Title: Relation Extraction

Hesam Farjad

Abstract

This report describes the implementation of a code block that reads map data from a relation file and prepares it for prediction. The code utilizes a tokenizer and converts the data into tokens for further processing. It also defines a dataset class and implements two loss functions for optimization. The report provides an overview of the code and explains the key steps involved in the process.

1 Introduction

In this section, we introduce the purpose of the code block and provide an overview of its functionality. The code aims to read map data from a relation file and predict a series of classes based on the data. It utilizes a tokenizer and applies various transformations to prepare the data for training and testing. The report will discuss the different parts of the code and their significance in achieving the desired outcome.

2 Reading and Processing Map Data

In this section, we focus on the first part of the code block, which involves reading the relation file and processing the map data. The code defines a function called "read_map_data" to read the relation file and output a JSON file. The JSON file is then read and converted into a dictionary called "order_dic." This dictionary contains classes that need to be predicted, and the number of classes is determined by the length of the relation file. The section explains the steps involved in reading and processing the map data.

3 Preparing the Data

The next section discusses the function "prepare" and its role in preparing the data for further processing. The code block defines a BERT tokenizer and applies it to the JSON file obtained from the previous step. The subject and object information are extracted from the JSON file and stored in a main data frame. The section explains how the subject and object data is processed and converted into the required format for training and testing.

4 Title: Train and Test Data Preparation

In this section, we focus on preparing the train and test data using the functions "prepare_train_df" and "prepare_test_df." These functions apply the previously defined "prepare" function to the train and test data, respectively. The section explains the steps involved in preparing the data, including counting the number of tokens and determining the maximum length. It also highlights the importance of the tokenizer in this process.

5 Title: Dataset and DataLoader

Here, we discuss the implementation of the dataset class and the data loader. The dataset class is defined to handle the input data and labels for training and testing. The data loader is responsible for loading the data in batches for efficient processing. The section explains the structure and functionality of the dataset class and how it interfaces with the data loader.

6 Title: Loss Functions / Optimization

In this section, we delve into the code block that defines two loss functions and performs simultaneous optimization. The section explains the purpose of the two labels and how they are used to calculate the loss functions. It also discusses the optimization process and the importance of optimizing both functions together for accurate predictions.

7 Title: Relation Model and Outputs

The next section focuses on the implementation of the relation model class and its outputs. The code initializes the model, which includes a BERT model, fully connected layers, and a

dropout layer. The section explains how the input tokens are processed by the model and the outputs obtained from the last hidden state and the pooler output. It also clarifies the differences between these outputs and their significance in the model's functioning.

8 Dataset and Model Training

Here, we discuss the dataset creation and model training process. The code block defines the dataset using the previously discussed dataset class and creates a data loader for efficient training. The model used is the relation model, and the criterion for training is cross entropy. The section explains the inputs and outputs of the data loader and the significance of the model's predictions (pred and pred1) in the training process.

9 Conclusion:

This report provided an overview of a code block that reads map data from a relation file and prepares it for prediction. The code implemented various functions to tokenize the data, process subject and object information, and create a dataset for training and testing. It also defined a relation model with multiple outputs and utilized two loss functions for optimization. The report discussed the key steps involved in each part of the code block.