Deep Convolution Neural Network for Extreme Multi-label Text Classification

Francesco Gargiulo, Stefano Silvestri and Mario Ciampi
Institute for High Performance Computing and Networking, ICAR-CNR, Via Pietro Castellino 111 - 80131, Naples, Italy

Keywords: Extreme Multi-label Text Classification, Deep Learning, Deep Convolutional Neural Networks, Word Embeddings.

Abstract: In this paper we present an analysis on the usage of Deep Neural Networks for extreme multi-label and multi-class text classification. We will consider two network models: the first one is formed by a word embeddings (WEs) stage followed by two dense layers, hereinafter *Dense*, and a second model with a convolution stage between the WEs and the dense layers, hereinafter *CNN-Dense*. We will take into account classification problems characterized by different number of labels, ranging from an order of 10 to an order of $30,000$, showing the different performances of the neural networks varying the total label number and the average number of labels for sample, exploiting the hierarchical structure of the label space of the dataset used for experimental assessment. It is worth noting that multi-label classification is an harder problem if compared to multi-class, due to the variable number of labels associated to each sample. We will even investigate on the behaviour of the neural networks as function of the training hyperparameters, analysing the link between them and the dataset complexity. All the result will be evaluated using the PubMed scientific articles collection as test case.

1 INTRODUCTION

The automatic classification of the semantic content of a media (image, text, video) has a paramount importance in many tasks and in different application domains. Usually, the classification result is one or more labels applied to each media. Many techniques have been already proposed in literature, ranging from ontology-based methods (Amato et al., 2014) to machine learning systems (Su et al., 2015), or using hybrid approaches (Alicante et al., 2016a) integrating ontological knowledge and machine learning. The availability of high computational power, in conjunction with the recent advances in the field of the Deep Learning (DL), have led the scientific community to develop Deep Neural Network (DNN) models able to outperform the previous state of the art systems. Among all different media that need to be automatically classified, textual documents have gained a growing importance, due to the application of this problem in various crucial tasks, like Information Retrieval, Question Answering or Natural Language Understanding. In this paper we consider the problem of DL textual documents classification.

In literature Natural Language text classification problem has been split in four different classes. The first one is the *binary classification*, i.e. sentiment analysis, usually applied both to short (i.e. tweets) and long texts (reviews, news, etc.). In this case, only one label belonging to positive or negative class must be assigned to each sample. A more complex problem is the *multi-class classification*, where the single classification label belongs to a set with more than two elements. When the labels belong to a multi-class domain, but differently from the previous case each document could be tagged with a variable number of labels, ranging from one to total classes number, this is a *multi-label classification* problem. Finally, when the latter scenario involves a very huge label set, it can be identified as an additional class, namely the *extreme multi-label text classification* (XMTC) (Liu et al., 2017). In details, XMTC refers to the automatic assignment of the most relevant subset of labels to a text document, but differently from classic multi-label and multi-class classification problems, where the number of the labels is usually in the order of ten, the peculiar aspect of XMTC is that the labels belong to extremely large set, in the order of thousands or even more. This huge label space raises research challenges, such as data sparsity and scalability. With

Gargiulo F., Silvestri S. and Ciampi M.
Deep Convolution Neural Network for Extreme Multi-label Text Classification.
DOI: 10.5220/0006730506410650
In *Proceedings of the 11th International Joint Conference on Biomedical Engineering Systems and Technologies (HEALTHINF 2018)*, pages 641-650
ISBN: 978-989-758-281-3

the availability of Big Data, the XMTC problem has gained a growing attention from the researchers from Machine Learning and DL fields.

Significant advances in multi-label classification methodologies have been made in recent years, thanks to the development of specific machine learning methods, such as tree induction with large-margin partitions of the instance spaces and label-vector embedding in the target space. However, Deep Learning (DL) has not been widely explored yet for this kind of problems. Despite the recent attention of the research community for the identification of NN topologies for XMTC task, the effects of different hyperparameters settings have not still deeply analysed, even if it could bring an improvement to the performance of each network (Sutskever et al., 2013).

In this paper we present an analysis of a Deep Neural Network performances, considering an increasing number of labels, starting from a classic multi-label problem to an XMTC problem. We also investigate on the effects of training hyperparameters (learning rate, momentum, batch size) and their link with the label space size. We consider as case study the PubMed papers repository, a Big Data collection of medical domain scientific papers. Each paper is labelled with a variable number of MeSH (Medical Subject Headings), namely labels from a controlled vocabulary used for indexing articles, manually assigned by human experts. Total MeSH number is about $30,000$ and all MeSHs lie in a hierarchical tree structure. To obtain a variable label set we consider each level of the hierarchical tree, obtaining in this way five different multi-label problems, ranging from 16 to $27,775$ classes, making possible to analyse the behaviour of the DNN with a simple multi-label classification problem and with XMTC problem, in a Big Data text source. The main contribution is the comparison of the effects in term of complexity and accuracy of different Stochastic Gradient Descent (SGD) hyperparameters settings. The proposed approach and the obtained results can be useful to define a baseline for the development of a fine tuned DL XMTC system.

2 RELATED WORKS

The problem of multi-label and multi-class classification involves many different fields. In particular, it is a challenging problem in computer vision and Natural Language Understanding (NLU) areas. In the first case, DL methodologies have been successfully applied in multi-label image classification. For example, in (Zhu et al., 2017) the authors proposed to exploit semantic relations between various labels of an image to improve multi-label image classification task. This improvement has been obtained using a Spatial Regularization Network (SNR) that generates attention maps for all labels and captures the underlying relations between them through learnable convolution. The regularized classification results have been applied to a ResNet-101 (He et al., 2016) network, significantly improving the baseline classification performances.

While DL multi-class and multi-label image classification produces state of the art performances, the same problem applied to text classification has many still open issues. Methodologies derived from Natural Language Processing, like Latent Dirichlet Allocation (Blei et al., 2003) have been used in text classification ((Zhang et al., 2017b), (Pavlinek and Podgorelec, 2017)) with good results in term of accuracy. More recently, alongside classic methodologies, various DNN-based solutions have been proposed. A simple but effective approach is presented in (Wang et al., 2015), where the authors approached the problem of multi-label text classification applied to keywords identification of scientific papers. They used a Word Embeddings (WEs) swallow neural network as an external knowledge base for both keyword extraction and generation, showing promising results. A similar approach based on WEs is described in (Qiang et al., 2017), where an effective model that uses WEs and Markov Random Fields to obtain topic modelling over short texts is proposed. In (Nam et al., 2014) a simple NN approach for large-scale multi-label text classification tasks is proposed, showing the effectiveness of cross entropy error function and demonstrating the usefulness of DL in this setting. The authors proved that simple NN models equipped with advanced techniques such as Rectified Linear Units, dropout, and AdaGrad outperform non NN approaches on six large-scale textual datasets with different characteristics.

A solution to multi-label text classification problem has been proposed even in (Hughes et al., 2017), where it is described the use of a Deep CNN applied to sentences of clinical NL texts, in order to sentence level classification. In this task, the use of the proposed DNN outperforms the WEs based methods. The authors of (Yogatama et al., 2017) characterized the performance of discriminative and generative Long Short Term Memory (LSTM) models for text classification, showing that generative models substantially outperform discriminative models. In (Yan et al., 2017) the authors proposed an innovative Recurrent Neural Network (RNN), namely a LSTM-based multi-label ranking model for document

classification, consisting of two LSTMs used respectively for adaptive data representation process and unified learning-ranking process. The first LSTM is used to learn document representation by incorporating the document labels, while in the latter the order of the documents labels is rearranged in accordance with a semantic tree, in which the semantics are compatible with and appropriate to the sequential learning of LSTM. Connectionist Temporal Classification is performed in rankLSTM to address the error propagation for a variable number of labels in each document. The experiments with document classification conducted on three typical datasets reveal impressive performance. The authors of (Schwenk et al., 2017) explored the use of Very Deep Convolutional Neural Networks (VDCNN) in multi-class text classification, proving the effectiveness of their proposed methodology with large scale training set. In (Wang and Tian, 2016) is analysed the use of Residual Network (ResNet) to improve the performance of LSTM RNN in multi-label text classification task, showing that direct adaptation of ResNet performs well in sequence classification. When combined with the gating mechanism in LSTM, residual learning significantly improve LSTM performances. In (Nigam, 2017), DL models have been applied to the multi-label classification task for assigning ICD-9 labels to textual medical notes, finding that a Recurrent Neural Network (RNN) and a RNN with Long Short-term Memory (LSTM) units show an improvement over the Binary Relevance Logistic Regression model.

It often happens that the classes lie in a hierarchical structure, such as a taxonomy. In (Baker and Korhonen, 2017) is applied a new method for hierarchical multi-label text classification that initializes a neural network final hidden layer such that it leverages label co-occurrence relations, such as hypernymy. The model has been assessed on two hierarchical multi-label text classification tasks in the biomedical domain, using both sentence and document-level classification, showing promising results.

Another common NN topology used for text classification is the one formed by both CNN and RNN. In (Chen et al., 2017) this model has been applied to multi-label text categorization. The proposed approach, through an ensemble application of convolutional and recurrent neural networks, is able to capture both the global and the local textual semantics and to model high-order label correlations, having at the same time a tractable computational complexity. The experimental assessment shows that this approach achieves the state-of-the-art performance when the CNN-RNN model is trained using a large size dataset.

In (Liu et al., 2017) extreme multi-label text classification (XMTC) with DL models is described. In details, a family of new CNN models tailored for multi-label classification is presented. The proposed model is tested with different datasets, producing very good results in all cases. Large scale multi-label text classification is analysed even in (Berger, 2015), where the problem of automatic PubMed articles labelling is approached. They explored how both a CNN and a RNN with a Gated Recurrent Unit (GRU) can independently be used with pre-trained WEs to solve the XMTC problem. On a data set with more than two million documents and $1,000$ potential labels, the authors demonstrated that a GRU provides substantial improvement over a Binary Relevance model with a bag-of-words representation. Similarly, in (Zhang et al., 2017a) a DL method applied to extreme multi-label learning (XML) is presented. The paper aims to better explore the label space by building and modelling an explicit label graph, proposing a practical deep embedding method for XMTC. The main contribution is the ideas of non-linear embedding and modelling label space with graph priors at the same time. Extensive experiments show that this method performs competitively against state-of-the-art systems. A comparison between different machine learning models for XMTC is presented in (Baumel et al., 2017), analysing the performances obtained in this task applied to ICD-9 codes of Electronic Health Records assignment. Support Vector Machines, Continuous Bag of Words, CNNs and GRUs have been considered, demonstrating that the latter two models provide the best results.

3 METHODOLOGY

Different DNNs have been proposed in literature for multi-label text classification. In all of them it is possible to identify the following three groups of layers, hereinafter modules:

- Word Embeddings module;
- Feature Extraction module;
- Classification module.

In this paper we consider this simple but effective paradigm applied to two network topologies: *Dense* and *CNN-Dense*, respectively depicted in the left-side and in the right-side of the Figure 1.

The Dense network is composed by a WEs module, which represents the input text, followed by a classification module composed by two fully connected dense layers, which classify the output. The CNN-Dense network has an additional Feature Extraction module composed by a *1D Convolutional*

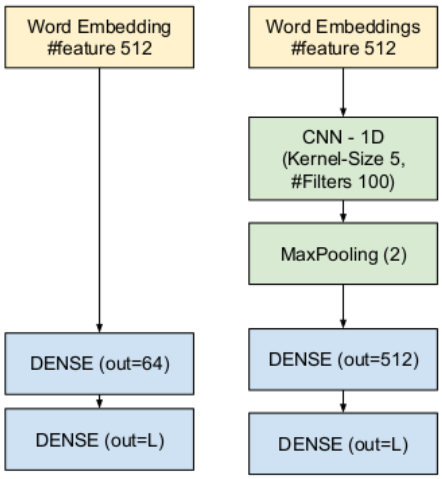

Figure 1: Graphical representation of the DNN models used. The yellow blocks represent the word embeddings module, the green blocks are the *feature extraction* layers and the blue blocks represent the classification layers.

layer followed by a *max pooling* layer which acts as feature extractor, sited between the WEs and classification modules.

In the following of this section we will describe the details of these networks and the methodology used to analyse their performances with increasing labels number. We also briefly explain the details of different Stochastic Gradient Descent (SGD) training hyperparameters considered in our analysis.

3.1 Word Embeddings Module

In this scenario the *Word Embeddings* is mapped conceptually into one layer and so we will refer to him as layer or module indifferently. The Word Embeddings (Mikolov et al., 2013) layer is a shallow neural network that maps the input text into a vector space. Previous experiments (see (Gargiulo et al., 2017b), (Gargiulo et al., 2017a), (Alicante et al., 2016b)) have shown the effectiveness of a Natural Language Processing (NLP) applied to input text before the training phase.

In order to obtain optimal performances, this layer has been pre-trained, applying on input text the following NLP: tokenization, punctuation and symbols removing, stop-words filtering, lemmatization. We used in WEs training the skip-gram algorithm with hierarchical softmax, due to the better performances observed with the dataset used in the experimental assessment, if compared with continuous bag of words and negative sampling.

The output of this layer is a dictionary of word vectors, containing a vector for each word from the training set. The word vectors will be represent the whole text formed by the words in title and abstract of each PubMed document.

3.2 Feature Extraction Module

The *Feature Extraction Module* is used only in the CNN-Dense network. It consists of two cascading layers, a 1D Convolution Neural Network (1D-CNN) followed by a Max Pooling (MP). The 1D-CNN layer takes as input the WEs vectors corresponding to each word of title abstract of all papers from PubMed dataset. The layer is formed by 100 filters with kernel size equals to 5 striding of one position at time; the activation function is a Rectified Linear Unit (ReLU).

The convolution input needs a fixed input size, but the total word of each sample is not constant. To overcome this problem, the maximum number of words has been chosen equals to 500, considering the average word number of dataset equals to 127 after the NLP preprocessing (see Section 4.1). In case of shorter word number, a zero padding is applied, while in case of samples with more than 500 words, we discard the last words of the text.

The 1D-CNN is followed by a MP layer of size 2. The result of this module is the automatic extraction of the features, which will be used into the next classification layer.

3.3 Classification Module

The classification module is composed by two dense fully connected layers: the first one has 64 and 512 outputs respectively for Dense and CNN-Dense network topologies, while the second and final layer of the whole DNN has an output number equals to the number of classes L to be predicted and so it varies depending on the different label number considered (see section 4.1).

3.3.1 Loss Function

Following (Nam et al., 2014) results, the chosen loss function is the sigmoid cross entropy 1:

$$loss(x,y) = -\sum_{l \in L} \left[\left(y_l \cdot \log \frac{1}{1+\exp(-x_l)} \right) + \right. \\ \left. + \left((1-y_l) \cdot \log \frac{\exp(-x_l)}{1+\exp(-x_l)} \right) \right] \quad (1)$$

where y_l and x_l are respectively the prediction and the target for each label $l \in L$. The sigmoid cross entropy loss function optimizes a multi-label one-versus-all loss based on max-entropy and then is well suited for multi-label problems.

3.3.2 SGD Hyperparameters

A key aspect in DNN training is the optimal hyperparameters setting, but despite its crucial importance, this is a very difficult task (Ilievski et al., 2017). Some hyperparameters are directly related to the the network structure and topology, i.e. the number of hidden layers, the number of hidden units and the choice of activation function. Other hyperparameters influence the training phase, because they are involved in Stochastic Gradient Descent (*SDG*) function, during the update of the parameters of the network.

The *SGD* algorithm updates the parameters θ of the objective $J(\theta)$ following the equation 2:

$$\theta = \theta - l_r \nabla_\theta J(\theta, x_i, y_i) \tag{2}$$

where x_i, y_i is a sample/label pair from the training set and l_r is the learning rate. Each parameter update in *SGD* is usually computed with a minibatch and not a single example. The previous equation 2 shows that the l_r and the batch size will directly influence the results of the DNN training, being the *SGD* a function of both them. In addition to that, due to the *SGD* iterative nature, another important hyperparameter is the number of iterations.

SGD can not solve easily ravines, namely the areas where the function surface curves much more steeply in one dimension than in another (Sutton, 1986). This happens around local optima and causes oscillations of the *SGD* and its very slow convergence. Momentum (Polyak, 1964) is a method to accelerate the function along the shallow ravine. The momentum update is given by the following equation 3:

$$\begin{aligned} v_t &= \mu v_{t-1} + l_r \nabla_\theta J(\theta, x_i, y_i) \\ \theta &= \theta - v_t \end{aligned} \tag{3}$$

where v_t is the *velocity* at iteration t and μ is the momentum coefficient. The effects of the modification of *SGD* equation with momentum can be explained in analogy with a ball pushed down a hill. The ball accumulates momentum, becoming faster and able to reach the top of the next hill that it founds on its way. The same thing happens with parameters update: the momentum term increases updates for dimensions whose gradient points in the same directions and reduces updates for dimensions whose gradient changes directions. In this way, the SGD can gain faster convergence and reduce its oscillations.

A further improvement to momentum SGD has been contributed by Nesterov with accelerated gradient (Nesterov, 1983), which is equal to:

$$\begin{aligned} v_t &= \mu v_{t-1} + l_r \nabla_\theta J(\theta - \mu v_{t-1}, x_i, y_i) \\ \theta &= \theta - v_t \end{aligned} \tag{4}$$

The term $\theta - \mu v_{t-1}$ in equation4 approximates the next parameters update (only the gradient calculation is missing, but in case of small differences this approximation is good). Thus, the gradient is calculated not to current parameters θ like in previous equation 3, but to an approximation of future position. The effects of this modification result in an increased responsiveness, because the velocity is corrected in order to take into account the next position. In this way, momentum increases or decreases adjusting itself to the future variations, helping the convergence of *SGD*.

The proposed methodology aims to analyse the effects of above described *SGD* hyperparameters (learning rate l_r, batch size, momentum μ and use of Nesterov method), examining their link with label set size in an XTMC problem trained with a Big Data source, as shown in next section 4.

4 EXPERIMENTAL ASSESSMENT

In this section we first describe the details and the features of the datasets used in the experimental assessment. Then, we report the parameters used in Word Embeddings layer to obtain pre-trained WEs and the evaluation metrics considered. Finally, we present the obtained results, considering an increasing labels number and showing the effects of different learning rate and momentum values on the performance of the DNNs.

4.1 Dataset Description

The experimental assessments has been performed using the PubMed papers collection[1]. PubMed is a free search engine maintained by US National Library of Medicine and specialized for medical and biological scientific articles. The Big Data architecture described in (Gargiulo et al., 2017a) has been used to extract from PubMed repository a dataset formed by a total number of $11,150,090$ papers. Only the titles and the abstracts of each article have been considered, in addition to the corresponding labels, named *MeSH* (Medical Subject Heading). Each document from the dataset is labelled with a variable number of classes, which belong to a large set of $27,755$ different MeSHs, organized in a hierarchical structure[2]. The same MeSH can be located in one or

[1] https://www.ncbi.nlm.nih.gov/pubmed/

[2] The MeSH hierarchical structure can be browsed at https://meshb.nlm.nih.gov/treeView

more branches of the tree. All these features identify a multi-class multi-label problem, as described in section 1. Starting from this dataset, we created five different training sets with an increasing number of classes. The next table 1 summarizes the text features of the original dataset.

Table 1: Overall Statistics of PubMed Papers Collection: N is the total document number, $E(Words)$ is the average number of words per document in the considered dataset, after the NLP.

Dataset	N	$E(Words)$
PubMed	11,150,090	127

In order to obtain different label sets, we process the labels of each document, exploiting the MeSH hierarchical structure and creating five different sets, with respectively 16, 116, 676, 6,339, and 27,755 classes. The first one is formed by the 16 MeSH classes corresponding to the root level of the MeSH hierarchical structure, which indexes the main categories. In this case, we substituted all labels with their corresponding root class label, obtaining the first training set. In the same way, we then considered the MeSHs respectively from the first, the second and the third level after the root of the tree structure, and like the first case, we substituted each label of the original dataset with its corresponding parent of first, second and third level label, obtaining respectively 116, 676 and 6,339 labels. At least, we considered the original dataset with MeSHs from all levels, formed by 27,755 classes. As we can see, the first dataset belongs to a multi-label problem, the second dataset is a simplified version of an XMTC problem, while the last three datasets can be properly considered as XMTC problems.

The text from the original dataset has been preprocessed using NLP techniques briefly described in previous section 3 (see (Gargiulo et al., 2017b) for further details), lower-casing, lemmatizing, removing punctuation and stop words. The preprocessed text has been used to train the WEs layers, using the parameters described in next subsection 4.2.

The obtained datasets have been split in training sets and test sets, randomly selecting the 99% of the samples for the training set and the remaining 1% for the test set; the latter has been further divided in ten smaller test sets, each one formed by approximately 2,500 samples. The following table 2 summarizes the main features of the obtained datasets in terms of class numbers, document numbers, average number of labels per document, average number of documents per label.

The multi-label PubMed article classification problem is one of the tasks of BioAsq[3], a research challenge on biomedical semantic indexing and question answering. The experimental results obtained in this paper can add further details to latest BioAsq results (Nentidis et al., 2017), helping the research community to solve this hard XMTC problem.

Table 2: Dataset statistics as function of deeper level of MeSH hierarchy selected. In the table L is the total class number, L^* is the average number of labels per document and L^o is the average number of documents per label.

$MeSHDepth$	L	L^*	L^o
0	16	5.88	4,100,528.63
1	116	8.57	824,077.34
2	676	10.17	167,800.57
3	6,339	11.12	19,557.78
All	27,755	12.91	5,188.15

4.2 Word Embeddings Parameters

The Word Embeddings (WEs) layer is a shallow neural network able to produce a vector representation of the words of the training set. The obtained vector space is used for all experiments with different datasets, because only label number changes between various training sets, while the text from title and abstract of the papers remains the same. Thus, the WEs layers is the same in all tested DNNs.

In our experimental assessment, the WEs has been pre-trained using Gensim framework (Řehůřek and Sojka, 2010), an effective Python implementation for WEs training. We used skip-gram algorithm with hierarchical softmax (Mikolov et al., 2013), setting a vector size equals to 512, a window size equals to 5, discarding the words that appears only one time in the training set and setting a threshold for higher-frequency words to be randomly downsampled equals to 0.0001.

4.3 Evaluation Metrics

To evaluate the performance of the models we use the commonly used F-measure metric, which is equal to the harmonic mean of recall (ρ) and precision (π). ρ and π (Özgür et al., 2005) are defined as follows:

$$\pi_i = \frac{TP_i}{TP_i + FP_i}, \qquad \rho_i = \frac{TP_i}{TP_i + FN_i} \tag{5}$$

where TP_i (True Positive) is the number of documents assigned correctly to class i; FP_i (False Positive) is the number of documents that do no belong

[3]http://bioasq.org/

to class i but are assigned to class i incorrectly by the classifier; and FN_i (False Negative) is the number of documents that are not assigned to class i by the classifier but which actually belong to class i.

In this paper we consider the micro-averaging: in this case the F-measure is computed globally over all category decisions and ρ and π are obtained by summing over all individual decisions:

$$\pi^{micro} = \frac{TP}{TP+FP} = \frac{\sum_{i=1}^{L} TP_i}{\sum_{i=1}^{L}(TP_i+FP_i)} \quad (6)$$

$$\rho^{micro} = \frac{TP}{TP+FN} = \frac{\sum_{i=1}^{L} TP_i}{\sum_{i=1}^{L}(TP_i+FN_i)} \quad (7)$$

$$F_1^{micro} = \frac{2\pi\rho}{\pi+\rho} \quad (8)$$

The F_1^{micro} values are in the interval $(0;1)$ and larger values correspond to better performances; in literature these values are usually expressed as a percentage, as we did in the following.

4.4 Experimental Results and Discussion

We evaluated the performances of the DNNs described in section 3, considering an increasing label number and analysing the effects of different *SGD* hyperparameters settings in terms of π^{micro}, ρ^{micro} and F_1^{micro} measures.

To give an idea of the models complexity, the next table 3 reports the number of parameters of the networks in function of the *MeSH Depth* and consequently of the number of classes involved.

Table 3: Model parameters as function of $MeshDepth$ and consequently of the labels number (L).

MeSHDepth	*L*	*#Parameters*	
		Dense	*CNN-Dense*
0	16	16,385,104	13,263,620
1	116	16,391,604	13,314,920
2	676	16,428,04	13,602,200
3	6,339	16,796,099	16,597,319
All	27,755	18,188,139	27,493,727

As described above, the label set size is increased exploiting the MeSH hierarchical structure. The corresponding DNNs parameters are influenced only in the last dense layer, which is directly related to the classes number. For this reason, only the last two XMTC problems, corresponding to MeSH depth 3 and *all*, show a significantly larger parameters number, if compared with the first three cases. This effect is more evident on the *CNN-Dense* models because the first dense layer has 512 units respect of the 64 of the *Dense* topology. The different number of the chosen hidden units (first dense layer) between the two topologies depends on the number of elements in input to the first dense layer that, for the CNN-Dense scenario, is considerably reduced by the Feature Extraction Module (CNN-MaxPooling).

We used the Nesterov correction (see section 3.3.2) in all cases, because our experiments confirmed that it produces a slight enhancements of about 1% in all values. Then, a first result is that in XMTC problem trained with Big Data the Nesterov momentum provides a little performances boost.

The details of the results of the experiments are summarized in Table 4 and Table 5, where the performances are reported in terms π^{micro}, ρ^{micro} and F_1^{micro} (see Section 4.3), for different $StepsPerEpoch$, *Learning Rate* l_r and *Momentum* values. The $StepsPerEpoch$ value is directly related to $MiniBatchSize$, being the latter one equals to the total samples number divided by the Steps Per Epoch.

$$MiniBatchSize = \frac{\#TotalSamples}{StepsPerEpoch} \quad (9)$$

The obtained results show that the best network topology between the two analysed is the *CNN-Dense*, demonstrating the need of the feature extraction module. It is worth noting that in all considered cases the best results are obtained using a $StepsPerEpoch$ value equals to $10,000$, confirming the usefulness of a smaller MiniBatch size with a Big Data training set.

It is interesting to analyse the hyperparameters values selected to achieve the best performances given the Depth Level, considering at the same time the dataset characteristics (see Table 2). When the problem is a simple multi-label for the $0-Level$ with 16 classes, the best results are obtained using a slower learning rate equals to 0.01 and a momentum equal to 0, which do not increases the convergence speed of the *SGD*. On the other hand, increasing the number of labels involved considering an higher Depth-Level corresponding to real XMTC problem, the results show a different behaviour of the performance in function of the hyperparameters. In fact, in the latter cases, the learning rate speed that achieves the best performances is equal to 0.1. In addition to that, the use of smaller batch size in conjunction with an higher momentum value equals to 0.5, considerably increases XMLT performances.

The overall results show that there is a significant performances boost (resulting in some cases in a F_1^{micro} improvement of more than 10%) with the use

Table 4: F_1^{micro}, π^{micro} and ρ^{micro} obtained using ***Dense*** network topology. The values are evaluated considering the datasets depicted in the Table 2 from which a training set and ten different test sets was extracted, the results are obtained as an average of the ones obtained on the ten test sets. The table shows the performance behaviour considering 100 epochs, two values of $StepsPerEpoch$, two learning rate (l_r) values and, for each of them, two momentum (μ) values. The best values per row are highlighted in bold.

$StepsPerEpoch = 1,000$												
Depth	$l_r = 0.01$						$l_r = 0.1$					
	$\mu = 0$			$\mu = 0.5$			$\mu = 0$			$\mu = 0.5$		
	F_1^{micro}	π^{micro}	ρ^{micro}	F_1^{micro}	π^{micro}	ρ^{micro}	F_1^{micro}	π^{micro}	ρ^{micro}	F_1^{micro}	π^{micro}	ρ^{micro}
0	**82.97%**	**82.59%**	83.35%	82.92%	82.15%	**83.70%**	81.66%	81.65%	81.67%	79.52%	77.98%	81.13%
1	56.48%	**74.70%**	45.40%	59.82%	74.04%	50.18%	**63.77%**	72.69%	56.80%	63.47%	70.98%	**57.40%**
2	33.71%	73.64%	21.86%	36.74%	73.35%	24.51%	44.94%	70.40%	33.00%	**48.05%**	69.68%	**36.67%**
3	13.86%	**84.19%**	7.55%	13.86%	**84.19%**	7.55%	27.29%	72.41%	16.81%	**28.96%**	70.79%	**18.20%**
All	9.21%	63.80%	4.96%	9.21%	63.80%	4.96%	9.21%	63.80%	4.96%	9.21%	63.80%	4.96%
$StepsPerEpoch = 10,000$												
Depth	$l_r = 0.01$						$l_r = 0.1$					
	$\mu = 0$			$\mu = 0.5$			$\mu = 0$			$\mu = 0.5$		
	F_1^{micro}	π^{micro}	ρ^{micro}	F_1^{micro}	π^{micro}	ρ^{micro}	F_1^{micro}	π^{micro}	ρ^{micro}	F_1^{micro}	π^{micro}	ρ^{micro}
0	**83.78%**	**84.79%**	**82.81%**	83.51%	84.61%	82.44%	81.16%	82.30%	80.06%	78.07%	77.02%	79.15%
1	65.40%	**76.58%**	57.06%	**66.46%**	76.09%	59.00%	66.00%	74.68%	**59.13%**	64.35%	72.53%	57.83%
2	44.12%	74.53%	31.33%	47.28%	74.16%	34.70%	52.52%	**73.82%**	40.76%	**53.62%**	73.64%	**42.15%**
3	25.14%	75.52%	15.08%	26.43%	76.38%	15.98%	31.28%	**77.88%**	19.57%	**34.81%**	77.02%	**22.49%**
All	9.21%	63.80%	4.96%	9.21%	63.80%	4.96%	18.57%	72.95%	10.64%	**20.15**	**73.54%**	**11.68%**

Table 5: F_1^{micro}, π^{micro} and ρ^{micro} obtained using ***CNN-Dense*** network topology. The values are evaluated considering the datasets depicted in the Table 2 from which a training set and ten different test sets was extracted, the results are obtained as an average of the ones obtained on the ten test sets. The table shows the performance behaviour considering 100 epochs, two values of $StepsPerEpoch$, two learning rate (l_r) values and, for each of them, two momentum (μ) values. The best values per row are highlighted in bold.

$StepsPerEpoch = 1,000$												
Depth	$l_r = 0.01$						$l_r = 0.1$					
	$\mu = 0$			$\mu = 0.5$			$\mu = 0$			$\mu = 0.5$		
	F_1^{micro}	π^{micro}	ρ^{micro}	F_1^{micro}	π^{micro}	ρ^{micro}	F_1^{micro}	π^{micro}	ρ^{micro}	F_1^{micro}	π^{micro}	ρ^{micro}
0	83.16%	**83.55%**	82.78%	**83.28%**	83.52%	**83.03%**	82.36%	82.15%	82.58%	81.33%	81.18%	81.47%
1	62.57%	73.05%	54.17%	64.49%	73.48%	57.46%	**65.66%**	73.43%	**59.38%**	64.98%	**73.88%**	58.00%
2	36.84%	66.81%	25.83%	40.87%	**69.25%**	28.99%	51.58%	69.04%	41.16%	**52.89%**	69.14%	**42.83%**
3	13.86%	84.19%	7.55%	13.86%	84.19%	7.55%	14.12%	**84.26%**	7.71%	**28.94%**	69.27%	**18.29%**
All	9.21%	63.80%	4.96%	9.21%	63.80%	4.96%	9.21%	63.80%	4.96%	0.01%	100.00%	0.00%
$StepsPerEpoch = 10,000$												
Depth	$l_r = 0.01$						$l_r = 0.1$					
	$\mu = 0$			$\mu = 0.5$			$\mu = 0$			$\mu = 0.5$		
	F_1^{micro}	π^{micro}	ρ^{micro}	F_1^{micro}	π^{micro}	ρ^{micro}	F_1^{micro}	π^{micro}	ρ^{micro}	F_1^{micro}	π^{micro}	ρ^{micro}
0	**84.33%**	**85.37%**	83.30%	84.17%	85.20%	**83.33%**	82.30%	83.07%	81.55%	80.79%	81.51%	80.09%
1	68.56%	77.19%	61.67%	**69.02%**	**77.33%**	62.32%	68.85%	76.62%	**62.51%**	67.56%	75.39%	61.21%
2	52.37%	74.57%	40.35%	54.67%	75.01%	43.01%	**57.06%**	**76.36%**	**45.55%**	57.01%	76.20%	45.54%
3	24.62%	74.31%	14.75%	27.81%	74.68%	17.09%	39.93%	**75.67%**	27.12%	**45.87%**	74.78%	**33.08%**
All	9.21%	63.80%	4.96%	9.21%	63.80%	4.96%	20.45%	69.00%	12.01%	**22.20%**	**72.11%**	**13.12%**

of a feature extraction module, the setting of a smaller MiniBatch size, the selection of a faster learning rate and an higher momentum value of *SGD*.

5 CONCLUSION AND FUTURE WORK

In this paper we presented an analysis on the usage of two typologies of DNN, *Dense* and *CNN-Dense* for extreme multi-label and multi-class text classification (XMTC). We considered multi-label classifi-

cation problems characterized by different number of labels, ranging from an order of about 10 to an order of about 30,000. The considered task is harder than a *normal* multi-class text classification due to variable label number (multi-label) associated to each sample. We analysed the performances and the behaviours of the networks considering the effects of the training hyperparameters of the *SGD* function, with an increasing classes number and average number of labels per sample, using a Big Data training source extracted from PubMed repository. We performed a preliminary empirical evaluation of the link between the *SGD* hyperparameters and the dataset complexity, providing an overview of the performances and the optimal settings of learning rate, momentum and batch size for the considered problem.

As future works we are planning to investigate the impact of the hyperparameters on other DNN topologies, and we are considering to build a completely new topology customized for the hierarchical XMTC problems.

In addition, we will investigate on the use of hierarchical label structure, exploiting the better performances of higher label levels to correct the results obtained with deeper cases.

REFERENCES

Alicante, A., Benerecetti, M., Corazza, A., and Silvestri, S. (2016a). A distributed architecture to integrate ontological knowledge into information extraction. *International Journal of Grid and Utility Computing*, 7(4):245–256.

Alicante, A., Corazza, A., Isgrò, F., and Silvestri, S. (2016b). *Semantic Cluster Labeling for Medical Relations*, pages 183–193. Springer.

Amato, A., Di Martino, B., Scialdone, M., and Venticinque, S. (2014). *Personalized Recommendation of Semantically Annotated Media Contents*, pages 261–270. Springer International Publishing, Prague, Czech Republic.

Baker, S. and Korhonen, A. (2017). Initializing neural networks for hierarchical multi-label text classification. In *BioNLP 2017*, pages 307–315, Vancouver, Canada. ACL.

Baumel, T., Nassour-Kassis, J., Elhadad, M., and Elhadad, N. (2017). Multi-label classification of patient notes a case study on icd code assignment. *arXiv preprint arXiv:1709.09587*.

Berger, M. J. (2015). Large scale multi-label text classification with semantic word vectors. Technical report, Stanford University.

Blei, D. M., Ng, A. Y., and Jordan, M. I. (2003). Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022.

Chen, G., Ye, D., Xing, Z., Chen, J., and Cambria, E. (2017). Ensemble application of convolutional and recurrent neural networks for multi-label text categorization. In *2017 International Joint Conference on Neural Networks, IJCNN 2017*, pages 2377–2383, Anchorage, AK, USA. IEEE.

Gargiulo, F., Silvestri, S., and Ciampi, M. (2017a). A big data architecture for knowledge discovery in pubmed articles. In *2017 IEEE Symposium on Computers and Communications, ISCC 2017*, pages 82–87, Heraklion, Greece. IEEE.

Gargiulo, F., Silvestri, S., Fontanella, M., Ciampi, M., and De Pietro, G. (2017b). A deep learning approach for scientific paper semantic ranking. In *International Conference on Intelligent Interactive Multimedia Systems and Services*, pages 471–481, Vilamoura, Portugal. Springer.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778.

Hughes, M., Li, I., Kotoulas, S., and Suzumura, T. (2017). Medical text classification using convolutional neural networks. *CoRR*, 235:246–250.

Ilievski, I., Akhtar, T., Feng, J., and Shoemaker, C. A. (2017). Efficient hyperparameter optimization for deep learning algorithms using deterministic rbf surrogates. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI-17)*, pages 822–829, San Francisco, California, USA. AAAI.

Liu, J., Chang, W., Wu, Y., and Yang, Y. (2017). Deep learning for extreme multi-label text classification. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 115–124, Shinjuku, Tokyo, Japan. ACM.

Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013). Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781.

Nam, J., Kim, J., Loza Mencía, E., Gurevych, I., and Fürnkranz, J. (2014). Large-scale multi-label text classification - revisiting neural networks. In *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2014*, pages 437–452, Nancy, France. Springer.

Nentidis, A., Bougiatiotis, K., Krithara, A., Paliouras, G., and Kakadiaris, I. (2017). Results of the fifth edition of the bioasq challenge. In *Proceedings of the BioNLP 2017 workshop*, pages 48–57, Vancouver, Canada. ACL.

Nesterov, Y. (1983). A method for unconstrained convex minimization problem with the rate of convergence o(1/kˆ 2). In *Doklady AN USSR*, volume 269, pages 543–547.

Nigam, P. (2017). Applying deep learning to icd-9 multilabel classification from medical records. Technical report, Stanford University.

Özgür, A., Özgür, L., and Güngör, T. (2005). Text categorization with class-based and corpus-based keyword selection. *Computer and Information Sciences-ISCIS 2005*, pages 606–615.

Pavlinek, M. and Podgorelec, V. (2017). Text classification method based on self-training and lda topic models. *Expert Systems with Applications*, 80(Supplement C):83 – 93.

Polyak, B. T. (1964). Some methods of speeding up the convergence of iteration methods. *USSR Computational Mathematics and Mathematical Physics*, 4(5):1–17.

Qiang, J., Chen, P., Wang, T., and Wu, X. (2017). Topic modeling over short texts by incorporating word embeddings. In *Advances in Knowledge Discovery and Data Mining - 21st Pacific-Asia Conference, PAKDD 2017*, pages 363–374, Jeju, South Korea. Springer.

Řehůřek, R. and Sojka, P. (2010). Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50. ELRA.

Schwenk, H., Barrault, L., Conneau, A., and LeCun, Y. (2017). Very deep convolutional networks for text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2017*, volume 1, pages 1107–1116, Valencia, Spain. ACL.

Su, F., Rong, C., Huang, Q., Qiu, J., Shao, X., Yue, Z., and Xie, Q. (2015). Attribute extracting from wikipedia pages in domain automatically. In *Information Technology and Intelligent Transportation Systems - Volume 2, Proceedings of the 2015 International Conference on Information Technology and Intelligent Transportation Systems, ITITS 2015*, pages 433–440, Xi'an, China. Springer.

Sutskever, I., Martens, J., Dahl, G. E., and Hinton, G. E. (2013). On the importance of initialization and momentum in deep learning. In *Proceedings of the 30th International Conference on Machine Learning, ICML 2013*, volume 28, pages 1139–1147, Atlanta, Georgia, USA. JMLR.

Sutton, R. S. (1986). Two problems with backpropagation and other steepest-descent learning procedures for networks. In *Proceedings of 8th annual conference of cognitive science society*, pages 823–831. Erlbaum.

Wang, R., Liu, W., and McDonald, C. (2015). Using word embeddings to enhance keyword identification for scientific publications. In *Databases Theory and Applications - 26th Australasian Database Conference, ADC 2015*, pages 257–268, Melbourne, VIC, Australia. Springer.

Wang, Y. and Tian, F. (2016). Recurrent residual learning for sequence classification. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016*, pages 938–943, Austin, Texas, USA. ACL.

Yan, Y., Wang, Y., Gao, W.-C., Zhang, B.-W., Yang, C., and Yin, X.-C. (2017). Lstm2: Multi-label ranking for document classification. *Neural Processing Letters*.

Yogatama, D., Dyer, C., Ling, W., and Blunsom, P. (2017). Generative and discriminative text classification with recurrent neural networks. *CoRR*, abs/1703.01898.

Zhang, W., Wang, L., Yan, J., Wang, X., and Zha, H. (2017a). Deep extreme multi-label learning. *CoRR*, abs/1704.03718.

Zhang, Y., Ma, J., Wang, Z., and Chen, B. (2017b). LF-LDA: A topic model for multi-label classification. In *Advances in Internetworking, Data & Web Technologies, The 5th International Conference on Emerging Internetworking, Data & Web Technologies, EIDWT-2017*, pages 618–628, Wuhan, China. Springer.

Zhu, F., Li, H., Ouyang, W., Yu, N., and Wang, X. (2017). Learning spatial regularization with image-level supervisions for multi-label image classification. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE.