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

A sentence splitting approach that works well for English may not work well for splitting text in another language.
In this project, we implemented two types of models for sentence splitting, using an Italian corpus as training and validation sets. The first kind is based on embedding models, the second is based on generative LLMs. We aim to analyze and compare the two approaches. We also plan to test the models out of the original domain of training. The training data consists of golden-annotated text from I Promessi Sposi (Quarantana) by Manzoni.

2 Methodology

2.1 Embedding-based

We fine-tuned a series of pretrained embedding models using the training and validation datasets provided by the homework guide.
The original dataset provides two large texts (one to use as training and the other as validation) along with the golden labels to mark the end of sentences (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, and to tokenize words into tokens, aligning the golden labels consistently.
With the constraint that the number of tokens of each sequence must fit the maximum 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:

1. 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).

2. 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 LLMs is often prohibitive on modest hardware, such as a single GPU with 16 GB or 24 GB RAM. Therefore, we applied two techniques

to reduce the memory requirements and training time: LoRA (Low-Rank Adaptation) and parameter quantization.
LoRA works by freezing the original model weights and introducing trainable low-rank decomposition matrices, significantly reducing the number of trainable parameters. We used 4-bit quantization (BNB-4bit) to further reduce memory usage by representing model weights with lower precision while maintaining performance.
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.

In the process, we also add the spaces between the words, paying attention not to introduce a space before a punctuation sign or after an apostrophe.
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 models are (as usual) pushed to the Hugging Face repository, so that they can be used for inference from anywhere.

3 Experiments

3.1 Embedding-based: Training Evaluation

The performance goal we set is to maximize the F1 score on the validation set. Our expectation is that a good result on the validation set can generalize nicely on the test set.
For the embedding-based solution, we tested the following base models:

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

For the LLM-based solution, we tested the following base models:

- Qwen3 4B
- Meta Llama 3.1 8B Instruct
- Mistral 7B Instruct v0.3 bnb 4bit
- Minerva 7B Instruct v1.0

As we said, both test and validation sets come from the same novel: *I Promessi Sposi*.

3.2 Out of Domain Evaluation: Eval Dataset

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.
We evaluated the F1 score of the predictions, having the annotated novels as ground truth.
We produced labels from both the annotated novels and from predictions. For simplicity of the implementation, for this phase, we produce a label for each character, instead of a label for each token.
The golden-annotated novel rows were grouped in chunks of equal size of lines, to produce sequences.
We tested the in-domain novel (Quarantana) and 3 novels (Pinocchio, I Malavoglia and Cuore) from different authors of the same period.

3.3 Out of Domain Evaluation: Test Dataset

Finally, we chose two embedding-based models (Italian BERT XXL and Italian ELECTRA) from the 6 we trained and one LLM-based model (Minerva 7B Instruct v1.0) from the 4 we trained to compute the inference on the test set.
The test set is still an out-of-domain dataset (with respect to the training and evaluation sets, of course), but this time it wasn't chosen by us. Unlike the Eval Dataset, which was chosen by us, it was assigned by the homework.

4 Results

4.1 Embedding-based: Training Evaluation

Table 1 reports the F1 scores and the losses on the validation set for the best-F1 epoch, training the

Table 1: Embedding-based Models: Best Epoch Performance on Validation Set

Models	Best Epoch	F1	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

Table 2: Out-Of-Domain Evaluation: Eval Dataset

Models	Quara.	Pinocc.	Malav.	Cuore
BERT base cased	0.9922	0.8058	0.7304	0.8757
ModernBERT ITA	0.9675	0.8495	0.8128	0.8864
BERT ITA XXL	0.9938	0.9947	0.9205	0.9591
XLM RoBERTa B	0.3953	0.5427	0.1980	0.2424
XLM RoBERTa L	0.5275	0.6618	0.3559	0.5
Electra ITA XXL	0.9923	0.9785	0.9171	0.9655

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).
On training, the F1s on the validation set look very good.

4.2 Out of Domain Evaluation: Eval Dataset

When we evaluate the same models on different novels of the Eval dataset, we get the results reported in table 2.
According to our tests, multilingual models (XLM-RoBERTa base and large) performed poorly, even in the same domain (Quarantana).
This poor performance might be attributed to the multilingual models' distributed attention across many languages, making them less effective for language-specific tasks compared to monolingual or Italian-specific models.
Interestingly, BERT base (not specifically trained on Italian) outperformed the multilingual models, suggesting that the general language understanding capabilities of BERT transfer better to Italian sentence splitting than the multilingual approach.
This base model, like ModernBERT ITA, works very well in domain (Quarantana) and decently out of domain (other novels).
BERT ITA XXL and Electra ITA XXL got great results both in domain and out of domain.
Among the out-of-domain novels, I Malavoglia appears to be the most challenging for sentence splitting across all models.
This could be due to Verga's distinctive writing style, which includes frequent use of dialogue and regional expressions that may differ significantly from Manzoni's more formal narrative style used in training.
As expected, the best results are consistently achieved on the in-domain novel (Quarantana), demonstrating the importance of domain similarity for optimal performance.

Table 3: Out-Of-Domain Evaluation: Test Dataset

Models	Type	F1
BERT ITA XXL	embedding-based	0.8770
Electra ITA XXL	embedding-based	0.8994
Minerva 7B Instruct v1.0	LLM-based	0.7445

4.3 Out of Domain Evaluation: Test Dataset

A special focus was given to the evaluation of the performance on the test set. Not to make any design choice but to provide a final evaluation of the trained models (seeing those as immutable instances). From the F1 scores we computed (see table 3), carefully chosen embedding-based models consistently outperform the LLM-based approach.
The Italian ELECTRA model achieved the highest F1 score of 0.8994, followed closely by Italian BERT XXL at 0.8770, while the Minerva 7B model reached 0.7445.
Furthermore, LLM-based models have a series of further issues:

- they are one order of magnitude slower than embedding-based
- we cannot use CPU hardware even for inference
- they can hallucinate

4.4 What are the recurrent errors committed by the models?

As inference notebook output, we produced the three CSV files:

- Lost_in_language_recognition-hw2_split-bert-base-italian-xxl-cased.csv
- Lost_in_language_recognition-hw2_split-electra-base-italian-xxl-cased.csv
- Lost_in_language_recognition-hw2_split-Minerva-7B-based.csv

Thanks to these files, we can analyze the misclassification of the models.
Looking at the errors it seems that the models tend most of the time to split the sentence where it is

not necessary.
For instance, "C'era una volta..." or "C'era una volta... -" has been misclassified by all our trained models as a sentence, but it is not. According to the golden labels, the sentence should have included other words.

Sometimes it can happen the opposite, namely the models do not recognize the beginning of a new sentence, as in the sentence:

- `"guardò dentro un armadio che stava sempre chiuso, e nessuno;"`

All the models classified ';' as 0. Looking at the output log of the 'sentence_splitter_out_of_-domain_test_generative.ipynb', we can see a few hallucinations (not possible with embedding-based models).
In the output it can be seen how some words have been transformed by the model, for instance:

- `messe` → `mise`
- `trasfigurito` → `trasfigurato`
- `imprestezziti` → `impresciuttiti`.

We can say that the model may use a more-common-used word in the output (violating the fact that the output should only refer to the input words!)
One hallucination is about portions of the sequence that are not present (they were not reported in the output), but this phenomenon seems to be quite rare.
A word ('Sì! –') in the input has been completely forgotten by the model when producing the output.

5 Conclusions

The embedding-based models demonstrate superior efficiency, requiring at least one order of magnitude fewer parameters than LLM-based approaches. When a well-suited Italian model is chosen (such as BERT ITA XXL or Italian ELECTRA), results are excellent both in-domain and out-of-domain, with F1 scores consistently above 0.87 on the test set.
The LLM-based approaches also provide reasonable results in terms of quality, but they suffer from hallucination issues, which are impossible with embedding-based approaches due to their discriminative nature.
Moreover, LLMs are significantly more resource-intensive to train and deploy, requiring specialized hardware even for inference.
LLM-based models take from 35 minutes to 1 hour to be trained on a GPU with maximum memory of 22.069 GB.
Moreover, we had to use LoRA and parameter quantization in order to train the models on modest hardware.
All in all, the team thinks that using an embedding-based model would be the natural choice to address the sentence splitting use case.
This follows the common software engineering principle: there is no universal solution for everything. LLMs work great for many use cases, but not in all cases.

A Further analysis

A.1 Possible mitigations for the recurring errors

Regarding the embedding-based approach, we thought about different possible mitigations in order to try to avoid some recurring errors. One idea that came to our mind is to use some confidence-aware rules (i.e. to add some filters that may help us to better understand risky punctuation):

- in the case of an ellipsis (...), it should require extra cues to accept EOS (i.e. check if next token is uppercase and not inside quotes or dashes);
- in the case of a dash dialogue (- [..] -), it shouldn't close the sentence if the dash is used as to open the dialogue, unless a proper case start follows and the clause ends before next dash;
- in the case of a semicolon (;), similar considerations should apply.

For the generative-based approach, one helpful mitigation may be to restrict the output to the exact input tokens plus <EOS> special token and to post-validate by checking if any non-input token appears or if any input token disappears. If this occurs, the produced text should be rejected and the model is re-prompted with an explicit constraint reminder, as to avoid hallucinations.

A.2 Why Electra and Bert ITA XXL definitely perform better?

Electra's pretraining task — Replaced Token Detection (RTD) — trains a discriminator to decide, for each token, whether it's "real" or "replaced."

This is a dense, per-token supervision signal on all positions and, thus, tends to yield sharper token-level representations, which works really well for EOS classification. Furthermore, many models are meant for cross-lingual transfer, but for monolingual, syntax-sensitive token tasks in a specific language (such as Italian 19th-century prose!), their huge multilingual vocabulary/parameters are partially "wasted" on non-Italian distribution. On the contrary, models such as Electra and BERT ITA XXL, which are trained on large, Italian-heavy corpora, have very good vocabulary coverage and distributional knowledge of Italian punctuation habits, apostrophes, dialogues with dashes/quotes, that really helps to reduce subword fragmentation and ambiguity.

B Notebook Descriptions Appendix

B.1 Notebook List

sentence_splitter_discriminative_training.ipynb: training of the embedding-based models.
sentence_splitter_generative_training.ipynb: training of LLM-based models.
ss_out_of_domain_eval_discriminative.ipynb: inference on the Eval dataset.
ss_out_of_domain_test_discriminative.ipynb: inference on Test dataset, using embedding-based models.
ss_out_of_domain_test_generative.ipynb: inference on Test dataset, using LLM-based models.

B.2 Google Colab Execution

The notebooks have been developed to be run on a JupyterLab environment.
To run those on Google Colab, you shall replace:

```
os.environ['HF_TOKEN']
```

with

```
userdata.get('HF_TOKEN')
```

You will also need to add the following line to the input:

```
from google.colab import userdata
```

Moreover, if you want to run the trainings part, you will need to change the Hugging Face repository (in the notebook *fax4ever*), providing consistently your Hugging Face token. Differently, you will not need to set the Hugging Face token in order to run the inference notebooks.

C Models and Datasets Appendix

All the fine-tuned models and the datasets were published to the Hugging Face Hub.

C.1 Embedding-based released models

- bert-base-cased-sentence-splitter
- ModernBERT-base-ita-sentence-splitter
- bert-base-italian-xxl-cased-sentence-splitter
- xlm-roberta-base-sentence-splitter
- xlm-roberta-large-sentence-splitter
- electra-base-italian-xxl-cased-discriminator-sentence-splitter

C.2 LLM-based released models

- qwen3-4b-unsloth-bnb-4bit-sentence-splitter
- meta-llama-3.1-8b-instruct-unsloth-bnb-4bit-sentence-splitter
- mistral-7b-instruct-v0.3-bnb-4bit-sentence-splitter
- Minerva-7B-instruct-v1.0-sentence-splitter

C.3 Datasets

The main datasets to train the embedding-based models and LLM-based models are:

- manzoni-192
- llm-manzoni-192
- sentence-splitter-ood-192
- sentence-splitter-ood-128

D Indeterminism Appendix

We tried to make the runs as deterministic as possible, but without affecting the performance.
It means that some output derived from indeterministic algorithms may change with the executions.
Making all deterministic would have affected too much the performance.
So, it means that if you run everything from scratch, you could find slightly different results.
For instance the second time we re-trained everything from scratch we got different results for the out of domain evaluations ()see Table 4).

Anyway, we can confirm the same analysis we made on the take-1. That's the hard life of the data scientists, even in the Deep Learning era!

Table 4: Out-Of-Domain Evaluation: F1 Scores on Different Novels (Second Run)

Models	Quara.	Pinocc.	Malav.	Cuore
BERT base cased	0.9859	0.8854	0.7826	0.9364
ModernBERT ITA	0.9130	0.5603	0.6667	0.8144
BERT ITA XXL	0.9939	0.8774	0.9121	0.9486
XLM RoBERTa B	0.3952	0.5426	0.2115	0.2424
XLM RoBERTa L	0.5275	0.6618	0.3559	0.5
Electra ITA XXL	0.9938	1.0	0.9348	0.9385

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.