Homework 1

Natural Language Processing 2018/2019

Contacts

[zinnai | bevilacqua | spadoni | scarlini | blloshmi] @di.uniroma1.it

If you write on the **Facebook group**

we will love you much more <3

What you will do

- Create your word segmenter for Chinese

- Participate in a competition with prizes!

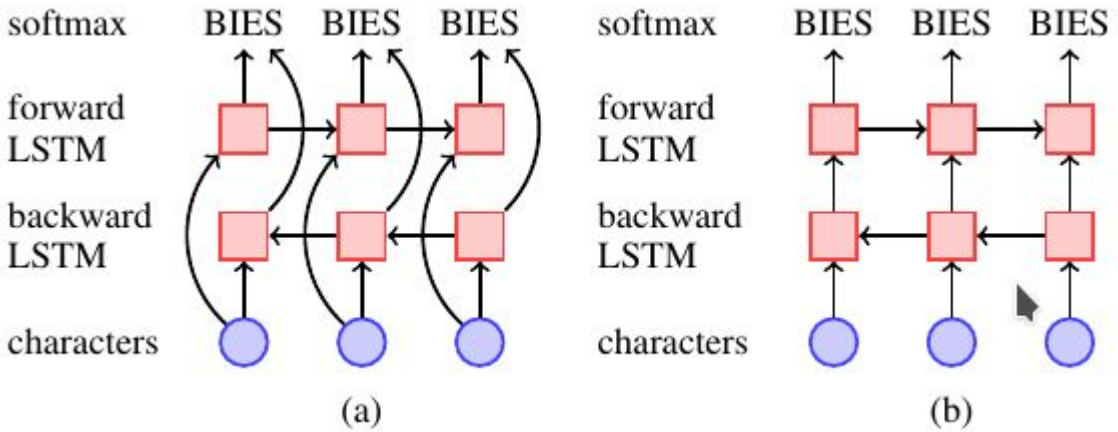
Homework outline

- Implement a state-of-the-art **word segmenter** model in **Tensorflow/Keras**
  - We will provide a scoring function, train and dev datasets, the original paper
- **Train** it using the provided datasets and find the best hyperparameters through **grid-search**
- Implement the **predict** function in the *predict.py* file to make predictions with your trained model
- Write a **report** where you describe your approach, your result analysis and the improvements you have made over the baseline (if any)
  - Text report **max 1 page**, text after the 1st page will not be considered
  - Images and tables **max 2 pages**
- **Submit your code and report**

What we provide

- Training and development **datasets** to train the model
- A **function to score** the accuracy of your model (`score.py`)
- A **skeleton for the predict** function (`predict.py`)
- These slides as guidelines

The model



State-of-the-art Chinese Word Segmentation with Bi-LSTMs

Ji Ma, Kuzman Ganchev and David Weiss, EMNLP 2018, link to paper

It is not mandatory to use the pretrained embeddings

The model

- The model to implement is a sequence tagging model
- The canonical model: **embeddings → rnn cell → dense output layer**
- Given a sequence of input characters the model has to predict one label for each character
- To speed up the learning phase, you have to process multiple sequences at once (**batch**)
- Each sequence has a different length so you should take that into consideration and use **padding** when needed.

BIES Format

The bies format is a way to encode the output of a word segmenter model

There are 4 classes the model has to predict.

- **b:** beginning
- **i:** inside
- **e:** end
- **s:** single

Example:

theanswerіs42! → BIEBIIIIEBEBES → the answer is 42 !

Datasets

Download the dataset: http://sighan.cs.uchicago.edu/bakeoff2005/

The full dataset contains **four smaller** datasets:

- AS (*Traditional Chinese*)
- CITYU (*Traditional Chinese*)
- MSR (*Simplified Chinese*)
- PKU (*Simplified Chinese*)

Datasets

- Each dataset has one train and one dev set
- You can find the train datasets in the ***training*** folder
- Your dev set is in the ***gold*** folder
- For each dataset use the file with the **.utf8** extension

Datasets

- You can use the train datasets as you **prefer**, you can **concatenate** them, use **just one**, **etc.**
- Normally more training data is **better,** but as you develop and tune your model, using too much data will **slow you down**
- Describe in the report how you have used the available datasets
- The development datasets are **quite big**, maybe you prefer to use just a **subset**

Preprocessing

- You are responsible to convert the *Traditional Chinese* datasets to *Simplified Chinese*
    - Install HanziConv and run the following command: `hanzi-convert -s infile > outfile`
- Implement a preprocessing script (**preprocess.py**) that reads the original file in the ***training*** or ***gold*** folder and produce an *input file* with no spaces and a *label file* in the BIES format.

Original file	Input File	Label File
共同 创造 美好 的 新 世纪 ——	共同创造美好的新世纪——	BEBEBESSBEBE
This file is the one you download. It contains spaces in order to get the labels from it.	*This is the input to your Tensorflow model. Same as original file BUT without spaces.*	*This is the label filename in the BIES format created based on the original file.*

Predict function

- The **predict.py** file contains the skeleton of the **predict function** you have to implement
- We will use your implemented predict function on our (**secret**) test set to score your model
- Be sure that it works well otherwise we will not be able to evaluate your model performances
- The *predict* function takes **three inputs**:
  - The path of the **input file** (the one you should predict)
  - The path of the **output file** (where you save your BIES predictions)
  - The path of the **resources folder** (the one containing your model and any additional data you might need)

Score function

We provide you with a score function in `score.py`. **You should not modify it!**

The score function takes **two inputs**:

1. The **output** of your system [list of lists]
2. The **gold** dataset [list of lists]

and returns the **precision.**

Both input and gold have to be in the BIES format.

[The function also checks if your predictions are **valid**]

Grid search

Tune your model varying the following hyper-parameters:

- Char embedding size
- Bigram embedding size
- Learning rate
- Batch size
- Input dropout rate
- RNN dropout rate
- RNN hidden size
- Number of stacked RNNs

Grid search

TIPS

- Find the best set of **hyper-parameters** by training the model on a subset of the train dataset and test it on the dev
- Test no more than 2-3 values for each hyper-parameter
- Take inspiration from the paper to choose reasonable values
- **Fix** the hyper-parameters set and use it for the **rest of the experiments**

How we will grade

The maximum grade for this homework is 34.5 (115% of 30) weighted as follows:

- Quality, comments and cleanness of code [40%]
- Report [50%]
- Overall performance of the system [10%]
    - This really depends on how much time you spend tuning hyper-parameters
    - Training only on the AS dataset your precision should be above **90%**
- **Creativity boost:** Improvements over the baseline model [15%]
    - You will get extra points if you add some effective features to your model to boost the performance, in this case we will expect a comparison with the baseline described in the paper

What we expect in the report

- **Describe** the model you implemented and the task
- Report the **best hyper-parameters** you found
  - Plot the accuracy and loss while changing the hyper-parameters
- **Describe** how you used the datasets
- **Describe** the decisions you made to improve the model performances
- Report some **interesting plots** regarding the training or screenshots from tensorboard

Implementation details

- After some tests, we found that implementing the model in pure **Tensorflow is at least 1.5X faster with respect to Keras**
- You could use Colab to run your model on GPUs (even TPUs are available)
- You can use in your project *only* the following libraries:
  - **Python 3.7**
  - **Tensorflow (compatible with 1.12.x)**
  - **Numpy**
  - **Sklearn**
  - **nltk**
  - **Maplotlib**
  - **gensim**
  - **All the standard libraries**

Submission

The project will have this **structure**:

- Your report in **pdf** format, **1 PAGE** for text and **2 PAGES** for images/tables/references, named **report.pdf**
- A folder with your source code named **code**, it also has to include the scripts we provided
- A folder containing the resources necessary to run your model and make predictions with the *predict.py* script, named **resources**
- You should submit **ONLY** the best model: if you made some improvements over the baseline of the paper you should submit that one

Submission

- Register to GitLab
- Create a new repository (**private**) with name: <**firstname**>_<**lastname**>_<**matricola**>_**nlp19hw1**
- **Share** the project with us:
  - use the emails on the second slide and
  - navigli@di.uniroma1.it
- The link where you have to submit:

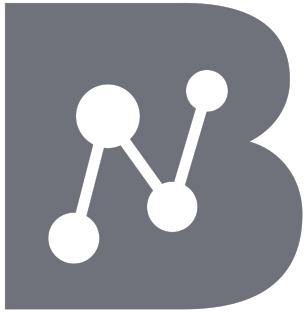
SUBMISSION FORM

Deadlines

- We will upload everything you need to complete this homework Sunday 31 March evening the latest, on the facebook group
- Your deadline for homework 1 will be: **24/04/2019, 23:59** **anywhere on Earth**
- If you push anything after the deadline you will be **out**

Competition results

- The competition scores will be published after the homework evaluations
- The **top 5** ranked students will receive a (super!) fancy **BabelNet T-Shirt!**



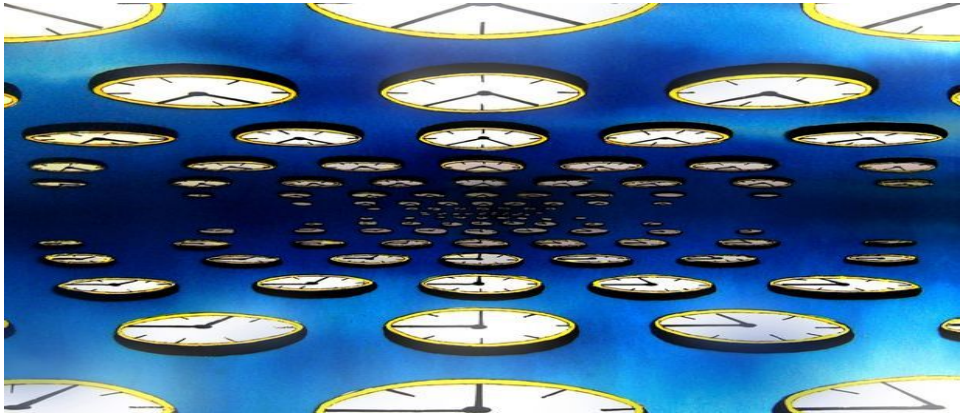
BabelNet

Competition results

- For results above the state of the art, we plan to submit a paper to EMNLP 2019 with you as co-author!

We will check all your submissions for **plagiarism**!

- If we find that you plagiarised you are **OUT** of this year's course and you cannot take the exam, you will have to sign up for the course next year
- We have a **zero-tolerance** policy for plagiarism!



- **Advice**: start as early as possible training the models!
- Training a Neural Network takes time, especially if you train on **CPU**
  - You can use Google Colab
- **Memory** is also an issue: you will probably not be able to load the whole training data into memory, e.g. you will have to create batches and load them lazily
- **Suggestion**: Start implementing the model **ASAP** and train it until you are satisfied with its performance. The rest should take less time.

Good job!

If you have any questions, do not hesitate to post them on the Facebook group