Deep Autoencoder Topic Model for Short Texts

Modelos de T'opicos para textos cortos mediante auto-codificadores de m'ultiples capas

Girish Kumar
NUS High School of Math and Science
Singapore 129957
girishvilla@gmail.com

Luis F. D'Haro
Institute for Infocomm Research
Singapore 138632
luisdhe@i2r.a-star.edu.sg

Resumen: En este trabajo presentamos un método para modelado de tópicos mediante el uso de auto-codificadores de múltiples capas (DATM por sus siglas en inglés). El objetivo principal de estos modelos es la extracción de distribuciones de tópicos en textos cortos. En un análisis comparativo, el método propuesto proporciona mejores resultados que otros métodos convencionales (LSA y LDA).
Palabras clave: Modelos de tópicos, Máquinas de Boltzmann Restringidas

Abstract: We present the Deep Autoencoder Topic Model (DATM) for the purpose of discovering topics from short texts. The DATM is trained in two steps: i) greedy layer-wise pre-training as Sparse & Selective Restricted Boltzmann Machines (RBMs) and ii) parameter fine-tuning with back-propagation. When benchmarked with the topic coherence metric, the DATM outperformed Latent Semantic Analysis and Latent Dirichlet Allocation.
Keywords: Topic models, restricted boltzmann machines, autoencoders, deep learning

1 Introduction

Topic models discover hidden topical structure in large sets of training documents by assuming that hidden topics (latent variables) generate the training documents (observed variables) through a generative process. Topics are usually described by a set of related words over a fixed vocabulary. Prominent topic modelling methods include Latent Semantic Analysis (LSA) and Latent Dirichlet Allocation (LDA). An introduction to LSA and LDA is given in the Appendix.
However, LDA and LSA do not perform well on short documents (in our case, sentences) as they do not model word-word co-occurrences well. As such, we propose a novel Deep Auto-encoder Topic model (DATM) that models both word-word and word-document co-occurrences.

2 Methodology

Figure 1 provides an overview of DATM training. To capture word-word and word-document co-occurrences, we chose the training input to be document vectors, $\mathbf{x} \in [0,1]^n$ where n is the vocabulary size. The DATM is then trained on input documents in two steps: i) greedy layer-wise pre-training as generative Restricted Boltzmann Machines (RBMs) and ii) parameter fine-tuning with back-propagation to learn the identity approximation of the input data for dimensionality reduction.
First, each RBM is trained greedily (left of Figure 1) using contrastive divergence (CD) learning [Hinton *et al.*2006b]. A short introduction to RBMs and contrastive divergence learning is provided in the Appendix. The RBMs consist of stochastic, binary units and are trained one by one starting from the bottom-most RBM which directly takes the input data. The upper RBMs take the output of the trained RBM below. The goal of CD learning is similar to that of LDA: tuning the RBM's weights $\mathbf{w}$ and biases $\mathbf{b}$ to find the set of latent variables, $\mathbf{h_1}, \mathbf{y}$, that maximise the probability of observing the documents. After RBM pre-training, the DATM is unrolled (right of Figure 1) to reconstruct the input data vectors for dimensionality reduction and to map topics to their constituent words. A softmax bottleneck layer is added to normalize the hidden output, $\mathbf{y}$, of the RBMs to a probability distribution. Note that each unit in the soft-max layer corresponds to each latent topic to be discovered. The stochastic binary activities of the other feature layers are replaced by the real-valued probabilities. To fine-tune the network, we back-propagate the gradient of the mean cross-entropy error (E) between $\widetilde{\mathbf{x}}$ and $\mathbf{x}$. d is the number of documents.

$$E(\widetilde{\mathbf{x}}, \mathbf{x}) = -\frac{1}{d}\sum_{k=1}^{d}[\mathbf{x}_k \log \widetilde{\mathbf{x}}_k + (1-\mathbf{x}_k)(1-\log \widetilde{\mathbf{x}}_k)] \quad (1)$$

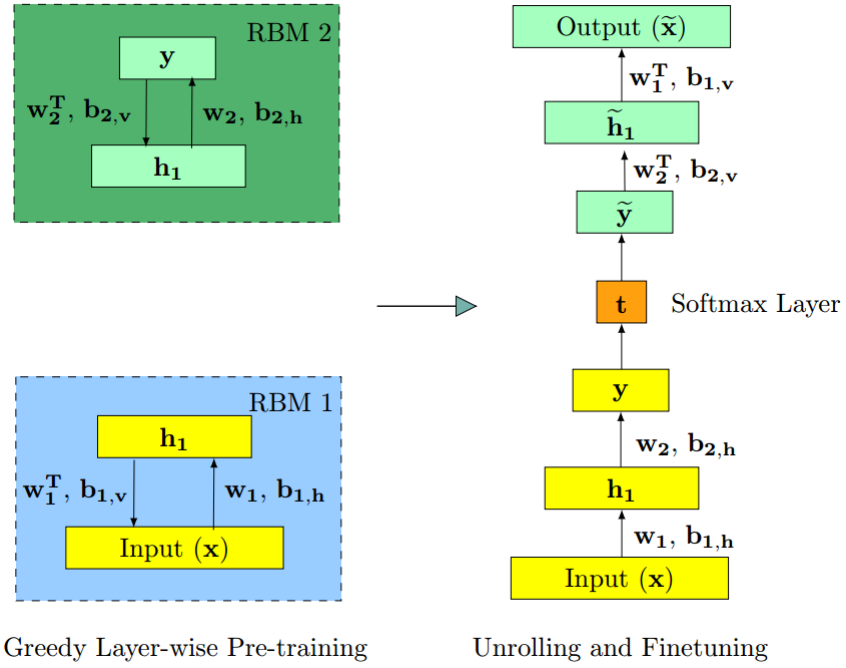

Figure 1: DATM Training Overview

Upon DATM training, we obtained the topic distribution, $\mathbf{t}$, of a document by first vectorizing it and then computing the soft-max layer activations (right of Figure 1, bottom part). For obtaining the words that describe each topic, we found words in the vocabulary that are associated with the activations of each softmax layer unit (right of Figure 1, upper part). To find the words that describe the k^{th} topic, we strongly activate the corresponding k^{th} soft-max unit by letting $\mathbf{t}[i] = \begin{cases} 0 & : i \neq k \\ 1 & : i = k \end{cases}$. We then compute the output activations with $\mathbf{t}$ and obtain the words that correspond to the output units with the highest activations.

2.1 Sparse and Selective RBMs

During testing, we found that all the topics decoded from the auto-encoder consisted of the exact same words. Interestingly, we found that these words were also the most frequently occurring terms as training was stuck in an undesirable local minimum of the cost function. Closer inspection found all of the hidden layer neurons being activated regardless of the input. We hypothesized that this was due to the sparsity of the input document vectors due to the short length of sentences. To solve this issue, inspired by [Lee *et al.*2008], we modified the RBM cost function to include sparsity and selectivity penalty terms. Sparsity ensures that each document belongs to at most a few topics. Selectivity ensure that each topic encodes for only a subset of the training documents. Hence, given d training examples $\{\mathbf{v}^{(1)}, ..., \mathbf{v}^{(d)}\}$, training a sparse and selective RBM, with n hidden units $\mathbf{h}$, is presented in the form of an optimization problem, as defined in Equation 2.1.

$$\text{minimize}_{w_i, b_{i,h}, b_{i,v}} \{ \underbrace{-\frac{1}{d}\sum_{l=1}^{d}\sum_{\mathbf{h}} \log(p(\mathbf{v}^{(l)}))}_{\text{RBM Log Likelihood}} + \underbrace{\lambda \cdot \frac{1}{n}\sum_{j=1}^{n} |\rho - \frac{1}{d}\sum_{l=1}^{d} \mathbb{E}[h_j^{(l)}|\mathbf{v}^{(l)}]|^2}_{\text{Selectivity Penalty}} + \underbrace{\mu \cdot \frac{1}{d}\sum_{l=1}^{d} |\tau - \frac{1}{n}\sum_{j=1}^{n} \mathbb{E}[h_j^{(l)}|\mathbf{v}^{(l)}]|^2}_{\text{Sparsity Penalty}} \} \quad (2)$$

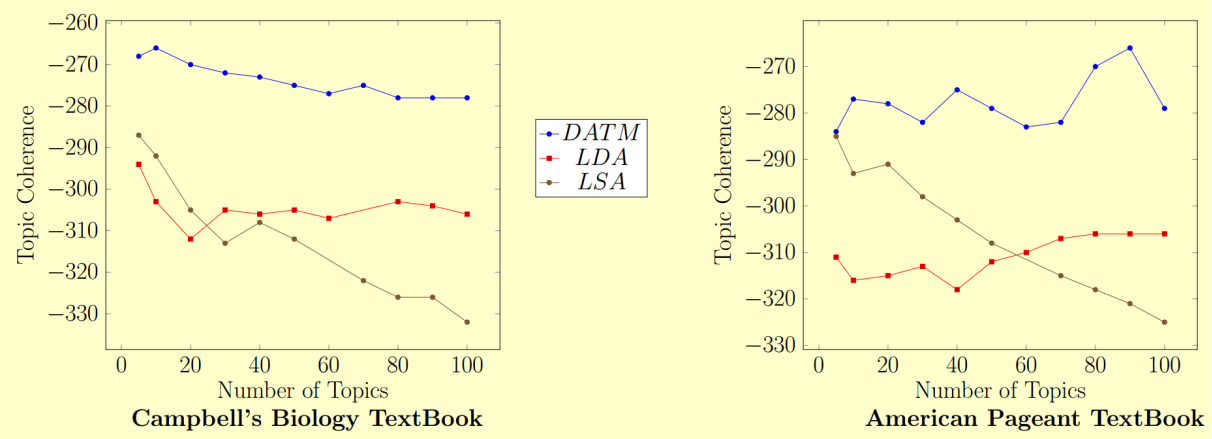

Figure 2: Average Topic Coherence for the various datasets and topic models

where $\mathbb{E}[h_j^{(l)}|\mathbf{v}^{(l)}]$ is the expected activation of hidden unit h_j given input $\mathbf{v}^{(l)}$; ρ, τ are the selectivity & sparsity targets and λ, μ are the penalty-term weights. Since computing the log-likelihood gradient is intractable, we deal with minimising the log-likelihood term and the penalty terms separately. Contrastive divergence was used to estimate the log-likelihood gradient to update the weights and the biases. Gradient descent is used for the penalty terms to update only the biases as they directly control the degree to which the hidden neurons are activated [Lee *et al.*2008].

3 *Experimentation & Results*

For benchmarking the proposed DATM with LDA and LSA, we used 2 corpora for which statistics are in Table 1. Campbell's Biology and The American Pageant are high school textbooks for biology and US history respectively. Here, each sentence is a document. The textbook datasets will allow us to to benchmark the performance of the DATM on short and informative sentences which is relevant to our work. We preprocessed all the datasets by removing stopwords and stemming. Also, we only consider the 2000 most frequent words in each dataset.

	Campbell Biology	**American Pageant**
No. of Documents	35621	22797
Words per Document	20	19

Table 1: Corpora Used for DATM Benchmarking & Evaluation

We use Average Topic Coherence (ATC) for performance benchmarking [Mimno *et al.*2011]. ATC computes a sum of pair-wise scores on the top n words, w, that describe a topic.

$$\text{ATC} = \frac{1}{T}\sum_{t=1}^{T}\sum_{i=2}^{n}\sum_{j=1}^{i-1}\log\frac{D(w_i, w_j)+1}{D(w_i)} \quad (3)$$

where T is the number of topics while $D(w_i)$ and $D(w_i, w_j)$ are the counts of training documents containing the word w_i and both the words w_i and w_j respectively. A better topic model will result in a less negative ATC. ATC was chosen as it was found to be strongly correlated to human judgement of topics [Mimno *et al.*2011]. Furthermore, ATC quantifies the extent to which the topic model captures word-word co-occurrences. Figure 2 shows the ATC scores for the DATM, LDA and LSA for the 2 datasets. Each topic model was used to discover various number of topics from each dataset, ranging from 5 to 100.

The LDA model was trained using the implementation in the Matlab Topic Modelling Toolbox [Griffiths and Steyvers2004]. 300 iterations were used for training on all the corpora. The *gensim* package was used for LSA [Řehůřek and Sojka2010]. We implemented our proposed DATM with Theano [Bastien *et al.*2012]. A hidden layer size of [500] and sparsity = selectivity = 0.03 were used. 50 iterations were used for training on all the corpora. Note that we chose parameters based on those proposed by Hinton et. al [Hinton2010b]. For ATC calculations, the top 20 words for each topic were used.

Evidently, our proposed DATM outperforms LDA and LSA on the ATC metric. However, better performance on ATC does not conclusively prove DATM's superiority. More tests

with multiple performance metrics are required to do so. Nonetheless, the results do manifest that DATM is better in modelling word-word co-occurrences.

4 Conclusion

In this paper, with the aim of topic modelling short and informative texts, we have proposed the Deep Autoencoder Topic Model (DATM). Training the DATM consists of two steps: 1) Greedy Layer-wise Pre-training & 2) Unrolling and fine-tuning via backpropagation. Furthermore, to deal with the issue of the sparsity, we added sparsity and selectivity penalties to the RBM cost function. The DATM was finally benchmarked with the topic coherence metric with textbook datasets, where it outperformed the widely-used LDA (Latent Dirichlet Allocation) and LSA (Latent Semantic Analysis).

Acknowledgement

The authors would like to thank Dr. Rafael E. Banchs from the Institute for Infocomm Research for his advice and support during the realization of this work.

References

Frédéric Bastien, Pascal Lamblin, Razvan Pascanu, James Bergstra, Ian J. Goodfellow, Arnaud Bergeron, Nicolas Bouchard, and Yoshua Bengio. Theano: new features and speed improvements. Deep Learning and Unsupervised Feature Learning NIPS 2012 Workshop, 2012.

Y. Bengio. Learning deep architectures for ai. *Foundations and Trends in Machine Learning*, November 2009.

Thomas L Griffiths and Mark Steyvers. Finding scientific topics. *Proceedings of the National academy of Sciences of the United States of America*, 101(Suppl 1):5228–5235, 2004.

G. E. Hinton, S. Osindero, and Y-W. Teh. A fast learning algorithm for deep belief nets. *Neural Computation*, 2006.

Geoffrey Hinton, Simon Osindero, and Yee-Whye Teh. A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554, 2006.

G. E. Hinton. A practical guide to training restricted boltzmann machines. *Momentum*, 2010.

Geoffrey Hinton. A practical guide to training restricted boltzmann machines. *Momentum*, 9(1):926, 2010.

J.J. Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Biophysics*, 1982.

Honglak Lee, Chaitanya Ekanadham, and Andrew Y Ng. Sparse deep belief net model for visual area v2. In *Advances in neural information processing systems*, pages 873–880, 2008.

David Mimno, Hanna M Wallach, Edmund Talley, Miriam Leenders, and Andrew McCallum. Optimizing semantic coherence in topic models. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 262–272. Association for Computational Linguistics, 2011.

Radim Řehůřek and Petr Sojka. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50, Valletta, Malta, May 2010. ELRA.

R. Salakhutdinov and G. E. Hinton. Deep boltzmann machines. In *International Conference on Artificial Intelligence and Statistics*, 2009.

A Appendix

A.1 LSA

Latent Semantic Analysis (LSA) is one of the most widely-used methods for learning latent topics from text and is often used for dimensionality reduction. Given a document-term matrix, $\mathbf{M} \in \mathbb{R}^{V \times N}$, where V is the number of words in the vocabulary and N is the number of input training documents, LSA factorizes $\mathbf{M}$ using Singular Value Decomposition to find a low-rank approximation given as follows. Rank lowering results in the combination of some dimensions which results in dependence on more than one term.

$$\mathbf{M} \approx \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$$

$\mathbf{U}$ and $\mathbf{V}$ represent word and document embeddings on the latent topic space.

A.2 LDA

Latent Dirichlet allocation (LDA) is a generative probabilistic model of a set of documents. Documents are represented as random mixtures over hidden topics, where each topic is characterized

by a distribution over words. The following generative process is assumed for each document in a corpus.

1. Choose number of words, $N \sim$Poisson(μ)
2. Choose topic mixture/distribution $\theta \sim$Dirichlet(α)
3. Choose topics $z_k \sim$Dirichlet(θ)
4. Choose words $w \sim$Multinomial(ϕ_k)

Expectiation-Maximization or Gibbs Sampling can then be utilized for inferring topics from the assumed generative process.

A.3 Restricted Boltzmann Machines (RBMs)

Restricted Boltzmann Machine (RBM), a bipartite graph variant of the boltzmann machine, is an energy-based probability model to infer hidden variables [Bengio2009]. The bipartite nature of the RBM means that it does not allow connections among units in each layer [Salakhutdinov and Hinton2009], which makes it efficient in learning [Bengio2009]. As a special form of the general second-order polynomial, the energy function of the RBM, formed by the joint configurations of both visible and hidden units $(\mathbf{v}, \mathbf{h})$, is given by [Hopfield1982]:

$$E(\mathbf{v}, \mathbf{h}) = -\sum_i c_i v_i - \sum_j b_j h_j - \sum_{i,j} w_{ij} v_i h_j \tag{4}$$

where $\mathbf{v}$ is the visible inputs, $\mathbf{h}$ consists of the hidden nodes or latent variables, i represents the number of dimensions for each input, and j represents the number of hidden nodes. RBMs are trained as probabilistic models by maximizing a log of the following likelihood of the visible vector [Hinton2010a].

$$p(\mathbf{v}) = \frac{1}{Z} \sum_{\mathbf{h}} e^{-E(\mathbf{v},\mathbf{h})} \tag{5}$$

where the partition function, Z, is given as follows

$$Z = \sum_{\mathbf{v},\mathbf{h}} e^{-E(\mathbf{v},\mathbf{h})} \tag{6}$$

Contrastive divergence (CD) is used to efficiently approximate the log-likelihood gradient of RBMs [Hinton *et al.*2006a]. The RBM learns in an unsupervised fashion with a stochastic element being introduced in the random sampling process. The CD algorithm updates the weights as in the following:

$$\mathbf{W}_{t+1} = \mathbf{W}_t + \epsilon(\mathbf{h}_t \mathbf{v}_t - \mathbf{h}_{t+1} \mathbf{v}_{t+1}) \tag{7}$$

where the subscript t represents the number of iterations, $\mathbf{v}$ is the visible inputs, $\mathbf{h}$ is the hidden vector, and ϵ is the learning rate.