FEATURE-RICH CONTINUOUS LANGUAGE MODELS FOR SPEECH RECOGNITION

Piotr Mirowski

Courant Institute of Mathematical Sciences
New York University
719 Broadway, 12th Floor
New York, NY 10034, USA

Sumit Chopra, Suhrid Balakrishnan, Srinivas Bangalore

Shannon Laboratories
AT&T Labs Research
180 Park Avenue
Florham Park, NJ 07932, USA

ABSTRACT

State-of-the-art probabilistic models of text such as n-grams require an exponential number of examples as the size of the context grows, a problem that is due to the discrete word representation. We propose to solve this problem by learning a continuous-valued and low-dimensional mapping of words, and base our predictions for the probabilities of the target word on non-linear dynamics of the latent space representation of the words in context window. We build on neural networks-based language models; by expressing them as energy-based models, we can further enrich the models with additional inputs such as part-of-speech tags, topic information and graphs of word similarity. We demonstrate a significantly lower perplexity on different text corpora, as well as improved word accuracy rate on speech recognition tasks, as compared to Kneser-Ney back-off n-gram-based language models.

Index Terms— Speech recognition, natural language, neural networks, probability

1. INTRODUCTION

A key problem in natural (both written and spoken) language processing is designing a metric to score sentences according to their well-formedness in a language, also known as statistical language modeling. In speech recognition applications, statistical language models are generally used to rank the list of candidate hypotheses that are generated based on acoustic match to the input speech. N-gram language models typically use a Markov approximation which assumes that the probability of a word w_t depends only on a short, fixed history $\mathbf{w}_{t-n+1}^{t-1}$ of $n-1$ previous words, and the joint likelihood of a sequence of T words is given by:

$$P\left(\mathbf{w}_1^T\right) = P\left(\mathbf{w}_1^{n-1}\right) \prod_{t=n}^{T} P\left(w_t | \mathbf{w}_{t-n+1}^{t-1}\right) \quad (1)$$

N-gram language models estimate conditional probability of a target word w_t, given its history $\mathbf{w}_{t-n+1}^{t-1}$, by keeping track of the n-gram counts in a corpus. Their main limitation is that as the size of the history increases, the size of the corpus needed to reliably estimate the probabilities grows exponentially. In order to overcome this sparsity, back-off mechanisms [1] are used to approximate n^{th} order statistics with lower-order ones, and sparse or missing probabilities may be further approximated by smoothing [2].

In contrast to the discrete n-gram models, the recently-developed Continuous Statistical Language Models (CSLM) [3], [4], [5], [6], [7], [8], [9], [10], [11], [12] *embed* the words of the $|W|$-dimensional vocabulary into a low-dimensional and continuously valued space $\Re^{|Z|}$, and rather than making predictions based on the sequence of discrete words $w_t, w_{t-1}, \ldots, w_1$ operate instead on the sequence of embedded words $\mathbf{z}_t, \mathbf{z}_{t-1}, \ldots, \mathbf{z}_1$. The advantage of such models over discrete n-gram models is that they allow for a natural way of smoothing for unseen n-gram events. Furthermore, the representations for the words are discriminatively trained in order to optimize the word prediction task.

In this paper, we describe a novel CSLM model that extends the previously presented models. First, our model is capable of incorporating similarity graph constraints on words in the language modeling task. Second, the model can efficiently use word meta-features, like part-of-speech tags. Finally, the model is also flexible enough to handle long range features such as those derived from topic models (see Figure 1). Thus, in essence our model and its architecture synthesize and extend many of the strengths of the state-of-the-art CSLMs. While language modeling is our task and hence test perplexity is a natural evaluation metric, we also evaluate our model on word accuracy for speech recognition (section 4.3).

2. OUR EXTENSIONS TO PREVIOUS APPROACHES

The best-known CSLM is the Neural Probabilistic Language Model (NPLM) [3], which consists of a neural network that takes as input a window of word embedding history $\mathbf{z}_{t-n+1}^{t-n}$ and is trained to directly predict the probability of the next word w_t for the entire vocabulary. Trainable parameters of this system are both the word embedding function (the way in which words, w_t are projected to their low-dimensional representations, $\mathbf{z}_t$) as well as the network combination weights (how the $\mathbf{z}$'s in the context are combined to make the prediction). A variant of this model has been successfully applied to speech recognition [5] and machine translation [6].

Since the NPLM architecture does not allow constraints to be added to the word embeddings, we only retain from this methods the non-linear architecture (single hidden layer neural network) and the trainability of the embedding. We considered instead the Log-BiLinear (LBL) architecture [8], [9], [10], i.e. a probabilistic energy-based model that is trained to predict the embedding $\bar{\mathbf{z}}_t$ of the next word w_t. The key elements of the LBL architecture are explained in Sections 3.1, 3.2 and 3.3. We demonstrate in Section 4.2, that LBL models outperform n-gram language models.

Other nonlinear classifiers (hierarchical logistic regression [7]) or state-space models (Tied-Mixture Language Models, [12]) used for CSLM have been considered that initialize the word representation by computing a square word co-occurrence matrix (bigrams) and reducing its dimensionality through Singular Value Decomposition to the desired number of latent factors $|Z|$. We follow this work in initializing the LBL word embeddings as explained in Section 3.4. We also explain in Section 3.5, how one can impose similarity constraints on the word representation, using for instance information about word similarity from the WordNet taxonomy.

A third, major extension of our LBL model (section 3.6), is

P.M. wishes to acknowledge AT&T Labs Research for summer funding.

our incorporation of part-of-speech tag features as additional inputs, similar to the Deep Neural Networks with Multitask Learning [11]. The latter study, however, addresses different supervised NLP tasks other than language modeling.

Finally, in Section 3.7, we investigate the influence of the long-range dependencies between words in the current and few previous sentences, or in the current document. For this reason, we integrate our CSLM with text topic models, in a spirit similar to HMM-LDA [13].[1] All the four proposed improvements over LBL are evaluated both in terms of language model perplexity and of speech recognition word accuracy in section 4.

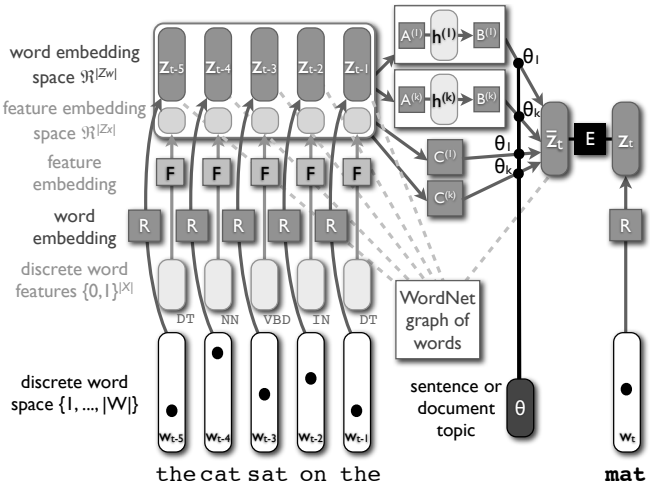

Fig. 1. Enhanced log-bilinear architecture. Given a word history $\mathbf{w}_{t-n+1}^{t-1}$, a low-dimensional embedding $\mathbf{z}_{t-n+1}^{t-1}$ is produced using $\mathbf{R}$ and is fed into a linear $\mathbf{C}$ matrix, as well as into a non-linear (neural network) architecture (parameterized by $\mathbf{A}$ and $\mathbf{B}$ to produce a prediction $\bar{\mathbf{z}}_t$. If one uses a topic model with K topics, the predictor becomes a mixture of K modules, controlled by topic weights $\theta_1, \ldots, \theta_K$ obtained for the current sentence of document from a topic model such as LDA. That prediction is compared to the embedding of all words in the vocabulary using a log-bilinear loss E, which is normalized to give a distribution. Part-of-Speech features can be also embedded using matrix $\mathbf{F}$, alongside the words, and the embeddings can have WordNet::Similarity constraints.

3. ARCHITECTURE

In a typical Continuous Statistical Language Model one tries to compute the probability distribution of the next word in a sequence using the distributed representation of the preceding words. One class of models try to achieve this by capturing the dependencies/interactions between the distributed representation of the next word and the distributed representations of the preceding words in the sequence. This is achieved by defining an *energy* function (a cost) between the variables that capture these dependencies. Learning in such models involves adjusting the parameters such that low energies are assigned to the valid sequences of words and high energies to the invalid ones. This is typically achieved by maximizing the likelihood of the training corpus [14].

3.1. Log-Bilinear Language (LBL) Model

Log-Bilinear models, recently proposed by Mnih et al. in [8, 9, 10] fall in the above class of models. Let us denote by $\mathbf{w}_1^T = [w_1 \ldots w_T]$ a discrete word sequence of length T, and its corresponding low dimensional real-valued representation by $\mathbf{z}_1^T = [\mathbf{z}_1 \ldots \mathbf{z}_T]$ (where $\forall t, z_t \in \Re^{|Z|}$). The LBL model tries to predict the distributed representation of the next word $\mathbf{z}_t$, using a linear function of the distributed representations of the preceding words $\mathbf{z}_{t-n+1}^{t-1}$, where $\mathbf{z}_{t-n+1}^{t-1}$ denotes a stacked history of the previous word embedding (a vector of length $(n-1)|Z|$):

$$\bar{\mathbf{z}}_t = \mathbf{C}\mathbf{z}_{t-n+1}^{t-1} + \mathbf{b}_\mathbf{C} = f_\mathbf{C}\left(\mathbf{z}_{t-n+1}^{t-1}\right) \tag{2}$$

Matrix $\mathbf{C}$ is a learnable parameter matrix that expresses the bilinear interactions between the distributed representations of the previous words and the representation of the current word. The vector $\mathbf{b}_\mathbf{C}$ is the corresponding vector of biases. For any word w_v in the vocabulary with embedding z_v, the energy associated with respect to the current sequence is a bilinear function and is given by

$$E(t, v) = -\bar{\mathbf{z}}_t^T \mathbf{z}_v - b_v \tag{3}$$

Intuitively, this energy can be viewed as expressing the similarity between the predicted distributed representation of the current word, and the distributed representation of any other word w_v in the vocabulary. The similarity is measured by the dot product between the two representations. Using these energies one can assign the probabilities to all the words w_v in the vocabulary:

$$P\left(w_t = w_v | \mathbf{w}_{t-n+1}^{t-1}\right) = \frac{e^{-E(t,v)}}{\sum_{v'=1}^{|W|} e^{-E(t,v')}}. \tag{4}$$

Training of LBL model involves iterating through the entire data set and maximizing, for each target word w_t, its likelihood, i.e. minimizing the negative log likelihood loss $\mathcal{L}_t$:

$$\mathcal{L}_t = E(t, v) + \log \sum_{v'=1}^{|W|} e^{-E(t,v')}. \tag{5}$$

3.2. Non-Linear Extension to LBL (LBLN)

The LBL model as described above is capable of capturing only linear interactions between representations of the previous words and the representation of the next word, conditioned on the matrix $\mathbf{C}$. However, expressing it as an energy-based model allows us to add more complex interactions among the representations just as easily. This is achieved by simply increasing the complexity of the energy function. For instance, one can capture non-linear dependencies among the representations of the previous words, and the next word by adding a single hidden layer neural network, as proposed in [10]. In particular let matrices $\mathbf{A}$ and $\mathbf{B}$ be the two learnable parameter matrices and the vectors $\mathbf{b}_\mathbf{A}$ and $\mathbf{b}_\mathbf{B}$ be the corresponding biases. Let the $\tanh$ sigmoid transfer function be denoted by σ and the set of hidden layer outputs by $\mathbf{h}$. Then the prediction given by this non-linear component, which captures non-linear dependencies among representations, is given by

$$f_{\mathbf{A},\mathbf{B}}\left(\mathbf{z}_{t-n+1}^{t-1}\right) = \mathbf{B}\sigma(\mathbf{A}\mathbf{z}_{t-n+1}^{t-1} + \mathbf{b}_\mathbf{A}) + \mathbf{b}_\mathbf{B}. \tag{6}$$

Then, prediction by both the linear and the non-linear component of the LBL (LBLN) is given by merely the sum of the two terms:

$$\bar{\mathbf{z}}_t = f_{\mathbf{A},\mathbf{B}}\left(\mathbf{z}_{t-n+1}^{t-1}\right) + f_\mathbf{C}\left(\mathbf{z}_{t-n+1}^{t-1}\right). \tag{7}$$

The energy of the system is defined in exactly the same way as in equation (3), and the loss function is defined in the same way as in equation (5). The system is again trained by maximizing the likelihood of the training corpus.

[1] The language model was a much simpler, discrete bigram in their case.

3.3. Training the LBLN Model

Throughout this study the dimensions of the distributed representation of words was set to $|Z| = 100$, and the number of hidden units in the neural network were set to 500.

As mentioned in the previous section, training of LBLN involves maximizing the log-likelihood of the target words in all the sequences of the training set, which is achieved by minimizing the negative log-likelihoods (equation 5) at each word. This minimization is accomplished by a stochastic gradient descent procedure on mini-batches of 1000 words, as given in [8, 10]. Typically, equation 5 is differentiated w.r.t. the prediction $\bar{\mathbf{z}}_t$, the target word representation $\mathbf{z}_w$ and the other word representations $\mathbf{z}_v$, and the gradients are propagated through the linear $\mathbf{C}$ and nonlinear $\mathbf{A}, \mathbf{B}$ modules up to the word representations $\mathbf{R}$ themselves, as well as to the respective biases. Following [8, 10], additional weight momentum μ is added to all parameters. In addition, the word embedding $\mathbf{R}$, and all weight matrices (but not the biases) are subject to L_2-norm regularization. Table 1 summarizes the various hyperparameter values, some of which were taken from [10] and others optimized by cross-validation on a small dataset.

Table 1. Hyperparameters (learning rates η, regularization λ and momentum μ coefficients and graph constraint coefficient) used in the LBL architecture of this article. Values in boldface are taken from [10].

η_C	η_A	η_B	η_R	η_F	λ	μ	γ
$\mathbf{10^{-3}}$	$\mathbf{10^{-1}}$	$\mathbf{10^{-5}}$	$\mathbf{10^{-4}}$	10^{-4}	$\mathbf{10^{-5}}$	**0.5**	1

3.4. Initializing the Latent Word Representation

All the parameters $\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{R}$ are initialized randomly. We use the rule of thumb of generating zero-mean normally distributed words of variance equal to the inverse of the matrix *fan-in* [15]. Biases are initialized to zero, with the exception of b_v, which are initially equal to the unigram statistics of the training data. Some CSLM architectures ([7] and [12]) are however dependent on the initial latent word representation. In order to evaluate this dependency, we followed a procedure similar to [12]. It involved computing a bi-gram co-occurrence matrix $\mathbf{O} = \{O_{w_t,w_{t-1}}\} \in \Re^{|W|\times|W|}$ over the words in the training data, and retrieving a $|Z|$-dimensional embedding of $\mathbf{O}$ through Singular Value Decomposition (equation 8). We used the *forward* term $\mathbf{R} \in \Re^{|Z|\times|W|}$ (equation 9), where each column (word) is represented in terms of a $|Z|$-dimensional influence of the past word w_{t-1} on the current word w_t.

$$\mathbf{O} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T \quad (8)$$

$$\mathbf{R} = (\mathbf{U}\mathbf{\Sigma})^T \quad (9)$$

Alternatively, we also considered initializing $\mathbf{R}$ using an SVD of the n-gram co-occurrence matrix $\mathbf{O} = \{O_{w_t,\mathbf{w}_{t-n+1}^{t-1}}\} \in \Re^{|W|\times n|W|}$.

We now discuss the various extensions to the NLBL model that we explored in the present study.

3.5. Graph Constraints on Latent Word Representations

As can be shown in section 4.4, the low-dimensional nature of the word embedding in CSLMs ($|Z| << |W|$, with $|Z| = 100$ and $|W|$ typically over 10,000) and the word co-occurrence in the text tend to cluster word representations $\mathbf{z}_w$ according to their syntactic co-occurrence and semantic equivalence. In order to speed-up the learning of our model and to potentially help achieve better performance, we considered imposing a graph constraint on the words. For each word w, we defined a word neighborhood N_w obtained through the hierarchical WordNet[2] tree and using the *WordNet::Similarity* module[3]. Specifically, we used the information theoretical-built Resnik similarity [16], keeping in N_w only words whose Resnik score was higher than 8. During learning time, the graph constraint was imposed by adding the following L_2-norm penalty to the total log-likelihood:

$$L_{graph} = \gamma \sum_{w=1}^{|W|} \| \mathbf{z}_w - \frac{1}{|N_v|} \sum_{v\in N_w} \mathbf{z}_v \|_2^2 \quad (10)$$

3.6. Incorporating Part-of-Speech Tag Features

The most important improvement over the LBL [8] and the LBLN [10] was the addition of Part-of-Speech (POS) tags to each word. Conceptually, this step is identical to the word embedding: for each word, discrete POS tags (out of a vocabulary of $|X|$, here between 30 and 52) are mapped into a low-dimensional embedding $\Re^{|Z_X|}$ through a linear operation (matrix $\mathbf{F}$). The matrix $\mathbf{F}$ was also initialized randomly in the same way as discussed in Section 3.4. We also considered the case $|X| = |Z_X|$, with an identity transform $\mathbf{F} = \mathbf{I}_{|\mathbf{X}|}$. Those tags can then be concatenated with the $|Z_W|$-dimensional word representations into a history of $n-1$ word and feature representations, and used as an input to the predictive model (Figure 1), just like in [11]. POS tag features can be trivially extended to accommodate other types of word features.

3.7. Incorporating topics models

A fourth improvement over the LBL and LBLN architecture that we considered consisted in a long-range dependency of the language model on the current topic, simplified as a dependency on the bag-of-words vocabulary statistics. Our main motivation was that such a context-dependent model would enable domain adaptation. This adaptation can be done at document-level (or paragraph-level). When proper document segmentation is not available, such as in broadcast transcripts, a "document" can be defined by considering the last D sentences, assuming that the speakers do not change topic too often.

We decided to implement a topic model based on the popular Latent Dirichlet Allocation[4] [17], a graphical model that is trained to extract a word-topic matrix from a collection of documents, and that can infer latent topic posterior distributions θ_d for each test document d. As can be seen on Fig. 1, the K-dimensional topic vector (where $\sum_k \theta_k = 1$) can be used as weights of a mixture model. Because the predictions made by each component of the mixture add-up for the final prediction $\bar{\mathbf{z}}_t$ (11), the implementation of the topic-dependent LBL(N) architecture is a simple extension of the previously described LBLN-based architectures.

$$\bar{\mathbf{z}}_t = \sum_{k=1}^{K} \theta_k \left(f_{\mathbf{C}_k}\left(\mathbf{z}_{t-n+1}^{t-1}\right) + f_{\mathbf{A}_k,\mathbf{B}_k}\left(\mathbf{z}_{t-n+1}^{t-1}\right)\right) \quad (11)$$

As can be seen in the next section, adding a topic model mixture holds promise in terms of language model perplexity but still requires additional experimental evaluation.

[2] See http://wordnet.princeton.edu

[3] Available at http://wn-similarity.sourceforge.net/

[4] We used the Gibbs-based implementation of LDA, available at http://gibbslda.sourceforge.net/

4. RESULTS

The following section summarizes several sets of experiments performed on four distinct datasets (section 4.1), aimed at assessing the test set perplexity of the respective language models (section 4.2), and at measuring the word accuracy performance for speech recognition tasks (section 4.3). Finally, we illustrate the power of clustering words with low-dimensional representations (section 4.4).

4.1. Datasets

We have evaluated our models on four distinct, public datasets: 1) the *Airline Travel Information System* (ATIS), a small corpus containing short sentences concerning air travel, 2) the *Wall Street Journal* (WSJ) set, containing sentences from business news, 3) the *Reuters-21578* corpus[5] of business news articles, which is normally used for text categorization, and 4) *TV broadcast news transcripts HUB-4* from the LDC (reference 2000S88), with audio information. Table 2 summarizes the statistics of each dataset.

For the WSJ set, we used POS tags to identify and replace all numbers (tag CD) and proper nouns (tags NNP and NNPS), as well as words with 3 or fewer occurrences, by generic tags, resulting in a considerable reduction in the vocabulary size.

For the Reuters set, each article was split into sentences using the Maximum Entropy sentence-splitter by Adwait Ratnaparkhi[6], and then tagged using the Stanford Log-linear Part-of-Speech Tagger[7]. We replaced numbers and rare words (i.e. appearing less than four times) by special tags, as well as out-of-vocabulary test words by $\{unk\}$.

For the HUB-4 corpus, we obtained 100-best hypotheses for each audio file in the test set using a speech recognition system comprising of a trigram language model that was trained on about 813,975 training sentences.

In all the experiments but ATIS, 5% of the training data were set apart during learning for cross-validation (the model with the best performance on the cross-validation set was retained). On the ATIS dataset, the best performance on the training set actually corresponded to the best test performance.

Table 2. Description of the datasets evaluated in this study: size of vocabulary $|W|$, number of POS tags $|X|$, number of training words T_{tr} and sentences/documents D_{tr}, number of test words T_{te} and sentences/documents D_{te}.

Dataset	$\|W\|$	$\|X\|$	T_{tr}	D_{tr}	T_{te}	D_{te}
ATIS	1,311	30	116k	11,294	23k	2,369
WSJ	10,222	52	1,079k	46,145	42k	2,266
Reuters	11,742	45	1,445k	9,598	462k	3,299
HUB-4	25,520	34	813k	19,049	32k	827

4.2. Perplexity on ATIS, WSJ, Reuters and HUB-4

Assuming a language model is defined by the conditional probability distribution q over the vocabulary, its perplexity intuitively corresponds to a word uncertainty given a context and on a sentence of T words, it is defined as:

$$p = \exp\left(-\frac{1}{T}\sum_{t=1}^{T}\log P\left(w_t|\mathbf{w}_{t-n+1}^{t-1}\right)\right) \quad (12)$$

In the absence of task-specific evaluation, such as word accuracy for speech recognition, perplexity is the measure of choice for language models. Therefore, and similar to [3], [8], and [10], we used perplexity to compare our continuous language models to the state-of-the-art probabilistic n-gram models. We chose the best performing n-gram models that include a *back-off* mechanism for handling unseen n-grams [1] and the Kneser-Ney smoothing of probability estimates [2], using an implementation provided by the SRI Language Modeling Toolkit[8] [18]. We did not consider n-gram models extended with POS tags. For each corpus, we selected the n-gram order that minimized the test set perplexity.

We performed an extensive evaluation of many configurations of the LBL-derived architectures and improvements. All the results presented here were achieved in less than 100 learning epochs (i.e. less than 100 passes on the entire training set), and with the set of hyperparameters specified in Table 1.

As can be seen in Tables 3, 4, 5, 6 and 7, most of the linear and all the non-linear LBL language models are superior to n-grams, as they achieve a lower perplexity. Moreover, using the POS tags lowers the perplexity even more. Conversely, various initializations (random or bi-gram/n-gram SVD-based) or *WordNet::Similarity* constraints do not seem to significantly improve the language model.

Taking advantage of the small size of the ATIS dataset, we investigated the influence of several hyper-parameters on the performance of the LBL model: the linear model learning rate η_C, as well as the word embedding learning rate η_R, the first layer η_A and second layer η_B nonlinear module learning rates. We conducted an exhaustive search on a coarse grid of the above hyper-parameters, assuming an LBL(N) architecture with $|Z_W| = 100$ dimensional word representation and $|H| = 0$, 50, 100 or 200 hidden nonlinear nodes, as well as $|Z_X| = 0$ or 3 dimensional embedding of POS tag features. Evidently, as suggested in [10], the number of hidden non-linear nodes had a positive influence on the performance, and our addition of POS tags were beneficial to the language model. Regarding the learning rates, the most sensitive rates were η_R and η_C, then η_A and finally η_B. The optimal results were achieved for the hyper-parameter values in Table 1. We then selected the optimal LBLN architecture with $|Z_W| = 100$ and $|H| = 200$ and further evaluated the joint influence of the feature learning rate η_F, the graph constraint coefficient γ, the dimension of the POS tag embedding $|Z_X|$, and the random or bigram initialization of the latent word representations. The most important factor was η_F, which needed to be smaller than 10^{-3}, and the presence or absence of POS features (larger embedding sizes did not seem to improve the model). Both the type of initializations for $\mathbf{R}$ and the strength of the *WordNet::Similarity* constraint seemed not to yield an impact on the final language model performance, as can be seen in Tables 4, 5 and 6. As summarized in Table 1, we selected a value of $\eta_F = 10^{-4}$ for the feature embedding learning rate and $\gamma = 1$ for the graph constraint.

In a subsequent set of experiments, we evaluated the benefit of adding a topic model to the (syntactic) language model, restricting ourselves to the Reuters dataset (organized in documents) and to the HUB-4 transcripts (a window of five consecutive sentences is treated as a document). We use the standard Latent Dirichlet Allocation topic model to produce a simplex of topic posteriors $\{\theta_{t,1}, \ldots, \theta_{t,K}\}$ for $K = 5$ topics, for each "document", and used these coefficients as weights of a 5-mixture model. We retained, for each mixture component, the same LBL and LBLN architectures as in the previous experiments, and experimented with adding POS features. As Table 5 suggests, adding a topic model improved the plain LBL perplexity on Reuters. Because the learning of the LBLN topic mixture did not converge in 100 epochs, reaching a perplexity slightly above the corresponding non-mixture LBLN architectures, we conjecture that further experiments are necessary on the Reuters

[5] Available at: http://disi.unitn.it/moschitti/corpora.htm

[6] Available on http://sites.google.com/site/adwaitratnaparkhi/publications

[7] Available at http://nlp.stanford.edu/software/tagger.shtml

[8] Available at http://www-speech.sri.com/projects/srilm/

Table 3. Language model perplexity results on the ATIS test set. LBLN with 200 hidden nodes, $|Z_W| = 100$ dimensions for the word representation and all $|Z_X| = 30$ POS tags achieved the lowest perplexity, outperforming the Kneser-Ney 4-gram model. It was not clear whether bi-gram SVD-derived initialization of word representation and *WordNet::Similarity* constraints significantly improved the results.

Method	Init.	WordNet	POS $\|Z_X\|$	Perplexity
KN 4-gram	-	-	-	13.5
LBL	-	-	-	15.19
LBL	-	-	3	14.24
LBLN	-	-	-	12.26
LBLN	bigram	-	-	12.29
LBLN	-	yes	-	12.11
LBLN	-	-	**30**	**11.4**
LBLN	bigram	-	**30**	**11.35**
LBLN	-	yes	**30**	**11.29**
LBLN	bigram	yes	**30**	**11.4**

Table 4. Language model perplexity results on the WSJ test set. A similar architecture to the one in Table 5 outperforms the Kneser-Ney 5-gram model.

Method	Init.	WordNet	POS $\|Z_X\|$	Perplexity
KN 5-gram	-	-	-	88.92
LBL	-	-	-	87.65
LBLN	-	-	-	79.72
LBLN	bigram	yes	-	78.79
LBLN	-	-	5	**76.88**
LBLN	bigram	yes	5	**76.1**

dataset. On the other hand, as can be seen in Table 6, adding a topic mixture model seemed to significantly increase the perplexity of the language model on the HUB-4 dataset.

4.3. Speech recognition (HUB-4)

In Table 6, we present the results of speech recognition experiments using our language model. We used AT&T Watson ASR [19] (with a trigram language model trained on HUB-4 training set) to produce 100-best hypotheses for each of the test audio files of the HUB-4 task. The 1-best and the 100-best oracle word accuracies are 63.7% and 66.6% respectively. Using a range of language models (including a 4-gram discrete LM), we re-ranked the 100-best hypotheses according to LM perplexity (ignoring the acoustic scores), and selected the top one from each list. The top-ranking hypothesis resulting from LBLN models had significantly better word accuracies than any discrete language models. We also see a decrease in perplexity when we include POS tags of a word. In order to measure the efficacy of the language model in selecting the correct hypothesis if it were present in the k-best list, we included the reference sentence as one of the candidates to be ranked. Table 7 shows that we significantly out-performed the best n-gram model on this task as well.

4.4. Word embedding (Reuters)

For the visualization of the word embedding, we chose the Reuters corpus (as it has relatively many words and a small vocabulary), and one of the best-performing language models, based on POS features. Table 8 illustrates the neighborhood of a few selected words, after training the language model and the word embedding. It is clear that functionally and semantically similar words are present in the neighborhood of these words.

Table 5. Language model perplexity results on the Reuters test set. LBLN with 500 hidden nodes, $|Z_W| = 100$ dimensions for the word representation and $|Z_X| = 5$ dimensions for the 40 POS features, a 5-gram SVD-derived initialization of word representation (*init*) and WordNet::Similarity constraints achieves the lowest perplexity, far outperforming the Kneser-Ney 5-gram model. Adding a $K = 5$ dimensional topic model based on LDA posteriors (i.e. creating a 5-mixture model of LBL and LBLN) seemed to improve the perplexity of the linear LBL but not of nonlinear LBLN (with the current set of hyperparameters, learning did not converge in 100 epochs).

Method	Init.	WordNet	POS $\|Z_X\|$	K	Perplex.
KN 5-gram	-	-	-	-	80.78
LBL	-	-	-	-	78.3
LBLN	-	-	-	-	63.92
LBLN	5-gram	-	-	-	63.67
LBLN	-	yes	-	-	63.22
LBLN	-	-	5	-	**60.34**
LBLN	5-gram	yes	5	-	**60.22**
LBL	-	-	-	5	73.12
LBLN	-	-	-	5	65.5
LBLN	-	-	5	5	61.85

4.5. Computational considerations

We implemented our LBL-derived architectures under Matlab. The training was linear in the size of the dataset (i.e. the number of words). As observed for previous CSLM models [3] or [8], the bulk of the computations consisted in evaluating the word likelihood (4) and in differentiating the loss (5), which was theoretically linear in the size of the vocabulary $|W|$. However, thanks to the BLAS and LAPACK numerical libraries, it was sublinear in practice. Typically, training our LBL architectures on moderately sized datasets (*WSJ*, *Reuters* and *TV broadcasts*) would take about a day on a multi-core server. Because of the possible book-keeping overhead that might arise from sampling approximations, or because of the decreased language model performance (higher perplexity) when hierarchical word representation are used [4], or of the LBL [9], we restrict ourselves to the exact solution.

5. ACKNOWLEDGMENTS

We would like to thank Diamantino Caseiro, Enrico Bocchieri and Yann LeCun for their insightful comments about our work.

6. CONCLUSIONS

In summary, in this paper we present an energy based statistical language model with a flexible architecture that allows for novel and diverse extensions of the log-bilinear model formulated in [8]. Consistent with independent findings in [10], we find that adding non-linearity improves predictions of the vanilla LBL significantly. Further, we find consistent and significant predictive improvements by incorporating part-of-speech tags as word features. We also explore other features like word similarity constraints via a word-graph and also long range (document level) topic features whose results are interesting but at this stage merit further investigation. Our results show that our proposed model significantly advances the state-of-the-art and easily bests typical n-gram models on test perplexity. Fi-

Table 6. Speech recognition and language model perplexity results on the HUB-4 task. For each target sentence, the candidate with lowest NLL score (combined acoustic AM and language LM models) from the 100-best hypotheses was chosen. LBLN with 34-dimensional POS tags and bigram SVD-derived initialization of word representation (*init*) achieves a higher word accuracy and the lowest perplexity, outperforming both the speech recognition and language model baseline, and the Kneser-Ney 4-gram model. We also indicate the best and worst possible accuracies that can be achieved with these hypotheses ("Oracle").

Method	AM weight	Accuracy	Perplexity
AT&T Watson	9	63.7 %	-
Back-off KN 4-gram	0	63.5 %	314.23
LBLN	0	**64.1 %**	299.59
LBLN+POS	0	**64.1 %**	294.66
LBLN+POS+init	0	**64.2 %**	**291.88**
LBLN+POS+init+LDA	0	**64.6 %**	382.84
100-best Oracle	-	*66.6 %*	-
Worst of 100-best Oracle	-	*57.8 %*	-

Table 7. Speech recognition results on TV broadcast transcripts, using the same training set and test set as in Table 6, but with the true sentence to be predicted included among the n-best candidates.

Method	Accuracy
Back-off KN 4-gram	86.9 %
LBLN+POS+init	**94 %**
"Oracle"	100 %

nally, we demonstrate the utility of this improved language modeling by obtaining better word accuracy on a speech recognition task.

7. REFERENCES

[1] S. Katz, "Estimation of probabilities from sparse data for the language moel component of a speech recognizer," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. ASSP-35, no. 3, pp. 400–401, 1987.

[2] S. F. Chen and J. Goodman, "An empirical study of smoothing techniques for language modeling," in *Proceedings of the Thirty-Fourth Annual Meeting of the Association for Computational Linguistics*. 1996, p. 310318, San Francisco: Morgan Kaufmann Publishers.

[3] Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin, "A neural probabilistic language model," *Journal of Machine Learning Research*, vol. 3, pp. 1137–1155, 2003.

[4] F. Morin and Y. Bengio, "Hierarchical probabilistic neural network language model," in *Advances in Neural Information Processing Systems NIPS*, 2005.

[5] H. Schwenk and J.-L. Gauvain, "Using continuous space language models for conversational speech recognition," in *IEEE Workshop on Spontaneous Speech Processing and Recognition SSPR*, 2003.

[6] H. Schwenk, "Continuous-space language models for statistical machine translation," *The Prague Bulletin of Mathematical Linguistics*, vol. 93, pp. 137–146, 2010.

[7] J. Blitzer, K. Weinberger, L. Saul, and F. Pereira, "Hierarchical distributed representations for statistical language modeling," in *Advances in Neural Information Processing Systems*, 2004.

Table 8. Examples of 10 closest neighbors in the latent word embedding space on the Reuters dataset, using an LBLN architecture with 500 hidden nodes, $|Z_W| = 100$ dimensions for the word representation and $|Z_X| = 5$ dimensions for the POS features representation. The notion of distance between any two latent word vectors was defined as the cosine similarity. Although word representations were initialized randomly and WordNet::Similarity was not enforced, functionally and semantically (e.g. both synonymic and antonymic) close words tended to cluster.

debt	aa	decrease	met	slow
financing	aaa	drop	introduced	moderate
funding	bbb	decline	rejected	lower
debts	aa-minus	rise	sought	steady
loans	b-minus	increase	supported	slowing
borrowing	a-1	fall	called	double
short-term	bb-minus	jump	charged	higher
indebtedness	a-3	surge	joined	break
long-term	bbb-minus	reduction	adopted	weaker
principal	a-plus	limit	made	stable
capital	a-minus	slump	sent	narrow

[8] A. Mnih and G. Hinton, "Three new graphical models for statistical language modelling," in *24th International Conference on Machine Learning ICML*, 2007, pp. 641–648.

[9] A. Mnih and G. Hinton, "A scalable hierarchical distributed language model," in *Advances in Neural Information Processing Systems NIPS*, 2008.

[10] A. Mnih, Y. Zhang, and G. Hinton, "Improving a statistical language model through non-linear prediction," *Neurocomputing*, vol. 72, no. 7-9, pp. 1414 – 1418, 2009.

[11] R. Collobert and J. Weston, "A unified architecture for natural language processing: deep neural networks with multitask learning," in *ICML '08: Proceedings of the 25th international conference on Machine learning*, 2008, pp. 160–167.

[12] R. Sarikaya, A. Emami, M. Afify, and B. Ramabhadran, "Continuous space language modeling technique," in *IEEE International Conference on Acoustics, Speech and Speech Processing ICASSP*, 2010, pp. 5186–5189.

[13] T. Griffiths, M. Steyvers, D. Blei, and J. Tenenbaum, "Integrating topics and syntax," in *Advances in Neural Information Processing Systems*, 2005.

[14] Y. LeCun, S. Chopra, R. Hadsell, and F.J. Huang, "A tutorial on energy-based learning," in *Predicting Structured Outputs*. 2006, MIT Press.

[15] Y. LeCun, L. Bottou, G. Orr, and K. Muller, "Efficient backprop.," in *Lecture Notes in Computer Science*. 1998, Berlin/Heidelberg: Springer.

[16] P. Resnik, "Semantic similarity in a taxonomy: An information-based measure and its application to problems of ambiguity in natural language," *Journal of Artificial Intelligence Research*, vol. 11, pp. 95–130, 1999.

[17] D. Blei, A. Ng, and M. Jordan, "Latent dirichlet allocation," *Journal of Machine Learning Research*, vol. 3, pp. 993–1022, 2003.

[18] A. Stolcke, "Srilm - an extensible language modeling toolkit," in *7th International Conference on Spoken Language Processing ICSLP*, 2002, pp. 901–904.

[19] V. Goffin, C. Allauzen, E. Bocchieri, D. Hakkani-Tur, A. Ljolje, S. Parthasarathy, M. Rahim, G. Riccardi, and M. Saraclar, "The AT&T WATSON Speech Recognizer," in *Proceedings of ICASSP*, Philadelphia, PA, 2005.