Multi-label Coding Questions Classification

Tianle Wang
Department of Computer Science
University of Toronto
tianle.wang@mail.utoronto.ca

Yifan Zhao
Department of Computer Science
University of Toronto
ethany.zhao@mail.utoronto.ca

Yuezhexuan Zhu
Department of Computer Science
University of Toronto
yuezhexuan.zhu@mail.utoronto.ca

Abstract

Inspired by the neural language models introduced in the lecture, we have decided to design a neural network that can predict the types of algorithms and data structures required to solve a coding question. To accomplish this, we have chosen to utilize the problems from Leetcode , a well-known online judge platform that provides coding questions for practicing programming skills. However, since the data size is relatively small, we plan to use techniques such as data augmentation and transfer learning to improve the accuracy of our model. Our goal is to create a model that can predict the most likely algorithm for a given coding question based on its text description. Additionally, the model will provide several top-ranking labels ordered by possibility. We will analyze multiple models, including BERT[1] and LSTM[2], to determine which model works best for our case.

1 Introduction

All members of our team share a common interest in solving coding questions, which we perceive as a means of enhancing our logical reasoning and programming proficiency. We have observed that the identification of an appropriate algorithm constitutes the initial and crucial step towards devising a solution. Any biased identification can lead to a series of unfortunate outcomes until the correct path is identified.

To address this issue, we have undertaken the development of a Neural Network that can effectively generate potential algorithms and data structure labels for a given coding question based on its textual description. Our proposed model is a specialized text classifier tailored to the domain of coding questions. The rationale behind this choice is twofold. Firstly, the existence of a plethora of prior research and frameworks in the field of text classification allows us to analyze various architectures and leverage pre-trained models to minimize the resources required for training while achieving superior performance. Secondly, our model seeks to enhance the accuracy and efficiency of algorithm identification, thereby facilitating the solution development process.

2 Data Preprocessing

Instead of fetching data from Leetcode, we decide to use the processed dataset from gzipChrist[3] on Kaggle. The dataset comprises 1825 problem descriptions and their corresponding required algorithms or data structures. To facilitate analysis, we developed a Python script to vectorize the labels associated with each problem. Following processing, the dataset contains 1825 rows, with the

first column representing the problem description and the remaining columns indicating the presence or absence of the corresponding labels, denoted by binary values of 0 or 1.

3 Model Architecture

3.1 BERT Pre-trained Model

3.1.1 Motivation

3.1.2 Design

3.1.3 Training

3.1.4 Result

3.2 LSTM Model

3.2.1 Motivation

3.2.2 Design

3.2.3 Training

3.2.4 Result

4 Comparison of Models

5 Limitations

In examining our dataset, several limitations must be acknowledged to ensure a comprehensive understanding of its potential impact on the analysis. Firstly, the dataset is relatively small, as generating LeetCode problems is a complex task and the total number of problems available is less than 3,000. Secondly, our model is limited to handling text input, rendering it incapable of addressing problems that necessitate image understanding. Thirdly, the labeling of LeetCode problems is inherently biased, as the majority of coding problems are associated with major data structures, such as arrays, or widely-used algorithms like dynamic programming and greedy techniques. Lastly, the presence of mathematical formulas and special characters in problem statements may hinder the model's tokenizer from optimally processing the input for training.

6 Conclusion

References

[1] Devlin, J., Chang, M., Lee, K., & Toutanova, K. (2018). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. ArXiv. /abs/1810.04805.

[2] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," in Neural Computation, vol. 9, no. 8, pp. 1735-1780, 15 Nov. 1997, doi: 10.1162/neco.1997.9.8.1735.

[3] gzipChrist. (2023, February). Leetcode Problem Dataset, Version 8.82. Retrieved April 10, 2023 from https://www.kaggle.com/datasets/gzipchrist/leetcode-problem-dataset.

A Appendix

Optionally include extra information (complete proofs, additional experiments and plots) in the appendix. This section will often be part of the supplemental material.