Assignment 3

GSN Team

May 2019

1 Data

In the Assignment 3, we will use the data from PolEval 2018 contest Language Models task. This dataset contains over $20M$ sentences in Polish language. As training models on such huge corpus might be computationally expensive - you can use no less than $50K$ examples of this dataset for preparation of the solution for this task. 20% of the dataset should be used as a separate hold-out set for the final validation.

2 Task

Your task is to implement a charwise word embedder for Polish language. The embedder should be trained as a part of the language model in a manner similar to [1] and [4] - where one (or more) word is masked and the model needs to determine the deleted word.

Below you can find the detailed description of what needs to be done.

2.1 Corpus Preprocessor - 0.5pts

You should implement a `CorpusPreprocessor` class which should have the following interface:

- `transform_text(text)` - a method which transforms every `text` from corpus into one which:
  - contains only letters and spaces,
  - removes double spaces and trims text.
- `mask_text(text)` - a method which is the base for creation of a language model dataset. It should:
  - Randomly choose a word (later referred as `orig_word`) from a sentence and exchange it with a predefined `MASK` word obtaining `masked_sent`.
  - Flip a coin (with a probability of success $p = 0.5$) and:

  - ∗ in case of a success return `(masked_sent, original_word, 1)`,
  - ∗ otherwise sample a random word from the corpus (later referred as `random_word`) and return `(masked_sent, random_word, 0)`.

Use the methods above as the base for preparation of your training and validation set for this task.

2.2 Language and embedding model - 2pts

The main task is to implement a model which will consist of two submodels:

- **Word embedder** - a model which will accept a word in a form of a sequence of characters. It should contain at least one convolutional or recurrent layer.
- **Main Language Model** - a model which will accept representation obtained by a **word embedder** and using bi-directional `RNN` (of any kind) will obtain a full sequence representation.

The final model should predict whether for a given `masked_sent` an additional `word` provided was the one which was masked or not.

The description of the model is intentionally left vague as we expect you to come up with your own ideas for model architectures. Because of that, we expect classifiers to surpass only the 60% level of accuracy (note that the lift from a random classifier is only 10% as the main classification problem is designed to be balanced). Remember that *characterwise* word embeddings are believed to be slightly worse than the one based on tokens or words [3], so results over 70% should be considered as really good.

2.3 Embedding visualization - 1.5pts

You should use a T-SNE visualization method and try to visualize the embedding obtained by the **Word embedder** from a previous task. We expect you to look for the various semantic relations between the words. The report should include examples of the found dependencies (or details of the experiments in case when no such dependencies were found).

3 Extra Exercises

You can try to solve the following exercises to get additional extra points:

- **Solving a cyberbulling detection task** - you can use either word representation or full text representation obtained using one of the models to build a solution for one (or two) of cyberbulling detection tasks from PolEval 2019, Task 6: Automatic cyberbullying detection. You may try to reuse a popular fine-tuning procedure presented in [2].

- **Context Embedding Visualization** - you can try to analyze contextual embeddings obtained by the **Main Language Model** and analyze how embeddings of the same word differ depending on the context. You can try to look for the words with the most and least ambiguous representations.
- **Named Entity Recognition** - you can try to pretrain a model for PolEval 2018 Task 2: Named Entity Recognition contest. Once again you may try to reuse a popular fine-tuning procedure presented in [2].
- **Training attention model** - you can exchange `CNN` or `RNN` implementation of any of models above to attention-based architecture [5].

4 Deadline

You should submit your solution by email by 23:59 on 04.06.2019 (Tuesday) to your lab teacher with email title ”Assignment 3 - Deep neural networks”. Your code will be inspected during the lab session following the deadline. Note that even if you are one minute late after the deadline, your solution will not be inspected. We have no mercy whatsover so you better not count on that.

References

[1] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv e-prints*, page arXiv:1810.04805, Oct 2018.

[2] Jeremy Howard and Sebastian Ruder. Universal Language Model Fine-tuning for Text Classification. *arXiv e-prints*, page arXiv:1801.06146, Jan 2018.

[3] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2018.

[4] Wilson L. Taylor. “cloze procedure”: A new tool for measuring readability. *Journalism Bulletin*, 30(4):415–433, 1953.

[5] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention Is All You Need. *arXiv e-prints*, page arXiv:1706.03762, Jun 2017.