CS 6120: Comparing Models for Multilabel Classification of PubMed Article Abstracts

Nithish Bhat and **Ashish Thomas** and **Jamie Tjia**
Northeastern University
Boston, MA, United States
{bhat.nithi, thomas.ash, tjia.j}@northeastern.edu

Abstract

This is a template for your project report. This template is used for submissions to the ACL conference. You can see the original files here (https://github.com/acl-org/acl-style-files/tree/master/latex) which have additional instructions on how to use this template.

1 Introduction

The growth of biomedical literature has created a need for automated methods to organize and classify research articles effectively. Manually labeling articles with relevant Medical Subject Headings (MeSH) is labor-intensive and time-consuming, making automated multi-label classification a helpful tool for researchers and clinicians.

This project aims to use a transformer-based language model, specifically BERT, to perform multi-label classification on biomedical research articles. Instead of traditional single-label classification, we use multi-label classification, as articles are typically associated with multiple MeSH tags.

Our goals are: (1) to train and evaluate a BERT model for accurately predicting MeSH root labels and (2) to compare its performance against more conventional models, including logistic regression, CNNs, and RNNs. By doing so, we hope to demonstrate the advantages of transformer-based models in capturing contextual information and label correlations in biomedical texts, ultimately providing a tool for organizing and analyzing the PubMed corpus.

2 Background/Related Work

Describe any background or related work

3 Data

The baseline and BERT models were all trained and tested on the dataset PubMed MultiLabel Text Classification Dataset MeSH. This dataset contains information on 50,000 articles from PubMed, including each article's title, abstract, and Medical Subject Heading (MeSH) labels.

MeSH labels are arranged into a nested hierarchy with 16 root labels (see Table 1). Two labels—Humanities [K] and Publication Characteristics [V]—have been excluded due to their infrequency in the available data. The 14 included labels each appear in at least ten percent of the sample articles (see Figure 1).

Each article has up to 13 root labels, with a median root label count of six (see Figure 2). The article abstracts are an average of 222 tokens long (see Figure 3).

4 Methods

We use BioBERT—a specialized language model based on the dmis-lab/biobert-base-cased-v1.2—as a starting point and fine-tuned the model. While regular BERT learns from general texts such as Wikipedia, BioBERT trains on vast biomedical data like PubMed summaries and PMC papers, so it understands medical meaning better. Instead of "and," it relies on connections like "while" or "whereas." It handles split-up words (limited to 128 pieces per summary) via 12 layered transformers. Rather than stacking ideas with "plus," it builds step by step. From the last processing stage, the [CLS] marker's combined signal feeds into a straight-line classifier that outputs scores across 14 units. Training adjusts weights using BCEWithLogitsLoss - not only efficient but also suited for cases when one document fits several MeSH labels at once. Each prediction passes through its own sigmoid function separately; therefore, overlapping tags are possible without conflict.

The performance of the fine-tuned BERT model was measured against three baseline models which were trained as follows.

The logistic regression model serves as a basic

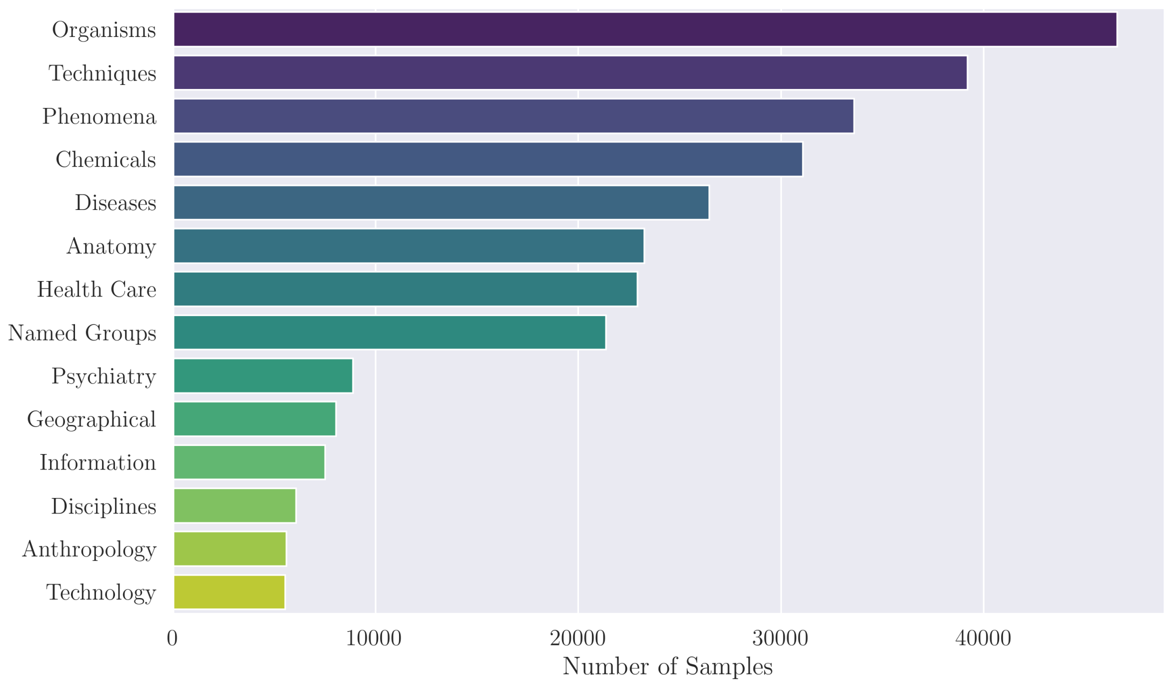


Figure 1: Frequency of MeSH Root Labels

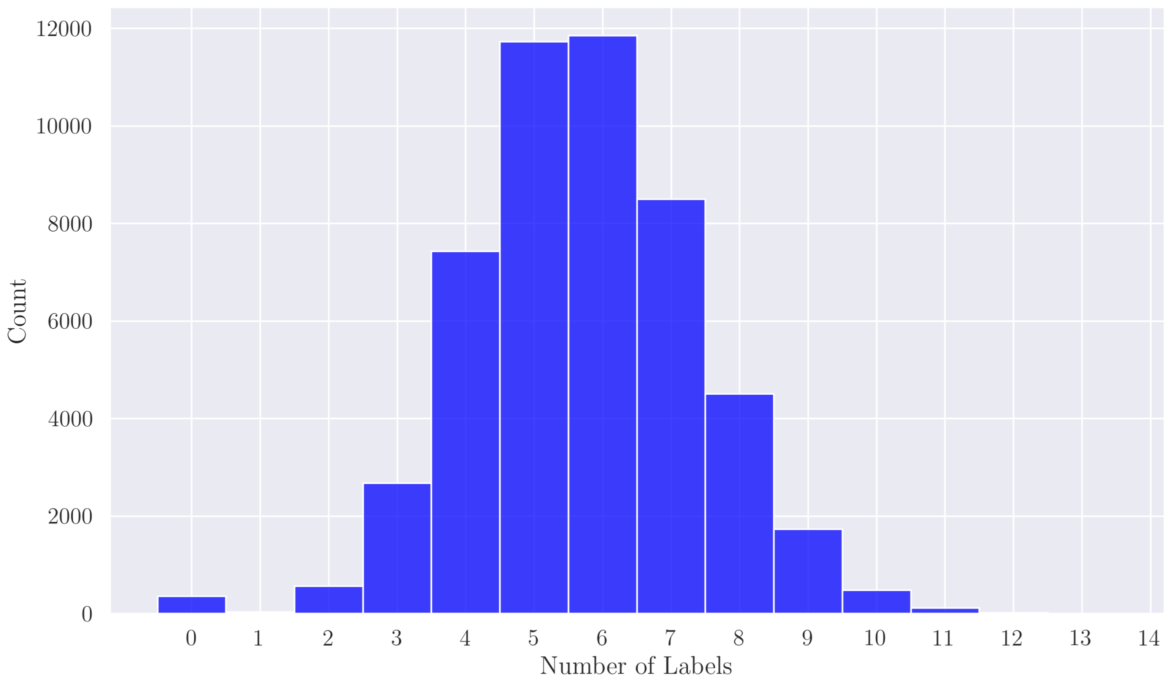


Figure 2: Number of Labels per Abstract

neural starting point while performing effectively like a "Neural Bag-of-Words" system. Although it uses an embedding layer set to 100 dimensions by default, this component converts word IDs into compact vector forms. Rather than tracking sequence position, it computes the average of these embedded values throughout the full text entry. As a result, one consolidated mean vector emerges, capturing the overall meaning of the abstract. This average form feeds straight into a linear layer, then uses BCEWithLogitsLoss for training. The approach estimates each label's likelihood separately,

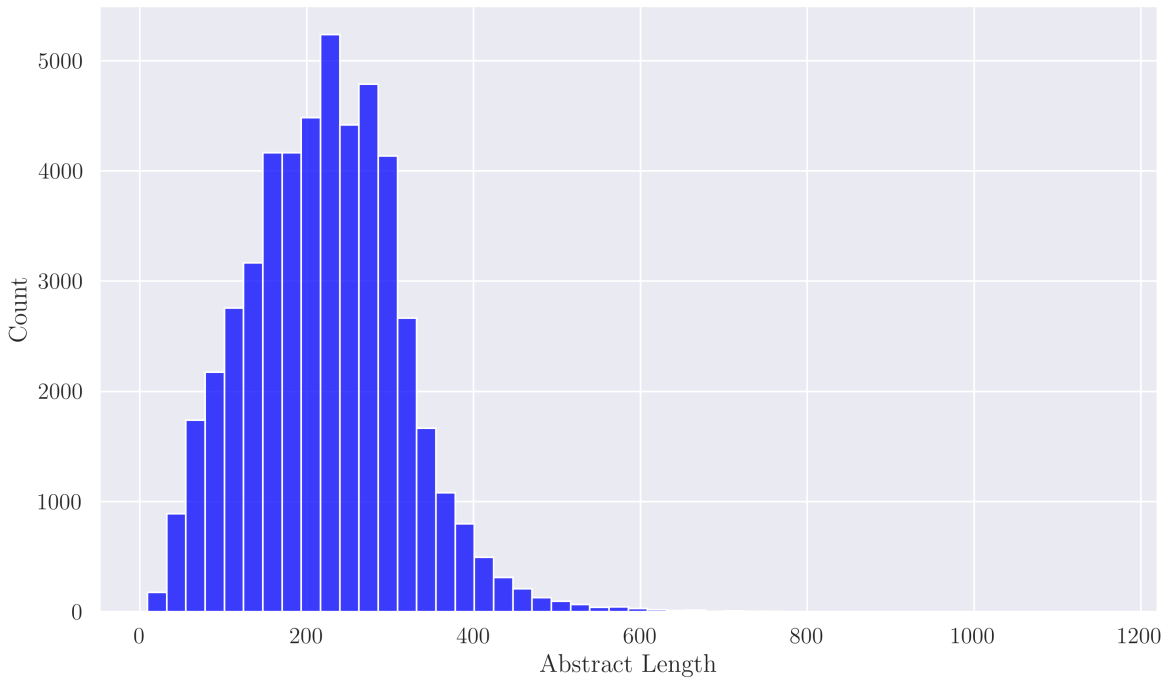


Figure 3: Distribution of Abstract Lengths

yet maintains fast response times.

The CNN approach detects local word sequences and meaningful phrases in text, no matter where they appear. Instead of focusing on exact positions, it processes embedded inputs using a 1D conv layer with a broad window size 50 and moves forward by steps of 20 to capture distinct signals from the abstract. Following this step, global max pooling selects the strongest activation per filter over the full input span. Then, these selected outputs use BCEWithLogitsLoss for training so that multiple labels can be predicted at once.

The RNN model handles word order by applying a Bi-LSTM setup. First, an embedding step converts split text into 256-element vectors. Then, these representations enter a Bi-LSTM block containing 128 memory cells. Instead of one direction only, processing occurs left-to-right and right-to-left to grasp full-sentence meaning. Afterward, outputs from both flows merge before entering a dense layer. That last component computes unnormalized scores directly. During inference, logits become probabilities using a sigmoid activation. With a 0.3 cutoff, multi-label outputs turn into binary results.

5 Experiments

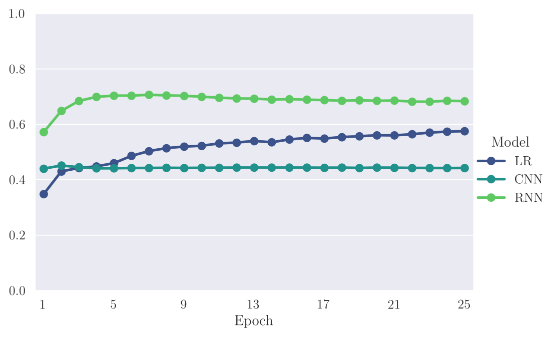
All of the baseline models were trained over 25 epochs, using an initial learning rate of 0.001 with Adam optimization.

At the end of each epoch, the micro F1, macro F1, weighted-average F1, exact match ratio, and Hamming loss were calculated. Micro F1 calculates the F1 score using precision and recall over all of the labels at once, while the macro F1 calculates the F1 for each label and returns the average across the labels. Similarly to macro F1, weighted-average F1 calculates the F1 for each label and returns the weighted average. The exact match ratio measures how many predictions exactly matched every label, and Hamming loss measures how many labels were predicted incorrectly across all labels. Higher F1 scores and exact match ratios indicate better performance, and lower Hamming loss indicates better performance. Together, these five metrics were used to measure the performance of each of the models.

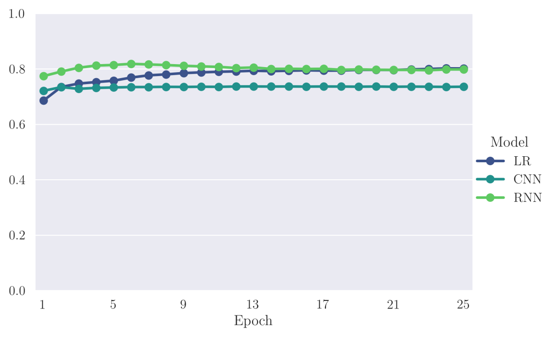
5.1 Results

The RNN model performed best of the three baseline models. It had the highest macro and weighted-average F1 scores of the three baseline models over all 25 epochs, and performed best with regard to micro F1, exact match ratio, and Hamming loss for most of training.

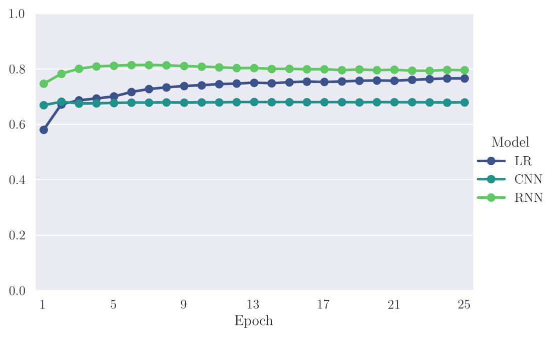
Although linear regression had worse performance than RNN across all five metrics for the



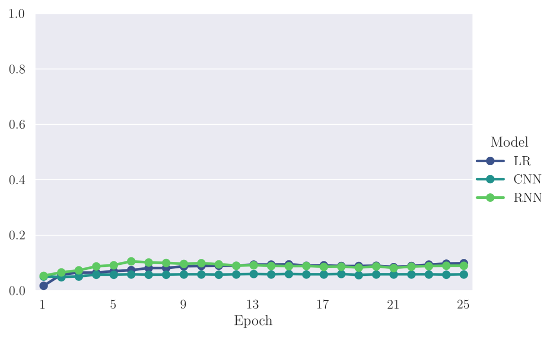
(a) Macro F1



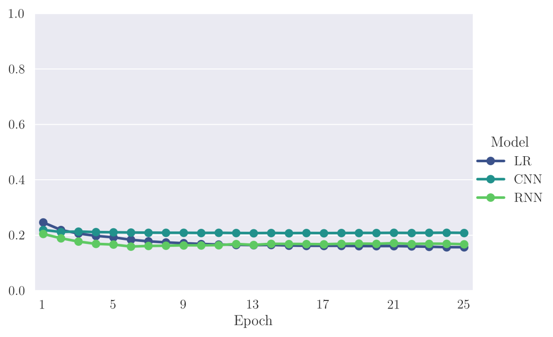
(b) Micro F1



(c) Weighted F1



(d) Exact Match Ratio



(e) Hamming Loss

Figure 5: Performance Metrics by Epoch

first several epochs of training, the model using linear regression had the marginally better performance with regard to micro F1, exact match ratio, and Hamming loss past 15 epochs of training.

The CNN model had by far the worst performance, with lower F1 scores, lower exact match ratios, and higher Hamming loss for the majority of training. CNN only performed better than logistic regression for the first two epochs, and never performed better than RNN for any evaluation metric.

6 Conclusions

Present your discussion/conclusions.

Label	Name
A	Anatomy
B	Organisms
C	Diseases
D	Chemicals and Drugs
E	Analytical, Diagnostic and Therapeutic Techniques, and Equipment
F	Psychiatry and Psychology
G	Phenomena and Processes
H	Disciplines and Occupations
I	Anthropology, Education, Sociology, and Social Phenomena
J	Technology, Industry, and Agriculture
K	Humanities
L	Information Science
M	Named Groups
N	Health Care
V	Publication Characteristics
Z	Geographicals

Table 1: Full Label Names