NLU project exercise 2 lab 10

Luca Zanolo (20029226)

University of Trento

luca.zanolo@studenti.unitn.it

1. Introduction

The report outlines the implementation of a BERT-based-uncased neural network for intent classification and slot filling tasks, using the ATIS dataset. Considering the approach described in the paper [1], I try to follow it and also to implement the Conditional Random Field (CRF) layer as described. The project began with preparing the dataset, ensuring the alignment of tokenized inputs with slot labels. Then, two experimental configurations were set up: one incorporating a CRF layer and another without it. The objective was to fine-tune the BERT model under these two configurations for intent prediction and slot filling.

2. Implementation details (max approx. 200-300 words)

The initial step was the preparation of the ATIS dataset, involving a process of aligning the tokenized inputs with their respective slot labels. Using `BertTokenizer`, each utterance in the dataset was tokenized. A problem of this process was managing multiple tokens representing a single word, especially for words labeled with 'B-' (begin) and 'I-' (inside) tags in slot labels. This was achieved in the `align_slots` method within the `IntentsAndSlots` class. For each word in an utterance, the corresponding slot label was expanded to match the number of sub-tokens generated by the tokenizer. The first sub-token was assigned a 'B-' label (if the original word began a slot) or retained its original label, while subsequent sub-tokens were labeled with 'I-' tags, indicating their continuity in the slot sequence. The dataset was then encapsulated into a `DataLoader` with batch sizes set at 128 for training and 64 for validation and testing phases.

The architecture of the model, named `jointBERT`, is based on the standard `BertModel`. It has two classifiers classes: `IntentClassifier` and `SlotClassifier`. The `IntentClassifier` is responsible for predicting intents from BERT's output, while the `SlotClassifier` focuses on slot filling tasks, utilizing the sequence output from BERT. Both classifiers include dropout layers for regularization and linear layers for classification. The model's architecture can be used with two different configurations: with or without the CRF layer, based on initialization parameters.

During the training phase, `jointBERT` was fine-tuned to optimize both intent and slot losses. The total loss is a summation of these individual losses. The intent loss is always calculated using a cross-entropy loss function. For slot loss, when the CRF layer is employed, it is calculated using this layer to enhance the model's sequence labeling capabilities. In the absence of the CRF layer, slot loss is also calculated using a cross-entropy loss function.

3. Results

The evaluation of the model was conducted through an `eval_loop` function, which calculated both intent and slot performance metrics. The model's performance was assessed based on intent accuracy, calculated using the scikit-learn classification report, and slot F1 score, retrieved using evaluate function from conll script.

Two sets of experiments were conducted: Experiment 1 without the CRF layer and Experiment 2 with the CRF layer. Each experiment was run five times to ensure consistency and reliability in the results. The table below summarizes the results of these experiments, showing the average intent accuracy and slot F1 score across all runs.

Experiment ID	Intent Accuracy	Slot F1 Score
JointBERT	0.96	0.9742
JointBERT with CRF	0.9604	0.9787

Table 1: *Best Performing Runs from Each Experiment*

The results indicate that the incorporation of the CRF layer in Experiment 2 slightly improved the performance on both intent accuracy and slot F1 score compared to Experiment 1.

In the next page are shown the metrics at run level detail and the charts of the training and dev losses.

4. References

[1] Q. Chen, Z. Zhuo, and W. Wang, "Bert for joint intent classification and slot filling," 2019.

Experiment ID	Run	Intent Accuracy	Accuracy Std.	Slot F1 Score	F1 Std.
Experiment 1	1	0.9561	0	0.9765	0
	2	0.9562	0	0.9720	0
	3	0.9584	0	0.9765	0
	4	0.9545	0	0.9754	0
	5	0.9600	0	0.9742	0
	AVG	0.9570	0.002	0.9749	0.0017
Experiment 2	1	0.9594	0	0.9787	0
	2	0.9575	0	0.9720	0
	3	0.9599	0	0.9720	0
	4	0.9604	0	0.9787	0
	5	0.9589	0	0.9709	0
	AVG	0.9592	0.001	0.9745	0.0035

Table 2: *Detailed Results of All Experiment Runs*

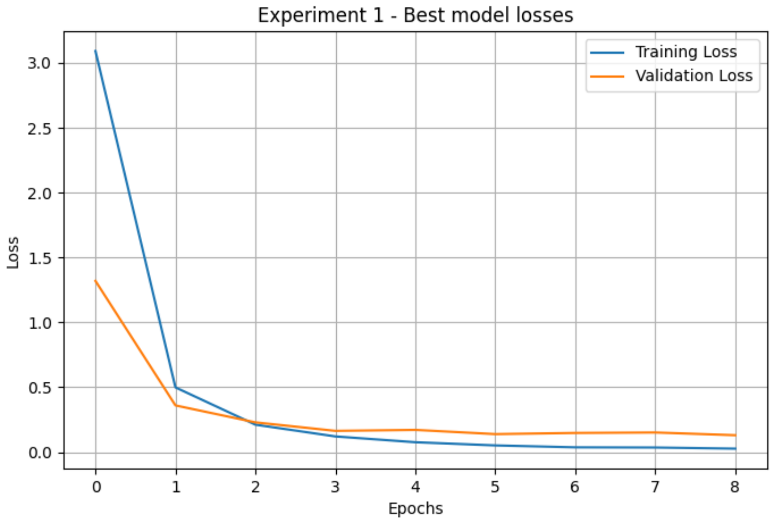

Figure 1: *Training and Dev Losses for Experiment 1*

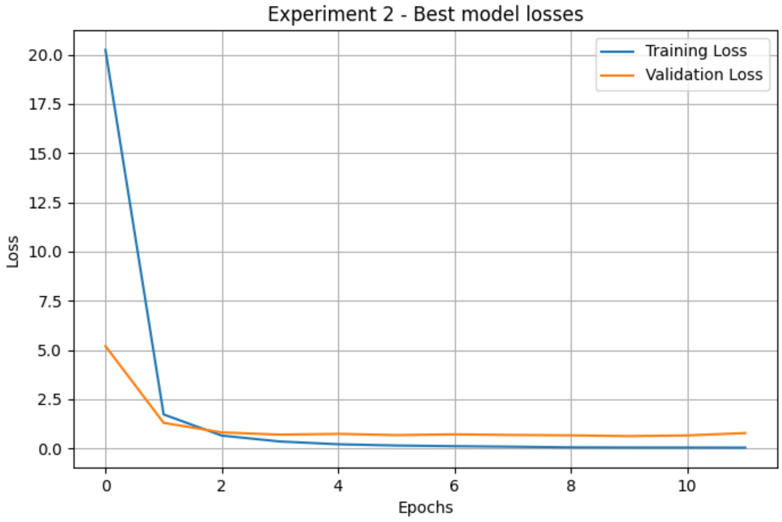

Figure 2: *Training and Dev Losses for Experiment 2*