Sentence Splitter
Multilingual Natural Language Processing
Homework 2

Ercoli Fabio Massimo
802397

Della Porta Nicolò
1920468

1 Introduction

Quoting [Redaelli and Sprugnoli, 2024] "Sentence splitting, that is the segmentation of the raw input text into sentences, is a fundamental step in text processing". According to [Frohmann et al., 2024] the main challenges are:

- robustness to missing punctuation
- effective adaptability to new domains
- high efficiency

According to [Redaelli and Sprugnoli, 2024] we can add to the list:

- multilinguality

Because a sentence splitting that works well for English may not work well to split another language.

In this project we implemented two models for sentence splitting, using an Italian corpus as train and validation set. The first one is based on an embedding model, the second one is based on a generative LLM. We aim to analyze and compare the two approaches. We also plan to test the models out-of-domain.

2 Methodology

2.1 Embedding-based

We fine-tuned a pretrained embedding model using the train and validation datasets provided by the homework guide.

The original dataset provides two large texts (one to use as train and the other as validation) together with the golden labels to mark the end of sentence (1) and all the rest of the words (0).

In order to make the datasets suitable for training, we had to apply some transformations. We needed to group words into sequences of tokens, aligning the golden labels consistently. The number of tokens of each sequence must fit the max length of the embedding models, for instance 512.

We generated a dataset using sequences of 192 words each. Notice that the number of tokens for each sequence will be strictly greater, since for each word the tokenizer of the model will produce one or more tokens.

Before using the datasets, we still need to align (as mentioned above) the labels to the tokens. The alignment strategy we applied consists of keeping 1 as the first token generated from a word having label 1, and using 0 for all the other cases.

One aspect that we had to address was the fact that the label distribution is very unbalanced for this use case. Most of the labels are 0s, and few are 1s. We applied 2 different strategies.

First, we set the *load_best_model_at_end* training argument to *true*, using *metric_for_best_model* set to F1. This implements early stopping based on F1, rather than on accuracy (which would be misleading for such an unbalanced dataset).

Second, we overrode the loss function to weight misclassification of class 1 (end of sentence) 30× more than class 0. This weighting factor was motivated by the label distribution (96.7% class 0 vs 3.3% class 1). We implemented this using a custom weighted trainer.

During inference, we don't need to align the labels, since the labels are generated by the model.

2.2 LLM-based

Fine-tuning an LLM is usually prohibitive or impossible using modest hardware, such as a single GPU with 8 GB / 16 GB RAM. So we applied two techniques to reduce the amount of memory required (and also the time) to fine-tune the models: LoRA (Low-Rank Adaptation) and parameter quantization.

A special focus was given to creating good prompts for both the training and inference phases.

We first generated new datasets from the ones we used for the embedding-based solution, in which for every sequence we have:

- *input_text*: here we simply concatenate all words of the sequence into a single text
- *output_text*: here we group all the words that belong to the same sentence on different lines, numbered sequentially

This is not a prompt yet. To create the final prompts, we created dictionaries of roles (system / user / assistant) with contents (we called them conversations) and passed them to the LLM tokenizer in order to have them converted into the proper chat templates.

We used the Supervised Fine-Tuning (TRL SFT) Trainer to train the model.

The fine-tuned model is (as usual) pushed to the Hugging Face repository, so that it can be used for inference from anywhere.

3 Experiments

3.1 F1 on different models - same domain

The performance goal we set is to maximize the F1 score on the validation set. The test set is unlabeled, so we cannot use it. Our expectation is that with a good result on the validation set can generalize nicely on the test set.

For the embedding-based solution, we implemented the approach using BERT-base-cased as the base model. While we considered several Italian and multilingual models during development:

- ModernBERT-base-ita (Dec 2024)
- Italian BERT XXL (most established)
- XLM-RoBERTa base
- XLM-RoBERTa large
- Italian ELECTRA

The actual implementation uses BERT-base-cased for demonstration purposes.

For the LLM-based solution, we implemented the approach using Qwen3-4B Instruct. We considered several instruction-tuned models during development:

- Qwen3-4B
- Meta-Llama-3.1-8B-Instruct
- mistral-7b-instruct-v0.3-bnb-4bit

Table 1: Best F1-epoch Embedding-based

Models	Best Epoch	F1	Valid loss
BERT base cased	6	0.9922	0.0014
ModernBERT base ITA	12	0.9938	0.0097
BERT base ITA XXL cased	11	0.9922	0.0021
XLM RoBERTa base	4	0.9953	0.0014
XLM RoBERTa large	12	0.9953	0.0087
Electra ITA XXL disc.	20	0.9953	0.0015

The actual implementation uses Qwen3-4B Instruct (via unsloth/Qwen3-4B) with 4-bit quantization and LoRA fine-tuning.

These tests are executed on the same domain, since all the datasets (train, validation and tests) come from the same novel: *I Promessi Sposi*.

3.2 Out-of-domain tests

We designed experiments to test the same models on different authors. For this purpose, we planned to use the dataset from [Redaelli and Sprugnoli, 2024], in which different authors from the same century are used. The dataset has already the gold results annotated by the authors of the paper.

4 Results

4.1 Embedding-based: in-domain

Tables 1 reports the F1 scores and the losses on validation set for the best-F1 epoch, training the network with 30 epochs. This is of course an in-domain evaluation, since the validation set is from the same author (Manzoni) and even also the same novel (I Promessi Sposi). The scores look not so bad, probably we can expect they will diverge more in case of out-of-domain evaluation.

4.2 Embedding-based: out-of-domain

4.3 LLM-based: in-domain

4.4 LLM-based: out-of-domain

4.5 Embedding vs LLM: comparison

This section is planned to compare the two approaches: embedding-based vs LLM-based. The comparison will be conducted once both implementations are fully executed and evaluated.

The planned comparison metrics include:

- F1 score - in-domain (Manzoni dataset)
- F1 score - out-of-domain (other 19th century Italian authors)
- Memory usage during training and inference

- Training time requirements
- Inference speed

Note: Actual results will be populated after running the experiments described in the notebooks.

A Additional details/describe Appendix

References

Markus Frohmann, Igor Sterner, Ivan Vulić, Benjamin Minixhofer, and Markus Schedl. 2024. Segment any text: A universal approach for robust, efficient and adaptable sentence segmentation. *Preprint*, arXiv:2406.16678.

Arianna Redaelli and Rachele Sprugnoli. 2024. Is sentence splitting a solved task? experiments to the intersection between NLP and Italian linguistics. In *Proceedings of the 10th Italian Conference on Computational Linguistics (CLiC-it 2024)*, pages 813–820, Pisa, Italy. CEUR Workshop Proceedings.