

# Homework 2

---

Natural Language Processing 2018/2019

# Contacts

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

If you write on the **Facebook group**  
we will love you much more <3



# What you will do

- Create your own sense embeddings
- Take a chance to win your first BabelNet t-shirt or maybe get one in another color!

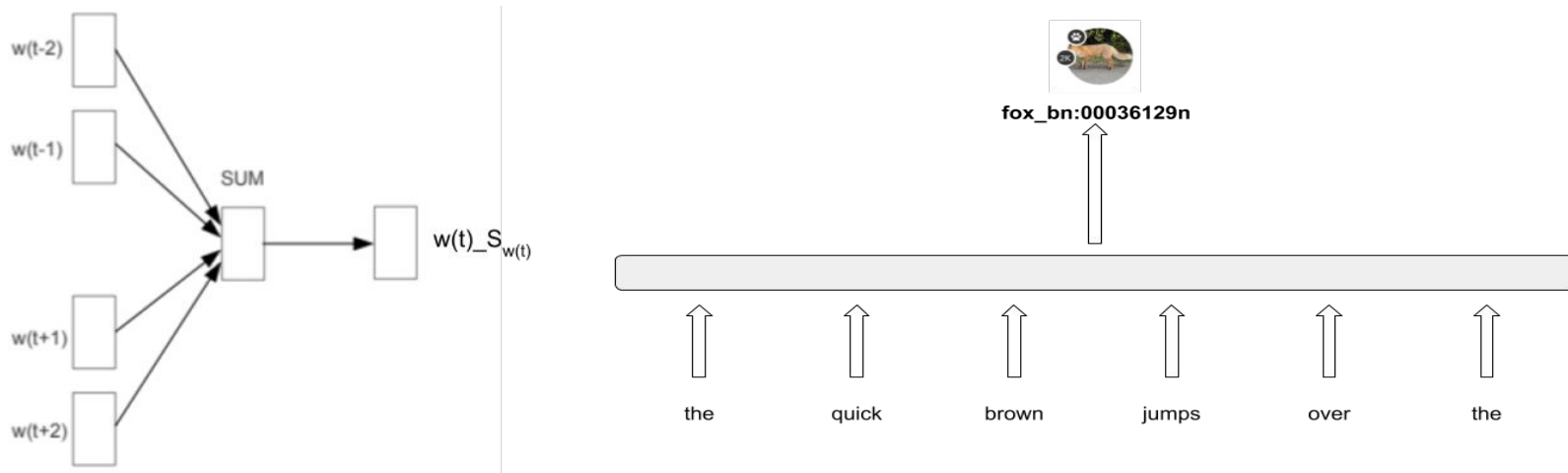
# Homework outline

- Train your own sense embeddings
  - We will provide a reference paper, training corpora and an evaluation task
- **Parse** the corpora to extract the information needed to train the sense embeddings
- **Restrict** the sense embeddings only to senses in WordNet
- **Train** a *word2vec* model to create the sense embeddings
- **Test** the sense embeddings in the word similarity task
- Write a **report** where you describe your approach, your result analysis and any interesting features you implemented
  - Text report **max 2 page**, text after the 2nd page will not be considered
  - Images and tables **max 2 pages**
- **Submit your code, sense embeddings and report**

# What we provide

- Sense tagged **training corpora**
- A reference **paper**
- An **evaluation dataset**
- A **mapping** from BabelNet synsets to WordNet offsets
- These slides as guidelines

# The model



## SENSEMBED: Learning Sense Embeddings for Word and Relational Similarity

Iacobacci, Pilehvar and Navigli, 2015 [link to paper](#)

It is *MANDATORY* not to use the pretrained embeddings

# The model

- The model to implement can be either CBOW or Skipgram
- You can use already implemented version of word2vec models
  - [Gensim](#)
  - [Word2Vec](#)
  - [Glove](#)
  - [Fasttext](#)
  - Or implement your own!
- Given a context of words surrounding the target word, the model has to predict the correct word sense represented as <lemma>\_<synset>
- Same as standard word embeddings but with word senses as outputs

# EuroSense corpus

**Download** the *mandatory* corpus: <http://lcl.uniroma1.it/eurosense/>

- The corpus consists of a single large XML file (21GB uncompressed for the high precision version)
- You might not be able to load the whole file in memory (you can take a look at the **iterparse** function in the *lxml.etree* library for python to solve the issue)
- It is a multilingual corpus but you can use only the **English sentences**
- It is up to you to choose between the high precision and the high coverage version



# EuroSense corpus

- Each sentence is translated into several languages identified by the **lang** attribute in the `<text>` tag. They are already tokenized so you just need to split by space

```
<sentence id="0">
  <text lang="de">Ich halte es für äußerst wichtig , die Grauzonen [...]</text>
  <text lang="en">I do believe that it is vital to minimise the grey areas [...]</text>
  ... // more languages

  <annotations>
    <annotation lang="en" type="NASARI" anchor="areas" lemma="area"
      coherenceScore="0.2247" nasariScore="0.9829"> bn:00005513n</annotation>

    ... //more annotations
  </annotations>
</sentence>
```

# EuroSense corpus

- Each sentence has its own annotations depending on the **lang**
- Each annotation marks the sense for a word in text identified by the **anchor** attribute

```
<sentence id="0">
  <text lang="de">Ich halte es für äußerst wichtig , die Grauzonen [...]</text>
  <text lang="en">I do believe that it is vital to minimise the grey areas [...]</text>
  ... // more languages

  <annotations>
    <annotation lang="en" type="NASARI" anchor=" areas" lemma="area"
      coherenceScore="0.2247" nasariScore="0.9829"> bn:00005513n</annotation>

    ... //more annotations
  </annotations>
</sentence>
```

# EuroSense corpus

- Remind that your sense embedding must be represented as **lemma\_synset**
- Each annotation provides you with the **lemma** of the word it is tagging and the **synset id**

```
<sentence id="0">
  <text lang="de">Ich halte es für äußerst wichtig , die Grauzonen [...]</text>
  <text lang="en">I do believe that it is vital to minimise the grey areas [...]</text>
  ... // more languages

  <annotations>
    <annotation lang="en" type="NASARI" anchor="areas" lemma="area"
      coherenceScore="0.2247" nasariScore="0.9829"> bn:00005513n</annotation>

    ... //more annotations
  </annotations>
</sentence>
```

# Additional Corpora (optional)

- You can use additional corpora to improve your sense embeddings
  - SEW (Semantically Enriched Wikipedia)
  - TOM (Train-O-Matic)
  - Anything                      you                      can                      come                      up                      with
- Make sure you write how you use them in your report

Remind that the use of ***EuroSense*** is still **mandatory**

# Sense Inventory

- You **MUST** create sense embeddings only for the BabelNet synset that are in WordNet
  - All WordNet synsets have a matching BabelNet synset
- We provide you with a file **bn2wn\_mapping.txt** which contains the **mapping** from BabelNet synset ids to WordNet offset ids
- For example, each `annotation` of EuroSense refers to a **BabelNet synset**, so you have to consider a synset only if it is in the mapping

# BabelNet WordNet mapping

- In order to access the synset in WordNet you can use the nltk API

```
from nltk.corpus import wordnet as wn
```

```
offset = "14512817n"
```

```
synset = wn.synset_from_pos_and_offset( offset[-1], offset[:-1] )
```

```
synset = wn._synset_from_pos_and_offset(offset[-1], int(offset[:-1]))
```

# Word Similarity Task

- You have to **test** your sense embeddings on the [WordSimilarity-353](#) dataset (use the *combined.tab* version)
- The task consists of measuring the **similarity** or **relatedness** of pairs of words
- Word similarity datasets consists of a list of pairs of words. For each pair you get a **score** of similarity established by human annotators

Word1	Word2	Gold
-----	-----	-----
tiger	cat	7.35
book	paper	7.46
computer	keyboard	7.62

# Word Similarity Task

- In order to perform this task with sense embeddings you have to:
  - For each pair  $w_1, w_2$ 
    - $S_1$  = all sense embeddings associated with word  $w_1$
    - $S_2$  = all sense embeddings associated with word  $w_2$
    - score = - 1.0
    - For each pair  $s_1$  in  $S_1$  and  $s_2$  in  $S_2$  do
      - score = max(score,  $\cos(s_1, s_2)$ )

(**cos** == cosine similarity of two vectors)



# Word Similarity Task

- Once you calculate the score for each pair you need to check your scores against the gold ones in the dataset
- To do so, you have to calculate the **Spearman** correlation between gold similarity scores and cosine similarity scores

Word1	Word2	Gold	Cosine
-----	-----	-----	-----
tiger	cat	7.35	0.452
book	paper	7.46	0.784
computer	keyboard	7.62	0.643

`Spearman([7.35, 7.46, 7.62], [0.452, 0.784, 0.643]) = 0.5`

# Word Similarity Task

- The cosine similarity might not be the best score function so feel free to experiment with other metrics (some of them are proposed in the paper)
- If you use some interesting ones make sure you write them in your **report**
- However, we will **evaluate** your embeddings by using the ***cosine similarity***
- We will test your sense embeddings on a **secret** (again!) word similarity dataset

# Submission

- You have to submit your sense embeddings by saving them in a single file named **embeddings.vec**
- The *embeddings.vec* file **MUST** respect the word2vec format:

```
number_of_senses embedding_dimension  
lemma1_synset1 dim1 dim2 dim3 ... dimn  
lemma2_synset2 dim1 dim2 dim3 ... dimn
```

- **space** (“ ”) is the separator character
- To make sure you save it correctly try to load them using

```
from gensim.models import KeyedVectors  
model = KeyedVectors.load_word2vec_format('embeddings.vec', binary=False)
```

- File formatted in other ways will be directly **discarded**

# Submission

The project will have this **structure**:

- Your report in **pdf** format, **2 PAGES** for text and **2 PAGES** for images/tables/references, named **report.pdf**
- A folder with your source code named **code**
- A folder with the *embeddings.vec* file named **resources** and the your model weights
- You should submit **ONLY** one *embeddings.vec* file: if you make some improvements make sure you submit only the best embeddings

# Submission

- Register to [GitLab](#) ( you should have already done that :D )
- Create a new repository (**private**) with name:  
**<firstname>\_<lastname>\_<matricola>\_nlp19hw2**
- **Share** the project with us (*MAINTAINER* role):
  - use ALL the emails on the second slide and
  - [navigli@di.uniroma1.it](mailto:navigli@di.uniroma1.it)
- The link where you have to submit:

[SUBMISSION FORM](#)

# 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 [30%]
- Report [63%]
- Overall performance of the system [7%]
  - We will evaluate you on a secret word similarity test set
- **Creativity boost:** Improvements over the model [15%]
  - You will get extra points if you add some effective features to your embeddings, model and similarity measure, in this case we will expect a comparison with the different approaches and meaningful observations on what you experimented

# What we expect in the report

- **Describe** any important step you carried out in the **preprocessing** phase (we do not care about small implementation details such as “how to parse an xml”)
- Report the **model** and the parameters you used, plus any implementation choice worth mentioning
- **Describe** the features you added to improve the sense embeddings
- Report some **qualitative analysis** such as nearest neighbours evaluation for a few interesting cases
- Report some **interesting plots** such as the visualization of a sample of the embeddings with t-SNE and/or PCA

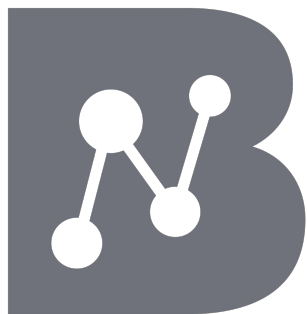
# Deadlines

- We will upload everything you need to complete this homework Sunday 5 May evening the latest, on the facebook group
- Your deadline for homework 1 will be: **31/05/2019, 23:59** anywhere on Earth
- If you push anything after the deadline you will be get **1 point less** for each day of delay



# Competition results

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



BabelNet





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 to parse the corpora!
- **Parsing** such large files could take days so make sure to perform that step ASAP
- **Memory** is also an issue: you will probably not be able to load the whole corpora into memory with standard libraries, e.g. you will have to load them as suggested

# Good luck!

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

