MNLP Homework 1.b Task 9 Report

Pisano Raffaele
`pisano.1959863@studenti.uniroma1.it`

1 Introduction

The Homework is about creating a model that classifies sentences as 'hateful' or 'neutral'.

1.1 Description of the Dataset

The dataset is divided in 3 '.jsonl' files, two for testing and one for training and validation. From the file 'train-taskA.jsonl' I've derived the files:

```
- train.jsonl
- validation.jsonl
```

80% of the strings are in the training and the remaining into the validation.

2 Model Architecture

I've used an embedding layer, a bidirectional LSTM layer, and a linear projection layer.

3 Design choices of your model

Embeddings: I've decided to use pretrained embeddings to capture better the context of each word. I've tried 3 different one embeddings:

```
- itwac218.sqlite| www.italianlp.it
/resources/italian-word-embeddings/
- cc.it.300.bin| fasttext.cc/
docs/en/crawl-vectors.html
- torchtext.vocab.FastText('it')
```

The first ones are italian embeddings with a dimension of 128. The last two are both Fasttext embeddings with a dimension of 300, the difference between the 2 is that the second one, that is the one used in the code, has more embedded words. So I've decided to use it with better performances as result.

Tokenizer: I've used 'Spacy' as a tokenizer, but observing the words missing in the pretrained embeddings, I've seen some strings not being split correctly. So, I've implemented some custom rules for better string splitting if the word splitted by spacy is not in the embeddings.

Learning rate and early stopping: I've observed through multiple tests that regardless of the model configurations, the best validation loss achieved is around 0.48 at some epoch. However, a recurring issue is that the loss tends to increase in subsequent epochs. To address this, I significantly reduced the learning rate if the loss dropped below 0.485 at any point, aiming to keep it within that range and allowing the model to continue learning slowly. Additionally, I implemented an early stopping mechanism to halt training prematurely if the validation loss continues to increase for several epochs (3 in the case of best performance).

4 Baselines

I've created two baseline models stored in the file 'baseline.ipynb'. The first one is random: it assigns a random class to each string and typically achieves an accuracy between 0.47 and 0.52. The second one assigns the most common class to each string and achieves a final accuracy of 0.638.

5 Results

with 'cc.it.300.bin' embeddings:
(Batch 32, layers 3, dropout 0.2, lr 5e-05)

```
valid_acc: 0.781 | loss: 0.472 |
tests: news_acc: 0.701 | tweets_acc: 0.729
```

Significantly higher than the baseline.

6 Instruction to run the code:

To run the code is sufficient to open the notebook 'hw1b.ipynb' and use the run all button and connect to drive when requested. But the file of the embeddings needs to be on drive in the path: 'MNLP_HM1.b/cc.it.300.bin'. The code will automatically stop at the 7th model trained (when the loss of the model goes under 0.47), thanks to a loop that break when the model has good results.

7 Bonus:

The extra baseline is a simle model using BiGRU instead of BiLSTM. It doesn't use a custom tokenizer but only spacy, it doesn't use pretrained embeddings and no scheduling of the learning rate. It is in the file 'hw1b_extra.ipynb'. This is the result with the same parameters:
(Batch 32, layers 3, dropout 0.2, lr 5e-05)

```
valid_acc: 0.781 | loss: 0.472 |
tests: news_acc: 0.701 | tweets_acc: 0.729
```