Report

1. Introduction

In this report, I'll introduce the models and results about how am I aiming to develop a NLP model for Commonsense Reasoning Task.

2. Methodologies and models

2.1 Recurrent Neural Networks (RNN)

Methodology:

Recurrent Neural Networks (RNN) are a class of neural networks that are particularly adept at handling sequential data. They leverage their internal state (memory) to process sequences of inputs, making them suitable for tasks involving temporal dynamics or ordered sequences, such as language modeling and commonsense reasoning.

Algorithm:

Initialization: Initialize the weight matrices and bias terms.

Forward Propagation:

For each time step t in the sequence:

1. Compute the hidden state of time **t** using the previous hidden state **t-1** and the current input **x** of time **t**.
2. Compute the output **y** of time **t**.

Loss Calculation: Compare the network output with the true labels to compute the loss.

Backpropagation Through Time (BPTT): Compute the gradients of the loss with respect to the network parameters by unrolling the RNN through time and applying backpropagation.

Parameter Update: Update the network parameters using an optimization algorithm such as gradient descent or Adam.

Iteration: Repeat the process for multiple epochs until the model converges.

2.2 DeBERTa (Decoding-enhanced BERT with Disentangled Attention)

Methodology:

DeBERTa is a transformer-based model designed to enhance the performance of BERT by introducing two main innovations: disentangled attention and enhanced mask decoder. It builds upon the transformer architecture by improving how attention scores are calculated and how masked tokens are predicted.

Algorithm:

Initialization: Initialize the model parameters, including the embedding layers and transformer blocks.

Embedding: Convert input tokens into embeddings, incorporating both content and positional information.

Forward Propagation through Transformer Blocks:

For each transformer block:

1. Apply multi-head disentangled self-attention.
2. Pass the results through a feed-forward neural network.

Masked Language Modeling : Predict masked tokens using the enhanced mask decoder.

Loss Calculation: Compute the loss based on the task (e.g., classification, regression, masked language modeling).

Backpropagation: Calculate gradients using backpropagation.

Parameter Update: Update the model parameters using an optimization algorithm.

Iteration: Repeat for multiple epochs until the model converges.

3. Data Preprocess

3.1 Overview

The project includes two jsonl files, one is train.jsonl for training and validation with size of [14912 x 5] and each column represent a Commonsense question. The five columns are **[Id,Question,Alternative1,Alternative2,Answer],** the target of this project is to build a model to the seminarist **sentence** with **Question** sentence from **Alternative1** and **Alternative2**, the **Answer** is the right sentence. Another file is eval.jsonl [4261 x 4] without **Answer** column compared to the train.jsonl, this file is for test and evaluation the model.

3.2 Training dataset

1.Split Dataset:

For train.jsonl file, it is split to training part and validation part with the ratio of 0.8(train) and 0.2(validation).

2.Extract Questions and Choices:

Extract each question and its corresponding two choices from the dataset, creating a list of question-choice pairs. This step aims to pair each question with its possible answer choices for further processing.

3. Adjust Labels:

Convert the labels from a 1/2 format to a 0/1 format. This is because most machine learning models expect zero-based indexing for labels. This conversion makes the labels more suitable for model processing.

4. Create Contexts:

Pair each question with its two choices, creating a list of all question-choice pairs. This step prepares each question and its possible answers for tokenization.

5. Tokenize Examples:

Use the pre-trained DeBERTa model's tokenizer to tokenize these question-choice pairs, ensuring all input sequences have the same length. Tokenization involves truncating sequences that are too long and padding shorter sequences to a specified maximum length.

6. Format Tokenized Data:

Group the tokenized input IDs and attention masks into pairs, corresponding to the choices for each question. This step ensures the tokenized data is provided to the model in the correct format.

By following these steps, I transform the raw data into a format suitable for input into the DeBERTa model. This ensures that the model can effectively process and learn from the data, performing well on the commonsense reasoning task.

3.3 Evaluation Dataset

Comparing to the training dataset, since eval.jsonl do not have a target column **Answer**, the step of preprocessing evaluation dataset is the same as training dataset except the Adjust Labels.

4. Experiment

During the experiment phase, I tried a lot of models including but not limited to building my own RNN and employing pre-trained model Deberta.

For **RNN** model, I used following parameters for training with 10 epochs:

```
embedding_dim = 200
hidden_dim = 256
output_dim = 1
vocab_size = len(vocab)
```

But the result for RNN on test get only **0.52** scores.

Then I tried to utilize pre-trained model like Deberta.

Following the training parameters for Deberta:

```
training_args = TrainingArguments(
    output_dir="./results",
    evaluation_strategy="epoch",
    save_strategy="epoch",
    save_total_limit=1,
    load_best_model_at_end=True,
    learning_rate=2e-5,
    per_device_train_batch_size=2,
    num_train_epochs=3,
    weight_decay=0.01,
)
```

With these parameters, I trained Deberta for 3 epochs and saved the best one which has least loss in validation dataset. Then I got **0.81** scores on test dataset.