Siamese Neural Network for document similarity and one-shot classification

Prateek Patel

14/06/2017

ME Project Report

Abstract

Applying deep learning techniques in limited data available scenarios is a challenging task. One shot learning recently introduced in literature for images tries to overcome this problem by learning discriminative features which will help to predict about data samples from new classes. In this work we introduce a new document similarity dataset, which will help to learn discriminative features for text classification problem. we present a siamese adaptation of Recurrent Neural Network (RNN) for labeled pairs of documents. We used the paragraph vectors model(doc2vec model) to generate the sentence-embeddings which encodes the underlying meaning expressed in a document and then feed these embeddings to the network to measure the similarity between documents. We then use the powerful discriminative features gained from the trained network to predict new classes. We present the effectiveness of the method with detailed experiment results.

1 Introduction and motivation

Training large networks in limited data available scenarios is a daunting task. Learning good features that generalize across data, learning tasks are hot topics of research in machine learning community. Transfer Learning, Domain Adaption are all trying to overcome this problem in various settings. One-shot learning is a mechanism recently introduced for image classification problems also addresses the same problem of less training data availability. One-shot is formulated under this setting, we have good amount supervised training samples for some classes, and we have to learn a model which can be tuned to predict new classes with very few training samples. The idea was introduced for images. In this work we are trying to extend the idea for natural language text. For training a suitable model, we had to create new dataset with pair of documents and a metric denoting how semantically similar they are. Text documents do not rigidly follow any structure or formal semantic theory making modeling documents to capture the meaning, a challenging task. Active/passive narration, sentence paraphrasing and syntactic variations are some of the other several issues , that a system needs to take into account. Most popular and contemporary approaches model documents as a mixture of latent topics which are learned from the data. But they fail to capture the semantic meaning of documents. So to overcome this drawback, we propose a model based on siamese-network which can capture semantic meaning of the documents and assess the similarity between the documents. Our model is trained on paired examples to learn a highly structured space of sentence-representations that captures rich semantics. Despite its simplicity, our approach exhibits good performance for evaluating similarity between documents. After tuning siamese-network, next is the one-shot learning setting, in which we try to correctly make predictions providing very few examples of new class to the network. Learning from limited examples is easy for humans but difficult for machine learning techniques. Even most effective machine learning techniques fail when forced to make predictions about data for which little supervised information is available. To generalize to these unfamiliar categories without full retraining of the model (which may be either expensive or impossible) is called one-shot learning[3]. In this work, we follow one particularly interesting task in which we try to classify data under the restriction that we may only observe a few examples of some class before making a prediction about a test instance. The model is trained to learn representations such that similar meaning documents have similar representations.

Contributions

The main contributions of this work can be summarized as follows,

- **Data set contribution:** We generated dataset for document similarity which contains Wikipedia article pairs and similarity score between them. We build the tree out of directed cyclic graph structure of Wiki dump and made pair of articles using the relationship stored in Wiki dump. This relationship are in the form of category with sub-category and category with article. The dataset is essential for one-shot learning.

- **Siamese network for measuring similarity between documents:** Before this work, siamese network were used for images and text sequence with small sequence length. We propose LSTM based siamese network for measuring document similarity which works for long sequence of text.

- **one-shot classification :** We illustrated the effectiveness of the model to learn discriminative features for one-shot learning for large text. We also showed that our model automatically acquired features from few examples which generalize successfully to all examples.

Rest of the work is organized as follows, Section 2 gives the explanation for the required deep learning constructs used. In Section 3, we place the work in the context of the relevant related works. Section 4 gives the details of how we created the dataset. Section 5 and 6 give the detailed overview of the siamese model and one-shot classification. Experiments and Results are detailed in Section 7 . We conclude in section 8 by giving some future work directions.

2 Preliminaries and Background

2.1 Recurrent Neural Networks

Recurrent Neural Networks (RNN), especially the Long Short-Term Memory model naturally suits the variable-length inputs like sentences. LSTMs are particularly successful in tasks such as text classification and language translation. RNNs adapt standard feedforward neural networks for sequence data $(x_1, \ldots, x_T)$, where at each $t \in \{1, \ldots, T\}$, it updates to a hidden-state vector h_t which are updated via

$$h_t = sigmoid(Wx_t + Uh_{t-1}) \tag{1}$$

Long Short Term Memory(LSTM), a RNN variant, is capable of learning long range dependencies through its use of memory cell units that can store/access information across lengthy input sequences. They work well on a large variety of problems and are now widely used. LSTMs adapt standard feedforward neural networks for sequence data $(x_1,.., x_T)$, where at each $t \in \{1,..., T\}$, it updates to a hidden-state vector h_t, but these steps also rely on a memory cell containing four components: a memory state c_t, an output gate o_t that determines how the memory state affects other units, an input gate i_t that controls what gets stored in memory and a forget gate f_t that controls what gets omitted from memory based on each new input and the current state. Below are the updates performed at each $t \in \{ 1,...,T \}$ in an LSTM parameterized by weight matrices $W_i, W_f, W_c, W_o, U_i, U_f, U_c, U_o$ and bias-vectors b_i, b_f, b_c, b_o :

$$\begin{aligned}
i_t &= sigmoid(W_i x_t + U_i h_{t-1} + b_i) && (2)\\
f_t &= sigmoid(W_f x_t + U_f h_{t-1} + b_f) && (3)\\
c_t^{'} &= tanh(W_c x_t + U_c h_{t-1} + b_c) && (4)\\
c_t &= i_t \odot c_t^{'} + f_t \odot c_{t-1} && (5)\\
o_t &= sigmoid(W_o x_t + U_o h_{t-1} + b_o) && (6)\\
h_t &= o_t \odot tanh(c_t) && (7)
\end{aligned}$$

Other RNN variants such as the simple gated recurrent unit (GRU)[1] have also been proposed. In extensive empirical analysis[2] compares various proposed variants with LSTM model, in which they showed LSTM outperforms others in most of the cases. In this work, we use a simple adaptation of the LSTMs in the form of siamese-network.

2.2 Paragraph Vectors

We extract the features of documents by training the paragraph vectors: distributed memory model proposed in [12]. Approach for learning paragraph vectors as described in [12] is inspired by the methods for learning the word2vec vectors [13]. Training of the model is done in such a way that paragraph vectors contribute to a prediction task about the next word in the document. The paragraph vectors are initialized randomly. These vectors capture semantics as an indirect result of the prediction task. In this framework (see Figure 1), we map every paragraph to a unique vector which is represented by a column in matrix D. We also mapped every word to a unique vector which is represented by a column in matrix W. To predict the next word in a context, paragraph vectors and word vectors are concatenated or averaged. More formally, given a sequence of training words $w_1, w_2, \ldots w_T$, the objective of the model is to maximize the average log probability

$$\frac{1}{T}\sum_{t=k}^{T-k} logp(w_t|w_{t-k},\dots w_{t+k}) \tag{8}$$

prediction of next word: this task is typically done via a multiclass classification using softmax :

$$p(w_t|w_{t-k},\dots w_{t+k}) = \frac{e^{y_{w_t}}}{\sum_i e^{y_i}} \tag{9}$$

For each output word i, each of y_i is un-normalized log-probability which is computed as :

$$y = b + Uh(w_{tk},\dots,w_{t+k}; W, D) \tag{10}$$

where h is constructed by a concatenation or average of word vectors extracted from W and D. U and b are the softmax parameters.

As mentioned in [12], the paragraph token acts as a memory that remembers the missing current context. These token are of fixed-length and sampled from a sliding window over the paragraph. To train the paragraph and word vectors, Stochastic gradient descent is used. A fixed-length context is sampled from a random paragraph at every step, then error gradient is computed from the network(see Figure 1) and then the gradient is used to update the parameters.

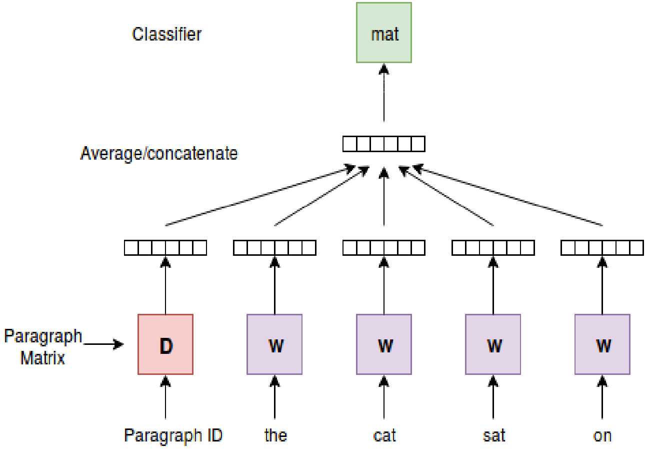

Figure 1: A framework for learning paragraph vector

3 Related work

Statistical topic models[16][17] have been proposed which identify latent topics that in a distributional sense are strongly associated with one another within a given document corpus. Recently, paragraph vector(doc2vec) technique is proposed in [12] which is based on word2vec that learns fixed-length feature representations from variable-length pieces of texts (such as sentences) by using an unsupervised algorithm. Similar to word2vec, it is trained to predict words in the document. [12] shows that paragraph vectors empirically outperform bag-of-words based measures. Because of this reason, in our model we use these paragraph vectors instead of BoW vectors as input to the network. Vector-space representations using latent semantic analysis performs well in semantic similarity evaluation task. But stronger demands are imposed in our task such as learning of semantically structured representation space such that simple metrics suffice to capture document similarity. Siamese neural networks which have been proposed for a number of metric learning tasks in [4] and [5], and is used in [6] for text(sentence similarity). [7] also used Siamese architecture for face verification/detection which utilizes symmetric ConvNets. One-shot learning algorithms have received limited attention by the machine learning community. A one-shot image classification algorithm is developed in [8] and [9] which uses variational Bayesian framework. Transfer learning approaches or domain adaptation techniques are also considered in this field. A generative HMM model for speech primitives combined with a Bayesian inference procedure to recognize new words by unknown speakers is used in [10]. In [11] one-shot learning is addressed in the context of path planning algorithms for robotic actuation. Recently, Siamese Neural Networks are used in [3] for image recognition. They used a convolutional architecture which achieved strong results and exceeded in performance as compared to other deep learning models with near state-of-the-art result on one-shot classification tasks. To the best of our knowledge, this is the first attempt to apply one shot mechanism for large sequence of natural langauge text.

4 Dataset Generation

For learning the discriminative features the siamese model needs to be trained on pair of documents along with the similarity measure. The closest dataset available to this is SICK dataset[18]. But it contained very few samples, and the the length of the text was small to use for any document classification task. So we needed to generate set of pairs of documents with similarity scores. We used Wikipedia category hierarchy structure for this. We will first explain how we did this.

4.1 Hierarchy construction from Wiki dump

Wikipedia is a set of pages that has relationships with one another. We used this relationship to generate scores in training and testing pairs. Every page in Wikipedia has a set of categories. Categories group related articles together. Wikipedia dump is a directed cyclic graph in which categories act as nodes. Every category has sub-categories and only categories at leaf level in wikipedia hierarchy do not have sub-categories. Also, every category has a set of pages. For example, Computer Science category has 17 sub-categories and 29 pages. Some of these sub-categories are Areas of computer science, Computer science conferences, History of Computer Science and some of these pages are Computer Engineering, Computer science in sports, Software etc. We build the graph of categories such as when we move towards leaf level in this graph we get more and more specific categories. For example, Heap is one of the leaf level category i.e it does not have any sub-category. It contains pages only related to Heap data structure (Fibonacci Heap, D-ary Heap, Binary Heap etc.) unlike other categories at upper level which has wide variety of pages. Therefore, we are only interested in article pages of leaf category. Wikipedia provides dumps namely Pages and Categorylinks. **Page:** The page table can be considered the textitcore of the wiki. Each page in wikipedia has an entry here which identifies it by title and contains some essential metadata. It has 34M rows approx that represent all the English Wikipedia pages. **Categorylinks:**This table represents the link between articles and categories, as well as categories and subcategories. We have 65M rows approx in this table. We build the tree of categories taking Computer Science category as root node. Since, Wiki contains huge number of articles and we can generate $O(n^2)$ pairs from n articles, we do not want to accumulate all the articles pages and therefore tree is constructed in such a way that it has maximum depth of five so that we only use limited number of articles. We only take article pages from leaf category for generating our dataset. We say, root category is at Level 0, sub-categories of root category are at Level 1 and so on. Hence, in our case, leaf categories are at Level 4.

4.2 Similarity score

We used LCA(Lowest common ancestor) node for giving similarity scores to pairs of documents. We first make pairs from all the documents at leaf level. Say, each pair has document D_1 and D_2. We find the level of LCA of D_1 and D_2.

$$score = \frac{d}{2^{d-l}} \tag{11}$$

where d is maximum depth of tree and $l = level(LCA(D_1, D_2))$. Score formula is designed in such a way that pairs which contain documents from same category get the highest score and as level of LCA node decreases similarity score also decreases. We normalize score so that we get all score in $[0, 1]$. Also, we want all score values to be uniformly distributed in our dataset, therefore we randomly choose pairs from each score value in such a way that we get approximately equal number of pairs for each score value. We discard all other left out pairs. Finally, we divide the dataset in training and test pair such that both set have equal number of pairs for each score value. We have total $79,703$ pairs in our dataset. We took $70,000$ pairs for training and reserved $9,703$ for testing. We also took only some example documents from this categories *Computer science awards*. We reserve all other articles from *Computer science awards* category for one shot classification experiments. We call this category as one-shot category.

5 LSTM Siamese Network

The first step is to encode the documents using paragraph vectors. We took every document from our corpus and divide it into sentences and generated doc2vec embedding(paragraph vectors) for each sentence of document. We feed these feature vectors directly to our siamese-network. An important advantage of paragraph vectors is that they are learned from unlabeled data. Paragraph vectors also address weaknesses of BoW models. First, they inherit the semantics of the words: an important property of the word vectors. The second advantage of the paragraph vectors is that they take the word order into consideration at least in a small context.

We had embeddings of sentences of every document as output from paragraph vector model. We used them as input to our siamese network here. The proposed model is outlined in Figure 2. There are two networks $LSTM_a$ and $LSTM_b$ which each process one of the document in a given pair. In siamese architectures, tied weights such that $LSTM_a = LSTM_b$ are used. Formally, we consider a supervised learning setting where each training example consists of a pair of sequences $(x_1^{(a)}, \dots, x_{T_a}^{(a)})(x_1^{(b)}, \dots, x_{T_b}^{(b)})$ along with a single label y which represent similarity score of that pair. Note that the sequences of text are not of the same length and can vary from document to document thus our network produces a mapping from a space of variable length sequences into a space of fixed dimensionality. In this case, each $x_i^{(a)}$ denotes the vector representation of a i^{th} sentence from the first document while the $x_j^{(b)}$ denote the j^{th} sentence vectors from the second. Thus, we apply LSTMs with the explicit goal to learn a metric that reflects semantics.

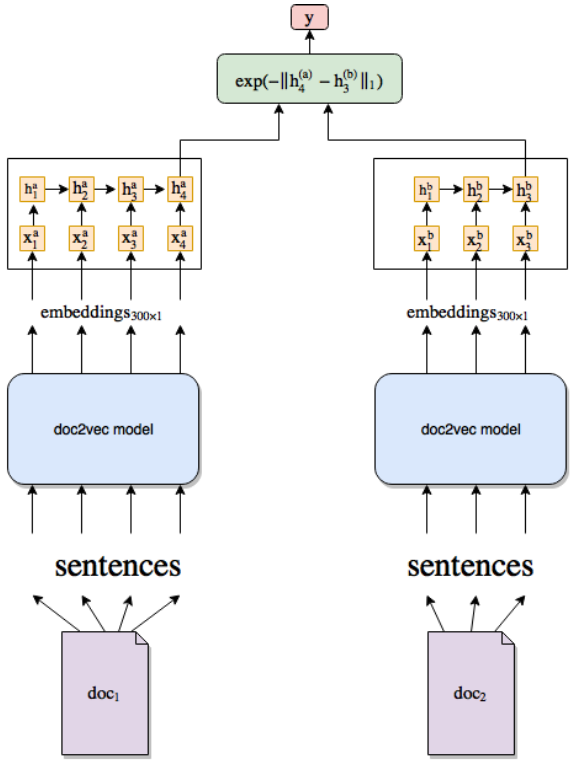

Figure 2: Model includes (a) **doc2vec** with input: sequence of sentences from documents, output: sentence embeddings. (b) **LSTM based siamese network** with input: sentence embeddings, output: similarity score.

LSTM learns a mapping from the space of variable length sequences of d_{in} = 300 dimensional vectors into $\mathbb{R}^{d_{rep}}$ where d_{rep} = 50 . More concretely, each document (represented as a sequence of sentences) $x_1, \ldots, x_T$, is passed to the LSTM, which updates its hidden state at each sequence-index via equations (2)-(7). The final representation of the sentence is encoded by $h_T \in \mathbb{R}^{d_{rep}}$, the last hidden state of the model. For a given pair of documents, our approach applies a pre-defined similarity function $g : \mathbb{R}^{d_{rep}} \times \mathbb{R}^{d_{rep}} \to \mathbb{R}$ to their LSTM-representations. Generally, typical language modeling RNNs are used to predict the next word given the previous text, but here our LSTMs simply function like the encoder. Thus, the error signal backpropagated during training shows how this predicted similarity deviates from ground truth relatedness. We used a simple similarity function,

$$g(h_{T_a}^{(a)}, h_{T_b}^{(b)}) = exp(-\|h_{T_a}^{(a)} - h_{T_b}^{(b)}\|_1) \in [0, 1] \qquad (12)$$

which is also used in [6]. In [7] it is pointed out that using a ℓ_2 rather than ℓ_1 norm in the similarity function can lead to undesirable plateaus in the overall objective function. We experiment with ℓ_2 also but empirically, our results using ℓ_1 are fairly stable and better across various types of similarity function.

6 One-shot Classification

After training the siamese network to master the similarity measurement task, we can use the discriminative potential of our network's learned features to demonstrate one-shot learning. Given a test document x, and C number of categories, we want to classify into one of these C categories. We are also given some other documents $\{x_c\}_{c=1}^{C}$, a set of column vectors representing examples of each of those C categories. We can now query the network using x, x_c as our input for a range of $c = 1, \ldots, C$ and then predict the class corresponding to the maximum similarity. To empirically evaluate one-shot learning performance, 10 documents were taken uniformly at random from each category. Similarity score is now measured between test document and 10 documents of each category. We calculate the average of these scores for each category. Also, similarity of test document with documents of one-shot class(few documents of that class whose all documents were not used for training) is calculated. Test document is now assigned the category which has maximum similarity score. This is repeated for every test document.

7 Experiments and Results

7.1 Visualizing the Paragraph Vectors

Not only we used doc2vec model to generate input(sentence embeddings) for siamese newtwork, we also performed experiments to better understand the behavior of the paragraph vectors. To achieve this, We took all the documents from our training corpus. We removed all stop words and words which are in less than four documents and created a vocabulary of 4,715 words. We trained paragraph vectors on these Wikipedia articles and visualized them using incremental PCA[19] in various Figures. The visualization confirms that articles having the same category are grouped together. **Results:** We took documents from categories : Heaps, Combinatorial Optimization, Computer Algebra System, Graph Algorithms, Sorting Algorithms from wikipedia and visualize them together in following figures.

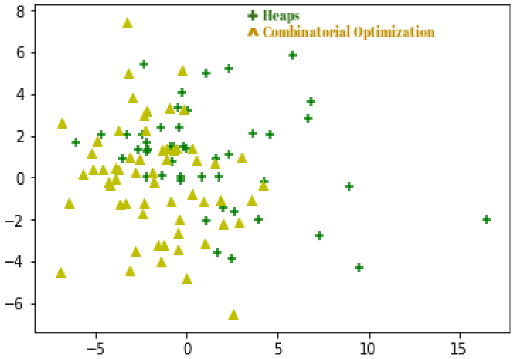

Figure 3: 2D Visualization of documents of Heaps and Combinatorial Optimization category documents.

Figure 3 visualizes clear separation between two categories: Heaps and Combinatorial Optimization.

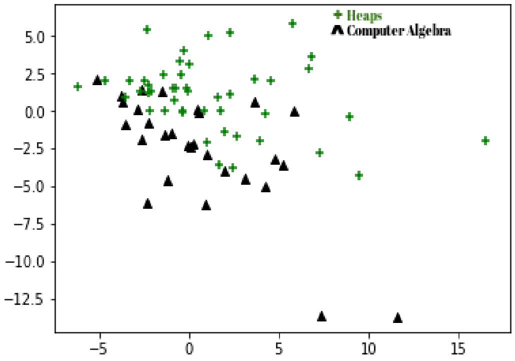

Figure 4: 2D Visualization of documents of Heaps and Computer Algebra System category documents.

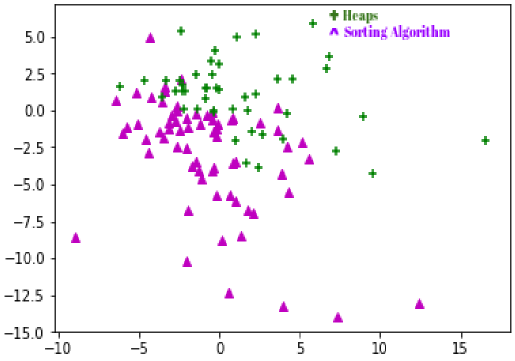

Figure 5: 2D Visualization of documents of Heaps and Sorting Algorithms category documents.

Similarly Figure 4: Heaps and Computer Algebra System and Figure 5: Heaps and Sorting Algorithms clearly visualize two different categories.

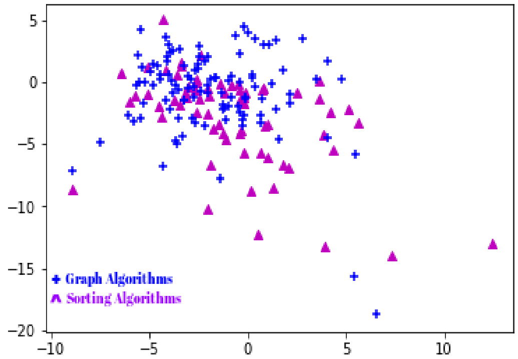

Figure 6: 2D Visualization of documents of Graph Algorithms and Sorting Algorithms category documents.

Since both the categories : Sorting Algorithms and Graph Algorithms comes from the parent category: Algorithm, their documents must be close to each other which are clearly shown in Figure 6.

7.2 Training details of siamese network

Input to siamese network are 300-dimension sentence embeddings which are obtained from doc2vec model. Our LSTM uses 50-dimensional hidden representations h_t. Adadelta method[14] with learning rate 0.0001 is used for optimization of parameters. gradient clipping[15] is used to avoid the exploding gradients problem. We initialize our LSTM weights with small random Gaussian entries since the success of LSTMs depends crucially on their initialization.

7.3 Results on Similarity Tasks

We empirically compare our model with LDA(latent Dirichlet allocation) [16]+ MLP(multi layer perceptron), BoW + MLP, LDA+SVR(Support Vector Regressor), BoW + SVR and our own Siamese network with similarity fuction taken as ℓ_2 norm. Since, we have 35 categories therefore we train LDA model for 35 topics. For, LDA+MLP, absolute difference of LDA vectors of both documents are given as input to 2 layer perceptron with 512 number of neurons in each layer to regress it with similarity score as output of MLP. Similarly in LDA+SVR, absolute difference of LDA vectors of both documents are given as input to SVR with 'rbf' kernel. Similar approach is applied for BoW+(MLP and SVR), only difference is well known dimensionality reduction technique incremental PCA is applied first to BoW vectors to reduce it to $dimension = 300$. Table shows the results of all the methods, and illustrate our model performs better for measuring similarity tasks.

#	Mean Absolute Error		Mean Squared Error	
#	Train	Test	Train	Test
LDA+MLP	0.3036	0.307	0.1203	0.1228
BoW+MLP	**0.0562**	0.2502	**0.0077**	0.1066
LDA+SVR	0.192	0.1936	0.074	0.0742
BoW+SVR	0.3553	0.3542	0.2036	0.2039
Siamese(exp(-L2))	0.152	0.354	0.0569	0.1468
Siamese(exp(-L1))	0.1334	**0.1331**	0.0302	**0.0305**

Table 1: Results comparison of our LSTM based Siamese network with other models

7.4 One-shot Classification Results

We tune the network with only few samples from one-shot category. We first started tuning with only one sample and computed accuracy for this category only. Then, we added few more samples to analyze the network, how when we increase the number of training samples from one-shot category, the accuracy increases. Figure 7 is the plot of number of training samples v/s accuracy on one-shot category. Figure 7 shows that by adding only few number of samples, accuracy increases with a very high rate and it also shows that how effectively network learns powerful discriminative feature from very few samples.

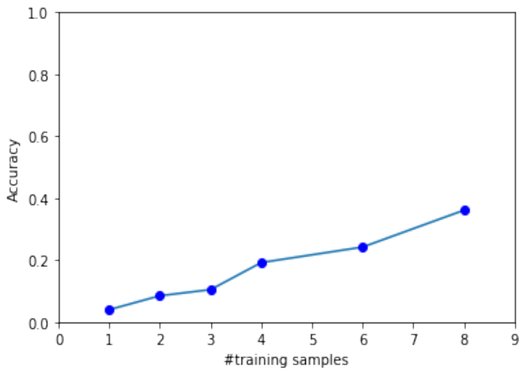

Figure 7: number of samples v/s accuracy on one-shot category

8 Conclusions and Future Work

In this paper, we propose a siamese network for learning semantic representation of documents using document similarity. We compared it with contemporary document modeling approaches and classical machine learning models. Our network achieved a high accuracy on measuring similarity. This work also demonstrates that a simple LSTM is capable of modeling complex semantics if the representations are explicitly guided. We also presented a strategy for performing one-shot classification using our trained model. To the best of our approach this is the first attempt at One-shot classification for text problems. We expected much better results from one-shot classification which we did not get here. But we found that network learned discriminative features from very few examples making our approach a promising one. We believe that one-shot classification results can be improved for text and research in this direction to be undertaken.

References

[1] Chung, Junyoung, et al. “Empirical evaluation of gated recurrent neural networks on sequence modeling.” arXiv preprint arXiv:1412.3555 (2014).

[2] Greff, Klaus, et al. “LSTM: A search space odyssey.” IEEE transactions on neural networks and learning systems (2017).

[3] Koch, Gregory. “Siamese neural networks for one-shot image recognition.” Diss. University of Toronto, 2015.

[4] Yih, Wen-tau, et al. “Learning discriminative projections for text similarity measures.” Proceedings of the Fifteenth Conference on Computational Natural Language Learning. Association for Computational Linguistics, 2011.

[5] Chen, Ke, and Ahmad Salman. “Extracting speaker-specific information with a regularized siamese deep network.” Advances in Neural Information Processing Systems. 2011.

[6] Mueller, Jonas, and Aditya Thyagarajan. “Siamese Recurrent Architectures for Learning Sentence Similarity.” AAAI. 2016.

[7] Chopra, Sumit, Raia Hadsell, and Yann LeCun. “Learning a similarity metric discriminatively, with application to face verification.” Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on. Vol. 1. IEEE, 2005.

[8] Fe-Fei, Li. "A Bayesian approach to unsupervised one-shot learning of object categories." Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on. IEEE, 2003.

[9] Fei-Fei, Li, Rob Fergus, and Pietro Perona. "One-shot learning of object categories." IEEE transactions on pattern analysis and machine intelligence 28.4 (2006): 594-611.

[10] Lake, Brenden M., et al. "One-shot learning of generative speech concepts." CogSci. 2014.

[11] Wu, Di, Fan Zhu, and Ling Shao. "One shot learning gesture recognition from rgbd images." Computer Vision and Pattern Recognition Workshops (CVPRW), 2012 IEEE Computer Society Conference on. IEEE, 2012.

[12] Dai, Andrew M., Christopher Olah, and Quoc V. Le. "Document embedding with paragraph vectors." arXiv preprint arXiv:1507.07998 (2015).

[13] Goldberg, Yoav, and Omer Levy. "word2vec explained: Deriving mikolov et al.'s negative-sampling word-embedding method." arXiv preprint arXiv:1402.3722 (2014).

[14] Zeiler, Matthew D. "ADADELTA: an adaptive learning rate method." arXiv preprint arXiv:1212.5701 (2012).

[15] Pascanu, Razvan, Tomas Mikolov, and Yoshua Bengio. "On the difficulty of training recurrent neural networks." ICML (3) 28 (2013): 1310-1318.

[16] Blei, David M., Andrew Y. Ng, and Michael I. Jordan. "Latent dirichlet allocation." Journal of machine Learning research 3.Jan (2003): 993-1022.

[17] Landauer, Thomas K. Latent semantic analysis. John Wiley Sons, Ltd, 2006.

[18] Marelli, Marco, et al. Semeval-2014 task 1: "Evaluation of compositional distributional semantic models on full sentences through semantic relatedness and textual entailment." SemEval-2014 (2014).

[19] Artac, Matej, Matjaz Jogan, and Ales Leonardis. "Incremental PCA for on-line visual learning and recognition." Pattern Recognition, 2002. Proceedings. 16th International Conference on. Vol. 3. IEEE, 2002.

[20] Gardner, Matt W., and S. R. Dorling. "Artificial neural networks (the multilayer perceptron)a review of applications in the atmospheric sciences." Atmospheric environment 32.14 (1998): 2627-2636.