Convolutional Neural Networks for Biomedical Text Classification: Application in Indexing Biomedical Articles

Anthony Rios
Department of Computer Science
University of Kentucky
Lexington, Kentucky
Anthony.Rios1@uky.edu

Ramakanth Kavuluru*
Division of Biomedical Informatics
Depts. of Biostatistics and Computer Science
University of Kentucky, Lexington, Kentucky
Ramakanth.Kavuluru@uky.edu

ABSTRACT

Building high accuracy text classifiers is an important task in biomedicine given the wealth of information hidden in unstructured narratives such as research articles and clinical documents. Due to large feature spaces, traditionally, discriminative approaches such as logistic regression and support vector machines with n-gram and semantic features (e.g., named entities) have been used for text classification where additional performance gains are typically made through feature selection and ensemble approaches. In this paper, we demonstrate that a more direct approach using convolutional neural networks (CNNs) outperforms several traditional approaches in biomedical text classification with the specific use-case of assigning medical subject headings (or MeSH terms) to biomedical articles. Trained annotators at the national library of medicine (NLM) assign on an average 13 codes to each biomedical article, thus semantically indexing scientific literature to support NLM's PubMed search system. Recent evidence suggests that effective automated efforts for MeSH term assignment start with binary classifiers for each term. In this paper, we use CNNs to build binary text classifiers and achieve an absolute improvement of over 3% in macro F-score over a set of selected hard-to-classify MeSH terms when compared with the best prior results on a public dataset. Additional experiments on 50 high frequency terms in the dataset also show improvements with CNNs. Our results indicate the strong potential of CNNs in biomedical text classification tasks.

Categories and Subject Descriptors

I.5.1 [**Pattern Recognition**]: Models—*Neural nets*; H.3.1 [**Information Storage and Retrieval**]: Content Analysis and Indexing—*Indexing methods*

Keywords

convolutional neural networks, text classification, medical subject headings

1. INTRODUCTION

Text classification is an important problem with many applications in biomedicine. Specific problems such as identifying reportable cases of cancer from pathology reports, recognizing particular phenotypes from clinical narratives, determining the correct sense for the usage of an ambiguous word given its context (word sense disambiguation), and assigning medical subject headings (MeSH terms) to biomedical articles, can all be modeled as instances of the general text classification problem. Traditional approaches to text classification involve applying conditional models (e.g., logistic regression (LR), support vector machines(SVMs)) trained on features derived from the text including n-grams and named entities. Further performance gains are usually achieved with feature selection, data selection, and ensemble approaches such as voting, bagging, stacking, and boosting [40]. Deriving more interesting and relevant features based on the particular domain of interest and the nature of text are also popular additional enhancements, typically termed as feature engineering. For example, emoticons and hashtags form important new features for sentiment analysis of Twitter data [20]. In biomedicine identifying negated mentions of named entities and automatically deriving regular expression features from a set of domain specific seed patterns have been shown to be effective [7]. Exploiting definitional knowledge (concept glosses) and inter-concept relations from an external domain specific knowledge base such as the unified medical language system (UMLS) can also provide performance gains [33]. Overall, applying linear classifiers with domain specific feature engineering and ensemble modeling still produces state-of-the-art results for text classification.

The resurgence of deep neural networks (or deep nets) has paved ways to more general alternatives to supervised learning, especially in object classification. Deep nets obviate the laborious process of feature engineering and take upon the burden of automatically learning high level representations of input instances that are better suitable for the classification problem at hand. Deep nets have been initially applied to problems in computer vision but have been recently adapted to natural language processing (NLP) tasks [3,9,22] especially through learning distributed representations of textual segments (words, sentences, documents) as vectors in $\mathbb{R}^d$. These vectors directly guide primitive tasks such as part-of-speech tagging and statistical parsing as well as high level tasks such as text classification and machine translation. Convolution neural networks (CNNs) take advantage

*corresponding author

of the so called "convolutional filters" to automatically learn features that are more suitable to the task at hand. They have been actively used in biomedicine in image classification even before deep learning became popular in the recent past. However, CNNs and deep nets in general have not been explored until recently for text classification, and are currently used for sentiment/opinion mining [17, 19, 28] for short texts with fairly balanced class distributions. Our motivation in this paper is to evaluate CNNs on longer texts with highly skewed class distributions, a typical scenario encountered in biomedicine. We specifically demonstrate the potential of CNNs for learning better binary classifiers to assign medical subject headings (MeSH terms) to biomedical articles when compared with more conventional approaches.

Indexing biomedical articles with concepts from the controlled hierarchical MeSH vocabulary is an important task that has significant impact on how researchers search and retrieve relevant information. This is particularly essential given the exponential growth of biomedical articles indexed by PubMed$^{®}$, the main search system developed by the National Center for Biotechnology Information (NCBI). PubMed lets users search over 22 million biomedical citations available in the MEDLINE bibliographic database curated by the National Library of Medicine (NLM) from over 5000 leading biomedical journals in the world. To keep up with the explosion of information on various topics, users depend on queries involving MeSH terms that are assigned to each biomedical article. Once articles are indexed with MeSH terms, users can quickly search for articles that pertain to a specific subject of interest instead of relying solely on key word based searches. Besides this direct application, recent efforts also demonstrated that using the set of MeSH terms assigned to an article as its semantic proxy can be helpful in high level applications such as literature based knowledge discovery [8].

To mitigate indexing consistency issues and expedite the indexing process, there have been many recent efforts by researchers to develop automatic ways of assigning MeSH terms for indexing biomedical articles including efforts in the on-going BioASQ indexing challenge [32]. However, automated efforts (including ours) mostly focused on predicting MeSH terms for indexing based solely on the abstract and title text of the articles. This is because most full text articles are only available based on paid licenses not subscribed by many researchers. In this paper, we utilize recent advances in text classification using CNNs for assigning MeSH terms to biomedical articles based on the title and abstract text of the article. Jimeno-Yepes et al. [35] recently identified a set of nearly 29 hard-to-classify terms based on automatic indexing efforts at the NLM. They use a variety of classifiers in an ensemble setup and achieve better results than NLM's medical text indexer (MTI) program. We use their dataset and employ a CNN model proposed by Kim [19] to achieve an absolute improvement of over 3% in macro F-score over these selected terms. In a very recent attempt [15], Jimeno-Yepes et al. demonstrate the use of an extensive set of features to obtain improved results over 50 high frequency terms in the same dataset used in their 2013 paper. On these 50 terms, we achieve about 1% improvement in micro F-score and a comparable macro F-score. We believe that our results are the first to demonstrate the utility of CNNs for biomedical text classification especially for the scenarios with extreme class imbalance.

We discuss essential background on automated efforts to MeSH term assignment in Section 2 along with related efforts that use neural networks for natural language processing. In Section 3, we elaborate the specifics of the CNN model we use in our paper. We describe the dataset used for experiments, outline the methods compared, and specify the settings used for the CNNs in Section 4. We present and discuss our main results in Section 5 for three specific groups of MeSH terms. We also provide a brief analysis on what is actually being captured by the convolutions learned. In Section 6, we provide new results based on our recent experiments conducted for the 50 high frequency terms in the dataset. Section 7 summarizes our results and identifies some limitations of CNNs and directions for future work.

2. BACKGROUND

Assigning MeSH terms to biomedical articles is an instance of the well known multi-label classification problem where multiple labels from the MeSH vocabulary need to be assigned to each input instance, which is a biomedical article. NLM initiated efforts in automatic MeSH term assignment with their MTI program that exploits terms from already tagged related citations in combination with named entity recognition (NER), unsupervised clustering, ad hoc indexing rules, and candidate term ranking heuristics in a pipeline [1]. MTI recommends MeSH terms for NLM indexers to assist in their efforts to expedite the indexing process. A few recent studies [13, 18] apply the k-NN approach to obtain MeSH terms from a set of top k, already indexed, nearest neighbors and use the supervised learning-to-rank approach with novel features to rerank the MeSH terms from the neighbors. Other researchers considered different machine learning approaches with novel feature selection [36] and training data sampling [29] techniques.

A recent effort by Jimeno-Yepes et al. [14] uses a large dataset and meta-learning to train custom binary classifiers for each MeSH term and indexes the best performing model for each label to be applied on new abstracts. This method is typically known as the binary relevance approach where separate datasets of positive and negative examples are built for each label. Specifically, let T be the set of labels and let $q = |T|$. Binary relevance learns q binary classifiers, one for each label in T. It transforms the dataset into q separate datasets. For each label T_j, we obtain the dataset for the corresponding binary classifier by considering each document–label-set pair $(D_i, \mathbf{Y}_i)$ and generating the document-label pair (D_i, T_j) when $T_j \in \mathbf{Y}_i$ and generating the pair $(D_i, \neg T_j)$ when $T_j \notin \mathbf{Y}_i$. The labels whose classifiers output a positive decision are finally considered the assigned labels. Recent state-of-the-art results [21] are obtained by combining the k-NN approach with the binary relevance approach, where the candidates from a few nearest neighbors are combined with top few predicted candidates from the binary relevance approach (based on classifier scores). This combined set of candidates is then ranked using learning-to-rank with a variety of features for each candidate that include neighborhood similarity scores from the k-NN approach and classifier scores from the binary relevance approach. Given this, we note that building high accuracy binary classifiers is crucial for MeSH term assignment and we use CNNs for this task in this paper.

Neural word representations have been shown to capture both semantic and syntactic information and a few recent approaches learn word vectors [3, 9, 22] (as elements of $\mathbb{R}^d$, where d is the dimension) in an unsupervised fashion from textual corpora. Deep nets have also been applied to sentence/document level classification problems especially sentiment analysis and opinion mining with relatively smaller and reasonably balanced datasets with few classes. Classification models were developed for sentiment analysis to take advantage of the structure of sentences. For instance in recursive neural networks [28], words of a sentence are recursively merged together using a nonlinear function until we have single vector. The elements of this vector are then used as features for classification. In our work, we build binary MeSH classifiers using the CNN model by Kim [19], which is closely related to time-delay neural networks and dynamic convolutional neural networks described by Kalchbrenner et. al [17]. Our main aims are to see if CNNs are helpful for longer narratives (200 tokens per biomedical citation vs 20-50 tokens per opinion/sentiment containing narratives) and whether they can deal with large class imbalance for assigning certain MeSH terms. In the next section, we present the details of the CNN model.

3. CNN MODEL DETAILS

The goal of the model described in this section is to build a binary classifier that outputs the probability estimates of unseen documents belonging to the positive class. This model is recently proposed by Kim [19]; we add additional inputs (see Section 6) to the softmax layer of this model besides traditional CNN features. In this section we furnish additional details of the model starting with a more intuitive explanation of CNNs and subsequently elaborating with a more detailed specification of the model and the training process.

The central concept in CNNs is the notion of *convolution filters* (CFs) that are traditionally used in signal processing. The general principle is to learn several CFs which are able to extract useful features from a document for the specific classification task based on the training dataset. This has proven to be very useful in computer vision [38] where convolutions learn high level features of an image (e.g., curves and faces). Before we go into the specifics, we provide a high-level overview of convolutions for text classification.

3.1 Convolutions and CFs

A convolution is a binary operation involving the operands: a segment of text and a specific CF, both of which are represented as real matrices for our purposes. The output is a single real number. The matrix representation of the text segment, which is typically a contiguous sequence of words in the document, is composed of the word vectors of tokens that constitute it. A CF is also a matrix of the same dimensions as the text segment matrix. A specific CF operates on all contiguous segments of a document using a sliding window producing as many real number outputs as there are contiguous segments of a certain length in the document. This sequence of real numbers is called the feature map associated with the particular CF being used. Different CFs produce different feature maps, which can then be used as features in text classification. The overview of the convolution operation and CFs explained here forms the convolutional layer of the CNN. There is a conventional softmax layer that takes as input the feature maps and outputs class probability estimates. The main idea is to learn CFs (that is, the elements in the corresponding matrices) that give better feature maps that optimize our objective function. The learning process is based on predicting classes for training instances and making adjustments to the CF elements through back propagation of the gradients of the objective function (conditional log-likelihood of training data) being optimized. In this intuitive explanation, we left out many details including the mathematical definition of the convolution operation, the objective function, regularization (to deal with overfitting), and the stopping protocol for the learning process. The following section discusses these in detail.

3.2 Model Specification

The architecture of the full CNN model used is shown in Figure 1. It has two layers including a single convolution layer and a fully connected output (softmax) layer.

The base component in the model is a word vector $\mathbf{x} \in \mathbb{R}^d$, where d is the dimension of the word vectors. A document is represented as a matrix $D \in \mathbb{R}^{n \times d}$, where n is the number of words in it and each row represents the word vector for the corresponding token. To simplify the equations in this section we will assume the ground truth for the document $Y \in \mathbb{R}^2$ such that $Y_2 = 1$ and $Y_1 = 0$ ($Y_2 = 0$ and $Y_1 = 1$) when we are training on a positive (negative) instance. This is more aligned with the two output nodes of the final layer for the two corresponding classes (positive/negative) for each binary classifier. Although a single node would have been sufficient for binary classification, we chose to build our model with multiple nodes to simply have the code set-up for multi-class classification for other text classification problems.

We define a CF $\mathbf{W} \in \mathbb{R}^{h \times d}$, where h is the number of words we wish the convolution filter to span, that is, the length of the sliding window. Let the 2-D convolution operation $*$ be defined as

$$\mathbf{W} * D_{j:j+h-1} = \sum_{i=j}^{j+h-1} \sum_{k=0}^{d-1} \mathbf{W}_{i,k} D_{i,k}.$$

We next map a length h word window, $D_{j:j+h-1}$, of the document to a real number $c_j \in \mathbb{R}$ using a non-linear function f as

$$c_j = f(\mathbf{W} * D_{i,j:j+h-1} + b), \tag{1}$$

where $b \in \mathbb{R}$ represents the bias term. In this work f is a rectified linear function [12, 23]. After convolving over the entire document using $\mathbf{W}$, we have the corresponding convolved feature map

$$\mathbf{c}(\mathbf{W}) = [c_1, c_2, \ldots, c_{n-h+1}].$$

To overcome the issue of varying document lengths we perform max-pooling [10] operation

$$\hat{c}_{\mathbf{W}} = \max_i \mathbf{c}(\mathbf{W})_i,$$

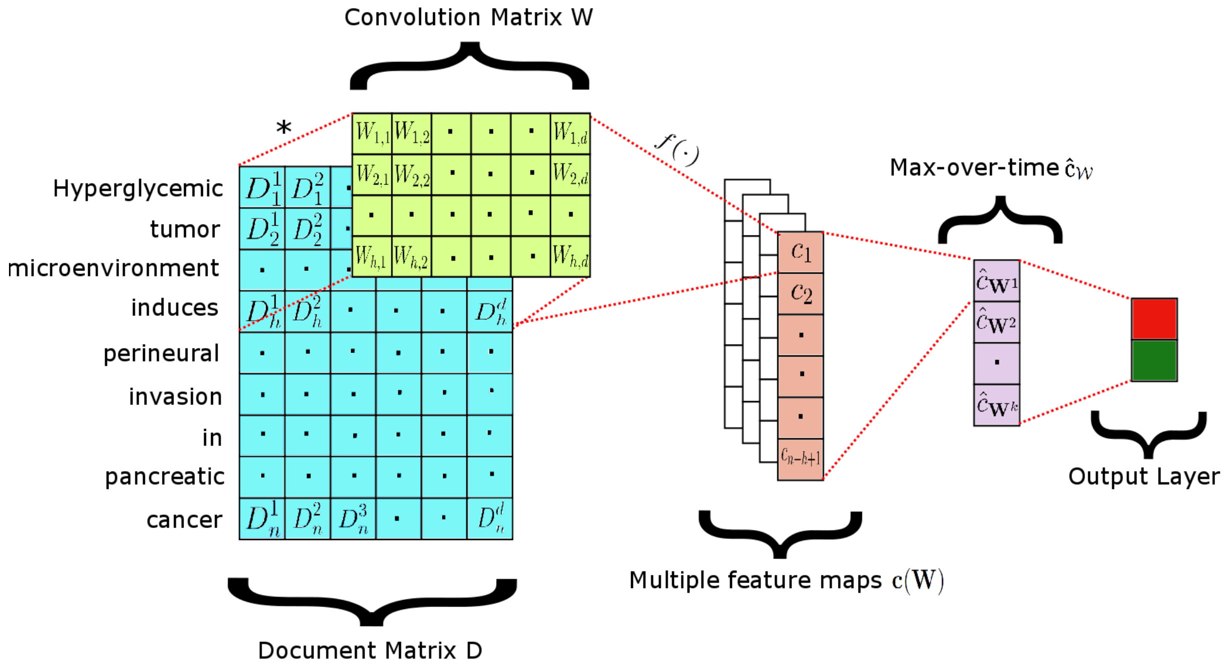

Figure 1: The CNN model layout

which gives a single feature $\hat{c}_{\mathbf{W}}$ corresponding to the feature map generated by $\mathbf{W}$. However, we will learn several CFs, say k of them, $\mathbf{W}^1, \ldots, \mathbf{W}^k$, to create multiple feature maps leading to the corresponding single max-pooled features $\hat{\mathbf{c}}_{\mathbf{W}^t}$, $t = 1, \ldots, k$. These form a final max-pooled feature vector

$$\hat{\mathbf{c}}_{\mathcal{W}} = [\hat{\mathbf{c}}_{\mathbf{W}^1}, \ldots, \hat{\mathbf{c}}_{\mathbf{W}^k}]^T,$$

where $\mathcal{W} = \{\mathbf{W}^1, \ldots, \mathbf{W}^k\}$.

After obtaining $\hat{\mathbf{c}}_{\mathcal{W}}$, we add a final softmax layer. Let $\mathbf{U} \in \mathbb{R}^{2\times k}$ and $b^U \in \mathbb{R}^2$ be the parameters of the softmax layer with weighted inputs

$$y_j = \mathbf{U}_j \hat{\mathbf{c}}_{\mathcal{W}} + b_j^U \tag{2}$$

and output label probability estimates

$$P(Y_j = 1 | D, \mathcal{W}, b, \mathbf{U}, b^U) = \frac{e^{y_j}}{\sum_i e^{y_i}}, \tag{3}$$

where $\mathbf{U}_j$ is the j-th row of $\mathbf{U}$, b_j^U is the j-th element of b^U, and Y_j is the j-th label for the document corresponding to matrix D.

If $\mathcal{D}$ is the set of training document matrices, to learn each binary classifier we minimize

$$-\sum_{D\in\mathcal{D}} \log(P(Y_{pos}^D = 1 | D, \mathcal{W}, b, \mathbf{U}, b^U)), \tag{4}$$

where $pos = 1$ ($pos = 2$) if the corresponding document is a negative (positive) instance and Y^D is the ground truth for document represented by D. The parameters of the CNN ($\mathcal{W}, b, \mathbf{U}, b^U$ from equation 4 and the word vectors) that minimize this are obtained by calculating the gradient and using back propagation with the stochastic gradient descent approach. A subtle but crucial aspect that makes CNNs for NLP tasks different from those used in computer vision is that the base input components to the CNN, that is, the word vectors, are also modified using back propagation in addition to the traditional network weights. This is done by treating the word vector elements as network weights of the first and so called projection layer [27, Chapter 2] and modifying them just like any other network weight. However, the vector elements for a given word only change when the current instance contains that word, which happens often if it is a common word or if the dataset is large. We used the popular mini-batches [25] approach instead of updating parameters for each example. CNNs also warrant multiple epochs where the learning process goes through the entire training dataset multiple times in optimizing equation 4.

Instead of using the well known l1 or l2 regularization we use the dropout [30] option to prevent over-fitting during training. Specifically, instead of passing y_j (from equation 2) to the softmax function in equation 3 during training, we actually pass

$$\hat{y}_j = \mathbf{U}_j(\hat{\mathbf{c}}_{\mathcal{W}} \circ \mathbf{r}) + b_j^U,$$

where $\circ$ refers to element-wise multiplication and $\mathbf{r} \in \{0,1\}^k$ is constructed with each r_i drawn from the Bernoulli distribution with parameter p (typically set to 0.5). Intuitively, this means that gradients are backpropagated only through unmasked elements where $r_i = 1$. During test time we scale the weights U such that

$$y_j = p\mathbf{U}_j \hat{\mathbf{c}}_{\mathcal{W}} + b_j^U.$$

This down weighting is essential since at training, on average, only half of the U edges were active, which is not true at test time. We also used early-stopping in order to help combat over-fitting. Typically early-stopping is done by simply terminating the training of the model when the desired score on a held out validation dataset does not increase in performance. However, we found this caused us to stop too early. To combat this we stopped training if there were 5 consecutive epochs in the training procedure that did not increase the validation score. We only saved the model on epochs that had an increase in F-score on the validation dataset. For example, if there was an increase in F-score in the second epoch on the validation dataset, and then training continued for five more epochs without any further increase, we kept the model from the 2nd epoch. Additional specifics on initialization of word vectors and CFs and other parameter choices are discussed in the next section.

4. EXPERIMENTAL SETUP

In this section we discuss the dataset used in our paper along with the MeSH terms for which we built CNN binary classifiers. We also discuss CNN model parameter settings and other approaches compared with them. For all our experiments we used the Python based Theano [4] platform.

4.1 Dataset and MeSH Terms Used

We use a publicly available[1] dataset consisting of MEDLINE citations from November 2012 to February 2013. It contains 89,942 biomedical citations for training, 5000 for validation, and 48,911 for testing. The dataset, the MeSH terms for which the classifiers are built, and the methods compared with CNNs are those used in a prior effort by Jimeno-Yepes et. al [35] to facilitate direct comparison. We compare our model with 29 MeSH terms used in [35] split into three groups. The three groups were created based on how NLM's MTI program performs on each term: check tags, low recall terms, and low precision terms. Check-tags are a special set of popular MeSH terms (e.g, `Humans`, `Female`, and `Adult`), which are typically checked for each article to be indexed. We built CNN models for the 12 popular check tags, the top 7 low recall terms, and the top 10 low precision terms based on MTI's judgments; so a total of 29 terms were considered based on the three groups in [35]. The goal was to see if the binary relevance approach based on supervised learning approaches can provide a better alternative to the k-NN based approach employed in MTI. In [35], the authors also conduct experiments on a fourth group of terms for which MTI had zero recall. These terms are very rare (about 1% of 48,911 citations in the test set) and given we use a validation dataset of only 5000 citations, we have excluded this group from our evaluation after preliminary results showed that there were not enough positive examples in the validation dataset for early stopping.

4.2 CNN Parameters of Section 3.2

For all experiments, we used a word vector of size $d = 300$. Each word vector is initialized with values drawn uniformly from $[-0.25, 0.25]$. The CF $\mathbf{W}$ values and softmax layer weights $\mathbf{U}$ are drawn uniformly from $[-0.1, 0.1]$. We used three different CF sizes with window lengths $h = 3$, 4, and 5. For each of these filter sizes we used 100 feature maps creating a total of 300 feature maps per classifier. The models were trained using AdaDelta [37], an adaptive learning rate method for stochastic gradient descent with a maximum of 15 epochs per classifier. We also used mini-batches of size 50 and we zero-padded the document at the beginning as needed. The dropout parameter p was set to 0.5. Some of these choices were based on settings used in the paper where this CNN model was first used [19] and others were based on our experimentation.

4.3 Methods Compared

The following list has a total of 13 types of models we compare in our effort. The last four are based on CNN models and the the rest of them are from Jimeno-Yepes et. al [35].

- Naive Bayes (NB)
- Logistic Regression (LR)
- Support Vector Machines (SVM)
- Support Vector Machines with Huber Loss (SVM HL)
- AdaBoostM1 (Ada)
- AdaBoostM1 with Oversampling (Ada Over)
- Medical Text Indexer (MTI)
- Vote 2 – This predicts a positive label if any two of the preceding models make a positive prediction for any given example.
- Vote 3 – This is the same as Vote 2 except it requires that at least 3 of the base algorithms NB, LR, SVM, SVM HL, Ada, Ada Over, or MTI predict any given label.
- CNN-rand – This is our first CNN model which uses randomized initial word vectors and is trained as described in Section 3.2.
- CNN-pre – This model initializes word vectors with those obtaining from running Word2Vec [22] on all biomedical citations in PubMed in the last decade except for instances in the test set.
- CNN-Vote 2 – This model ensembles five CNN-rand models (with different word vector initializations) and one CNN-pre model for each term. Predictions are made just as in Vote 2, when at least two models make a positive prediction.

5. MAIN RESULTS AND DISCUSSION

In this section we present and discuss results for each group of terms using models in Section 4.3. The F-scores of the models for the three MeSH term groups (from Section 4.1) are shown in Tables 1–3. In these tables the 'positive' column refers to the number of positives examples in the test dataset. The 'prior-best' column refers to the maximum F-score taken over all non-CNN models (from [35]) enumerated in Section 4.3 except MTI (we list MTI separately since it is our baseline). However, the non-CNN models that involve voting (Vote 2 and Vote 3) include MTI as a component of the ensemble.

5.1 CNN Model Performance

For check-tags results (Table 1), MTI is not included because for check-tags MTI has been modified to use AdaBoostM1 with oversampling instead of the k-NN method used for other terms. The check tags group has an average of 8701 positive examples per label in the test set and is a popular set of terms frequently used; most citations have at least

[1] `http://ii.nlm.nih.gov/MTI_ML/index.shtml`

MeSH Term	Positive	Prior-best	CNN-rand	CNN-pre	CNN-Vote 2
Adolescent	3824	0.4708	0.5028	0.4951	**0.5335**
Adult	8792	0.6225	0.6593	0.6491	**0.6759**
Aged	6151	0.6005	0.6378	0.6427	**0.6516**
Aged, 80 and over	2328	0.3753	0.3909	0.3932	**0.4389**
Child, Preschool	1573	0.5129	0.4963	0.5296	0.5417
Female	16483	0.7764	0.7820	**0.7945**	0.7863
Humans	35967	0.9337	0.9322	**0.9407**	0.9339
Infant	1281	0.4796	0.4912	0.4926	**0.5378**
Male	15530	0.7582	0.7690	**0.7797**	0.7748
Middle Aged	8392	0.6731	0.7019	0.6911	**0.7058**
Swine	285	**0.7323**	0.7233	0.7220	0.7282
Young Adult	3807	0.3973	0.4007	0.4207	**0.4536**

Table 1: The F-scores on test set for the check tags group

MeSH Term	Positive	MTI	Prior-best	CNN-rand	CNN-pre	CNN-Vote 2
Age Factors	889	0.0844	0.1748	0.2344	**0.2469**	0.2406
Brain	823	**0.5201**	0.4700	0.4640	0.4472	0.5014
Cell Line	781	0.2876	**0.3059**	0.2243	0.2847	0.2989
Cells, Cultured	1079	0.3046	0.3894	0.2893	0.3347	**0.3979**
Models, Molecular	851	0.4292	0.4763	0.3764	0.3740	**0.4914**
Molecular Sequence Data	1527	**0.5495**	0.5118	0.4047	0.4532	0.5178
RNA, Messenger	628	0.4477	0.4626	0.4675	0.3782	**0.4743**
Severity of Illness Index	751	0.1824	0.2415	0.2299	0.2152	**0.2489**
Time Factors	2153	0.0980	0.1513	0.1608	0.1574	**0.2041**
United States	2658	0.3585	0.4128	0.3956	0.4693	**0.4613**

Table 2: The per-term F-scores for the low-precision MeSH term group

Mesh Term	Positive	MTI	prior-best	CNN-rand	CNN-pre	CNN-Vote 2
Child	2780	0.5836	0.5854	0.6202	0.6154	**0.6410**
Follow-Up Studies	1470	0.0407	0.2741	0.2989	0.3337	**0.3382**
Reproducibility of Results	1206	0.3191	0.3722	0.3316	0.3344	**0.3923**
Retrospective Studies	2183	0.6608	0.6592	0.6695	0.6681	**0.6756**
Risk Assessment	1014	0.2556	0.2189	0.2369	0.2252	**0.2661**
Risk Factors	2365	**0.4989**	0.4774	0.4748	0.4716	0.4878
Treatment Outcome	2999	0.4202	0.4421	0.3875	0.3985	**0.4553**

Table 3: The per-term F-scores for the low-recall MeSH term group.

one check tag. Our results for this group in Table 1 show that CNN models improve the F-score for all but one term in the group. Interestingly, a single CNN with pretrained word vectors seems to perform better than using an ensemble of CNNs on very frequent terms such as Humans, Male, and Female. This is not unexpected since our voting scheme requires at least two models predicting a term. However, for less frequent check tags, ensemble models prove to be very useful. Finally, we also note that the our check tag scores in Table 1 also show considerable improvement over another prior effort that uses deep belief networks [34, Table 2].

The low precision group has an average of 1214 examples per term. The low precision/recall situations are common in unbalanced datasets due to the inherent complexity in the particular class being predicted and also owing to underfitting/overfitting issues in certain algorithms. However, in Table 2 voting with CNNs consistently outperforms other methods. Next, we look at the performance of the low recall group in Table 3. This group has an average of 2002 examples per label. Again we see that CNNs models perform better than other models except for one term for which MTI performs better. Finally, we consider the macro average scores for all models in each term group in Table 4 to assess their consistency. We can see that CNN ensembles provide the best results. Overall, CNNs seem to perform better than any other individual method. Ensembled CNNs perform better than any individual method and better than any ensemble method without CNNs.

Model	Check Tags	Low P	Low R
MTI	–	0.3262	0.3970
NB	0.4191	0.1504	0.2588
LR	0.5441	0.2682	0.3653
SVM	0.5549	0.2606	0.3647
SVM HL	0.5437	0.2654	0.3587
Ada	0.5237	0.1720	0.3398
Ada Over	0.5646	0.2764	0.3819
Vote 2	0.6072	0.3596	0.4315
Vote 3	0.5885	0.2825	0.3933
CNN-rand	0.6240	0.3247	0.4313
CNN-pre	0.6293	0.3361	0.4353
CNN-Vote 2	**0.6469**	**0.3837**	**0.4652**

Table 4: The macro-averaged F-scores for each individual method for all three groups.

The main difference between the CNN models we used and other methods we compared with, such as the Linear SVM and Logistic Regression, is that our method creates high level abstractions using CFs. However, even in cases when there is a small number of positive examples, CNN based models perform better than all other methods. In Table 4, CNN-Vote 2 is an ensemble containing only CNN models. However, even in the low precision group which has, on average, the least number of positive examples per term, CNN-Vote 2 still achieves 2% improvement over the best model that does not use CNNs. In all our voting models in this section, we purposefully avoided including MTI as a component model given the potential bias as the test set articles are already indexed in PubMed when we conducted our experiments. However, we believe using MTI as part of the voting ensembles for new citations will yield better performance since MTI also incorporates nearest neighbor based predictions and as such offers complementary predictive power.

5.2 Visualizing Convolutions

Given the results are promising, we wanted to see what the CFs are actually capturing at a high level. For this we applied each of the CFs obtained after training the `Humans` model to all trigrams in the test dataset. We then identified the top five trigrams when ranked based on the first layer output score in equation 1. Table 5 shows four manually selected CFs of the 100 we used. CF-2 seems to have learned to capture information about small things, such as nanoparticles and cells. Raman images are actually an imaging/fingerprinting method for molecules. CF-4 seems to capture information from citations that are performing different types of analysis, which may have to do with numerical properties. These results are interesting in that instead of just learning to capture very specific important n-grams (like in logistic regression), CFs seem to capture different topics or aspects of the input citations to be classified.

6. ADDITIONAL EXPERIMENTS WITH FREQUENT TERMS

In a recent paper [15], Jimeno-Yepes et al. used the same dataset used in their prior effort and in our current effort to conduct a comprehensive feature engineering study over a set of 50 randomly selected high frequency terms from 63 terms that have a minimum frequency of 1500 citations in the dataset. They select a large set of feature types including named entities (and their hypernymic variants), journal, author, and author affiliations of the citation, noun phrases, and argumentative structure enhanced n-grams (based on methods, results, and discussion components of structured abstracts). They also treat title and abstract as separate fields and consider the corresponding features as having different types. They also include as features term sets from NLM's MTI indexing program and its components including the PubMed related citations (PRC) component that in a sense fetches nearest neighbors of the input citation. Besides the conventional SVMs with linear kernels, they also used SVM^{perf} [16] to directly optimize F-score (actually, a function that lower bounds F-score), which is a non-linear measure that cannot be directly optimized with traditional classifiers.

We conducted experiments with our CNN models over the same set of 50 high frequency terms (those used in [15]) whose results are shown in Table 6. In addition to the straightforward macro average discussed earlier, we also compute micro F-score over the 50 terms (where the contingency table counts of all classifier outputs are pooled to compute a single F-measure). As we can see, the single model with pre-trained word vectors without any of the feature engineering described earlier has a better micro F-score compared with the prior best score. However, the macro F-score is six

Convolution Filter 1	**Convolution Filter 2**	**Convolution Filter 3**	**Convolution Filter 4**
generate strong evidence	antibody conjugated nanoparticles	variety of both	mathematical model
academic performance in	nanoparticles on intact	temperament and character	analysis thus we
of chemical reactions	to raman images	studies revealed survival	like climate ph
theory and experiment	of cells and	electrochemical biosensor was	of tumor uptake
techniques and vacuum	membranes where the	features of individuals	benthic cyanobacterial mats

Table 5: Top five trigram activations from four different convolutions from the Humans check tag model

Model ID	**Model**	**Micro-F**	**Macro-F**
1	Jimeno-Yepes Best Micro (SVMLight +MTI)	0.7135	0.5128
2	Jimeno-Yepes Best Macro (SVM^{perf} with all features)	0.6922	**0.5565**
3	CNN-pre	0.7175	0.4963
4	CNN-pre+MetaMap+PRC	0.7170	0.5185
5	CNN-Vote 2 with Model **4**	0.7200	0.5473
6	Model **5** and best single models for top four terms	**0.7217**	0.5482

Table 6: The micro and macro averaged F-scores for top 50 high frequency terms

points lower with the single pre-trained CNN. We note that the outer softmax layer in Figure 1 can take other features in addition to the CFs. Out of more than ten different types of features used in [15], we use two of them: PRC component based terms and named entities by running the named entity recognition tool MetaMap [2]. Both these are components of the MTI program. We did not use MTI predictions given the articles in the test set are already indexed in PubMed. Adding these two features to the softmax layer improves the macro F-score over the pre-trained model by two percentage points with negligible loss in micro F-score.

From the vote-2 model 5 in Table 6, our macro F-score is lower by less than 1% compared with the best score produced by extensive feature engineering with a 2.5% increase in micro F-score over the corresponding feature engineered model that produced the best macro F-score. Model 6 is same as model 5 except voting is not used for the top four frequent terms, which is a natural choice given voting might decrease the performance of very frequent terms. This improves both micro and macro F-scores by 0.1% compared with model 5. Thus, our analysis shows that a single CNN based voting model (model 6) with two straightforward features (besides the CFs) almost achieves (macro score) or improves (micro score) over two separate feature engineered models that do not achieve comparable macro-micro score combinations.

7. CONCLUDING REMARKS

In this effort, we demonstrated that CNNs with CFs over word sequences are effective for biomedical text classification in achieving new state-of-the-art scores over more traditional linear classifiers especially when there is significant label imbalance. Specifically, using a well known public gold standard dataset, we achieved a macro F-score improvement of 3% over previous best scores on three specific groups of hard to classify MeSH terms (Table 4). Additional experiments over 50 high frequency terms also reveal that elaborate feature engineering can also be minimized with CNNs (Table 6) and that new features can be simply passed to the outer softmax layer of the CNN to achieve better results compared with using those features in traditional classifiers. We believe our results will further improve if we incorporate predictions from the MTI program given it offers a complementary approach to our methods. We, however, chose to exclude MTI terms from our experiments given potential bias that could occur with our particular test dataset. Overall, our results demonstrate the strength of CNNs in capturing high level features that lead to superior classification performance over other approaches that involve several well known linear classifiers with elaborate feature engineering and ensembling.

Next, we outline a few research directions we plan to explore in the immediate future. First, we will study deeper CNNs utilizing more sophisticated max-pooling procedures. In this effort we only have one convolutional layer followed by a softmax layer. It would be interesting to look at the types of higher order features CNNs generate at deeper layers for text classification. CNNs for image classification have been widely studied and have been shown to produce very interesting higher order features [38]. For text, topic models [5, 6, 31] have been extensively studied based on their ability to represent documents as a distributions of 'topics'. Can deep models learn 'topics' or some other higher order representation to help with different tasks? In this paper, we only perform max-over-time pooling procedure after a CF operates on sliding windows of a chosen size. However, dynamic max-pooling methods have been proposed for text classification [17] with CNNs. Using a max-over-time method loses a significant amount of information, especially for long documents and we would like to explore these other alternatives.

A caveat of CNNs is that even with early stopping they typically take at least an order of magnitude more time than conventional classifiers. This is typically not an issue for building high quality classifiers for few labels. However, in multi-label classification scenarios over large label spaces, using CNNs as part of the binary relevance approach, which requires learning a different classifier per label, can be prohibitive. So we will look at methods for expanding CNNs to simultaneously predict over the entire label space of MeSH avoiding the binary relevance approach. We will explore different loss functions that are already proposed for general neural networks for multi-label classification [24, 39]. We would like to see if a deep CNN with a multi-label output can perform well on MeSH. Finally, we will explore methods that take advantage of label correlations and prior structure we know about the MeSH label space. While there have been several methods that take advantage of label correlations [26], few look at very large label spaces. There has been work in computer vision that exploits prior information about labels to improve prediction on classes that have very few examples [11]. We would like to explore whether they can be applied to a multi-label CNN for MeSH term assignment.

Acknowledgments

We are grateful to an anonymous reviewer who correctly pointed out the potential for recall oriented bias in our results when we include NLM's MTI program in the voting ensemble. Our experiments show that the Web based MTI program resulted in high recall as the test instances are already indexed and are potentially being incorporated in MTI results. We have removed MTI from the voting ensembles and also from inputs to the softmax layer in Section 6. We updated the scores and performance improvement estimates accordingly, given this valid criticism. We also thank three other reviewers for their thorough assessment of our contribution and specific comments that helped improve the quality of this paper. This publication was supported by the National Center for Research Resources and the National Center for Advancing Translational Sciences, US National Institutes of Health (NIH), through Grant UL1TR000117. The content is solely the responsibility of the authors and does not necessarily represent the official views of the NIH.

8. REFERENCES

[1] A. Aronson, J. Mork, C. Gay, S. Humphrey, and W. Rogers. The NLM indexing initiative's medical text indexer. In *Proceedings of MEDINFO*, 2004.

[2] A. R. Aronson and F.-M. Lang. An overview of metamap: historical perspective and recent advances. *Journal of the American Medical Informatics Association*, 17(3):229–236, 2010.

[3] Y. Bengio, R. Ducharme, P. Vincent, and C. Janvin. A neural probabilistic language model. *The Journal of Machine Learning Research*, 3:1137–1155, 2003.

[4] J. Bergstra, O. Breuleux, F. Bastien, P. Lamblin, R. Pascanu, G. Desjardins, J. Turian, D. Warde-Farley, and Y. Bengio. Theano: a CPU and GPU math expression compiler. In *Proceedings of the Python for Scientific Computing Conference (SciPy)*, June 2010. Oral Presentation.

[5] D. Blei and J. Lafferty. Correlated topic models. *Advances in neural information processing systems*, 18:147, 2006.

[6] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *the Journal of machine Learning research*, 3:993–1022, 2003.

[7] D. D. A. Bui and Q. Zeng-Treitler. Learning regular expressions for clinical text classification. *Journal of the American Medical Informatics Association*, pages amiajnl–2013, 2014.

[8] D. Cameron, R. Kavuluru, T. C. Rindflesch, A. P. Sheth, K. Thirunarayan, and O. Bodenreider. Context-driven automatic subgraph creation for literature-based discovery. *Journal of biomedical informatics*, 54:141–157, 2015.

[9] R. Collobert and J. Weston. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, pages 160–167. ACM, 2008.

[10] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12:2493–2537, 2011.

[11] J. Deng, N. Ding, Y. Jia, A. Frome, K. Murphy, S. Bengio, Y. Li, H. Neven, and H. Adam. Large-scale object classification using label relation graphs. In *Computer Vision–ECCV 2014*, pages 48–64. Springer, 2014.

[12] X. Glorot, A. Bordes, and Y. Bengio. Deep sparse rectifier networks. In *Proceedings of the 14th International Conference on Artificial Intelligence and Statistics. JMLR W&CP Volume*, volume 15, pages 315–323, 2011.

[13] M. Huang, A. Névéol, and Z. Lu. Recommending mesh terms for annotating biomedical articles. *Journal of the American Medical Informatics Association*, 18(5):660–667, 2011.

[14] A. Jimeno-Yepes, J. G. Mork, D. Demner-Fushman, and A. R. Aronson. A one-size-fits-all indexing method does not exist: Automatic selection based on meta-learning. *Journal of Computing Science and Engineering*, 6(2):151–160, 2012.

[15] A. Jimeno Yepes, L. Plaza, J. Carrillo-de Albornoz, J. G. Mork, and A. R. Aronson. Feature engineering for medline citation categorization with mesh. *BMC Bioinformatics*, 16(1):113, 2015.

[16] T. Joachims. A support vector method for multivariate performance measures. In *Proceedings of the 22nd international conference on Machine learning*, pages 377–384. ACM, 2005.

[17] N. Kalchbrenner, E. Grefenstette, and P. Blunsom. A convolutional neural network for modeling sentences. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 655–665, Baltimore, Maryland, June 2014. Association for Computational Linguistics.

[18] R. Kavuluru and Y. Lu. Leveraging output term co-occurrence frequencies and latent associations in predicting medical subject headings. *Data & Knowledge Engineering*, 94(Part B):189–201, 2014.

[19] Y. Kim. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference*

on Empirical Methods in Natural Language Processing (EMNLP), pages 1746–1751, Doha, Qatar, October 2014. Association for Computational Linguistics.

[20] S. Kiritchenko, X. Zhu, and S. M. Mohammad. Sentiment analysis of short informal texts. *Journal of Artificial Intelligence Research*, pages 723–762, 2014.

[21] K. Liu, J. Wu, S. Peng, C. Zhai, and S. Zhu. The fudan-uiuc participation in the BioASQ challenge task 2a: The antinomyra system. *Proceedings of Question Answering Lab at the Conference and Labs of the Evaluation Forum (CLEF)*, 2014.

[22] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, pages 3111–3119, 2013.

[23] V. Nair and G. E. Hinton. Rectified linear units improve restricted Boltzmann machines. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 807–814, 2010.

[24] J. Nam, J. Kim, E. L. Mencía, I. Gurevych, and J. Fürnkranz. Large-scale multi-label text classificationâĂŤrevisiting neural networks. In *Machine Learning and Knowledge Discovery in Databases*, pages 437–452. Springer, 2014.

[25] J. Ngiam, A. Coates, A. Lahiri, B. Prochnow, Q. V. Le, and A. Y. Ng. On optimization methods for deep learning. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 265–272, 2011.

[26] J. Read, B. Pfahringer, G. Holmes, and E. Frank. Classifier chains for multi-label classification. *Machine Learning*, 85(3):335–359, 2011.

[27] R. Socher. *Recursive Deep Learning for Natural Language Processing and Computer Vision*. PhD thesis, Department of Computer Science, Stanford University, 2014.

[28] R. Socher, A. Perelygin, J. Y. Wu, J. Chuang, C. D. Manning, A. Y. Ng, and C. Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the conference on empirical methods in natural language processing (EMNLP)*, volume 1631, page 1642. Citeseer, 2013.

[29] S. Sohn, W. Kim, D. C. Comeau, and W. J. Wilbur. Optimal training sets for bayesian prediction of MeSH assignment. *Journal of the American Medical Informatics Association*, 15(4):546–553, 2008.

[30] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[31] M. Steyvers, P. Smyth, M. Rosen-Zvi, and T. Griffiths. Probabilistic author-topic models for information discovery. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 306–315. ACM, 2004.

[32] G. Tsatsaronis, G. Balikas, P. Malakasiotis, I. Partalas, M. Zschunke, M. R. Alvers, D. Weissenborn, A. Krithara, S. Petridis, D. Polychronopoulos, et al. An overview of the bioasq large-scale biomedical semantic indexing and question answering competition. *BMC bioinformatics*, 16(1):138, 2015.

[33] A. J. Yepes and R. Berlanga. Knowledge based word-concept model estimation and refinement for biomedical text mining. *Journal of biomedical informatics*, 53:300–307, 2015.

[34] A. J. Yepes, A. MacKinlay, J. Bedo, R. Garnavi, and Q. Chen. Deep belief networks and biomedical text categorisation. In *Proceedings of the Twelfth Annual Workshop of the Australasia Language Technology Association*, page 123, 2014.

[35] A. J. Yepes, J. G. Mork, D. Demner-Fushman, and A. R. Aronson. Comparison and combination of several MeSH indexing approaches. In *AMIA Annual Symposium Proceedings*, volume 2013, page 709. American Medical Informatics Association, 2013.

[36] M. Yetisgen-Yildiz and W. Pratt. The effect of feature representation on medline document classification. In *Proceedings of AMIA Symposium*, volume 2005, pages 849–853. American Medical Informatics Association, 2005.

[37] M. D. Zeiler. ADADELTA: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, 2012.

[38] M. D. Zeiler, D. Krishnan, G. W. Taylor, and R. Fergus. Deconvolutional networks. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 2528–2535. IEEE, 2010.

[39] M.-L. Zhang and Z.-H. Zhou. Multilabel neural networks with applications to functional genomics and text categorization. *Knowledge and Data Engineering, IEEE Transactions on*, 18(10):1338–1351, 2006.

[40] Z.-H. Zhou. *Ensemble methods: foundations and algorithms*. CRC Press, 2012.