NLU course projects: Lab. 5 (NLU)

Alessandro Lorenzi (247177)

University of Trento

alessandro.lorenzi-1@studenti.unitn.it

1. Introduction

This report contains the work done to solve the **NLU task**, in particular to implement and improve the performance of different architectures for the task of **Intent classification** and **Slot filling**. *Intent Classification* is a particular task of text classification, consisting in deducing (classify) the specific purpose of a whole sequence given. *Slot filling* (or *Concept tagging*) is a case of shallow parsing task that aims to identify meaningful slots in a text. The first part consists in modifying the baseline architecture based on LSTM, adding some regularization techniques, while the second part refer to fine-tune a pre-trained BERT model using a multitask learning setting on both the task.

2. Implementation details

The used dataset is the ATIS (Airline Travel Information Systems). Regarding the joint training the criterion is the cross-entropy loss for both the tasks and the final loss is given by $loss = loss_{intent} + loss_{slot}$. Both the parts' performance were evaluated under 5 runs of 100 epochs each, using AdamW as optimizer, with a learning rate of 0.0001.
The architectures and all the methods used are then briefly described.

2.1. Part 1

The baseline model with LSTM backbone was modified, investigating some design choices and regularization techniques applied incrementally. In particular:

- Bidirectionality: the model processes data in two directions: forward and backward. This allows it to capture dependencies in both directions, which is particularly useful when each element of a sequence depends not only on past data but also on future data. This implies doubling the size of both the slot and intents linear layers of the model.
- Dropout: technique for neural networks that drops a unit (along with connections) at training time with a specified probability *p*. The dropout is added to both the embeddings and the sequence. The probability used is 0.5.

2.2. Part 2

Following [1], to solve the joint intent classification and slot filling, a model based on BERT was implemented.
Regarding the first task, by using the hidden state of the initial special token (`[CLS]`), which captures contextual information for the entire input, the intent is determined through a softmax layer applied to this representation. For the slot filling, instead, it's needed to deal with the sub-tokenization issue: BERT uses *WordPiece* tokenization, which can break words into smaller subword units, especially for rare or complex words, resulting in a problem of label alignment and ambiguity. To address this, the implemented solution maps slot labels only to the first sub-token of each word while maintaining consistency across sentences. In particular, the method uses the tokenizer from `AutoTokenizer` [2] that allow to map tokens back to their original words through the `word_ids()` function [3]. This function provides a list in which each token is assigned an ID corresponding to its original word in the sentence. If a word is split into multiple tokens, all these tokens are assigned the same ID, helping us identify all parts of a word. To ensure that only the first sub-token of each word is mapped to a slot label, keeping the alignment straightforward the solution also check for spaces between words, so it's possible to confirm when a new word starts and align it correctly. By creating a mapping that links each word with just its first token, it ensures that slot labels correspond clearly to each word. This method allows maintaining consistency in slot labeling, even with sub-tokenized words, ultimately improving the model's ability to correctly assign slot labels across different inputs.

3. Results

The metrics evaluated are the accuracy for the intent classification and the F1 score for slot filling.
Table 1 and Table 2 show the results of all the experiments conducted in the first and second part. Figure 1 graphically shows the different performance.

- In **Part 1**, the three configurations of the ModelIAS architecture were evaluated. The results indicate that the standard model was already able to solve the task with good performance. Adding bidirectionality, it demonstrated improved results, with also the addition of dropout, it enhanced again its performance, resulting in a slot F1 score of 0.942 and an intent accuracy of 0.951, that are the best performance within this part of the study.
- In **Part 2**, the JointModel using a BERT-base-uncased architecture was assessed. This configuration outperformed all previous models, achieving a Slot F1 score of 0.954 and an intent accuracy of 0.973.

.

4. References

[1] Q. Chen, Z. Zhuo, and W. Wang, "Bert for joint intent classification and slot filling," 2019. [Online]. Available: https://arxiv.org/abs/1902.10909

[2] H. Face, "Transformers documentation: Auto classes." [Online]. Available: https://huggingface.co/docs/transformers/model_doc/auto

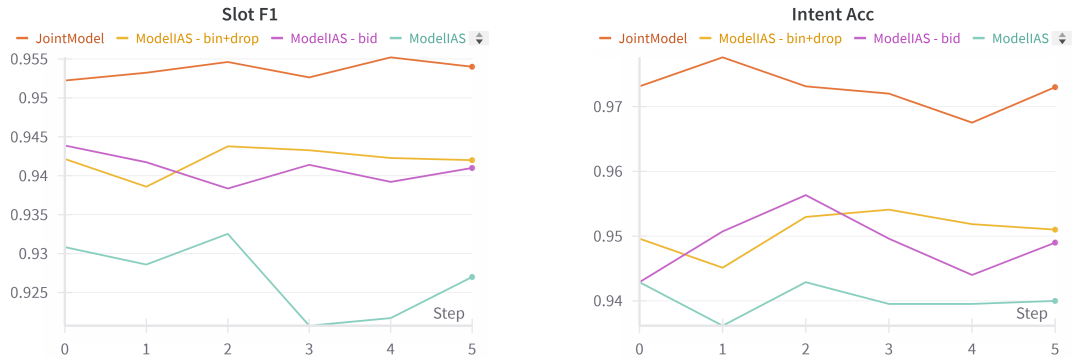
[3] ——, "Transformers documentation: Batchencoding.word_ids method." [Online]. Available: https://huggingface.co/docs/transformers/main_classes/tokenizer

Architecture	Slot F1	Intent Acc
ModelIAS	0.927 ± 0.005	0.940 ± 0.005
ModelIAS bidirectional	0.941 ± 0.002	0.949 ± 0.002
ModelIAS bidirectional + dropout	**0.942** ± 0.002	**0.951** ± 0.002

Table 1: *Part 1 Results. The terms in **bold** indicate the best performance obtained for the corresponding configuration.*

Architecture	Slot F1	Intent Acc
JointModel (BERT-base-uncased)	**0.954** ± 0.001	**0.973** ± 0.001

Table 2: *Part 2 Results. The terms in **bold** indicate the best performance obtained for the corresponding configuration, while the underlined term represents the best overall results (considering both parts).*



(a) *Slot F1*

(b) *Intent Acc*

Figure 1: *Report Results*