International Journal of Computer Science and Engineering (IJCSE)
ISSN(P): 2278-9960; ISSN(E): 2278-9979
Vol. 8, Issue 5, Aug - Sep 2019; 1-12
© IASET

International Academy of Science, Engineering and Technology
Connecting Researchers; Nurturing Innovations

CONVOLUTIONAL NEURAL NETWORK WITH LOCAL CLASSIFICATION PER NODE APPROACH FOR HIERARCHICAL TEXT CLASSIFICATION

Milan Krendzelak & Frantisekjakab

Research Scholar, Department of Computer Science, Technical University of Kosice, Slovakia, Europe

ABSTRACT

The focus of this paper is to explore the possibilities of using convolutional neural networks for hierarchical local text classification. One of the most researched models for hierarchical text classification is h-LR and h-SVM models, usually, using one of the hierarchical approaches. Moreover, recent research has demonstrated the ability to using CNN baseline model not only for image classification but for text classification, as well. Based on these observations, the authors proposed a novel solution to tackle the hierarchical text classification problem by hierarchical CNN model that reuses features maps to create additional fully-connected layers incorporating hierarchical local classification per node approach. Furthermore, multiple experiments are conducted in order to demonstrate the feasibility of proposed solutions and evaluate achieved performance by empirical comparison with the previously reported performance of h-LR and h-SVM using local per node approach trained with 20Newsgroup training datasets.

KEYWORDS: *Convolutional Neural Networks, Hierarchical Text Classification, Local Classification per Node Approach*

Article History

Received: ***28 Jun 2019*** **|** ***Revised:*** ***08 Jul 2019*** **|** ***Accepted:*** ***23 Jul 2019***

INTRODUCTION

Due to the recent technological progress of enterprise information systems generate, process and collect big data of information. Usually, this will come in the different shape and form of arbitrary uncategorized text such as emails, chat logs, service tickets, feedback surveys, and so on. Such hierarchically organized categories into ataxonomy have become the most frequent way of organizing a large quantity of information. For instance, enterprise customer support, product knowledge databases, help centers and e-learning system will benefit from having anability to automatically classifying new text information hierarchically.

A flat classification treats ataxonomy as set of unique classes that represent leaf nodes in the hierarchy of classes implemented as binary classifier trained for each leaf node to discriminate from the remaining leaf nodes. Due to this fact, it’s not possible to learn parent categories on higher levels because the relationship between the classes is lost [1].

One of the most researched models for hierarchical text classification (HTC) is hierarchical logic regression (h-LR) and hierarchical support vector machine (h-SVM) models. These models are applied, usually, using one of the hierarchical approaches such as global or local approaches, which is based on how the hierarchical relationships are utilized for the training of the selected model [2].

A local per node hierarchical classification approach utilizes a splitting of the hierarchical structure by each node in the tree, except the root node. Each node is implemented as a binary classifier that learns from the subset of the training dataset, represented with only two labels, to recognize whether it is the class of the node or it is not.In order to perform hierarchical classification, it is required to iterate selectively through the entire collection of all trained binary classifiers, and having an ability to stop at desired hierarchical level of the taxonomy, to evaluate a prediction of the classes represented as a tree.

Furthermore, authors propose a novel solution of improved convolutional neural networks architecture, implementing local per level approach used for hierarchical text classification.

Related Work

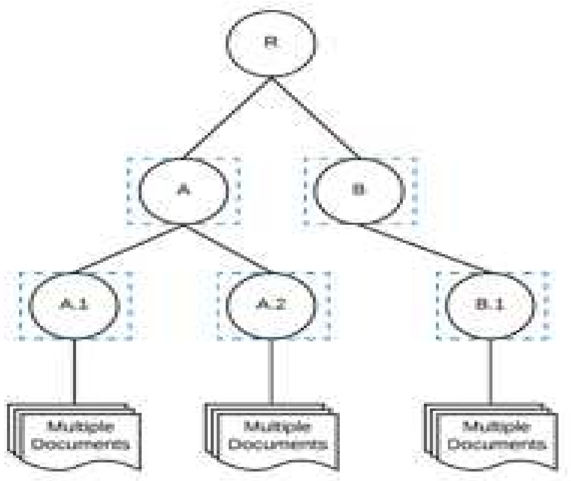

Figure 1: Local Classifier Per Code(LCN)

There are numerous works that focus on solving hierarchical text classification. As an example, well known hierarchical top-down approach with level-based support vector machine models for text classification suggested by Sun and Lim [3]. After a similar route, Sokolov et al.[4] proposed the model for ontology terms forecast by explicitly simulating the construction hierarchy using kernel techniques for structured output. Cerrietal. [5] proposed an approach for hierarchical multi-label text classification that trains a multi-layer perception for each level of the classification hierarchy. Predictions created by a neural network in a given level serve as inputs to the neural network responsible for the forecast within the following level. Their method has been evaluated against several datasets with persuasive results. Within the context of neural networks, Kurata et al. [6] proposed a strategy for initializing neural networks hidden output by considering multi-label co-occurrence. Their method treats a number of those neurons in the final hidden layer as committed neurons for every pattern of tag co-occurrence. There are also several important works that proposed the inclusion of multi-label co-occurrence into loss functions such as pair wise standing loss by Zhang and Zhou [7] and also more recent work by Nam et al.[8], who reported binary cross-entropy can outperform the pair wise ranking reduction by minding rectified linear units (ReLUs) for nonlinearity.

Hierarchical Local Classification per Node Approach

Given, a hierarchy H defined over the output (label) space Y and a set of N training examples composed of pairs $\mathrm{D} = \{(\mathrm{x(i)}, \mathrm{y(i)})\}_{i=1}^{N}$, where $(\mathrm{x(i)}, \mathrm{y(i)}) \in$ XxY, the goal of the hierarchical classification is to learn a mapping function $f: X \in R^{d} \rightarrow Y$that maps the inputs in the input space X to outputs in the output space Y,, such that the function f is accurately able to predict the output y of an input instance x and generalizes well to data that is not observed during the training [9].

Hierarchical Local Classification Per Node (LCN) approach is comprised from a set of binary classifiers, each ϕ_{n} is learned for every node $\mathrm{n} \in \mathbb{N}$, except the root nodeR, in the hierarchy $\mathcal{H}$ as demonstrated in Fig 1. The dashed squares represent the binary classifiers that are assembled into top-down manner execution.

The binary classifier is formulated in Eq. 1.As trying to minimize weight vectors for each label l, where $\lambda > 0$ is the penalty parameter, $\mathcal{L}$ denotes the loss function such as hinge loss or logistic loss and $||\cdot||_2^2$denotes the squared$l_2 - norm$.

$$\sum_{i=1}^{N} \mathcal{L}\,(w_l, x(i), y(i)) + \lambda ||w_l||_2^2 \qquad (1)$$

The training of a binary classifier at the node is performed by feeding the model with positive and negative examples available in the training dataset. Specific to LCN approach, all documents belonging to the n-th node and its descendants are considered as the positive examples and those belonging to the n-th node siblings and their descendants as the negative examples [10].

In order to make a predict unseen new text, binary classifiers for each node are evaluated in the top-down ensemble, first selecting the node at the highest level and recursively traversing the tree toward the leaf nodes, choosing the nodes with best.

```
Result: Repeat recursively until last leaf node is reached
initialization $n := \mathrm{Root}$;
while $n \notin L$ do
        $n := argmax_{q \in C(n)} f_q(\mathrm{x})$;
end
return n;
```

Algorithm 1: Top-Down Ensemble of Hierarchical Predication Score to be Considered for the Evaluation of the Entire Hierarchical Tree Path Prediction as Shown in Algorithm 1

TD ensemble of prediction evaluation has computational advantages where only the subset of classes representing the tree path starting at the tree root, located at the top, down to the leaf node located at the bottom of the hierarchical tree. Additionally, TD ensemble handles well problems with prediction inconsistencies due to the fact that the best node is selected at each hierarchical level of the prediction path.

Hierarchical Evaluation Metrics

In regards to hierarchical classification performance, hierarchical metrics should be considering the hierarchical distance between the true class and predicted class. The idea is to penalize misclassification differently, compared to flat metrics that penalizes each of the misclassified examples equally. Generally speaking, the misclassifications that are closer to the actual class are less penalized than misclassifications that are farther from it with respect to the hierarchy.

The hierarchical metrics include hierarchical-$F_1(hF_1)$ score as defined in Eq. 2, hierarchical precision (hP) as defined in Eq. 3: Hierarchical Recall (hR) As Defined in Eq. 4.

$$hF_1 = \frac{2hPhR}{hP+hR} \tag{2}$$

$$hP = \frac{\sum_{i=1}^{N}|A(\hat{y_i}) \cap A(y_i)|}{\sum_{i=1}^{N}|A(\hat{y_i})|} \tag{3}$$

$$hR = \frac{\sum_{i=1}^{N}|A(\hat{y_i}) \cap A(y_i)|}{\sum_{i=1}^{N}|A(y_i)|} \tag{4}$$

where, $A(\hat{y_i})$ *and* $A(y_i)$ are respectively the set of ancestors of the predicted and true labels including the class itself, but not the root node, and$\delta(\hat{y_i}, y_i)$provides the length of the undirected path between two classes$\hat{y_i}$ and y_i in the graph [11].

Convolutional Neural Networks

Convolution neural networks (CNN) has been adopted from computer vision field [18] and gained much of the respect among researches for the consistent state of the art results, and been proven for its practical application for text classification similar results just with baseline settings of its hyper parameters [12].

Let $x_i \in R^k$be the $k - dimensional$ word vector relevant to the $i - th$ word in the sentence. A sentence of length n, maybe be padded if such required is represented in Eq.6.

$$x_{1:n} = x_1 \oplus x_2 \oplus \dots \oplus x_n \tag{6}$$

where $\oplus$ is the concatenation operator. Furthermore, let $x_{i:i+j}$ refer to the concatenation of words $x_i, x_{i+1}, \dots, x_{i+j}$. By applying a filter $w \in R^{hk}$ in strides will define a convolution operation, which is applied to the selected window of hterms to compute a new feature. Therefore, a feature c_i is produced for each window of words $x_{i:i+h-1}$as depicted by Eq.7.

$$c_i = f(w \cdot x_{i:i+h-1} + b) \tag{7}$$

Let $b \in R$be a bias and $f$some non-linear function, for example, the *hyperbolic* tangent or *relu* functions. Then a filter is used to stride through each possible window in the given sentence$\{x_{1:h}, x_{2:h+1}, \dots, x_{n-h+1:n}\}$to create a feature map, such as$c = [c_1, c_2, c_3, \dots, c_{n-h+1}]$, where$c \in R^{n-h+1}$. Afterward, max-overtime pooling operation is applied over the set of feature map, in order to select a maximum value$C = \max\{c\}$as the feature. The general idea is to be able to capture the most important set of the features for all feature maps [13].

Above process describes how one feature is extracted from one filter. However, the model uses multiple filters, typically with varying window sizes, to extract multiple features. Therefore, these features form next to the last layer that is directly followed by a fully connected soft max layer whose output is the probability distribution over labels.

Feature Selection

One of the CNN advantages compared to other neural networks is that it is designed to extract features automatically from the given text corpora [14]. It does so by applying convolutional layers in a specific predefined manner as it was described in the previous section. Therefore, such extracted feature maps are more efficient than manually constructed, thus these maps are less error-prone and assure a high level of the accuracy in capturing the most important details for given examples text data.

Generally speaking, the convolution layers could be considered as a feature extractor, whose output is then fed into a fully connected layer for a purpose of a simple decision making, such as classification or ranking. Because CNN created local features for each word in the sentence, it is possible to combine or stack features up to produce a global feature vector. In order to some major key points of the CNN ability to create and extract text features are:

- Internally creates features that can be extracted and used as input for another custom-defined internal layers or external models.
- Automatically creates features that do not rely on the handcrafted process. It adapts well to the specifics of the training dataset in a supervised manner.
- Hierarchy of local features are considering during feature creation; therefore, it captures well the context.

Convolution

An$1-\text{D}$ convolution is an operation between a vector of weights m, there $m \in \mathbb{R}^m$and a vector of input sequences s, there $s \in \mathbb{R}^s$. The vector m is the filter of the convolution. Considering s as the input sentence and $s_i \in \mathbb{R}$, is a single feature value associated with the i-th word in the sentence, then a $1-\text{D}$ convolution produces the dot product of the vector m with each m-gram in the sentence s to obtain another sequence c as depicted by Eq.8, which represents two possible types of convolution, such as narrow and wide, depending on the range of the index j.

$$c_j = m^T s_{j-m+1:j} \tag{8}$$

The narrow type of convolution requires that $s \geq m$and yields a sequence $c \in \mathbb{R}^{s-m+1}$ with j ranging from m to$s$.

The wide type of convolution does not constrain $m$nor $s$and yields a sequence $c \in \mathbb{R}^{s+m-1}$where the index j ranges from 1 to$s+m-1$. Values of s_i outside of the range are considered to be zero, where $i < 1$ or $i > s$.

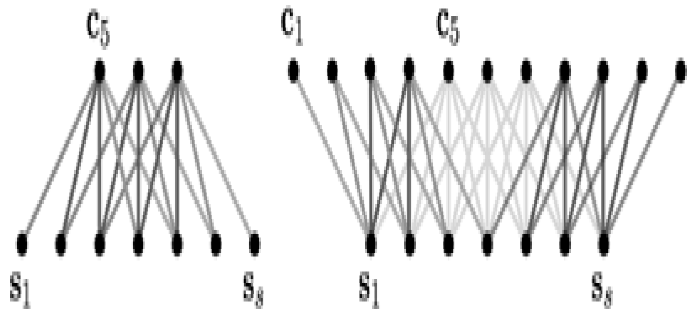

Figure 2: Wide Convolutional Layers.Filters has Size *M*=5

The result of the narrow convolution is a subsequence of the result of the wide convolution. The two types of one dimensional convolution are illustrated in Fig. 2. The trained weights in the filter m correspond to a linguistic feature that learns to recognize a particular class of n-grams. These$n$-grams have size $n \leq m$, where m is the width of the filter. Applying the weights m in a wide convolution has some advantages over applying them in a narrow one. A wide convolution ensures that all weights in the filter reach the entire sentence, including the words at the margins and paddings.

Fully Connected Layer

Fully-connected (FC) layer contains neurons that have full connections to all activations, in the preceding layer, similar to multiple-perception. In order to calculate their activations a matrix multiplication with a bias offset is required. The neurons in the CONV layer are connected only to its local regions in the input and usually many of them do share parameters [15] in contrast with FC layer. The dot products are computed for the neurons in both layers because of the same identity of their functional form.

Therefore, FC and CONV Layers are Fully Convertible from One Layer to Another, Under the Following Conditions:

- FC layers compliments any CONV layer, if both have the same forward function. A large matrix that contains mostly zeros except only for particular blocks because of the local regions with equal weights for many blocks due parameter sharing represents the weight matrix.
- Any FC layer can be converted to a CONV layer if both have the same filter size that corresponds to the size of the input volume

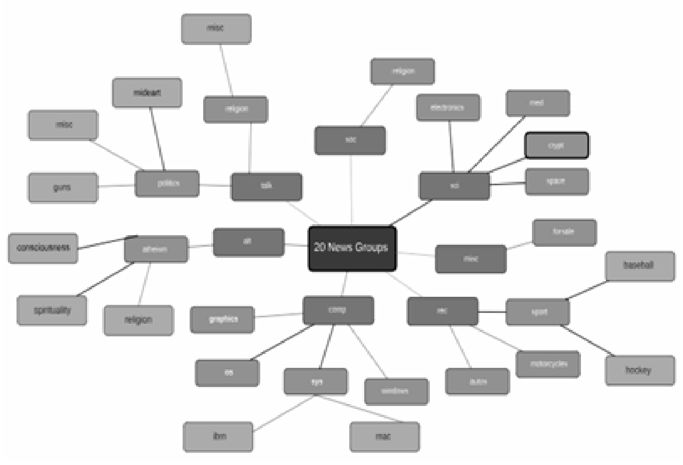

Figure 3: 20Newsgroup Training Dataset Taxonomy.

Experimental Setup

In order to provide a comprehensive assessment of CNN application for hierarchical text classification using local per node approach, we conduct multiple experiments to observe, measure and compare the outcomes.

It is important to emphasize that multiple constraints were taken into consideration and applied throughout the experimentation. These constraints are the following:

- the same training dataset outlined in Fig. 3is used; thus, the results are relatively comparable with each other and with the previously reported application of h-SVN and h-LR models using the same local approach, and prediction is evaluated using a top-down ensemble of binary classifiers.
- For all experiments in this study, we use the 300-dimensional word vectors trained by Mikolov et al. [16] on roughly 100 billion words from Google News. the same variation of CNN baseline model is used; thus, the outcome will not depend on the model specifics but on the effectiveness of the local approach as a strategy itself.

- For all experiments in this study, we use the 300-dimensional word vectors trained by Mikolov et al. on roughly 100 billion words from Google News.

Training Dataset

20Newsgroup training dataset comprises of 11314 documents and more than 30 categories, which are hierarchically organized into a taxonomy with 3 hierarchical levels as illustrated in Fig. 3. The category size corresponds to the number of available training documents per that category, as depicted in Fig. 4.

Original training dataset has to be preprocessed in order to relabel documents according to a new hierarchy derived from the type of selected hierarchical approach. For example, the LCN approach requires that all classes represented as nodes in the hierarchical tree, except the root node, are implemented as binary classifiers and trained accordingly. This will ensure that newly derived and relabeled training dataset provides the optimal collection of examples the machines could learn from.

CNN Word Embeddings

Among a large number of recent researches made on word-based CNN for text classification, one of the aspects is the choice of using pre-trained word2vec or learned word embedding from scratch. In our experiments we use two variants of the CNN baseline model that differ in the initialization of word embedding:

- CNN-rand, which learns the word embedding from scratch during the training
- CNN-stat, which is initialized with pre-trained word2vec word embedding

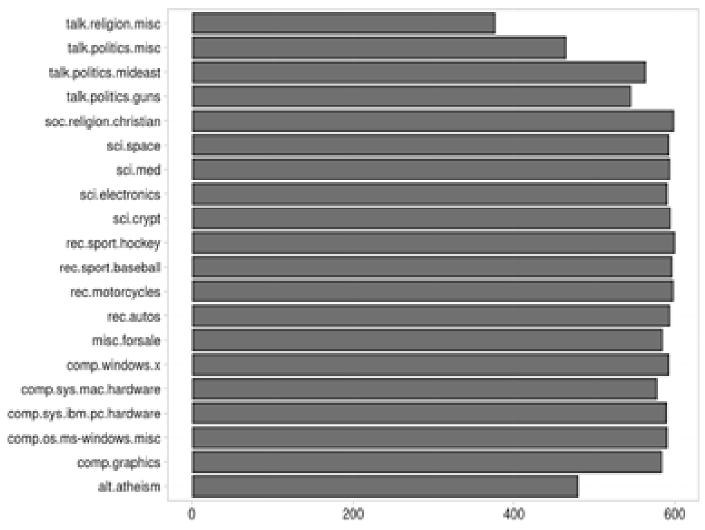

Figure 4: 20 Newsgroup Training Dataset Categories with Size (Documents per Category)

Hyper Parameters Setting

For all conducted experiments with the proposed hierarchical CNN level per node, the values of hyper parameters are predefined and constant. In other words, our intention is to test and evaluate general possible application of CNN for hierarchical text classification, therefore, optimal known values for the hyper parameters:

- Optimizer. CNN model comes with Adam Optimizer initialized with a learning rate of 0.001

- Batch size. CNN model comes with Batch Size of 64, optimal variant for hierarchical text classification.
- Embedding size. By default, embedding size is 128, optimal variant for hierarchical text classification
- A number of filters with filter size. CNN model contains 3 convolutional layers, each layer has 128 different filters. Each filter has a size of 3, 4, 5 strides, respectively.

Hierarchical CNN LCN Model

Reportedly, hierarchical local classification per node approach consists of a set of binary classifiers, each classifier is trained independently for each node selected from the taxonomy based on the type of hierarchical approach.

Typically, a CNN is comprised of the multiple convolutional modules stacked on top of each other in order to perform extraction of the features from the text. The module itself is constructed from a convolutional layer followed by a pooling layer. The dense layers follow after the last convolutional module and could contain more dense layers that perform classification. Therefore, in order to incorporate the LCN approach, it is required to modify the baseline model architecture to add additional dense layers connected with features maps to represent the rest of binary classifiers.

The model that was constructed to perform hierarchical local per node prediction is comprised of the set of 27 classifiers trained for each selected node. The conceptual schema of the model is represented in Fig. 5. The major difference between CNN baseline and current is custom classification module comprises from additional fully-connected layers connected to specific features maps. These additionally added fully-connected layers represent the classifiers hierarchically organized according to local classification per node hierarchical strategy.

For this hierarchical approach, binary one-vs-rest classifiers are trained for each node of the tree which represents the hierarchy, except the root node. In order to make a prediction, each classifier is called separately but the results are evaluated and processed in the top-down ensemble, recursively traversing the hierarchical tree nodes from the root down to the leaf nodes.

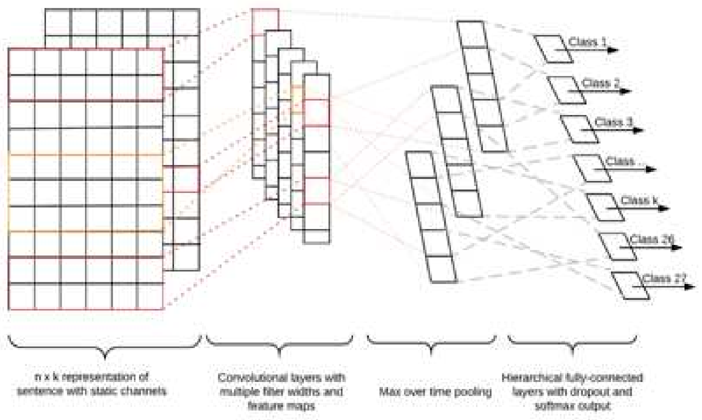
Class 1
Class 2
Class 3
Class ...
Class k
Class 26
Class 27
n x k representation of sentence with static channels
Convolutional layers with multiple filter widths and feature maps
Max over time pooling
Hierarchical fully-connected layers with dropout and softmax output

Figure 5: Hierarchical CNN for Local per Node Classification.

CNN LCN Experimental Performance Evaluation

Collected results from conducted experiments are given either in the form of the graph or a table and only selected metrics with a limited range of values are listed in this thesis. The list of metrics listed is the same for every experiment and contains hF1 which is hierarchical-F1 score, accuracy and error loss score. Moreover, the major focus is on hierarchical-F1 score because this is a core metric used for benchmarking the experiments and comparing the results with the reported performance of h-LR and h-SVM.

We observe that different size of datasets, with an arbitrary selection of the documents per category, do have a direct impact on the accuracy of the classifiers as presented in Table 1.

Table 1: *hF$_1$*Score Comparison of Hierarchical LCN Approach using CNN-rand and CNN-Stat with h-LR and h-SVM

Dataset Size	h-LR	h-SVM	CNN-Rand	CNN-Stat
1000	0.659	0.761	0.632	0.733
2000	0.722	0.812	0.635	0.735
3000	0.767	0.826	0.645	0.745
4000	0.772	0.831	0.685	0.755
5000	0.777	0.839	0.693	0.753
6000	0.789	0.845	0.747	0.794
7000	0.793	0.811	0.778	0.799
8000	0.800	0.832	0.798	0.801
9000	0.801	0.844	0.799	0.809
10500	***0.811***	0.854	0.800	**0.811**
11314	0.804	***0.866***	**0.801**	0.808

Based on the outcome of the classification, it is noticeable, that smaller size of training dataset results in a lower hF1 score. For example, with the training dataset size of 1000 documents, measured performance of CNN-rand is 0.632 and CNN-stat is 0.733.

In contrary, the training dataset with the maximum available size of 11314 needed to train CNN-rand model in order to achieve the highest possible hF1 0.801 score and for CNN-stat achieves the best performance hF1 0.811score with slightly less training dataset size of 10500 documents.

It is important to mention that CNN-stat performance degrades if the model is trained with an entire dataset size of 11314 documents. This is caused mostly probably by over fitting the model with the training dataset.

On comparing the performance of proposed CNN-rand and CNN-stat models with h-LR and h-SVM models using hierarchical local classification per node strategy, the latter models achieve better results than former. The best results achieved by h-LR and h-SVM using LCN approach is 0.811 with the training dataset size of 10500 and 0.866 with the size of 11314, respectively.

The main drawback of top-down prediction evaluation is error propagation caused by misclassifications at higher levels of the hierarchy which cannot be corrected at the lower levels, thus contributes to performance downgrade.

The reason CNN LCN approach was outperformed by h-LR and h-SVM is mostly probably due to the top-down prediction ensemble. To predict the labels of instances, top-down local hierarchical methods proceed by selecting the most relevant node at the topmost level and then recursively selecting the best child node until a leaf category is reached, which is the final predicted label.

CONCLUSIONS

This paper comprises empirical observations of the experiments with proposed modifications of CNN baseline model designed to more closely represent a structured hierarchy class of a given taxonomy. It was demonstrated that such modified default CNN architecture obtains an additional ability to perform the hierarchical type of classification.

The author conducted multiple experiments applying local hierarchical classification per node approach using top-down prediction ensemble. Furthermore, the results of performed experiments with the 20Newsgrouptraining dataset are captured and included in this paper for analysis. This enabled to benchmark with different CNN variants like CNN-rand and CNN-stat with hierarchical local per node approaches and hierarchical Logistic Regression and hierarchical Support Vector Machine model.

According to the results, provided in this paper, authors conclude that CNN-rand model with local per node hierarchical classification approach underperforms h-LR by 2% and h-SVM models by 7.5%. Regarding CNN-stat model with local per node approach, it achieves exactly the same performance (hF1 0.811 score) as h-LR and underperforms h-SVM by 6.35%.

REFERENCES

1. *Zimek, F. Buchwald, E. Frank, S. Kramer, "A study of hierarchical and flat classification of proteins" inIEEE/ACM Trans Computational Biology and Bioinformatics 7(3), pp.563-571, 2010.*
2. *Joulin, A., Grave, E., Bojanowski, P., & Mikolov, T. (2016). Bag of tricks for efficient text classification. arXiv preprint arXiv:1607.01759.*
3. *R. Babbar, I. Partalas, E. Gaussier, M. Amini, "On flat versus hierarchical classification in large-scale taxonomies" in NIPS'13 Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2,pp.1824-1832, 2013.*
4. *Sun, E. Lim, W. Ng, "Performance Measurement Framework for Hierarchical Text Classification" in Journal of the American Society for Information Science and Technology 54(11), pp.1014-1028, September 2003.*
5. *Sokolov, C. Funk, K.Graim, K.Verspoor, A. Ben-Hur, "Combining heterogeneous data sources for accurate functional annotation of proteins" in BMC bioinformatics 14(3): S10, 2013.*
6. *R. Cerri, R. C Barros, A. CPLF De Carvalho, "Hierarchical multi-label classification using local neural networks" in Journal of Computer and System Sciences 80(1), pp.39-56, 2014.*
7. *G. Kurata, B. Xiang, B. Zhou, "Improved neural network-based multi-label classification with better initialization leveraging label cooccurrence" in Proceedings of NAACL-HLT, pp.521-526, 2016.*
8. *M. Zhang, Z. Zhou, "Multilabel neural networks with applications to functional genomics and text categorization" in IEEE transactions on Knowledge and Data Engineering 18(10), pp.1338-1351, 2006.*
9. *J. Nam, J. Kim, E. L.Mencıa, I.Gurevych, J. Furnkranz, "Large-scale multi-label text classification revisiting neural networks" in Joint European Conference on Machine Learning and Knowledge Discovery in Databases, Springer, pp.437-452, 2014.*
10. *J. Demsar, "Statistical Comparisons of Classifiers over Multiple Data" in Journal of Machine Learning Research 7,pp.1-30, 2006.*
11. *S. Dumais, H. Chen, "Hierarchical classification of Web content" in SIGIR '00 Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval", pp.256-263, 2000.*

12. *K. Wang, S. Zhou, Y. He, "Hierarchical classification of real-life documents" in Proceedings of the 2001 SIAM International Conference on Data Mining, 2001.*
13. *Y. Kim, "Convolutional Neural Networks for Sentence Classification" in Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), 2014.*
14. *N. Kalchbrenner, E. Grefenstette, P. Blunsom, "A Convolutional Neural Network for Modeling Sentences", in Proceedings of ACL 2014. 2014.*
15. *R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuglu, P. Kuksa, "Natural Language Processing (Almost) from Scratch" inJournal of Machine Learning Research 12, pp. 2493-2537. 2011.*
16. *Y. Shen, X. He, J. Gao, L. Deng, G. Mesnil, "Learning Semantic Representations Using Convolutional Neural Networks for Web Search" in Proceedings of WWW. 2014.*
17. *T. Mikolov, I. Sutskever, K. Chen, G.CorradoandJ. Dean. "Distributed representations of words and phrases and their compositionality" in Proceedings. Advances in Neural Information Processing Systems 26, pp. 3111-3119, 2013.*