute sequence reduces $p(i)$ below a fraction of the average $p(i)$, we can disregard all permutes which share the sequence $w_{0..i}$ and move to the next lexicographic sequence which has a different word at index i. Let us consider the example sequence *a b e d c* again. The next sequence in lexicographic order as we desribed earlier is *a c b d e*. In this case, if $p(1)$ i.e $p(a, c)$ is low then we can disregard all the $(5 - 2) = 3!$ permutes *a c * * **. The next sequence that needs to be considered is *a d b c e*.

For large blocks of words, even with lexicographic ordering and pruning, the calculation of the permute sum becomes computationally demanding. Some possible ways to compute the sum in such a case include, random walks with a bag-of-word constrained graph, sampling of the permutation space, trellis based computation of all paths. In practice for the case of language modeling for ASR, we have found that chunking text into blocks of uniform size and calculating the permute sum individually for each chunk provides results comparable to the full computation. The size of the blocks serves as a tradeoff between the computation cost(which is approximately m! for a block of size m) and the approximation error. We discuss this tradeoff and our experimental evaluation of the proposed composite BOW-normalized n-gram model in the next section.

4. Experiments

The LVCSR system used for our experiments is based on the 2007 IBM Speech transcription system for GALE Distillation Go/No-go Evaluation. The transcription system operates in four stages: audio diarization, speaker-independent decoding, adaptation, and speaker-adapted decoding. The speaker-independent transcription system was trained on 430 hours of audio from the 1996 English Broadcast News Speech corpus (LDC97S44), the 1997 English Broadcast News Speech corpus (LDC98S71), and the TDT4 Multilingual Broadcast News Speech corpus (LDC2005S11). The speaker-adapted transcription system was trained on 6096 hours of audio that included the three sources listed above as well as the English portion of the EARS BN03 data. For the 1996 and 1997 speech corpora there were companion transcript corpora (LDC97T22 and LDC98T28). To obtain training transcripts for the TDT4 and BN03 audio, we used a lightly supervised training method in which we decoded the audio with a biased language model and used the resulting transcripts as references. The features

Block Size	Perplexity	Computation time(normalized)
6	202	1
8	195	14
10	178	170
12	175	1300
14	173	9800
16	172	15000

Table 2: Computation time and perplexity of the Dev-04 set as function of block size. The computation time shown is normalized by the computation time for a block size of 6

used by the speaker-independent system are PLP features that are mean normalized on a segment-by-segment basis, while the features used by the speaker-adapted system are PLP features that are mean- and variance-normalized on a speaker-by-speaker basis, with normalization statistics accumulated only for frames labeled as speech in the speaker-independent pass. Both the speaker-independent and speaker-adapted systems use LDA+MLLT transforms to project from high-dimensional supervectors of spliced raw features to a 40-dimensional recognition feature space. The LDA projection is computed by accumulating full-covariance statistics for each context-dependent state in a forced alignment of the training data; that is, we use a single full-covariance model for each state. The LDA projection is then computed from these statistics. Both the speaker-independent and speaker-adapted systems have acoustic models with roughly 6000 states and 250000 Gaussians. Both transcription systems are discriminatively trained using fMPE and MPE with backing off to MMI estimates in I-smoothing .For both fMPE and MPE, we use a lattice-based framework for training. Details on various aspects of the system can be found in [18].

We used the 1996 CSR Hub4 Language Model data (LDC98T31) for building the models in our experiments. Kneyser-ney smoothing was used for building the baseline 4-gram language model. The size of the training text was 140M words and the baseline model had 40M n-grams. The recognition vocabulary had 84K word tokens with an average of 1.08 pronunciation variants per word. Where possible, pronunciations were based on PRONLEX. We report Word Error Rate (WER) results on the RT-04 task (40K words) while the perplexity results are on the RT-04 Development set (15K words) . The baseline language model had a perplexity of 204 on the RT-O4 Dev set.

We used Gibbs sampling [3] and LDA to build a 40 topic LDA-based BOW model. The choice of number of topics was based on previous literature [3, 2, 12]. The sequence model (Section 3) was built from the baseline n-gram model. We used blocking (Section 3.1) to efficiently generate the permute sum required to normalize the n-gram model. We experimented with different block sizes to study the tradeoff between perplexity and computational complexity. Table 2 compares the computation time and the perplexity on the RT-04 Development set for different block sizes. For ease of comparison, the computation time in Table 2 is shown as factor of the time required for perplexity computation with a block size of 6. We chose a block size of 10 as a good tradeoff between the perplexity and computational complexity. At a block size of 10, the composite BOW-normalized n-gram model gives a relative perplexity improvement of 12%.

Next, we generated n-best lists (1000 best) by using a static graph decoder with the graph built from the baseline language

model. The n-best lists were then rescored using the composite BOW-normalized n-gram model. We chose the same acoustic model weight for rescoring as the n-best list generation. For each n-best hypothesis a LDA BOW likelihood score was computed and combined with the corresponding sequence model probability. For computing the sequence model probability we used a block size of 10 based on our perplexity results to compute the permute sum. Rescoring with the proposed composite BOW/n-gram model reduced WER from 17.1% to 16.8% on the RT-04 set. This corresponds to a 1.7% relative reduction in WER.

5. Conclusion

5.1. Summary

In this paper we proposed a composite BOW/n-gram model to merge probabilistic BOW models with n-gram models. The composite model combines the BOW model with a sequence model which assigns probability to a word sequence given the BOW vector. We also presented some efficient ways to compute the normalization factor in the sequence model. Using the proposed model, we saw significant gains on an English broadcast news task in both perplexity (12% relative) and WER (1.7% relative) terms.

5.2. Scope of this work

The composite modeling framework presented in this paper can be used in conjunction with a wide range of probabilistic BOW models such as LDA [2] and Author Topic Models [19]. Although the experiments reported in this paper are based on a n-best rescoring framework, we believe it should be possible to extend the model to a decoding framework.

5.3. Directions for future work

BOW and n-gram models capture different kind of dependencies between words. BOW models cover long range word-word correlations by modeling co-occurrences as topic mixtures, whereas n-gram models look at short range sequence dependencies. We believe that the gains that can be achieved by merging the two text modeling techniques depend on the nature of the domain, the language and amount of training material available. By experimenting with the proposed model in other languages and domains other than broadcast news we hope to come up with guidelines on choosing between n-gram models and merged n-gram/BOW models. We also plan to investigate the effectiveness of the proposed approach in other NLP domains such as Statistical Machine Translation which rely on n-gram models.

One of the key elements that needs to be addressed to make decoding efficient with our composite model is the fast computation of the permute sum. We are investigating methods to represent the permute sum in terms of subsequences which can be effectively precomputed.

6. Acknowledgements

The authors would like to thank Stanley Chen for helpful discussions and providing the resources necessary to develop and implement the ideas presented in this paper.

7. References

[1] S. Deerwester, S. Dumais, G. Furnas, T. Landauer, and R.Harshman, "Indexing by latent semantic analysis," *Journal of American Society of Information Sciences*, 1990.

[2] D. Blei, A. Ng, and M. Jordan, "Latent Dirichlet Allocation," *Journal of machine Learning Research*, pp. 993–1022, 2003.

[3] T. L. Griffiths and M. Steyvers, *A road to meaning*. Laurence Erlbaum, 2007, ch. Probabilistic topic models.

[4] T. Hoffman, "Probabilistic latent semantic analysis," in *Proceedings of Uncertainty in Artificial Intelligence*, 1999.

[5] X. Wei and W. B. Croft, "LDA-based document models for ad-hoc retrieval," in *Proceedings of ACM SIGIR*, 2006.

[6] J. Bellegarda, "Statistical language model adaptation: review and perspectives," *Speech Communication*, pp. 93–108, 2004.

[7] Y. C. Tam and T. Schultz, "Unsupervised language model adaptation using latent semantic marginals," in *Proceedings of ICSLP*, 2006.

[8] D. Mrva and P. Woodland, "Unsupervised language model adaptation for mandarin broadcast conversation transcription," in *Proceedings of ICSLP*, 2006.

[9] W. Wang and A. Stolcke, "Integrating map, marginals, and unsupervised language model adaptation," in *Proceedings of Eurospeech*, 2007.

[10] Y. C. Tam and T. Schultz, "Correlated latent semantic model for unsupervised lm adaptation," in *Proceedings of ICASSP*, 2007.

[11] T. Kawahara, Y. Nemoto, and Y. Akita, "Automatic lecture transcription by exploiting presentation slide information for language model adaptation," in *Proceedings of ICASSP*, 2008.

[12] A. Heidel and Lin-Shan Lee, "Robust topic inference for latent semantic language model adaptation," in *Proceedings of ASRU*, 2007.

[13] B. Ramabhadran, O. Siohan, and A. Sethy, "The IBM 2007 speech transcription system for European parliamentary speeches," in *Proceedings of ASRU*, 2007.

[14] Y. Liu and F. Liu, "Unsupervised language model adaptation via topic modeling based on named entity hypotheses," in *Proceedings of ICASSP*, 2008.

[15] B.-J. Hsu and J. Glass, "Style and topic language model adaptation using HMM-LDA," in *Proceedings of EMNLP*, 2006.

[16] T. L. Griffiths, M. Steyvers, D. Blei, and J. B. Tenenbaum, "Integrating topics and syntax," in *NIPS*, 2005.

[17] M.-K. Shen, "Algorithm 202: generation of permutations in lexicographical order," *Communications of ACM*, p. 517, 1963.

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

[19] M. Steyvers, P. Smyth, M. Rosen-Zvi, and T. Griffiths, "Probabilistic author-topic models for information discovery," in *Proceedings of ACM SIGKDD*, 2004.