Incorporating Figure Captions and Descriptive Text in MeSH Term Indexing

Xindi Wang
Department of Computer Science
The University of Western Ontario
London, Ontario, Canada
xwang842@uwo.ca

Robert E. Mercer
Department of Computer Science
The University of Western Ontario
London, Ontario, Canada
mercer@csd.uwo.ca

Abstract

The goal of text classification is to automatically assign categories to documents. Deep learning automatically learns effective features from data instead of adopting human-designed features. In this paper, we focus specifically on biomedical document classification using a deep learning approach. We present a novel multichannel TextCNN model for MeSH term indexing. Beyond the normal use of the text from the abstract and title for model training, we also consider figure and table captions, as well as paragraphs associated with the figures and tables. We demonstrate that these latter text sources are important feature sources for our method. A new dataset consisting of these text segments curated from 257,590 full text articles together with the articles' MEDLINE/PubMed MeSH terms is publicly available.

1 Introduction

Text classification is a process that assigns labels or tags to text according to its contents. It can be done manually or automatically. Most text classification tasks were done by human annotators prior to the information age. A human annotator reads and interprets the content of the text and then classifies it into certain categories. Traditional text classification is time consuming and expensive, especially when dealing with a large number of documents.

Currently, there is a trend to support text classification through automatic tools as it does the same job as human annotators, but accomplishes it in more accurate and efficient ways. Automatic text classification is an important application and research topic in natural language processing because of the exponentially increasing number of online documents. It saves time and money in general, leading to its continued and enthusiastic usage in both business and research.

MEDLINE[1] and PubMed[2] are databases that can access publications of life sciences and biomedical topics. They are maintained by the United States National Library of Medicine (NLM).

The MEDLINE database includes bibliographic information for articles in various disciplines of life sciences and biomedicine, such as medicine, health care, biology, biochemistry and molecular evolution. The database contains more than 25 million records in over 5,200 worldwide journals. More than 800,000 citations were added to MEDLINE in 2017, which is more than 2,000 updates daily[1].

PubMed has a web server that can freely access the MEDLINE database of references and abstracts. Some PubMed records have full text articles available on PubMed Central[3]. Journal articles in MEDLINE are indexed according to Medical subject headings (MeSH)[4], which are the NLM's controlled vocabulary thesaurus.

MeSH is a hierarchically-organized terminology indexing system that categorizes biomedical documents in the NLM databases. It is updated annually. The 2018 version of MeSH contains 28,939 headings[5]. Among these MeSH terms, there are 29 check tags which are a special group of MeSH terms describing subjects of research (human or animal; mice or rats, etc.). MeSH terms are distinctive features of MEDLINE, which are great tools for indexers and searchers. Indexers from NLM use MeSH terms to classify documents based on the contents of journal articles in the MEDLINE database. Searchers and researchers use MeSH terms to assist subject searching in MEDLINE, PubMed and other databases.

[1] https://www.nlm.nih.gov/bsd/medline.html
[2] https://www.nlm.nih.gov/bsd/pubmed.html
[3] https://en.wikipedia.org/wiki/PubMed_Central
[4] https://www.nlm.nih.gov/mesh/meshhome.html
[5] https://www.nlm.nih.gov/pubs/techbull/nd17/nd17_mesh.html

Proceedings of the BioNLP 2019 workshop, pages 165–175
Florence, Italy, August 1, 2019.

Currently MeSH term indexing is performed by a large number of human annotators, who review full text documents and assign suitable MeSH terms to each article. Human annotation is time consuming and costly. Research shows that the average cost of annotation per document is around $9.40 (Mork et al., 2013), which translates into a huge cost for indexing a large number of documents. Meanwhile, there is a large number of documents uploaded to MEDLINE and PubMed databases every day (approximately 2,000–4,000 on a daily basis)[2]. It is challenging to annotate all new incoming documents in a relatively short time. Therefore, a computational system that can assist the indexing of a large number of biomedical articles is highly desired.

In this paper, we focus on the task of automatic MeSH indexing. We propose a novel deep learning based discriminative method, multichannel TextCNN, which uses convolutional neural network based feature selection to extract important information from the article to be indexed. In addition to extracting information from the title and abstract of the article, our innovation integrates figure and table captions, as well as relevant paragraphs into the indexing process. We summarize the most major contributions as follows:

- We explore the use of multichannel deep learning architectures for the automatic MeSH indexing task.
- Experimental results show that incorporating figure and table information improves the performance of automatic MeSH indexing.
- We make available a labeled full text biomedical document dataset (including title, abstract, figure and table captions, as well as paragraphs related to the figures and tables) to the research community.

2 Related Work

Due to the growth in the number of documents in MEDLINE, and the increasing number of MeSH terms every year, automatic MeSH indexing is a difficult challenge. The Medical Text Indexer (MTI) (Aronson et al., 2004) produced by the U.S. National Library of Medicine (NLM), is the first program that automatically produces MeSH indexing recommendations. Given the title and abstract for an article in MEDLINE, MTI will provide a ranked list of MeSH terms. The initial MTI system was developed in 2002, and has been continuously improved over the years. There are two main components in MTI, namely, MetaMap (Aronson and Lang, 2010), and PubMed Related Citations (PRC) (Lin and Wilbur, 2007). MetaMap analyzes documents and annotates them using the Unified Medical Language System (UMLS)[6]. The PRC algorithm[7] with k-nearest neighbours (k-NN) uses document similarity to find MeSH terms. MTI is an important tool in MeSH indexing, and indexers can use MTI suggestions for documents that they are annotating. Another method, Restrict-To-Mesh (Kin-Wah Fung, 2007) also maps from UMLS to MeSH terms.

BioASQ[8], a European Union-funded project, has organized challenges on automatic MeSH indexing since 2013. Participants are required to annotate unlabelled PubMed citations with abstracts and titles using their models before these articles are indexed by human annotators. The winning system in 2013, for example, used the MetaLabeler algorithm (Tang et al., 2009) to learn two models, one for ranking and the other for predicting the number of related labels. MeSHLabeler (Liu et al., 2015) won first place in 2014. It also has two components: MeSHRanker and MeSHNumber. MeSHRanker returns a ranked list of candidate MeSH terms. MeSHNumber predicts the number of output MeSH terms. DeepMeSH (Peng et al., 2016) was the best system in 2017. It incorporates deep semantic information into MeSHLabeler using a dense semantic representation for documents, namely document to vectors (D2V). In addition, DeepMeSH has a second classifier to find the number of MeSH terms returned. AttentionMeSH (Jin et al., 2018), also proposed in 2017, uses a bi-direction recurrent gated unit (Bi-GRU) architecture to capture contextual features, and attention mechanisms to select MeSH terms from the candidate list.

Rios and Kavuluru (2015) used a convolutional neural network (CNN) to classify the 29 most frequent MeSH terms on a small dataset comprised of 9,000 citations. Gargiulo et al. (2018) applied deep CNN on the abstracts and titles of 1,115,090 articles. Besides deep learning approaches, other

[6] https://www.nlm.nih.gov/research/umls/
[7] https://ii.nlm.nih.gov/MTI/Details/related.shtml
[8] http://bioasq.org

machine learning algorithms have also been explored in the hopes of solving MeSH indexing tasks. A few examples are: Naïve Bayes (NB), support vector machines (SVM), linear regression, and AdaBoost (Jimeno-Yepes et al., 2012, 2013).

3 Proposed Model

3.1 Problem Statement

Multi-label classification studies the problem where each document is associated with a set of labels (Zhang and Zhou, 2014). In the MeSH indexing problem, each MeSH term can be treated as a class label and each biomedical article can have multiple MeSH terms. Because of the large number of MeSH terms we regard automatic MeSH term indexing as an extreme multi-label classification problem.

The learning framework is defined as follows. Suppose $\mathcal{X}$ is a set of biomedical documents (at this point we won't prejudice how these documents are represented, these representational details are discussed below) and $\mathcal{Y}$ is the set of MeSH terms. Multi-label classification studies the learning function $f : \mathcal{X} \rightarrow 2^{\mathcal{Y}}$ using the training set $\mathcal{D} = (x_i, Y_i), i = 1 \ldots D$, where D is the number of documents in the set $\mathcal{X}$. Each instance x_i is an n-dimensional vector, where n is the number of words in document x_i, and $Y_i \subseteq \mathcal{Y}$ is the set of labels associated with instance x_i. The objective of multi-label classification is to predict the proper label set Y_k for any unseen instance x_k (Zhang and Zhou, 2014).

Two challenges should be considered when solving automatic MeSH indexing tasks (Zhai et al., 2015). First, the number of MeSH terms is large and they have widely varying occurrence frequencies. There are around 29,000 MeSH term and they are updated annually. The frequency of each MeSH term appearing as a document label is quite biased. For instance, of the 29,000 MeSH terms, the most frequent term "Humans", appears in 8,152,852 citations; and "Pandanaceae", on the other hand, only appears in 31 documents (Zhai et al., 2015). Second, the number of MeSH terms assigned to each document varies. Some documents have more than 30 MeSH terms and some have fewer than 5. In this paper, we have used the 2018 version of MeSH which contains 28,939 headings in total.

3.2 Model Overview

We propose multichannel TextCNN, a novel deep learning approach to assign proper MeSH terms to given documents. To make use of multimodal features, our model has two input channels:

- Channel 1: word embeddings from abstract and title
- Channel 2: word embeddings from figure and table captions and corresponding paragraphs that mention the figures and tables

As promised above, we now discuss the representational details of a document. A document is composed of n words. We use d-dimensional word embeddings to represent the words. The word embedding matrix e for each document is then $e \in \mathbb{R}^{d \times n}$. For each document, we have two texts: the abstract and title, and the captions and paragraphs. These two texts are represented by two embedding matrices, namely e_{AT}, the word embedding matrix for the abstract and title text, and e_{CP}, the word embedding matrix for the captions and paragraphs.

The model structure, shown in Figure 1, is a variant convolutional neural network (CNN) with multichannel inputs, which is inspired by TextCNN (Kim, 2014). We have chosen the CNN-based model because it has been successful in various text classification tasks. For each channel, the architecture is similar to TextCNN. The representation of abstract and title, e_{AT}, is input to one channel. The representation of captions and paragraphs, e_{CP}, is input to the other. We also use a single channel architecture by concatenating these two representations as input to one of the channels.

The model learns feature representations by passing embedded documents to the convolutional layer. The entire input document in each channel can be represented as $e_{1:n} = [e_1, e_2, \ldots, e_n] \in \mathbb{R}^{d \times n}$, where n is the length of the document and $e_i \in \mathbb{R}^d$, where e_i represents the i-th word in the document. The convolutional layer is composed of 128 convolutional filters each with sizes 3, 4, and 5. Recalling, we have d-dimensional word embedding vectors. So, the convolutional windows are $m \times d$, where $m \in \{3, 4, 5\}$.

In the convolutional layer, we have 128 feature maps for each filter size. The feature maps are then passed to a pooling layer which takes the maximum value for each associated feature map. After pooling, we get the feature map for each chan-

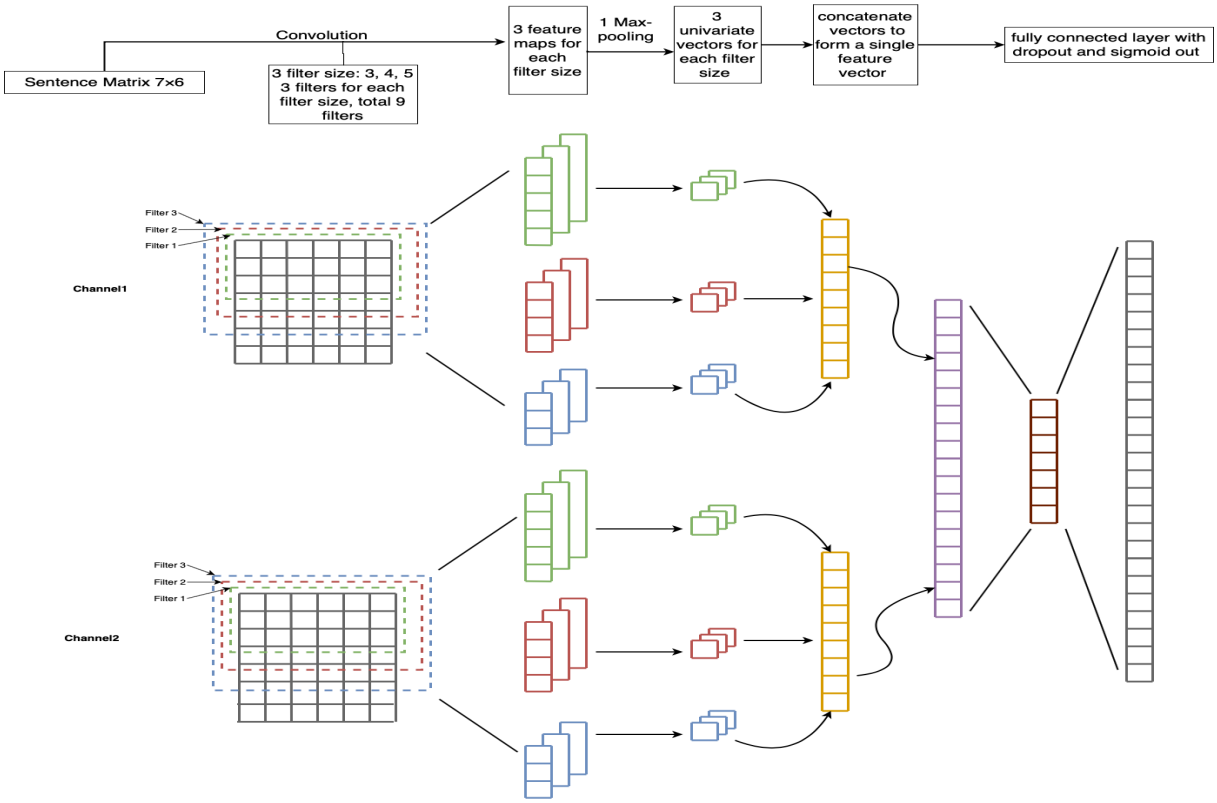

Figure 1: Multichannel TextCNN Architecture - filter 1, filter 2, and filter 3 indicate convolutional filters of size 3, 4, and 5, respectively. In this figure, we characterized our model with a 7×6 input document, where the number of words in the document n is 7, and the dimensionality of the word embedding d is 6.

nel and we concatenate these two feature maps to form a single feature vector. This feature vector is then passed to a fully connected bottleneck layer with 512 hidden units followed by a sigmoid classifier that returns a probability value for each of the 28,939 MeSH terms.

The training of our proposed methods uses binary cross-entropy as the loss function on the sigmoid classifier. We use the sigmoid function to return the probability score of each class. The sigmoid function is defined as:

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

Binary cross-entropy is formulated as:

$$H(q) = -\frac{1}{L}\sum_{i=1}^{L} y_i \cdot log(\sigma(y_i)) + (1 - y_i) \cdot log(1 - \sigma(y_i))$$

where σ is the sigmoid function, L is the total number of labels, y_i is the original label of document i, and $\sigma(y_i)$ is the predicted probability of label y for document i. The sigmoid binary cross-entropy optimizes a label one-versus-all loss based on max-entropy.

We have also experimented with multichannel XMLCNN, which is inspired by XMLCNN (Liu et al., 2017), a variant CNN model developed for extreme multi-label classification; multichannel biLSTM, a bidirectional long short term memory neural network (Schuster and Paliwal, 1997) with multichannel inputs; and multichannel attention based convLSTM, which is a stacked CNN and LSTM followed by an attention layer. The experimental results indicate that the multichannel TextCNN model performed best among all of the models mentioned above in both execution time and evaluation metrics. Details of these experiments are available (Wang, 2019).

3.3 Setup and Model Hyper-parameters

In our proposed multichannel TextCNN model, we used rectified linear units (ReLU) as the activation function, convolutional filter windows of size 3, 4, and 5 with 128 filters each, dropout rate of 0.5, 512 hidden units in the bottleneck layer, batch size of 10, and learning rate of 0.001. The number of epochs in training is 20. These hyper-parameters are fixed across the different

datasets. In each dataset, we used 90% of the data as the training set and 10% to test the performance of the model. We reserved 20% of the training data, chosen randomly, as the validation set, and the remaining 80% is used for training the model. All experiments are performed on the Nvidia GeForce 1080Ti GPU. Models for Fulltext (Large) are trained on 2 GPUs and training with the other datasets is performed on a single GPU.

For word embeddings in our proposed model we used the pre-trained 200-dimensional BioASQ word embedding vectors (Pavlopoulos et al., 2019) to represent the words in our vocabulary. These pre-trained word vectors are trained on 10,876,004 English biomedical abstracts from PubMed, and represent 1,701,632 distinct words.

4 Experiments

4.1 Datasets

Most existing approaches in automatic MeSH indexing are performed on datasets with abstracts and titles only. In this paper, we created a full text dataset which is composed of table and figure captions as well as associated paragraphs, as we believe figures and tables might provide important MeSH features for classification. The two datasets that were used to build our four datasets are described below:

- **2015 Subject Extraction Test Collection (SETC2015)**: SETC2015 contains 14,828 PMC full text articles used by Demner-Fushman and Mork (2015). We used this dataset to create the following two Small (S) datasets:
  - **AT (S)**: labelled documents from SETC2015 which contain abstract and title only
  - **Full (S)**: labelled documents from SETC2015 which contain abstract, title, figure and table captions, and associated paragraphs
- **PMC Full Text Collection**[9] **(PMC Collection)**: We used a downloaded dataset of 257,590 PMC full text documents in XML format, and used this dataset to create the following two Large (L) datasets:
  - **AT (L)**: labelled documents from PMC Collection which are composed of abstract and title only
  - **Full (L)**: labelled documents from PMC Collection which are composed of abstract, title, figure and table captions, and associated paragraphs

[9] https://www.ncbi.nlm.nih.gov/pmc/tools/ftp/

Datasets	D	F	L	$\bar{L}$	$\tilde{L}$
AT (S)	14828	63004	14365	13.15	13.5
Full (S)	14828	148330	14365	13.15	13.5
AT (L)	257590	188693	22881	13.34	150
Full (L)	257590	669999	22881	13.34	150

Table 1: Statistics of Datasets: D is the total number of documents (90% training, 10% testing); F represents the number of unique tokens contained in all of the documents; L is the number of class labels; $\bar{L}$ is the average number of labels per document; $\tilde{L}$ is the average number of documents per label

Table 1 provides statistical information for the described datasets. Our labeled datasets are using 28,939 MeSH terms in total. To assist in our understanding of the hierarchical evaluation, we explored the MeSH hierarchical structure and split them into 5 levels to see how many MeSH terms exist at each level (it should be noted that there is some overlap of MeSH terms between levels). The number of MeSH terms in the first, the second, the third, the fourth and the fifth level, are: 16, 120, 1903, 6,808, and 11,127, respectively.

4.2 Data pre-processing

The full text source files from PMC are in XML format. We extracted article information (including PMID, abstract, title, captions, and figure and table related paragraphs) from these downloaded XML files. Paragraphs are considered related to figures or tables if they contain the words "Figure" or "Table". MeSH terms for each article were scraped from its citation on PubMed by locating the citation using its PMID, the unique article identifier number used in PubMed.

In pre-processing, we first did word level tokenization of our input documents, to split the documents into a list of words. Then we prepared our data by using the following process: set all characters to lowercase; convert numbers to "NUM", percentage sign "%" to "PERCENTAGE", chemical notations (i.e., H_2O) to "CHEM", and relation symbols, namely "=", "<", ">", "≤", "≥",

to "EQUAL", "LESS", "GREATER", "LessAndEqual", "GreaterAndEqual"; remove punctuation.

After the above process, we utilized the Keras (Chollet et al., 2015) Tokenizer API to vectorize our data into a sequence of integers. Each integer represents the index of a token in the dictionary generated from the dataset.

4.3 Evaluation Metrics

There is generally no accepted standard for the evaluation of multi-label classifications. Evaluation metrics adopted from multi-class classification and binary classification are used to measure multi-label classification in an effective way. In automatic MeSH indexing, even if the label space is very large, only relatively few MeSH terms match each document. To evaluate the performance of our proposed model, we present three groups of measurements suggested by Tsoumakas et al. (2010) and Kosmopoulos et al. (2015), namely bipartition-based, ranking-based and hierarchy-based evaluation.

To set the stage to discuss the three metrics, we define a test set of N document-label pairs $\{x_i, y_i\}_{i=1}^{N}$ taken from the dataset, where x_i is the document text and $y_i \in \{0,1\}^L$. The vector y_i denotes the set of true labels (i.e., MeSH terms) for each document i (0 meaning the label is not in the set, 1 meaning it is in the set), N denotes the number of test examples, and L is the total number of labels. Given a document x_i, the set of labels predicted by the classifiers is denoted as $\{\hat{y}_i\}_{i=1}^{N}$, where $\hat{y}_i \in \{0,1\}^L$, and the ranking indexes of predicted labels among the top k is denoted as $r_k(\hat{y})$, where $\hat{y} = \{\hat{y}_i\}_{i=1}^{N}$.

Bipartition evaluation is further divided into example-based and label-based metrics. Example-based measurements calculate precision, recall, and F-score over (in our evaluation) the top 5, top 10, and top 15 ranked labels over all of the documents of the test set. The measurements include example-based precision (*EBP*), example-based recall (*EBR*) and example-based F-score (*EBF*). The metrics are defined as:

$$EBP = \frac{1}{N}\sum_{i=1}^{N}\frac{|y_i \cap \hat{y}_i|}{|\hat{y}_i|}$$

$$EBR = \frac{1}{N}\sum_{i=1}^{N}\frac{|y_i \cap \hat{y}_i|}{|y_i|}$$

$$EBF = \frac{1}{N}\sum_{i=1}^{N}\frac{2 \times |y_i \cap \hat{y}_i|}{|y_i| + |\hat{y}_i|}$$

Label-based evaluation is calculated for each label in the label set. The measurements include macro- and micro-average precision (*MaP, MiP*), macro- and micro-average recall (*MaR, MiR*),and macro- and micro-average F-score (*MaF, MiF*). The metrics are defined as:

$$MaP = \frac{1}{L}\sum_{j=1}^{L}\frac{TP_j}{TP_j + FP_j}$$

$$MiP = \frac{\sum_{j=1}^{L} TP_j}{\sum_{j=1}^{L} TP_j + \sum_{j=1}^{L} FP_j}$$

$$MaR = \frac{1}{L}\sum_{j=1}^{L}\frac{TP_j}{TP_j + FN_j}$$

$$MiR = \frac{\sum_{j=1}^{L} TP_j}{\sum_{j=1}^{L} TP_j + \sum_{j=1}^{L} FN_j}$$

$$MaF = \frac{2 \times MaR \times MaP}{MaR + MaP}$$

$$MiF = \frac{2 \times MiR \times MiP}{MiR + MiP}$$

where TP_j, FP_j and FN_j as true positives, false positives, and false negatives respectively for each label l_j in the set of total labels L.

Ranking-based evaluation, including precision at k ($p@k$), and normalized discounted cumulative gain (*nDCG*), ranks the predicted labels and aims to rank the relevant labels higher than the irrelevant ones. The metrics are defined as follows:

$$p@k = \frac{1}{k}\sum_{l \in r_k(\hat{y})} y_l$$

$$DCG@k = \sum_{l \in r_k(\hat{y})}\frac{y_l}{log\,(l+1)}$$

$$IDCG = \sum_{l=1}^{min(k, \|y\|_0)}\frac{1}{log\,(l+1)}$$

$$nDCG@k = \frac{DCG@k}{IDCG}$$

Hierarchy-based evaluation, including hierarchical precision (*HP*) and hierarchical recall (*HR*), is used to measure a hierarchical classification that classifies elements into a hierarchy of classes. It measures performance based on the gold standard labels and the predicted labels augmented with

their ancestors and descendants within distances 1 and 2. The augmented gold standard labels Y_{aug} and predicted labels $\hat{Y}_{aug}$ are used in the hierarchical evaluation. *HP* and *HR* are defined as follows:

$$HP = \frac{1}{N}\sum_{i=1}^{N}\frac{|\hat{Y}_{aug} \cap Y_{aug}|}{|\hat{Y}_{aug}|}$$

$$HR = \frac{1}{N}\sum_{i=1}^{N}\frac{|\hat{Y}_{aug} \cap Y_{aug}|}{|Y_{aug}|}$$

In ranking-based evaluation, for $p@k$, $k \in \{1, 3, 5, 10, 15\}$, and $k \in \{1, 3, 5\}$ for *nDCG@k*. In example-based, label-based, and hierarchy-based evaluations, the calculation is done with the top 5, 10, and 15 predicted labels. In hierarchical evaluation, we used distances 1 and 2 for *HP* and *HR*. The example-based, ranking-based, and hierarchical evaluation metrics are calculated for each document. An average score over all documents in the test set is returned. Likewise, the label-based evaluation is calculated for each label and averaged over all labels in the test set.

5 Results

We first conducted our experiments on datasets with titles and abstracts only (designated AT), passing the appropriate word embeddings to the single channel TextCNN. Next, we did our experiments on the full text datasets (designated Full), passing the word embeddings for titles, abstracts, captions and paragraphs to the single channel TextCNN. Finally, we conducted our experiments on the full text datasets using the multichannel model: we passed word embeddings for titles and abstracts to the first channel, and word embeddings for captions and paragraphs to the second channel. Four datasets have been used: two Small datasets (comprised of text from SETC2015)—AT (S) and Full (S)—and two Large datasets (comprised of text from PMC Collection)—AT (L) and Full (L).

The $p@k$ and *nDCG@k* performance of the single channel TextCNN and the multichannel TextCNN on all four datasets is summarized in Table 2. Each row in the table compares all datasets on a specific metric, where the best score for each metric (the Small and Large datasets being observed separately) is in boldface. The results clearly indicate that when dealing with the same dataset, multi-channel TextCNN performs the best, which indicates that integrating captions and paragraphs indeed helps to improve the performance of classification. Also, the multichannel TextCNN outperforms the single channel TextCNN, suggesting that the multi-channel TextCNN architecture has an advantage over the single channel TextCNN architecture. The reason for this could be that the single channel model misses some important features in the captions and related paragraphs in the convolutional and pooling layers. To be more explicit, the single channel model may be extracting insignificant features in the convolutional layer from which the max-pooling layer can take only one value in each filter. Further observation indicates that for the Small dataset, the Fulltext single channel TextCNN outperforms the AT TextCNN model (with only the $p@10$ and $p@15$ results differing from this general trend), while interestingly for the Large dataset, the AT TextCNN model outperforms the Fulltext single channel TextCNN model by a wide margin.

Now looking only at the multichannel TextCNN model when comparing the results, using data that comes from abstracts and titles only (AT) with the Fulltext data, the Small dataset shows the

		Datasets					
Metrics		AT (S)	Full (S) Single_Channel	Full (S) Multichannel	AT (L)	Full (L) Single_Channel	Full (L) Multichannel
p@k	$p@1$	0.76197	0.78220	**0.80512**	0.87600	0.72305	**0.87907**
	$p@3$	0.58283	0.59699	**0.62980**	0.70951	0.51016	**0.72139**
	$p@5$	0.47633	0.48901	**0.52057**	0.60532	0.41908	**0.61479**
	$p@10$	0.39641	0.38815	**0.41958**	0.51000	0.32631	**0.51793**
	$p@15$	0.37281	0.35910	**0.39587**	0.47127	0.28318	**0.48009**
nDCG@k	*nDCG*@1	0.76197	0.78219	**0.80512**	0.87600	0.72305	**0.87907**
	nDCG@3	0.62306	0.63744	**0.66982**	0.74737	0.55327	**0.75737**
	nDCG@5	0.53918	0.55227	**0.58409**	0.66640	0.48009	**0.67521**

Table 2: Results for TextCNN in $p@k$ and *nDCG@k*. Boldface indicates the best result on the each dataset.

		Metrics										
Datasets	top_*k*	*EBP*	*EBR*	*EBF*	*MiP*	*MaP*	*MiF*	*MaF*	HP_1	HR_1	HP_2	HR_2
AT (S)	@5	0.477	0.187	0.261	0.498	0.499	0.456	0.478	0.521	0.135	0.605	0.144
	@10	0.349	0.253	0.288	0.473	0.498	0.456	0.478	0.381	0.208	0.456	0.225
	@15	0.301	0.278	0.288	0.458	0.497	0.453	0.478	0.325	0.241	0.399	0.262
Full (S) Single Channel	@5	0.490	0.185	0.261	0.499	0.500	0.456	0.478	0.521	0.126	0.578	0.129
	@10	0.345	0.245	0.281	0.472	0.499	0.454	0.478	0.338	0.193	0.377	0.212
	@15	0.291	0.267	0.277	0.455	0.498	0.449	0.478	0.281	0.221	0.329	0.250
Full (S) Multi-channel	@5	0.521	0.200	0.282	0.503	0.498	0.460	0.478	0.539	0.161	0.608	0.176
	@10	0.377	0.270	0.309	0.478	0.495	0.460	0.477	0.386	0.237	0.442	0.265
	@15	0.325	0.298	0.310	0.463	0.494	0.457	0.477	0.326	0.268	0.380	0.304
AT (L)	@5	0.606	0.239	0.332	0.575	0.502	0.364	0.398	0.632	0.196	0.685	0.197
	@10	0.462	0.334	0.380	0.479	0.500	0.407	0.401	0.478	0.304	0.532	0.315
	@15	0.404	0.372	0.386	0.434	0.497	0.414	0.403	0.413	0.352	0.468	0.371
Full (L) Single Channel	@5	0.420	0.162	0.227	0.446	0.500	0.282	0.396	0.394	0.100	0.405	0.091
	@10	0.290	0.206	0.236	0.338	0.500	0.287	0.396	0.313	0.119	0.336	0.105
	@15	0.240	0.220	0.228	0.290	0.500	0.277	0.396	0.223	0.135	0.262	0.137
Full (L) Multi-channel	@5	0.616	0.243	0.338	0.581	0.503	0.369	0.400	0.640	0.199	0.702	0.199
	@10	0.468	0.339	0.386	0.484	0.500	0.413	0.403	0.492	0.307	0.558	0.319
	@15	0.411	0.379	0.392	0.440	0.497	0.420	0.405	0.426	0.357	0.491	0.378

Table 3: Flat and Hierarchical Measures for TextCNN on Different Datasets. top_*k* indicates the top *k* labels returned by the classifier; *EBP*, *EBR*, *EBF* are example based precision, recall, and F-score, respectively; *MiP* and *MiF* are micro precision and F-score; *MaP* and *MaF* are macro precision and F-score; HP_m and HR_m are hierarchical precision, where m denotes the maximum distance from the original label to its ancestors and descendants.

greater improvement, approximately 2-5 percentage points for each *p@k* and each *nDCG@k* value. The improvement for the Large dataset is typically closer to 1 percentage point. It should be noted that the Large dataset has a somewhat higher, thus more difficult to improve upon, abstract and title baseline for each metric (10 percentage points or more than the Small dataset). Another reason for this difference could be that more training examples simply gives better models, so the extra information provided by the new data sources does not have as significant an effect as the increase in the number of training examples. Comparing AT (L) to AT (S) and Full (L) to Full (S) shows an approximately 7-13 percentage point improvement for each *p@k* and *nDCG@k*. Another possibility could be that the Small and Large datasets were generated from documents with different attributes. We have not investigated this possibility.

Table 3 reports the performance of flat and hierarchical evaluations on all datasets giving a further assessment of introducing the extra information sources. When comparing AT to Full Multi-channel in the Small and Large datasets, we see an approximate .5-5 percentage point improvement in all of the measures except *MaP*. Most importantly, there is improvement in precision without a decrease in recall. The obtained results further suggest that our hypothesis that adding captions and paragraphs indeed provides valuable information in automatic MeSH indexing. Comparing *EBP*, which is the same as HP_0, with the *HP* values, an approximate 1-5 percentage point improvement in all cases at HP_1 and an approximate 6-13 percentage point improvement in all cases at HP_2 can be seen. These observations indicate that some of the predicted MeSH terms are not exactly the same as the gold standard labels, but the model has suggested MeSH terms that are in the correct branch of the MeSH term hierarchy. With this latter observation we have investigated how the predicted results correspond to the gold standard results. To do this investigation, we look at the parents above and the children below the predicted labels.

An in-depth analysis of the hierarchical evaluation on the AT (L) and Full (L) datasets are reported in Table 4. We have computed the average number of gold standard MeSH term labels in common with the predicted labels including *m* levels up and *n* levels down over all documents, where $m \in \{0, 1, 2\}$ and $n \in \{0, 1, 2\}$. Each row in the table compares model performance at a certain MeSH hierarchy, where C_m indicates the predicted label augmented with children with dis-

	top_5_predicted			top_10_predicted			top_15_predicted		
	AT (L)	Full (L) Single Channel	Full (L) Multi-channel	AT (L)	Full (L) Single Channel	Full (L) Multi-channel	AT (L)	Full (L) Single Channel	Full (L) Multi-channel
$C_0\_P_1$	0.0256	0.0021	0.0748	0.0236	0.0008	0.1107	0.0290	0.0067	0.1424
$C_1\_P_0$	0.1119	0.4154	0.4551	0.2420	0.5040	0.8545	0.2791	0.5763	1.0379
$C_0\_P_2$	0.2387	0.1988	0.4904	0.2933	0.1536	0.6983	0.3405	0.2741	0.8269
$C_2\_P_0$	0.1591	0.4860	0.6528	0.3202	0.6480	1.1166	0.3681	0.7296	1.3542
$C_2\_P_1$	0.1847	0.4881	0.7276	0.3438	0.6489	1.2267	0.3971	0.7363	1.4954
$C_1\_P_2$	0.3506	0.6142	0.9455	0.5354	0.6576	1.5528	0.6196	0.8504	1.8649

Table 4: Hierarchical Analysis on TextCNN - top_k_selected indicates the top k labels return by the classifier

tance m, and P_n is predicted label augmented with parents with distance n. As an example: $C_0\_P_1$ on AT (L) with the top 5 predicted labels indicates that if the predicted labels are augmented with their parents with distance *1*, the number of common labels between true labels and predicted ones will increase on average by *0.0256* over all documents in the test set. For each top_k_predicted labels returned by the TextCNN model, comparisons within the same dataset but expanded augmentations show that the number of common MeSH terms between the gold standard and predicted ones increase in all but four cases: two instances of an increased window size for the multichannel TextCNN, the single channel TextCNN augmented with two parent labels, and the AT (L) dataset for top_5_predicted. Observing each column, this can a ten-fold increase or more. Comparing AT with Full multichannel TextCNN the increase is approximately three times when adding captions and related texts. This observation gives us confidence in concluding that the multichannel TextCNN model gives MeSH terms that are in the correct branch of the MeSH hierarchy and adding figure captions and related texts does provide valuable improvement in automatic MeSH indexing.

6 Conclusions and Future Work

This paper has presented a novel multichannel TextCNN model for MeSH term indexing. In addition, this paper has included figure and table information for the automatic MeSH indexing task. Notably, our deep learning model introduced a variety of features obtained from different parts of the document. The experimental results indicate that adding more features obtained from captions and related paragraphs indeed improve the performance of our proposed multi-channel TextCNN model, supporting the initial hypothesis that figure and table captions as well as associated paragraphs provide valuable evidence in automatic MeSH indexing. In addition, introducing the extra information in a separate channel appears to have a positive effect compared with presenting all of the information in one channel.

We have contributed a labeled text-enhanced biomedical document dataset for the research community. It includes title, abstract, figure and table captions, and paragraphs related to figures and tables. This dataset and our software is available at https://github.com/xdwang0726/Mesh.

In the future, we first intend to extend our experiments on different optimizers, learning rates and classifiers in order to improve the performance of our models. Secondly, in this paper, we focused on finding a classifier to capture important features in the document. We manually set the number of MeSH terms returned from the model, i.e., in this work, we asked our model to return the top k predicted MeSH terms, where $k \in \{5, 10, 15\}$. We plan to improve our model by implementing a ranking system module which can be added right after the classifier. The ranking module would automatically suggest the number of labels returned for each document, which could help the indexing system to return more accurate MeSH terms. Thirdly, we also aim to develop a tool which could help human annotators locate the places in the document that has text important for determining MeSH terms in order to improve the efficiency of computer assisted human MeSH indexing.

Acknowledgements This research is partially funded by The Natural Sciences and Engineering Research Council of Canada through a Discovery Grant to Robert E. Mercer. We also acknowledge the helpful comments provided by the reviewers.

References

Alan R Aronson and François-Michel Lang. 2010. An overview of MetaMap: Historical perspective and recent advances. *Journal of the American Medical Informatics Association*, 17(3):229–236.

Alan R. Aronson, James G. Mork, Clifford W. Gay, Susanne M. Humphrey, and Willie J. Rogers. 2004. The NLM Indexing Initiative's Medical Text Indexer. *Studies in Health Technology and Informatics*, 107 Pt 1:268–272.

François Chollet et al. 2015. Keras. https://github.com/fchollet/keras.

Dina Demner-Fushman and James G. Mork. 2015. Extracting characteristics of the study subjects from full-text articles. In *Proceedings of the American Medical Informatics Association (AMIA) Annual Symposium*, pages 484–491.

Francesco Gargiulo, Stefano Silvestri, and Mario Ciampi. 2018. Deep convolution neural network for extreme multi-label text classification. In *Proceedings of the 11th International Joint Conference on Biomedical Engineering Systems and Technologies – Volume 5: AI4Health*, pages 641–650.

Antonio Jimeno-Yepes, James G. Mork, Dina Demner-Fushman, and Alan R. Aronson. 2012. 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.

Antonio Jimeno-Yepes, James G. Mork, Dina Demner-Fushman, and Alan R. Aronson. 2013. Comparison and combination of several MeSH indexing approaches. In *Proceedings of the American Medical Informatics Association (AMIA) Annual Symposium*, pages 709–718.

Qiao Jin, Bhuwan Dhingra, and William W. Cohen. 2018. AttentionMeSH: Simple, effective and interpretable automatic MeSH indexer. In *Proceedings of the 2018 EMNLP Workshop BioASQ: Large-scale Biomedical Semantic Indexing and Question Answering*, pages 47–56.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751.

Olivier Bodenreider KinWah Fung. 2007. Utilizing the umls for semantic mapping between terminologies. In *Proceedings of the American Medical Informatics Association (AMIA) Annual Symposium*, pages 266–270.

Aris Kosmopoulos, Ioannis Partalas, Eric Gaussier, Georgios Paliouras, and Ion Androutsopoulos. 2015. Evaluation measures for hierarchical classification: A unified view and novel approaches. *Data Mining and Knowledge Discovery*, 29(3):820–865.

Jimmy Lin and W. John Wilbur. 2007. Pubmed related articles: A probabilistic topic-based model for content similarity. *BMC Bioinformatics*, 8(1):423.

Jingzhou Liu, Wei-Cheng Chang, Yuexin Wu, and Yiming Yang. 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.

Ke Liu, Shengwen Peng, Junqiu Wu, ChengXiang Zhai, Hiroshi Mamitsuka, and Shanfeng Zhu. 2015. MeSHLabeler: Improving the accuracy of large-scale mesh indexing by integrating diverse evidence. *Bioinformatics*, 31(12):i339–i347.

James G. Mork, Antonio Jimeno-Yepes, and Alan R. Aronson. 2013. The NLM Medical Text Indexer system for indexing biomedical literature. In *Proceedings of the first Workshop on Bio-Medical Semantic Indexing and Question Answering (BioASQ)*. CEUR-WS.org, online http://CEUR-WS.org/Vol-1094/bioasq2013_submission_3.pdf.

Ioannis Pavlopoulos, Aris Kosmopoulos, and Ion Androutsopoulos. 2019. Continuous space word vectors obtained by applying word2vec to abstracts of biomedical articles. Retrieved from http://bioasq.lip6.fr/info/BioASQword2vec/.

Shengwen Peng, Ronghui You, Hongning Wang, ChengXiang Zhai, Hiroshi Mamitsuka, and Shanfeng Zhu. 2016. Deepmesh: deep semantic representation for improving large-scale mesh indexing. *Bioinformatics*, 32(12):i70–i79.

Anthony Rios and Ramakanth Kavuluru. 2015. Convolutional neural networks for biomedical text classification: Application in indexing biomedical articles. In *Proceedings of the 6th ACM Conference on Bioinformatics, Computational Biology and Health Informatics*, pages 258–267.

Mike Schuster and Kuldip K. Paliwal. 1997. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681.

Lei Tang, Suju Rajan, and Vijay K. Narayanan. 2009. Large scale multi-label classification via MetaLabeler. In *Proceedings of the 18th International World Wide Web Conference (WWW)*, pages 211–220.

Grigorios Tsoumakas, Ioannis Katakis, and Ioannis P. Vlahavas. 2010. Mining multi-label data. In *Data Mining and Knowledge Discovery Handbook (2nd ed.)*, pages 667–685. Springer, Boston, MA.

Xindi Wang. 2019. Incorporating figure captions and descriptive text in mesh term indexing: A deep learning approach. Master's thesis, The University of Western Ontario.

Chengxiang Zhai, Hiroshi Mamitsuka, Junqiu Wu, Ke Liu, Shanfeng Zhu, and Shengwen Peng. 2015. MeSHLabeler: Improving the accuracy of large-scale MeSH indexing by integrating diverse evidence. *Bioinformatics*, 31(12):i339–i347.

M. Zhang and Z. Zhou. 2014. A review on multi-label learning algorithms. *IEEE Transactions on Knowledge and Data Engineering*, 26(8):1819–1837.