A Neural Network for Text Representation

Mikaela Keller and Samy Bengio

IDIAP Research Institute, 1920 Martigny, Switzerland*
{mkeller,bengio}@idiap.ch

Abstract. Text categorization and retrieval tasks are often based on a good representation of textual data. Departing from the classical *vector space model*, several probabilistic models have been proposed recently, such as PLSA. In this paper, we propose the use of a neural network based, non-probabilistic, solution, which captures jointly a rich representation of words and documents. Experiments performed on two information retrieval tasks using the TDT2 database and the TREC-8 and 9 sets of queries yielded a better performance for the proposed neural network model, as compared to PLSA and the classical TFIDF representations.

1 Introduction

The success of several real-life applications involving tasks such as text categorization and document retrieval is often based on a good representation of textual data. The most basic but nevertheless widely used technique is the *vector space model* (VSM) [1] (also often called *bag-of-words*), which makes the assumption that the precise order of the words is uninformative.

Such representation neglects potential semantic links between words. In order to take them into account, several more recent models have been proposed in the literature, mostly based on a probabilistic approach, including the Probabilistic Latent Semantic Analysis (PLSA) [2]. They in general factor the joint or conditional probability of words and documents by assuming that the choice of a word during the generation of a document is independent of the document given some hidden variable, often called *topic* or *aspect*.

In this paper, we would like to argue that while the basic idea behind probabilistic models is appealing (trying to extract higher level concepts, such as topics, from raw texts), there is no need to constrain the model to be probabilistic. Indeed, most of the applications relying on text representation do not really need precise probabilities. It was recently argued [3] that in such a case, one should probably favor so-called *energy-based models*, which associate an unnormalized energy to each target configuration, instead of a proper probability, and then simply compare energies of competing solutions in order to take a final

* This work was supported in part by the Swiss NSF through the NCCR on IM2 and in part by the European PASCAL Network of Excellence, IST-2002-506778, through the Swiss OFES.

decision. It is argued that this scheme enables the use of architectures and loss functions that would not be possible with probabilistic models.

We thus propose here a neural network based representation that can be trained on a large corpus of documents, and which associates a high score to pairs of word-document that appear in the corpus and a low score otherwise. The model, which automatically induces a rich and compact representation of words and documents, can then be used for several text-related applications such as information retrieval and text categorization.

The outline of the paper is as follows. Section 2 briefly summarizes current state-of-the-art techniques used for text representation, mostly based on probabilistic models. Then, Section 3 presents our proposed neural network based model. This is followed in Section 4 by some experiments on two real information retrieval tasks. Finally, Section 5 concludes the paper.

2 Related Work

In most Textual Information Access applications, documents are represented within the *Vector Space Model* (VSM) [4]. In this model, each document d is represented as a vector $(\alpha_1, ..., \alpha_M)$, where α_j is a function of the frequency of the j^{th} word w_j in a chosen dictionary of size M. To be more concrete, let us consider the Document Retrieval task, - used in the experimental section - and how VSM is implemented there. In a Document Retrieval task, a user formulates a query q addressed to a database, and the database documents d are then ranked according to their *Relevance Status Value*, $RSV(q, d)$, which is defined so that documents relevant to q should have higher values than non-relevant ones. In the VSM, $RSV(q, d)$ is defined as the scalar product of the query's and document's representations: $RSV(q, d) = \sum_{j=1}^{M} \alpha_j^q \cdot \alpha_j^d$, where α_j^d (resp. α_j^q) is the weight in the document (resp. query) representation of the j^{th} dictionary word. A simple way to implement this value function is to choose α_j^q as a binary weight stating the presence or absence of the word in the query, and d_j as the well-known TFIDF weight [5]:

$$\alpha_j^d = tf_j(d) \cdot \log(\frac{N}{df_j}) ,$$

where $tf_j(d)$ corresponds to the number of occurrences of w_j in d, N is the number of documents in the database and df_j stands for the number of documents the term w_j appears in. It is designed to give more importance to terms frequent in the document while penalizing words appearing in too many documents.

In the VSM, two documents (or a query and a document) are considered similar if they are composed of the same words. However, a property of most human languages is that a certain topic can be expressed with different words (*synonyms*). Moreover, a given word can often be used in totally different contexts (*polyseme*). The VSM does not model these links between words. Several attempts have been proposed to take that into account, among which the use of the so-called Latent Semantic Analysis (LSA)[6]. LSA tries to link words together according to their co-occurrences in a database of documents by performing a Singular Value Decomposition. The more recent Probabilistic Latent Semantic Analysis (PLSA) model [2] seeks a generative model for word/document co-occurrences. It makes the assumption that each word w_j in a given document d_δ, is generated from a latent aspect t taking values among $\{1, \dots, K\}$, K being a chosen hyper-parameter, and δ a variable picking one document among the others in the database. The joint probability of a word w_j and a document d_δ is then:

$$P(d_\delta, w_j) = P(\delta) \sum_{k=1}^{K} P(t = k|d_\delta) P(w_j|t = k) \ . \tag{1}$$

PLSA can then be used to replace the original VSM document representation by a representation in a low-dimensional "latent" space. In [2], the components of the document in the low-dimensional space are $P(t = k|d), \forall k$; for each unseen document or query these are computed by maximizing the log likelihood of (1) with $P(w_j|t = k)$ fixed. Successful Document Retrieval experiments have been reported in [2], for which documents were ranked according to a combination of their cosine similarities with the query in the latent space and in the VSM.

Several other probabilistic models have also been proposed lately, including a hierarchical version of PLSA [7], Latent Dirichlet Allocation [8], multinomial Principal Component Analysis [9], Theme Topic Mixture Model [10], etc. Kernel methods pursuing the same goal have been also proposed [11].

3 Proposed Model

As seen so far, most recent research work have concentrated on novel probabilistic models for document representation. But is the probabilistic framework really necessary at all? In the resolution of a machine learning task, such a probabilistic framework appears necessary in two cases: either some probabilities are involved in the final decision, or probabilities are to be used as a tool for exploring the solution space. The tasks related to Information Access do not necessarily belong to the first case; for example in a Document Retrieval task, what we seek is a ranking of *RSV*, for which a probabilistic setting is not particularly needed. Regarding the second case, as it has been suggested in [3], while probabilities are a useful tool, they establish constraints, *eg* of normalization or cost function to be minimized, which are not always justified and are difficult to deal with. In addition, in a non-probabilistic framework, a lot of powerful tools allowing different kinds of exploration are available, among which the well-established margin and kernel concepts as well as the stochastic approximation.

The model we propose in this paper is designed to take advantage of the huge amount of unlabeled textual documents, using them as a clue *per se* to the links between words. The basic idea is to train a Neural Network using couples *(word, document)* as inputs and the absence or presence of the word in the document as targets.

A similar approach has been first proposed successfully in the context of statistical language modeling under the name of *Neural Probabilistic Language*

Model (NPLM) [12], which learns a distributed representation for each word alongside with the probability of word sequences in this representation.

Here we adapt the same idea and call our model *Neural Network for Text Representation* (NNTR). As illustrated in Figure 1, there are two input vectors in an NNTR: the first one is a word w_j represented by a one-hot encoding, and the second one is a document d_i represented as a VSM with TFIDF weighting. The output is a score which target is high if w_j is *in the context of* d_i, and low otherwise. As depicted in Figure 1, the word (resp. document) vector is first passed through a Multi-Layer Perceptron MLP_W (resp. MLP_D) that extracts a richer and more distributed representation of words (resp. documents); these two representations are then concatenated and transformed non-linearly in order to obtain the target score using MLP_T, as summarized in (2):

$$\text{NNTR}(w_j, d_i) = \text{MLP}_T \left\{ [\text{MLP}_W(w_j), \text{MLP}_D(d_j)] \right\} . \tag{2}$$

All the parameters of the model are trained jointly on a text corpus, assigning high scores to pairs (w_j, d_i) corresponding to documents d_i containing words w_j and low scores for all the other pairs.

A naive criterion would be to maximize the likelihood of the correct class, however, doing so would give the same weight to each seen example. Note that our data presents a huge imbalance between the number of *positive* pairs and the number of *negative* pairs (each document only contains a fraction of words of the vocabulary). Thus, the model would quickly be biased towards answering negatively and would then have difficulties in learning anything else. Another kind of imbalance specific to our data is that, among the positive examples, a few words tend to appear really often, while a lot appear only in few documents, which would bias the model to give lower probabilities to pairs with infrequent words independently of the document.

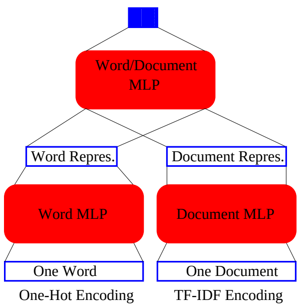

Fig. 1. The NNTR model

Task	TFIDF	PLSA	NNTR
Retrieval	0.170	0.199	**0.215**
Filtering	0.185	0.189	**0.192**

Fig. 2. Compared mean Averaged Precisions (the higher the better) for Document Retrieval and Batch Document Filtering tasks.

The approach we thus propose does not try to obtain probabilities but simply unnormalized scores. Thanks to this additional freedom, we can help the optimization process by weighting the training examples in order to balance the total number of positive examples (words w_j that are indeed in documents d_i)

with the total number of negative examples. We can also balance each positive example independently of its *document frequency* (the number of documents into which the word appears). The criterion we thus optimize can be expressed as follows:

$$\mathcal{C} = \frac{1}{L^-}\sum_{l=1}^{L^-} Q(x_l, -1) + \frac{1}{M}\sum_{j=1}^{M}\frac{1}{\mathrm{df}_j}\sum_{i=1}^{\mathrm{df}_j} Q((d_i, w_j), 1), \tag{3}$$

where L^- is the number of negative examples, M the number of words in the dictionary extracted from the training set, df_j the number of documents the word w_j appears in, and $Q(x, y)$ the cost function for an example $x = (d, w)$ and its target y. Note that using this weighting technique we do not need to present the whole negative example set but a sub-sampling of it at each iteration, which makes stochastic gradient descent training much faster. Furthermore, we use a margin-based cost function $Q(x, y) = |1 - y \cdot \mathrm{NNTR(x)}|_+$, as proposed in [13], where $|z|_+ = \max(0, z)$.

4 Experiments

TDT2 is a database of transcripted broadcast news in American English. For this experiment we used 24 823 documents from a manually produced transcription and segmentation, referred to in the following as TDT2-clean. Two sets of 50 queries for documents of TDT2, called TREC-8 and TREC-9 were collected during TREC SDR evaluation. In this **Document Retrieval** classical setting the database documents are available as development data as well as the TREC-8 queries and their corresponding relevance judgements, while the TREC-9 queries are for evaluation. Using TDT2-clean, we trained a PLSA model with 1000 aspects and a NNTR with the following architecture: the word and document sub-MLPs had 25 438 inputs each (corresponding to the size of the training set vocabulary), no hidden unit, and 10 outputs each; the joint word-document MLP_T had 20 inputs, 25 hidden units, and one output unit. Similarly to [2], the relevance of a document d to a query q was computed as $\lambda \cdot RSV_{tfidf}(q, d) + (1 - \lambda) \cdot RSV_{model}(q, d)$, where $RSV_{tfidf}(q, d)$ is a normalized version of the scalar product described in Sect. 2, $RSV_{model}(q, d)$ in the case of PLSA is the cosine similarity and in the case of NNTR the normalized sum, over words w of q, of $\mathrm{NNTR}(w, q)$. All hyper-parameters of the compared models, including λ, were tuned by cross-validation using TREC-8 queries, and we report the mean averaged precision of each model for TREC-9 queries in Figure 2.

Another Document Retrieval setting, described in [14], is the **Batch Filtering** task. In this application, the targeted documents are not immediately available. Thus, we trained our models using a parallel corpus, which we called TDT2-par, of 28 843 documents from other medias covering the same period of news as TDT2-clean. Using TDT2-par, we trained a PLSA model with 500 aspects and a NNTR with the word and document sub-MLPs having 63 736 inputs each, no hidden unit, and 10 outputs each, and the joint word-document MLP_T having 20 inputs, 10 hidden units, and one output unit, with all hyper-parameters tuned using cross-validation over the training set. Note however that in that case there were no data available to tune λ, and it was thus set to 0.5 arbitrarily for both models. Figure 2 reports the results in terms of mean averaged precision for the TREC-9 queries.

5 Conclusion and Discussion

In this paper, we have proposed a novel, non-probabilistic, text representation model, which yields rich internal representations of both words and documents. It has been applied to two text-related tasks, namely **document retrieval** and **batch filtering**. In both cases, the proposed neural network yielded a better mean averaged precision than the well-known PLSA model. Several extensions of NNTR are currently investigated, including representing a full query/document relation instead of a word/document relation, with shared parameters between all word sub-MLPs of the query.

References

1. Sebastiani, F.: Machine learning in automated text categorization. ACM Computing Surveys **34** (2002) 1–47
2. Hofmann, T.: Unsupervised learning by Probabilistic Latent Semantic Analysis. Machine Learning **42** (2001) 177–196
3. LeCun, Y., Huang, F.J.: Loss functions for discriminative training of energy-based models. In: Proc. of AIStats. (2005)
4. Salton, G., Wong, A., Yang, C.: A Vector Space Model for Automatic Indexing. Communication of the ACM **18** (1975)
5. Salton, G., Buckley, C.: Term-weighting approaches in automatic text retrieval. Information Processing and Management **24** (1988) 513–523
6. Deerwester, S.C., Dumais, S.T., Landauer, T.K., Furnas, G.W., Harshman, R.A.: Indexing by Latent Semantic Analysis. Journal of the American Society of Information Science **41** (1990) 391–407
7. Gaussier, E., Goutte, C., Popat, K., Chen, F.: A hierarchical model for clustering and categorising documents. In: Advances in Information Retrieval – Proceedings of the 24th BCS-IRSG European Colloquium on IR Research (ECIR-02), Glasgow. (2002) 229–247
8. Blei, D., Ng, A., Jordan, M.: Latent Dirichlet Allocation. JMLR **3** (2003) 993–1022
9. Buntine, W.: Variational extensions to em and multinomial pca. In: ECML. (2002) 23–34
10. Keller, M., Bengio, S.: Theme topic mixture model: A graphical model for document representation. In: PASCAL Workshop on Learning Methods for Text Understanding and Mining. (2004)
11. Cristianini, N., Shawe-Taylor, J., Lodhi, H.: Latent semantic kernels. J. Intell. Inf. Syst. **18** (2002) 127–152
12. Bengio, Y., Ducharme, R., Vincent, P., Gauvin, C.: A Neural Probabilistic Language Model. JMLR **3** (2003) 1137–1155
13. Collobert, R., Bengio, S.: Links between perceptrons, MLPs and SVMs. In: Proceedings of ICML. (2004)
14. Lewis, D.D.: The trec-4 filtering track. In: TREC. (1995)