PubMeSH: Extreme Multi-label Classification of Biomedical Research

Kevin Thomas
kevin.a.thomas@stanford.edu

Rohan Paul
ropaul@stanford.edu

Mia Kanzawa
mkanzawa@stanford.edu

Abstract

Medical Subject Headings (MeSH) is a comprehensive controlled vocabulary, which has been developed and maintained by National Library of Medicine (NLM). Over 29,000 MeSH terms are used to label multiple topics of each scientific abstract found in the PubMed database. Accurate MeSH indexing of documents is crucial in order for biomedical researchers to discover and disseminate new knowledge. This task is currently performed by human experts at the NLM for thousands of abstracts per year. This report presents work to automate the process of assigning MeSH terms to articles in PubMed. We build on past work on this task with two novel strategies: (1) accounting for the context of MeSH labels when making predictions and (2) ensembling several models that each specialize on predicting MeSH labels of a certain frequency range. Results show that these strategies yield a final model that makes predictions comparable to state-of-the-art for common MeSH terms and demonstrates effective few-shot learning for rare MeSH terms. We also provide evidence that the model may be approaching human-level performance in certain contexts.

1 Introduction

MEDLINE is a freely available database maintained by the U.S. National Library of Medicine (NLM) that contains references to over 50 years of life science articles, totaling more than 29 million citations [1]. Access to this database is achieved through PubMed, a free search engine that uses Medical Subject Headings (MeSH) for information retrieval, and in 2017 3.3 billion MEDLINE/PubMed searches were made. Accurate labelling of articles with MeSH terms is therefore critical to enable discovery and spread of new scientific knowledge. Currently, a large part of this process of assigning labels is done manually, with experts at the NLM selecting typically 10-15 MeSH terms from the database of over 29,000 for each journal article. This is an incredibly time consuming task that is estimated to cost $9.40 per article on average, with more than 800,000 articles indexed per year [2].

Automatically annotating PubMed abstracts with MeSH terms is challenging for several reasons. There are over 29,000 MeSH labels and each abstract can have a variable number of labels. In addition to the large class space, the frequencies of different MeSH terms varies tremendously. The most frequent label, "Humans", is used in over half of all abstracts while 73 MeSH labels are only found once across the full training set of 14 million abstracts. This introduces a difficult class imbalance challenge as well as the need for few-shot learning for many MeSH terms. Professional human indexers assess the full article when assigning MeSH terms, but only the abstracts of the articles are publicly available for training a model. Lastly, the task of choosing from 29,000 MeSH terms with a perfectly consistent strategy and careful consideration of every possible label is presumably difficult for any human, regardless of experience. MeSH labels for individual articles may therefore be influenced by human bias, making it difficult to objectively measure the true performance of any model.

Most work on automatic MeSH labelling, including the work presented here, uses the BioASQ Challenge Task A dataset. This data challenge has been ongoing for several years and has been used to encourage work on this task. It includes roughly 14 million training set abstracts. Each of the abstracts has an average of 13 MeSH labels and there are over 29,000 unique MeSH labels. Until recently, the majority of challenge participants and publications on this task have relied on traditional machine learning approaches. However, gains in performance have been achieved over the past two years through deep learning approaches. The capacity for deep learning models to scale with increased data represents the potential to fully leverage this large biomedical corpus. Recent work has framed it as a multi-class, multi-label problem and has generated a predicted probability for each of the thousands of MeSH labels for each abstract. In our work, we propose two new strategies. First, by using pre-trained word embeddings to embed the MeSH terms in addition to the abstract text, we improve our predictions for rare MeSH terms by using their learned context. Second, by training separate models for common terms, moderate-frequency terms, rare terms, and super-rare terms, and then ensembling them, we avoid many of the difficulties of training a model with severe class imbalance.

2 Related Work

Several automatic annotation models have been developed for MeSH labeling. We discuss the leading model that takes a traditional machine learning approach and one that takes a deep learning approach. As a baseline, we will compare our own model with the leading deep learning model, AttentionMesh.

2.1 MeSH Now

A recent non-deep learning approach is MeSH Now [5]. The authors of this method use a ranking based system to automate the indexing of MeSH terms for the BioASQ challenge. The input to the MeSH Now model is the preprocessed abstract and a list of candidate MeSH terms. The authors acquired these candidates using three approaches:

1. K-Nearest neighbours: Nearest neighbours are found for each abstract using the Pubmed related articles algorithm [4]. 2. Binary text classifiers were trained for 20,000 of the most common MeSH terms. The authors used a weighted huber loss for a support vector machine (SVM) classifier. Each abstract was fed through all classifiers and the positive MeSH predictions were added to the candidate list 3. MeSH Text Indexer (MTI), a baseline rule-based model provided by the BioASQ challenge, was also used to augment the list of candidate MeSH terms.

After this list of candidates was gathered, a ranking algorithm, LambdaMART [6], was used to rank the MeSH terms by neighbourhood features, unigram/bigram overlaps, transition probabilities, query-likelihoods and synonyms. The authors achieved an F1 score of 0.612, the highest of any model at the time. While the performance of the model was good, it required extensive use of hand-engineered features, heuristic post-processing of the results and the development and ensembling of 20,000 individually trained models. As the MeSH vocabulary continues to grow and change, this method would have to be continually re-engineered and retrained to remain effective.

2.2 AttentionMeSH

The current state-of-the-art for automated MeSH classification is AttentionMeSH [3]. AttentionMeSH represents the first end-to-end deep learning model to assign MeSH terms. It uses a Bidirectional GRU encoder to obtain a "context-aware" representation of each word, $\widetilde{w_i}$, in the abstract using pretrained word embeddings as initial input, w_i. These context-aware embeddings are the concatenated forward and backward nodes' hidden states for each input word.

To constrain the number of possible labels for a given input abstract, the authors employ a form of masking. They do so by selecting labels from only the k-nearest neighbor articles to the query article. A more in-depth discussion of this is available in our project proposal.

Attention scores are then calculated between every $\widetilde{w_i}$ and each candidate MeSH term's embedding using multiplicative attention and normalized using a softmax. The embeddings for the MeSH terms were learned from scratch. This differs from our approach, which initializes MeSH embeddings from pretrained word vectors for faster training and increased context for rare MeSH terms. After summing the attention weighted word vectors, the authors used a linear layer with sigmoid activations to get probabilities for each of the candidate MeSH terms. The threshold for each MeSH term is selected to maximize the primary performance metric, Micro-F1 (MiF), for its corresponding term in the validation set. This is calculated as a harmonic mean of micro-precision (MiP) and micro-recall (MiR). Equations are provided further below.

AttentionMeSH achieved a maximum MiF of 0.684 which is the highest in published literature and just under the unpublished state-of-the-art performance of 0.688. Training the AttentionMeSH required 4 days of training on 2 GeForce GTX TITAN X GPUs.

3 Approach

3.1 Preprocessing

We developed a preprocessing pipeline and a data loader for interfacing with models. Preprocessing of input training examples involved removing punctuation, concatenating the journal name, title, year and body of the abstract, and then tokenizing the resulting text. We limited the maximum length of the input to 330 tokens as this value could capture 95% of all abstracts without truncation. Inputs shorter than 330 tokens were padded with a <pad> character. These tokens were then mapped to pre-trained BioASQ PubMed word vectors to create our initial abstract word embedding layers $\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_D)$. $\mathbf{X} \in \mathbb{R}^{D \times e}$, where D represents the 330 length input concatenated text for each article, and e is the dimension of the word embedding.

MeSH label preprocessing included removal of punctuation and tokenization for transformation into pre-trained vector representations. However, since some MeSH terms are composed of several words, we retrieved PubMed embeddings for each word separately and averaged them to result in a single embedding vector. These embeddings were used to initialize MeSH embedding layers. Although the PubMed embeddings were originally trained on article text and not MeSH terms, approximately 99% of MeSH terms were contained within the PubMed embeddings vocabulary. The 1% of MeSH terms that were not included were initialized with random vectors. Both MeSH embeddings and abstract word embeddings were trained separately after initialization.

3.2 Our modeling approach

We implemented (coded ourselves from scratch) a model similar to the baseline model described above, but with an LSTM instead of a GRU, an additional linear layer at the end, and without the neighbor masking approach. This model is based off "AttentionXML", which is described in [7] by You et. al.

The first layer is the abstract word embedding layer described above. The next layer is a bidirectional LSTM. Let $\mathbf{h}^{(t)} \in \mathbb{R}^h$, and $\mathbf{c}^{(t)} \in \mathbb{R}^h$ be the hidden state and cell state respectively that will be calculated from the input $\mathbf{x}^{(t)} \in \mathbb{R}^e$ at time step t. Since the LSTM is bidirectional, the forward and backward hidden states are concatenated for the output $\mathbf{H} \in \mathbb{R}^{D \times 2h}$ of the LSTM, where $\mathbf{H} = (\mathbf{h}_1, \mathbf{h}_2, ..., \mathbf{h}_D)$.

Next is the MeSH embedding layer. To the best of our knowledge, using pre-trained vectors to embed MeSH terms has never been done before. Previous models therefore performed classification by developing a representation of abstract text but without knowing how the context of that text relates to the context of MeSH terms. By representing each MeSH term simply as a classification class without accounting for its actual meaning, these past approaches forfeit the opportunity to directly detect MeSH concepts present in the abstract via attention.

An attention layer was then applied. The output of the attention layer for the j-th label will be denoted as $\mathbf{m}_j \in \mathbb{R}^{2h}$. It is calculated by taking a weighted sum of the LSTM hidden layer outputs from each node. The weights of this weighted sum are referred to as attention coefficients. Attention coefficients are obtained by taking the softmax of the dot product of each hidden layer with each MeSH embedding. These calculations are summarized below:

$$\alpha_{ij} = \frac{e^{\mathbf{h}_i \mathbf{w}_j^T}}{\sum_{l=1}^{D} e^{\mathbf{h}_l \mathbf{w}_j^T}}, \ \mathbf{m}_j = \sum_{i=1}^{D} \alpha_{ij} \mathbf{h}_i$$

Where α_{ij} is the normalized attention coefficient of the i-th hidden vector output and $\mathbf{w}_j$ is the trainable embedding vector for the j-th MeSH term.

Finally, the output of the multi-label attention layer is input into two fully connected layers. The first fully connected layer has ReLU activation and the second, a sigmoid activation. The linear layer weights are shared across all labels within a given model.

To overcome the challenge of class imbalance, we trained 4 models with the same architecture. Each model was responsible for classifying MeSH terms with a specified range of class frequencies within the corpus. The four frequency ranges are specified in Table 1. For example, the "Common" model was trained to identify the common labels within abstracts, such as "Human" and "Animal". Additionally, within each model, we weight the loss of each MeSH term prediction according to the inverse of its frequency in the training corpus. Splitting prediction responsibilities across models based on class frequency is a novel approach for this task. This approach may indirectly allow different models to specialize in detecting concepts at different levels in the MeSH hierarchy. At test time, all 4 models are used to evaluate an abstract and their collective predictions are concatenated into a single vector to provide a complete set of MeSH predictions.

4 Experiments

4.1 Data

We are using the BioASQ Challenge Task A dataset. This includes roughly 14 million training set abstracts with an average of 14 MeSH labels. This represents approximately 15 GB of data. There are a total of 29,351 unique MeSH labels. The MeSH terms are organized into a hierarchy; however, each MeSH term may be placed at several different positions within the hierarchy. Almost all MeSH terms have at least 2 different parents (up to 20) as well as multiple children. Abstracts are labeled with both general parent MeSH terms as well as more specific leaf MeSH terms. However, abstracts are not generally given all labels along the path from the root to the leaf of the label hierarchy. The multi-parent tree structure and the lack of a full path of labels for each observation make it difficult to leverage the hierarchical structure in order to systematically rule out sub-trees and reduce the number of potential labels that the final model must classify over. However, both parent and child terms are considered when evaluating the performance of models.

This dataset poses several interesting challenges. In addition to the large number of potential labels and their unique hierarchical structure, there is large variation in the frequency of different MeSH terms; 90% of MeSH labels apply to <0.08% of articles while the top 3 most common MeSH labels apply to 41-64% of articles. Additionally, human curators have access to full articles when assigning MeSH labels while the gold standard challenge data set for this task only includes titles and abstracts.

The data is provided by the hosts of the challenge as a JSON file with the following format:
["articles": [["abstractText":"text..", "journal":"journal..", "meshMajor":["mesh1",...,"meshN"], "pmid":"PMID", "title":"title..", "year":"YYYY"],..., [...]]]

4.2 Evaluation method

We use MiF to evaluate our model to enable direct comparison with our baseline, the AttentionMeSH model from literature. MiF considers both MiP and MiR and weights them evenly in a final evaluation metric. MiP and MiR describe the precision and recall, respectively, calculated by considering each label output from our model to the ground truth labels:

$$\text{Micro F1} = \frac{2 \times \text{MiP} \times \text{MiR}}{\text{MiP} + \text{MiR}}, \text{ where MiP} = \frac{\sum_{i=1}^{N_a} \sum_{j=1}^{N} y_{ij}\hat{y}_{ij}}{\sum_{i=1}^{N_a} \sum_{j=1}^{N} \hat{y}_{ij}}, \text{ and MiR} = \frac{\sum_{i=1}^{N_a} \sum_{j=1}^{N} y_{ij}\hat{y}_{ij}}{\sum_{i=1}^{N_a} \sum_{j=1}^{N} y_{ij}}$$

Here y_{ij} is the ground-truth jth MeSH term for the ith article, $\hat{y}_{ij}$ is the corresponding predicted MeSH term, N_a is the number of articles in the test set, and N is the number of MeSH terms.

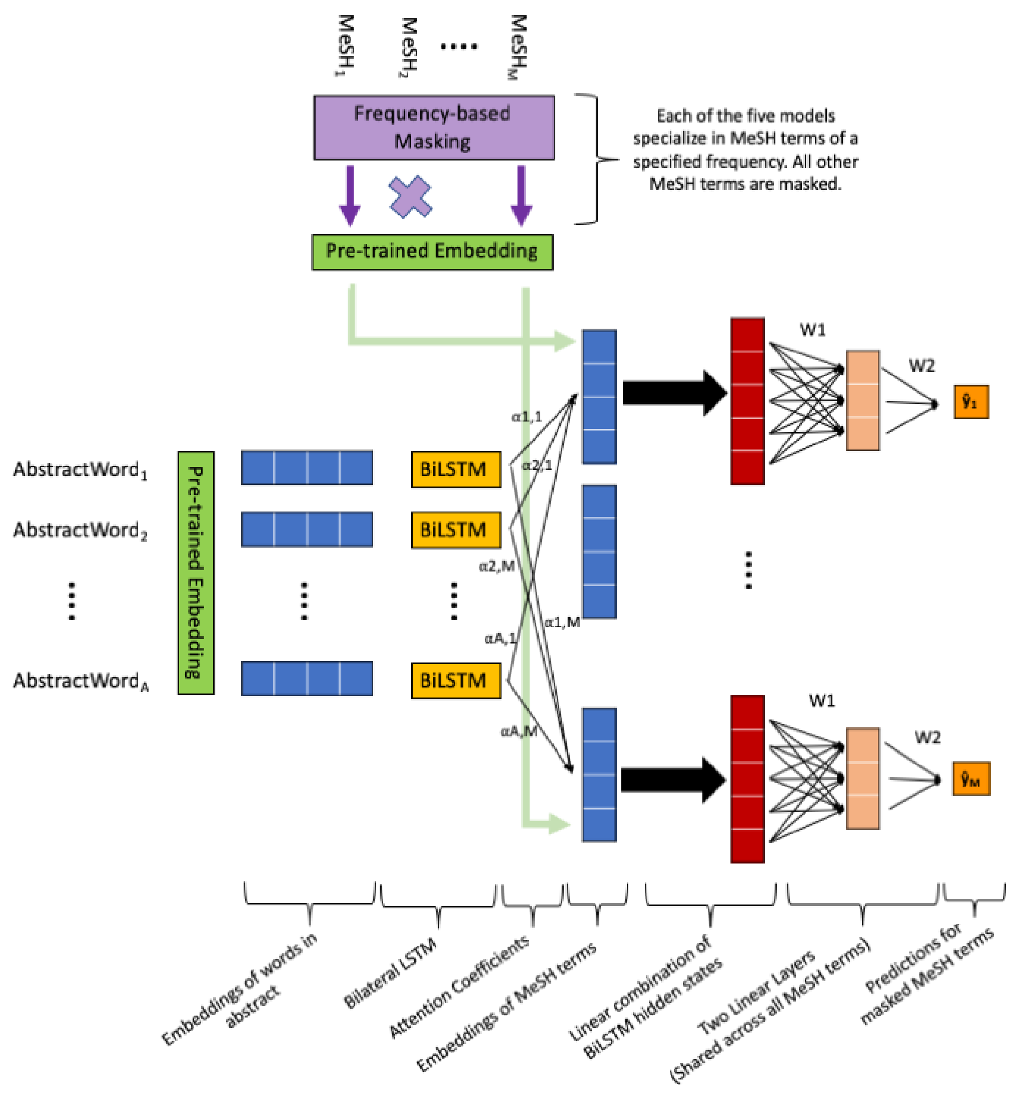
MeSH1
MeSH2
MeSHM
Frequency-based Masking
Each of the five models specialize in MeSH terms of a specified frequency. All other MeSH terms are masked.
Pre-trained Embedding
AbstractWord1
AbstractWord2
AbstractWordA
Pre-trained Embedding
BiLSTM
BiLSTM
BiLSTM
α1,1
α2,1
α2,M
αA,1
α1,M
αA,M
W1
W2
W1
W2
ŷ1
ŷM
Embeddings of words in abstract
Bilateral LSTM
Attention Coefficients
Embeddings of MeSH terms
Linear combination of BiLSTM hidden states
Two Linear Layers (Shared across all MeSH terms)
Predictions for masked MeSH terms

Figure 1: PubMeSH architecture schematic. Each abstract is run through 4 models of the same architecture to predict labels within each frequency category. The model includes an abstract word embedding layer, one bidirectional LSTM layer, an attention layer, a MeSH embedding layer, and 2 fully connected layers. Both the abstract word and MeSH embedding layers are initialized with pre-trained PubMed embeddings.

4.3 Experimental details

The 14 million abstracts were randomly split into training, development, and test sets using a 90/10/10% split. Each model was trained on the subset of the training abstracts that contained at least one MeSH term within their class frequency range and were only trained to make predictions on terms within this range (e.g. the "Super Rare" model was trained on many abstracts that had the very common "Human" MeSH term as one of their labels but was not trained to predict this label).

Training set abstracts were randomly ordered and all four models were trained for part of one epoch on one NVIDIA Tesla K80 GPU. This required approximately 48 hours per model. The models responsible for classifying rarer MeSH terms required more training time because there were drastically more labels to classify (e.g. there were 109 Common MeSH terms but over 10,000 Super Rare terms). Additional training was not possible due to limitations in computational resources and the large size of the training dataset (over 12M abstracts). A batch size of 64, learning rate of

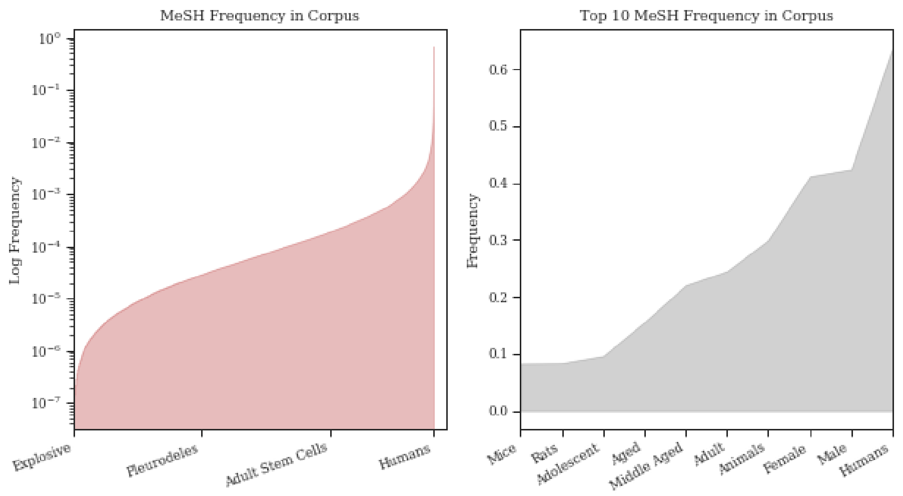

Figure 2: Proportion of articles containing each MeSH term. Left plot is the log frequency distribution for all MeSH terms. Right plot is the zoomed in frequency distribution of the top 10 most common MeSH terms.

Table 1: Model responsible for each class frequency range

MeSH Frequency	**Model Name**
0.01 - 1	Common
0.001 - 0.01	Moderate
0.0001 - 0.001	Rare
0 - 0.0001	Super Rare

Table 2: Model validation results

Model Name	**MiF**
Common	0.670
Moderate	0.384
Rare	0.229
Super Rare	0.171
Ensemble	0.484

0.0005, and Adam optimization were used for each model. Pilot studies were used to compare the cross entropy (CE) loss to the soft dice loss. While the soft dice loss allowed us to directly optimize the evaluation metric (micro F1), it was more prone to overfit to the most common labels within a frequency range, even with our four-model approach. We therefore used a class-weighted CE loss.

4.4 Results

The final results can be found in Table 2, with out final model having a micro F1 of 0.484. This is relative to a state-of-the-art of 0.68. The training loss progression for each of the four models is depicted in figure 4. We can observe that the loss plateaus for each of the models with the possible exception of Moderate. This could be attributed to the fact this model covered the fewest number of batches and could continue to learn if allowed to run longer. All models appear to plateau around 400 batches, likely due to a local minimum where probabilities for all uncommon MeSH terms are uniformly low and the model makes few predictions.

4.5 Analysis

4.5.1 Performance for common and rare MeSH terms

The "Common" model performs at near state-of-the-art levels with an F1 score of 0.670. This is the most important contributor to the F1 score as the labels predicted by this model can be found in nearly all abstracts on Pubmed. Performance suffers significantly in the rarer models. This is likely

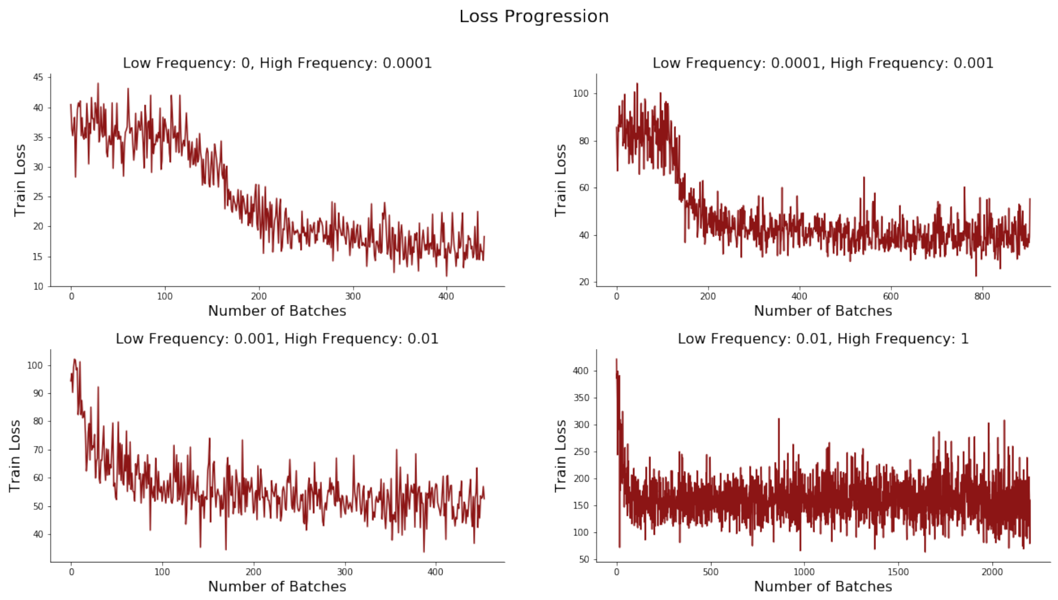

Figure 3: Loss Progression for each of the 4 models. Top left: Super Rare, Top right: Rare, Bottom Left: Moderate, Bottom right: Common

due to the fact that we were unable to train the model for a whole epoch - therefore the model likely did not see most rare MeSH terms even once.

4.5.2 Error analysis and the challenges of human bias

Our model disagreed with human experts on at least one label for many abstracts. In some of these cases, the model seems to have failed to identify important, relevant MeSH topics. However, there is also evidence that the model has learned to identify relevant MeSH terms that are not noted by the expert. This is to be expected considering how challenging the task of selecting multiple labels from a list of over 29,000 labels would be for any human. Disagreements between the model and the experts may often arise due to the hierarchical structure of the labels; It is likely that different human experts disagree with one another regarding how general or specific labels should be. Unfortunately, only one set of labels per abstract is available so we cannot assess our model's performance relative to inter-expert or intra-expert agreement. We provide examples below of situations in which the model disagreed with humans and discussion of why these disagreements may have occurred:

Examples in which the model produces reasonable labels that are missing from the human labels

1. Model: [Oxygen, Exercise] Expert: [Oxygen Consumption, Physical Endurance, Heart Rate]
   Analysis: Oxygen and oxygen consumption are extremely similar terms, as are exercise and physical endurance. This article discussed physical endurance in the context of exercise. It is unclear why the human would not also include exercise as a topic of the paper too. The paper inherently discussed oxygen when discussing oxygen consumption. While this particular human chose slightly more specific labels than the model, the next example illustrates the opposite phenomenon.

2. Model: [Curriculum, Education, Internship and Residency, Students] Expert: [Clinical Competence, Education]
   Analysis: This paper focused on the assessment of clinical education for medical students, interns, and residents. The model correctly identified the "Internship and Residency" and "Student" topics that the human expert failed to identify. By specifying curriculum, the model gives a more specific sub-category of "Education" in addition to also predicting the "Education" MeSH term.

3. Model: [Fracture Fixation, Osteoporosis] Expert: [Lumbar Vertebrae, Pain Measurement]
   Analysis: Both fracture fixation and osteoporosis are mentioned in the abstract. These are more detailed topics that relate to the lumbar vertebrae and are sources of the pain that the abstract authors seek to measure. Disagreements among human experts regarding specificity of topics are likely in case like this.

Examples in which the model misses important labels

1. Model: [Brazil, Food, Fruit] Expert: [Brazil, Fruit, Meat, Vegetables, Energy Intake, Eating, Feeding Behavior, Obesity, Body Mass Index]
   Analysis: The model correctly identifies the significance of Brazil and Fruit. It also identifies the general topic of food, which is arguably missing from the human labels. However, it fails to identify the clinical aspect of the work; It does not predict obesity, body mass index, or energy intake. It is also possible that the food word embeddings used in the abstract are relatively similar to the clinical MeSH embeddings and numerous in the abstract. This could then obscure the clinical words when attention is applied.
2. Model: [] Expert: [Amiodarone]
   Analysis: A common mistake committed by the model was failing to predict any labels for a given abstract. This was particularly true of the sub-model trained to classify super rare MeSH terms. It is likely that choosing not to predict any MeSH terms in many situations represents a significant local minimum in the loss. When the prior probability for each label is very low, the model faces a difficult task in attempting to maintain good recall without making drastic sacrifices to precision.

4.6 Conclusion

In this study we proposed novel methods to automatically index MeSH terms for PubMed abstracts: 1) using pre-trained MeSH embeddings, 2) separately training models for labels from different frequency categories. While the results did not surpass the current state-of-the-art models, we believe our approach to be theoretically sound and promising. With additional training time, we expect to match and potentially outperform current methods with the goal of fully automating MeSH indexing.

4.6.1 Future Directions

The size of the class space and corpus along with the rarity of many classes required significant model training time. With more time and computational resources, we would have performed these additional analyses:

- **Ablation studies for the linear layer:** The final component of our current model architecture features two linear layers. The first linear layer takes in the attention output and the second linear layer outputs a probability for one class. This differs from previous works [3, 7] with similar architectures in that it features an additional linear layer. This change was made in light of the fact that this single module is responsible for converting attention outputs to predictions for all weights (i.e. the weights are shared for all labels)
- **Training vs. Freezing Layers** For this model, we chose to make the MeSH and word embedding layers trainable. While we believe this method is likely to yield the best results for the task, keeping resource limitations in mind, it may be possible to allow training on many more batches in a shorter amount of time if these weights were kept frozen. This approach may be especially useful for the Rare and Super Rare models as exposure to a rare MeSH term multiple times may be more beneficial to performance than optimizing word embeddings in the short run.
- **Transfer Learning** We ran all models on separate GPUs in order to minimize clock time. It is however conceivable that using weights trained for one model may boost performance for another model since the features to be learned are likely to be similar across label frequencies. With additional time, we would like to explore this strategy by feeding the weights of the "Common" model to the "Moderate" and so on.
- **Hyperparameter Tuning** The size of the model and the dataset were prohibitive to any kind of hyperparameter tuning with the available resources. Specially tuned hyperparameters would almost certainly show gains in performance. This includes parameters such as the size and number of LSTM layers, size and number of fully connected layers, learning rate, and individual label thresholds.

4.7 Additional information and references

Our mentor is Annie Hu. We have no external collaborators and are not sharing projects between classes.

References

[1] MEDLINE®: Description of the Database.

[2] Alan R Aronson, James G Mork, Clifford W Gay, Susanne M Humphrey, Willie J Rogers, et al. The nlm indexing initiative's medical text indexer. *Medinfo*, 89, 2004.

[3] Qiao Jin, Bhuwan Dhingra, William Cohen, and Xinghua Lu. AttentionMeSH: Simple, Effective and Interpretable Automatic MeSH Indexer. In *Proceedings of the 6th BioASQ Workshop A challenge on large-scale biomedical semantic indexing and question answering*, pages 47–56, Brussels, Belgium, November 2018. Association for Computational Linguistics.

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

[5] Yuqing Mao and Zhiyong Lu. Mesh now: automatic mesh indexing at pubmed scale via learning to rank. *Journal of Biomedical Semantics*, 8(1):15, Apr 2017.

[6] Qiang Wu, Christopher J. C. Burges, Krysta M. Svore, and Jianfeng Gao. Adapting boosting for information retrieval measures. *Information Retrieval*, 13(3):254–270, Jun 2010.

[7] Ronghui You, Suyang Dai, Zihan Zhang, Hiroshi Mamitsuka, and Shanfeng Zhu. AttentionXML: Extreme Multi-Label Text Classification with Multi-Label Attention Based Recurrent Neural Networks. November 2018.