NLU course project - Assignment 2 - NLU

Daniele Della Pietra (258646)

University of Trento

daniele.dellapietra@studenti.unitn.it

1. Introduction

This project addresses joint natural language understanding (NLU) on **ATIS**, combining intent classification (utterance-level) and slot filling (token-level). In Part 2.A, I extend the LSTM baseline by making the encoder bidirectional and applying dropout, while keeping a simple multi-task head: a token classifier for slots and a sentence classifier for intents. In Part 2.B, I fine-tune a pre-trained BERT model [1] in a multi-task setting, using the [CLS] representation for intent and per-token representations for slots, with careful handling of sub-tokenization. Models are trained with early stopping on a stratified dev split; evaluation uses intent accuracy and CoNLL chunk-level F1 for slots. Results show steady gains from BiLSTM and the strongest performance from BERT.

2. Implementation details

2.1. Data and split

I use the standard ATIS train/test files and create a 10% dev set by stratified sampling on the intent label (rare, single-occurrence intents are kept in train). Vocabulary, intent, and slot label spaces are built from the union of train/dev/test.

2.2. Part 2.A (LSTM baselines)

The encoder is a single-layer LSTM with optional bidirectionality. Inputs are embedded with a learnable lookup table; I apply dropout after the embeddings and again to the timewise encoder outputs when enabled. Variable-length sequences are packed during the forward pass.

For **slot filling**, a linear classifier is applied to each time step of the encoder output (H features for unidirectional, $2H$ for bidirectional). For intent, I concatenate the forward and backward hidden states of the top BiLSTM layer (2H) before the intent classifier; the slot classifier operates over time-step features (H/2H for uni/bi).

Training minimizes the sum of the two cross-entropies (slot loss ignores padding with **PAD_TOKEN**). I use Adam, gradient clipping at 5, mini-batches of 128, and early stopping on dev slot F1 (patience 3–5). Representative hyperparameters are: hidden size 200, embedding size 300, learning rate $5 \cdot 10^{-4}$, dropout $p \in \{0, 0.1\}$, and bidirectionality toggled per configuration (UniLSTM, BiLSTM, BiLSTM+Dropout).

2.3. Part 2.B (BERT fine-tuning)

I replace the embedder+LSTM with a pre-trained **bert-base-uncased**. The multi-task head mirrors Part 2.A: the pooled [CLS] vector feeds the intent classifier; per-token hidden states feed the slot classifier. Sub-tokenization is handled with **BertTokenizerFast** and **DataCollatorForTokenClassification**: gold slot labels are aligned to tokens and inner sub-tokens are masked with **label_pad_token_id = -100**, which is ignored by the slot loss. I train with AdamW, linear warmup and decay, gradient clipping, and early stopping on dev F1. Typical settings: max length 128, batch size 32, learning rate $5 \cdot 10^{-5}$, clip $= 1.0$, patience $= 3$, for up to 50 epochs (best checkpoint on dev is used for test).

2.4. Evaluation

Intent is measured by accuracy; slot filling uses CoNLL chunk-level F1 via the standard evaluator. For all runs, I report test scores from the checkpoint with the best dev F1. Parameter counts are computed over trainable parameters only.

3. Results

Table 1 summarizes performance on the ATIS test set. Moving from a unidirectional LSTM to a BiLSTM produces consistent gains for both tasks, confirming the benefit of leveraging right-context for slot labeling and intent prediction. Adding dropout on top of the BiLSTM provides a further, smaller boost by regularizing the encoder and the task heads.

Fine-tuning **bert-base** delivers the best results overall, with clear improvements on both intent accuracy and slot F1 compared to the BiLSTM variants, while also converging in fewer epochs. This is expected: pre-training equips BERT with strong lexical and syntactic priors that transfer well to sequence labeling and sentence classification. In practice, most of the intent gains appear early in training, while slot F1 continues to improve for longer, which supports the design choice of early stopping on slot F1.

Implementation details matter for stability. Gradient clipping prevents occasional exploding gradients in **LSTM** training; the stratified dev split avoids skew from rare intents and masking inner sub-tokens with **-100** is crucial for fair slot evaluation under WordPiece tokenization. The parameter counts also illustrate the trade-off between capacity and efficiency: while ˜110M parameters in BERT produce the best accuracy/F1, the ˜1.1M BiLSTM with dropout gets surprisingly close given its size, and may be preferable in constrained settings.

Overall, the progression LSTM $\rightarrow$ BiLSTM $\rightarrow$ BiLSTM+dropout $\rightarrow$ BERT shows a clear and monotonic improvement on ATIS.

Learning-curve evidence. Dev F1 (Fig. 1) shows BERT reaching the highest F1 the fastest; by the end of training the bidirectional LSTM without dropout slightly edges the dropout variant, and both outperform the unidirectional baseline. Dev loss (Fig. 2) mirrors this ranking; a slight late uptick for BERT suggests overfitting if training continued, which motivates early stopping on dev F1. Training loss (Fig. 3) illustrates optimization behavior: BERT descends most rapidly, and the bidirectional variants train faster and lower than the unidirectional LSTM.

Table 1: *ATIS test results: intent accuracy and slot F1 (higher is better).*

Experiment	#Params	Intent Acc	Slot F1
LSTM	0.7 M	94.40	92.43
BiLSTM	1.1 M	94.85	93.64
BiLSTM + Dropout	1.1 M	95.74	94.36
BERT	109.6 M	**97.09**	**95.97**

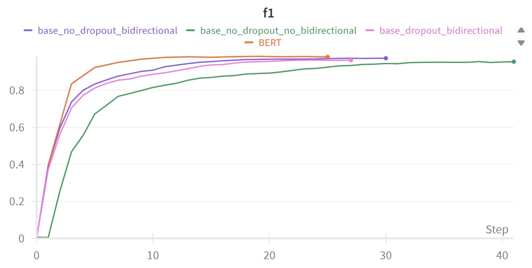

Figure 1: *Dev F1 over training (higher is better).*

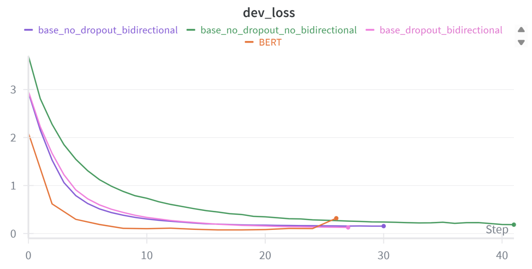

Figure 2: *Dev loss over training (lower is better).*

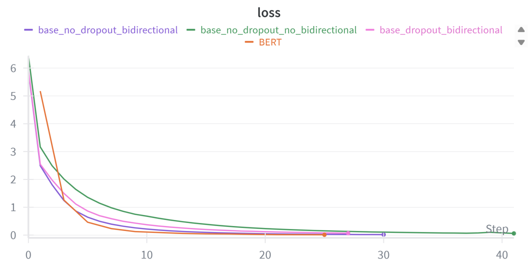

Figure 3: *Training loss over steps.*

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