Sentiment Analysis with BiLSTM

Matteo Manias
manias.1822363@studenti.uniroma1.it

Abstract

Report for the HM1b assignment, create a sentiment analysis model using a BiLSTM architecture on the HASPEEDE dataset. The model achieves an accuracy of 70% on the test set, demonstrating the effectiveness of the approach for sentiment analysis.

1 Introduction

Sentiment analysis is an important natural language processing task that involves classifying text as expressing positive, negative, or neutral sentiment.
For this specific case i developed a BiLSTM model to predict the sentiment label of the HASPEEDE dataset, which consists of Italian text data labeled as positive or negative sentiment.
The main goal of this work is to demonstrate the superior performance of an RNN model over several baseline classification methods and a more advanced bag of words (BoW) model.

2 HASPEEDE Dataset

```
{
  "text": "È terrorismo anche questo, per mettere in uno
           stato di soggezione le persone e renderle
           innocue, mentre qualcuno... URL",
  "choices": ["neutrale", "odio"],
  "label": 0
}
```

The generic entry of the HASPEEDE dataset is shown above.
The text field contains the sentence to be classified, the choices field contains the possible sentiment labels and the label field contains the index of the ground truth sentiment label.
The dataset is expressed in the JSONL format, with each line representing a single data entry and is split into training, validation, and test sets.

3 Model architecture

To capture the underlying sentiment of the text, i used an RNN model that can successfully capture the context of the text by leveraging the sequential nature of text data and the long-term dependencies between words, the contex of the text is preserved in the model by storing it in a memory cell.
The specific RNN model used is the Bidirectional Long Short-Term Memory (BiLSTM) model, which processes the text sequence in both forward and backward directions.
The BiLSTM model is initialized as follows:

```
BiLSTMModel(
  (embedding): Embedding(23699, 128, padding_idx=0)
  (bilstm): LSTM(128, 128, num_layers=4, batch_first=True, dropout=0.3,
   bidirectional=True)
  (layer_norm): LayerNorm((256,), eps=1e-05, elementwise_affine=True)
  (hidden_layer): Linear(in_features=256, out_features=128, bias=True)
  (projection): Linear(in_features=128, out_features=2, bias=True)
  (relu): ReLU()
)
```

3.1 BiLSTM Desing

The BiLSTM model consists of the following components:

- An embedding layer that converts the input tokens into dense vectors, the embedding dimension is determined by the size of the vocabulary and the desired vector size.
- A BiLSTM layer that processes the input sequence in both forward and backward directions, the LSTM layer has 128 hidden units and 4 layers.
  More complex values migth lead to overfitting as the dataset is relatively small.
- A layer normalization layer that normalizes the output of the BiLSTM layer. This helps stabilize the training process and improve the model's generalization.
- A hidden layer that reduces the dimensionality of the output. This is the first layer in the final classification network.
- A projection layer that maps the output to the number of sentiment classes, to make the final prediction, using multiple layers helps the model to learn more complex representations of the LSTM output.

- A ReLU activation function that introduces non-linearity into the model.

3.2 Baselines implemented

To gauge the performance of the BiLSTM model, i used the following baselines:

- Majority class baseline: Predicts the majority class label for all instances.
- Random baseline: Predicts a random label for each instance.
- BoW baseline: Uses a bag of words model with a FNN final classifier.

4 Results

Model	Test Loss	Test Accuracy
BoW Baseline	0.6595	0.6502
Majority Baseline	0.6931	0.6465
Random Baseline	0.9093	0.4707
BiLSTM	0.6920	0.6938

Table 1: Results of baselines classification techniques and the BiLSTM model on the HASPEEDE test dataset.
the BiLSTM model outperforms all oother baselines

code available at: https://github.com/maniaa1822/HM1b