Journal of Modern Technology and Engineering
Vol.8, No.3, 2023, pp.189-195

BENCHMARKING DYNAMIC CONVOLUTIONAL NEURAL NETWORK WITH LANGUAGE MODELING PRE-TRAINING FOR SENTIMENT AND QUESTION CLASSIFICATION TASKS

Ali Mert Ceylan

Department of Computer Engineering, Ege University, Izmir, Turkiye

Abstract. One-dimensional convolutional models are used for various natural language processing tasks. This study revisits Dynamic Convolutional Neural Network (DCNN) architecture. The study investigates the effect of language modeling pre-training on Wikicorpus on published experiment results for DCNN. Therefore, the reference study integrates a top layer for the language-modeling training into DCNN. Also, benchmarks were reported for the original DCNN compared to the pre-trained language model version. The revisited model was then benchmarked for sentiment classification and question classification tasks. Benchmarks include transfer learning from pre-trained DCNN for language modeling and ground-up trained versions of DCNN on Stanford Sentiment Tree Bank and TREC Question Classification datasets.

Keywords: DCNN, TDNN, k-max pooling, language modeling, sentiment classification, question classification.
AMS Subject Classification: 68T50, 68T01, 68Q32.
Corresponding author: Ali Mert Ceylan, Department of Computer Engineering, Ege University, Izmir, Turkey, e-mail: *91180000031@ogrenci.ege.edu.tr*
Received: 23 March 2023; Revised: 25 September 2023; Accepted: 1 October 2023;
Published: 30 December 2023.

1 Introduction

With the introduction of Time Delay Neural Networks (TDNN) (Waibel et al., 1989), the natural language processing field developed a branch of similar models related to TDNN. TDNN is an ancestor to some of the current popular convolutional models. Its work scheme is based on arranging the data input to the model. Data arranged as time series and weight/kernel of TDNN convolve over inputs corresponding rows. After that, Bengio et al. (2003) and Collobert & Weston (2007) have used the concept of concatenation of word vectors to form a feature vector for sentence representation. In Bengio et al. (2003), the Language Modeling (LM) task has been inspected statistically.

On the other hand, Collobert & Weston (2007) has mainly focused on the Semantic Role Labeling (SRL) task. In Collobert & Weston (2007), they feed a fixed window size of words and classify the label for the middle word. Later, TDNN architecture was used in Collobert & Weston (2008) to learn a series of word vectors instead of time-series data. However, more importantly, they have introduced k-max-pooling operation over the rows of concatenated word vectors. This allowed variable-length sentences to be processed without recurrent states. They have used this model in various tasks, namely, Multi-Task Learning (MTL). It has been covered in Collobert & Weston (2008). In the context of MTL, they have used the same model for SRL, LM, Part of Speech (POS) Tagging, Named Entity Recognition (NER), Chunking, and Synonyms with different tops for each. Amongst all the tasks mentioned, only LM trained alone. For this task, they have used the Wikipedia. To achieve that, they have processed all sentences in a window size of 11, $wsz = 11$. Each window's middle word(Word of Interest) gets replaced with a random word from their vocabulary and presents the model as the wrong sentence, whereas unmodified ones are right. They take advantage of the complete context of a word rather than learning to estimate the occurrence probability of the next word (Collobert & Weston, 2008). Although the approach they have followed in Collobert & Weston (2008) has worked out well, it was not novel; it has been used by Okanohara & Tsujii (2007) before.

Furthermore, similar to Collobert & Weston (2007), Collobert et al. (2011) utilized a similar approach over the feature matrix of word vectors. However, they aim to solve multiple tasks with a single model with shared weights this time. They tried to excel in various tasks while avoiding task-specific engineering in their work (Collobert et al., 2011). Later, Kalchbrenner et al. (2014) introduced a pooling operation similar to the one used in Max-TDNN by Collobert & Weston (2008). They have proposed a heuristic to calculate the number of features selected from each row of the feature matrix; thus, the longer the sentence, the more features to be extracted from it. The heuristic is given in Equation 1.

$$k_l = max\left(k_{top}, \lceil \frac{L-l}{L} s \rceil\right) \tag{1}$$

In Equation 1, the calculation of k value is explained. L is the total number of convolutional layers, l is the number of convolutional layers which the pooling operation is applied after, s is the length of the sentence, k_{top} is the fallback value picked up for the last dynamic k-max pooling layer thus input will be reduced to a fixed size, and can be processed by a fully connected layer. The difference from Max-TDNN is to apply k-max pooling regardless of the output size.

As an extra to the dynamic pooling layer, Kalchbrenner et al. (2014) has applied a b bias value after the dynamic pooling layer. Other than that, Kalchbrenner et al. (2014) has almost identical architecture to Max-TDNN proposed in Collobert & Weston (2008) and achieved relatively higher scores than Max-TDNN and some statistical baselines. However, Kalchbrenner et al. (2014) has shared their interesting insight into the convolutional architecture applied over sentences. They propose that the trained model, with its convolutional and pooling layers, induces an acyclic graph over the input sentence. In this induced graph, the convolution kernel applies a weighted processing mechanism over feature matrices, and dynamic k-max pooling operation drops or selects features. This computational structure continues until the last pooling layer, where the variable length feature matrix is reduced to a fixed length, which is considered the root of the tree (Kalchbrenner et al., 2014).

Later, some diverging studies were introduced. A convolutional-recurrent hybrid model inspired by Kalchbrenner et al. (2014) was presented by Yan et al. (2016) for the Chinese language. It has introduced using the output of the Bidirectional LSTM (Bi-LSTM) (Graves & Schmidhuber, 2005) model as input to a Dynamic Convolutional Neural Network (DCNN) like convolutional neural network with a fixed pooling. However, the model isn't utilized on common datasets considered in this study. In Zhou et al. (2017), it has introduced a model that uses Bi-LSTM outputs as input to DCNN-like one-dimensional convolution to extract higher-level features to be reduced by "max-over-time" pooling as also used by Collobert & Weston (2008). However, they have utilized a sigmoid layer instead of softmax since they aim to extract features. A Long Short-Term Memory (LSTM) (Hochreiter & Urgen Schmidhuber, 1997) decoder with Softmax activation (Bridle, 1989) layer for NER task uses extracted features. In another study, Tian et al. (2019) has utilized DCNN directly with a Recurrent Neural Network (RNN) (Rumelhart & McClelland, 1987) at the end for a generative conversational system. They have utilized k-max pooling to ensure a non-empty feature map and added a dropout layer for regularization. In addition, the model incorporates an intention vector over the attention weights for distributing attention.

In this study, DCNN architecture is used as the base architecture for benchmarking the model for comparison to reported results and measuring the effect of language modeling pre-training on reported results in Kalchbrenner et al. (2014). The main contributions of this study are,

- Since DCNN does not support language modeling by design, we created a top layer for the task as described in Collobert & Weston (2008) and compared it to published language modeling results.
- With language modeling capability added to DCNN, additional benchmarks are also reported for comparison to the original DCNN in Kalchbrenner et al. (2014).
- The model has trained for sentiment and question classification tasks from Kalchbrenner et al. (2014) with and without transfer learning from language modeling to observe the effects of language modeling.

Datasets used for sentiment classification are Stanford Sentiment Tree Bank (Socher et al., 2013) and for question type classification TREC QC (Li & Roth, 2002). Stanford Sentiment Tree Bank contains both sentences and their syntactic tree structures. Therefore, sentences and their corresponding sentiments are considered only in the scope of this study.

TREC QC dataset consists of 6 types of questions and will be used for benchmarking purposes with original DCNN results reported in Kalchbrenner et al. (2014).

The structure of the paper continues with Section 2: Model and experiments, where related convolutional architectures are compared concerning their differences and experiments are documented. Section 3: Conclusion, summarized goals of the study, and stated comments regarding the performance of experiments.

2 Model and Experiments

In this study, the model proposed in Kalchbrenner et al. (2014) is kept as the base model, and any alterations done so far are explained in their respective experiment section. The architecture of the original DCNN is given in Figure 1 as depicted by Kalchbrenner et al. (2014).

We have experimented with DCNN on various datasets that are also used by Max-TDNN of Collobert & Weston (2008) and DCNN of Kalchbrenner et al. (2014). Those datasets are Wikicorpus (Reese et al., 2010) , SSTB (Socher et al., 2013) , Trec QC (Li & Roth, 2002) . We have concluded that Collobert & Weston (2008)'s specification of "entire Wikipedia" was unclear from the point of data pre-processing (stop words and the number of sentences involved weren't mentioned). Thus, we have used Wikicorpus from Reese et al. (2010) for the language modeling task.

To use a language model for transfer learning, word vector size is kept $d = 16$. In addition to replicating the original DCNN benchmarks with transfer learning, the model has also trained ground-up for that specific task to compare the difference between transfer learning. Those benchmarks are labeled as "*ground-up*" in all tables.

Another difference in experiments is caused by the missing information. Due to the original study not publishing batch size, we kept the batch size as 1 for SSTB and Trec QC benchmarks.

2.1 Language Modeling

In reported experiments, explanations of Collobert & Weston (2008) have followed for our version of the Language Modeling task. Because Collobert & Weston (2008) has used Wikipedia to create a custom dataset. Therefore, the number of sentences is not mentioned in the original study. For this reason, our benchmarks use the Wikicorpus dataset Reese et al. (2010). In the Wikicorpus dataset, 1000000 sentences are used for the language modeling training, dataset is split by 80%-20% for train and testing. Similar to the Collobert & Weston (2008), a dictionary of 30000 most common words is used from the Wikicorpus dataset. These sentences are used

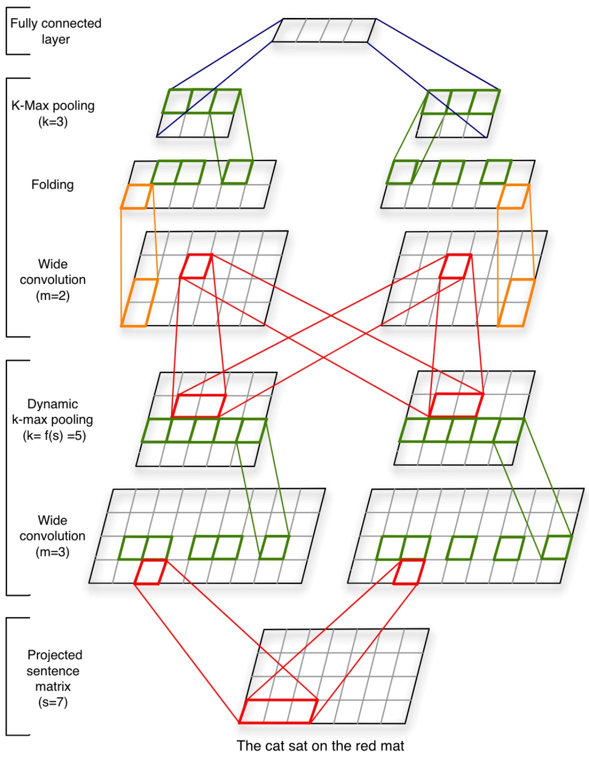
Fully connected layer
K-Max pooling (k=3)
Folding
Wide convolution (m=2)
Dynamic k-max pooling (k= f(s) =5)
Wide convolution (m=3)
Projected sentence matrix (s=7)
The cat sat on the red mat

Figure 1: DCNN architecture. As originally published in Kalchbrenner et al. (2014).

for creating fixed window size sub-sentences as mentioned in Collobert & Weston (2008). For each sentence window, a pseudo-negative instance is created by replacing word-of-interest with a randomly selected word from the lookup table.

Gradient descent optimization has been used with learning rate $lr = 10e{-1}$. The model configuration that Kalchbrenner et al. (2014) has used to classify the SSTB dataset kept. The only architectural modifications are the descriptions of Collobert & Weston (2008) for top layers. Therefore, a 100 node layer and a dropout layer with 0.5 probability are added after the penultimate layer. The model is referred to as LM in all tables. However, word vectors are restricted to $d = 16$ and $k_{top} = 3$ for this study. This approach uses weights for trained word vectors and convolutional layers and trains further for sentiment prediction and question classification tasks. The only exception is that the Semantic Role Labeling task before training wasn't included in the Language Modeling test. Since stand-alone language modeling accuracy was not reported in Collobert & Weston (2008), we couldn't measure the effectiveness of the architectural difference. In addition, we have repeated original experiments as much as possible from the explanations from Kalchbrenner et al. (2014) for sentiment prediction and question classification tasks.

Table 1: Test results for Wikicorpus dataset. Abbreviations "w" and "w/o" correspondingly refer to "with" and "without". LM stands for Language Model pre-training replicated in this work, Max-TDNN (*ground-up*) refers to the original Max-TDNN model from Collobert & Weston (2008). The parameter d refers to word vector size and lr for learning rate.

Model Name, Training	k_{top}	d	lr	Algorithm	Accuracy %
LM (*ground-up*) w/o SRL	3	16	$1e{-1}$	SGD	74.6
Max-TDNN (*published*) w SRL	N/A	15	N/A	SGD	85.6

2.2 Stanford Sentiment Tree Bank

This task uses default train and test splits of the SSTB dataset as also used by Kalchbrenner et al. (2014). The experiment setup replicates DCNN from the Kalchbrenner et al. (2014), with an exception. That exception is the language model training beforehand. However, in our experiments, we couldn't receive the same results reported by Kalchbrenner et al. (2014). Repeated experiments use Xavier initialization (Glorot & Bengio, 2010) for weights and uniform random initialization for word vectors since the extent of the "random" has not been explained in the original study.

Table 2: Test results for Stanford Sentiment Tree Bank dataset. LM stands for Language Model, DCNN (*ground-up*) refers to the original DCNN model from Kalchbrenner et al. (2014), replicated in this work. The parameter d refers to word vector size and lr for learning rate. The term "fine-grained" is the same architecture. However, with the filters having sizes 10 and 7, the top pooling parameter k is 5, and the number of maps is, respectively, 6 and 12, as also reported in Kalchbrenner et al. (2014).

Model Name, Training					
Models	k_{top}	d	lr	Algorithm	Accuracy %
LM (*transfer*) Finegrained	3	16	$1e-2$	SGD	35.2
LM (*transfer*) Binary	3	16	$1e-2$	SGD	71.17
LM (*transfer*) Finegrained	3	16	$1e-2$	AdaGRAD	32.08
LM (*transfer*) Binary	3	16	$1e-2$	AdaGRAD	69.19
LM (*ground-up*) Finegrained	3	16	$1e-2$	SGD	32.31
LM (*ground-up*) Binary	3	16	$1e-2$	SGD	67.44
LM (*ground-up*) Finegrained	3	16	$1e-2$	AdaGRAD	30
LM (*ground-up*) Binary	3	16	$1e-2$	AdaGRAD	67.44
DCNN (*ground-up*) Finegrained	3	48	$1e-2$	SGD	32.22
DCNN (*ground-up*) Finegrained	4	48	$1e-2$	SGD	31.58
DCNN (*ground-up*) Finegrained	3	48	$1e-2$	AdaGRAD	33.17
DCNN (*ground-up*) Finegrained	4	48	$1e-2$	AdaGRAD	34.21
DCNN (*ground-up*) Binary	3	48	$1e-2$	SGD	65.95
DCNN (*ground-up*) Binary	4	48	$1e-2$	SGD	65.51
DCNN (*ground-up*) Binary	3	48	$1e-2$	AdaGRAD	71.44
DCNN (*ground-up*) Binary	4	48	$1e-2$	AdaGRAD	70.06
DCNN (*published*) Finegrained	4	48	N/A	AdaGRAD	48.5
DCNN (*published*) Binary	4	48	N/A	AdaGRAD	86.8
Max-TDNN (*published*) Finegrained	N/A	48	N/A	AdaGRAD	48.5
Max-TDNN (*published*) Binary	N/A	48	N/A	AdaGRAD	86.8

2.3 Trec QC

For this task, two different experiment setups were arranged. The language model that has previously been pre-trained, trained over the Trec QC datasetLi & Roth (2002). The replaced top layer is the only modification different from the language model. We have put a 6-node output layer with softmax activation for this purpose. Also, we kept the dropout layer with a 0.5 probability right before the classification layer concerning the DCNN. We refer to it LM in the Table 3. On the other hand, we have repeated the experiment setup described in the Kalchbrenner et al. (2014) and reported its results in the same table.

Table 3: Test results for Trec Question Classification dataset. LM stands for Language Model, DCNN (*ground-up*) refers to the original DCNN model from Kalchbrenner et al. (2014) replicated in this work. The parameter d refers to word vector size and lr for learning rate. This experiment showed no significant improvement when transfer learning was applied. Also, repeated experiments with the configuration published in Kalchbrenner et al. (2014) did not achieve the published results. The lower value of d with transfer learning shows up to 3% improvement in the accuracy.

Model Name, Training					
Models	k_{top}	d	lr	Algorithm	Accuracy %
LM (*transfer*) Finegrained	3	16	$1e{-2}$	SGD	86.8
LM (*transfer*) Binary	3	16	$1e{-1}$	AdaGRAD	89.2
LM (*transfer*) Finegrained	3	16	$1e{-2}$	AdaGRAD	86.8
LM (*transfer*) Binary	3	16	$1e{-3}$	AdaGRAD	42.4
LM (*ground-up*) Finegrained	3	16	$1e{-2}$	SGD	86
DCNN (*ground-up*) Finegrained	3	32	$1e{-2}$	SGD	86.6
DCNN (*ground-up*) Finegrained	4	32	$1e{-2}$	SGD	86.2
DCNN (*ground-up*) Finegrained	3	32	$1e{-2}$	AdaGRAD	83.8
DCNN (*ground-up*) Finegrained	4	32	$1e{-2}$	AdaGRAD	82.2
DCNN (*published*) Finegrained	4	32	*N/A*	AdaGRAD	93
Max-TDNN (*published*) Finegrained	*N/A*	32	*N/A*	AdaGRAD	84.4

3 Conclusion

In this work, experiments with the model proposed in the Kalchbrenner et al. (2014) repeated as much as possible with the information available from the reference study. We have also aimed to improve results by training DCNN architecture on the language modeling task described in the Collobert & Weston (2008). Due to limited experimentation capabilities, the state-of-the-art results with the language modeling task couldn't be achieved.

Even with the non-perfect language modeling base training, our modified DCNN scored more than the version without language modeling training. Another downside of this study is that the results obtained by the attempt to repeat the original DCNN experiments do not match the ones provided. We don't think it relates to our setup, but how Kalchbrenner et al. (2014) has processed data to perform training using batches.

References

Bengio, Y., Ducharme, R., Vincent, P., & Jauvin, C. A neural probabilistic language model. 3, 1137–1155.

Bridle, J. S. (1989). Training stochastic model recognition algorithms as networks can lead to maximum mutual information estimation of parameters. In *Proceedings of the 2nd International Conference on Neural Information Processing Systems*, NIPS'89, pages 211–217, Cambridge, MA, USA. MIT Press.

Collobert, R., Weston, J. (2007). Fast semantic extraction using a novel neural network architecture. In *ACL*. Association of Computational Linguistics.

Collobert, R., Weston, J. (2008). A unified architecture for natural language processing: deep neural networks with multitask learning, 160–167. ACM.

Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., & Kuksa, P. (2011). Natural language processing (almost) from scratch, 45.

Glorot, X., Bengio, Y. (2010). Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, 249–256. JMLR Workshop and Conference Proceedings.

Graves, A., Schmidhuber, J. (2005). Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *Neural Networks: The Official Journal of the International Neural Network Society*, *18*(5-6), 602–610.

Hochreiter, S. and Urgen Schmidhuber, J. (1997). Long Short-Term Memory. *Neural Computation*, *9*(8), 1735–1780.

Jang, Y., Kim, H. (2020). Document Re-Ranking Model for Machine-Reading and Comprehension. *Applied Sciences-Basel*, *10*(21).

Kalchbrenner, N., Grefenstette, E., & Blunsom, P. (2014). A convolutional neural network for modelling sentences, 655–665.

Li, X., Roth, D. (2002). Learning Question Classifiers. In *COLING 2002: The 19th International Conference on Computational Linguistics.*

Okanohara, D. & Tsujii, J. (2007). A discriminative language model with pseudo-negative samples. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 73–80. Association for Computational Linguistics.

Reese, S., Boleda, G., Cuadros, M., Padro, L., and Rigau, G. (2010). Wikicorpus: A Word-Sense Disambiguated Multilingual Wikipedia Corpus. *In Proceedings of 7th Language Resources and Evaluation Conference*, page 4.

Rumelhart, D. E., McClelland, J. L. (1987). Learning Internal Representations by Error Propagation. In *Parallel Distributed Processing: Explorations in the Microstructure of Cognition: Foundations*, pages 318–362. MIT Press. Conference Name: Parallel Distributed Processing: Explorations in the Microstructure of Cognition: Foundations.

Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C. D., Ng, A. Y., & Potts, C. (2013). Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank, 1631–1642.

Tian, Y., Jia, Y., Li, L., Huang, Z., & Wang, W. (2019). Research on Modeling and Analysis of Generative Conversational System Based on Optimal Joint Structural and Linguistic Model. *SENSORS*, *19*(7).

Waibel, A., Hanazawa, T., Hinton, G., Shikano, K., & Lang, K. J. (1989). Phoneme recognition using time-delay neural networks. *37*(3), 328–339.

Yan, R., Song, Y., Wu, H., & Assoc Comp Machinery (2016). Learning to Respond with Deep Neural Networks for Retrieval-Based Human-Computer Conversation System, 55–64.

Zhou, P., Zheng, S., Xu, J., Qi, Z., Bao, H., & Xu, B. (2017). Joint Extraction of Multiple Relations and Entities by Using a Hybrid Neural Network, 10565, 135–146.