Deep Learning of Semantic Word Representations to Implement a Content-based Recommender for the RecSys Challenge'14

Omar U. Florez[1], Lama Nachman[1]

Intel Labs, Santa Clara CA 95053, USA,
{Omar.U.Florez, Lama.Nachman}@intel.com

Abstract. In this paper, we will discuss a recommender system that exploits the semantics regularities captured by a Recurrent Neural Network (RNN) in text documents. Many information retrieval systems treat words as binary vectors under the classic bag-of-words model; however there is not a notion of semantic similarity between words when describing a document in the resulting feature space. Recent advances in neural networks have shown that continuous word vectors can be learned as a probability distribution over the words of a document [3, 4]. Surprisingly, researchers have found that algebraic operations on this new representation captures semantic regularities in language [5]. For example, $Intel + Pentium - Google$ results in word vectors associated to $\{Search, Android, Phones\}$. We used this deep learning approach to discover the continuous and latent semantic features describing content of documents and fit a linear regression model to approximate user preferences for documents. Our submission to the RecSys Challenge'14 obtained a Root Mean Squared Error (RMSE) of 0.902 and ranked 6th for Task 1. Interestingly enough, our approach provided better vector representations than LDA, LSA, and PCA for modeling the content of book abstracts, which are well-known techniques currently used to implement content-based recommender systems in the recommendation community.

Keywords: deep learning, recommender systems, neural network language models

1 Introduction

Latent representation of text is an important task in context-based recommender systems. Especially cold-start problems impose a need to trust the content to infer accurate recommendations when few (or non-existing) user ratings are provided. Beyond are the n-grams and bag-of-words models to demonstrate text, continuous representations techniques such as Latent Dirichlet Allocation (LDA), Latent Semantic Analysis (LSA), and Principal Component Analysis (PCA) which have been used to describe the content of a document as a probability distribution of latent variables known as topics.

The idea is that a sparse matrix M that characterizes the user preferences (rows) in items (columns) can be factorized into two matrices U and V of joint

latent factor space of dimensionality K. This way the user preference u of item v can be approximated by the dot product $u^T v$. This method is known as *matrix factorization* and has been proved to be effective in the Netflix Prize competition combining better scalability and predictive accuracy than Collaborative Filtering methods [1]. We have followed a similar approach, but have not assume a random initialization for matrix V. Our hypothesis is that the features describing a document can be learned in an unsupervised way considering how words form a document in sentences. This provides a context for each word and thus not every word is independent of each other as in the bag-of-words model.

2 Approach

Our method consists of two steps: a) feature learning, and b) user preference learning. While the second step is the traditional matrix, factorization with stochastic gradient descend, the feature learning step uses a neural network to learn a continuous representation of words according to its context in a sentence. Google has shown Word2Vec as a similar deep-learning approach to model semantic word representations[1], but the task of recommendations on this new representation is still unexplored.

2.1 Feature Learning

We describe a topic representation by learning the word topic proportions with a Recurrent Neural Network (RNN) as in [5]. This architecture is illustrated in Figure 1 and consists of an input layer $w(t)$, hidden layer $s(t)$, output layer $y(t)$, and the corresponding weigh matrices U, V, and W. The hidden layer $s(t)$ provides recurrent connections to $s(t-1)$ forming a short-term memory that models the context of a word in a document. Thus, the input layer consists of the current word $w(t)$ and of the previous values of the hidden layer $s(t-1)$. This layer is projected to a lower dimensional space using matrices U and W and a non-linear activation function with a dimensionality of K.

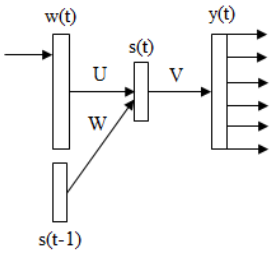


Fig. 1: a) A Recurrent Neural Network (RNN)

[1] https://code.google.com/p/word2vec/

The network is trained by stochastic gradient descend using back propagation algorithm as in [5]. Hidden and output states are computed as follows:

$$s_j(t) = f(\sum_i w_i(t)U_{ji} + \sum_l s_l(t-1)W_{jl})$$
$$y_k(t) = g(\sum_j s_j(t)V_{kj})$$

where $f(x)$ and $g(x)$ are sigmoid and softmax activity functions:

$$f(x) = \frac{1}{1+e^{-x}} \qquad g(x_m) = \frac{e^{x_m}}{\sum_k e^{x_k}}$$

When the RNN is trained, the output layer $y(t)$ contain $P(w_{t+1}|w_t, s(t-1))$, the probability distribution of a word given a history of words (context) stored in time $t-1$. To obtain a per document topic distribution, we match the empirical distribution of words in a document d by using a continuous distribution over these words indexed by a random variable θ.

$$P(d) = \int P(d,\theta) = \int P(\theta^*) \prod_{i=1}^{N} P(w_i|\theta)\, d\theta$$

where N is the number of words in document d and w_i is the i-th word in d. As in[2], we use a Gaussian prior on θ. Note that the term $P(w|\theta)$ is approximated by the output of the RNN, so the problem consists of estimating θ^*. The MAP of θ^* for a document d is $max_{\theta_d^*} P(\theta_d^*) \prod_{i=1}^{N} P(w|\theta_d)$. We find θ_d^* with stochastic gradient descend for each document with a fixed number of iterations.

2.2 User Preference Learning

Given a matrix V learned in the previous step with RNN for each document (cf. Feature Learning), the problem of predicting user preferences in a subset of documents $v \in V$ consists of finding the regression weights u_i for user i such that their inner product approximates the real user preferences y_{u_i} in the training dataset with $y_{u_i} \approx u_i v^T$. After taking the derivative of the squared error and setting it to zero ($\frac{d}{dv}(y_{u_i} - u_i v^T)^2 = 0$), we solve for each user u_i the following equation:

$$u_i = (v^T v)^{-1} v^T y_{u_i}$$

and then predict preference values for new documents within V.

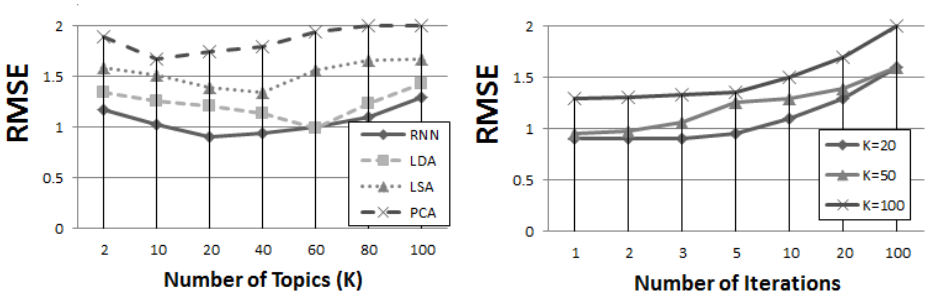
3 Why is the System Innovative?

The popular bag-of-words model used in natural language processing has been outperformed by neural networks language models in recent years [3, 4]. The

novelty of our attempt in the RecSys Challenge strives on the way how features are learned to describe content. The weights of a trained RNN provide high quality word vectors that show semantic relationships in large datasets [5]. This way, words are not only similar or belong to the same topic, but share a similar context. As far as we know, none of the previously proposed architectures has been successfully trained for the problem of recommendation based on content.

4 Results

Given the 8,170 DBpedia URIs provided in the competition, we extract the abstract of each book with SPARQL to obtain a vector of size $(8,170 \times V)$, where V is the number of words in the vocabulary. After stop-removal and removing words with low frequency, we ended up with $V \approx 1,500$. We then trained 99 RNNs with hidden layers ranging from 2 to 100 nodes to cover a broad range in the number of latent features K to describe content in the DBbook dataset. Similarly, we have set the same number of topics for LDA, LSA, and PCA in the experiments (Figure 3 (a)). Initially we had considered gradient descend to evaluate the regression weights during user preference learning, but this approach (with 1, 2, 3 5, 10, 20, and 100 iterations) didn't provide lower Root Mean Squared Error (RMSE) values than using the normal equation for linear regression (Figure 3 (b)), so we used the normal equation as the best (and fastest) approach. When gradient descend was considered, we measured 100 values for the regularization parameter λ in the range of 10^{-6} and 10^{3}, so in total $9,900$ deep learning-based recommenders were implemented.



(a) (b)

Fig. 3: a) Effect of the number of topics b) Effect of the number of iterations

We compare the quality of different content representations in terms of RMSE. When a Recurrent Neural Network (RNN) is well trained, the resulting recommender will be able to linearly combine the latent features learned during training to better approximate the user preferences. Our submission to the RecSys Challenge'14 obtained a RMSE of 0.902 and ranked number 6 for

Task 1 with $K = 20$. Interestingly enough our approach has always provided better vector representations than LDA, LSA, and PCA for modeling the content of book abstracts in our experiments, as shown in Figure 3 (a). Those techniques are well-established methods currently used when implementing content-based recommender systems in the recommendation community.

5 Learned Lessons

During the implementation the recommender system, we come up with the following findings:

- Modeling word contexts provides a semantic relationship between words that improves the latent representation of documents.
- When the matrix V contains enough information to describe the structure in the content of documents, updating U and V with coordinate ascend provides minor improvements. Thus, we can train U with a very small number of iterations if enough effort has been devoted to train V or both steps can be performed independently without loss of RMSE.
- The above approach requires a more aggressive regularization parameter to control overfitting in matrix U. Empirically, large numbers for the regularization parameters λ provide the smallest RMSE with this setting.
- Surprisingly, projecting LDA topics to an orthogonal space with PCA notably improved the prediction results. We believe this is because the linear combination of features and weights provided by the regression algorithms is better suit in a space with low correlation between latent variables.

6 Conclusions

We presented a recommender system that uses the semantic word properties of a type of deep learning algorithm called Recurrent Neural Network. This method provides a lower and less sparse representation of content of text documents. Our results and submission to the RecSys Challenge shows that a RNN provides a lower RMSE values than a latent representation with LDA, PCA, and LSA.

References

1. Robert M. Bell and Yehuda Koren. Lessons from the netflix prize challenge. *SIGKDD Explor. Newsl.*, 9(2):75–79, December 2007.
2. Andrew L Maas and Andrew Y Ng. A probabilistic model for semantic word vectors. In *NIPS*, volume 10, 2010.
3. Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *Computing Research Repository*, 2013.
4. Tomas Mikolov and et al. Distributed representations of words and phrases and their compositionality. In *NIPS*, 2013.
5. Tomas Mikolov, Wen tau Yih, and Geoffrey Zweig. Linguistic regularities in continuous space word representations. In *Conference of the North American Chapter of the Association for Computational Linguistics*, pages 746–751, 2013.