

NLP course 2023

# Homework 2

---

## Coarse-Grained WSD

Prof. Roberto Navigli

Teaching assistants:

Edoardo Barba, Tommaso Bonomo,  
Karim Ghonim, Giuliano Martinelli,  
Francesco Molfese, Stefano Perrella,  
Lorenzo Proietti



**SAPIENZA**  
UNIVERSITÀ DI ROMA



# Word Sense Disambiguation

An introduction

# Word Sense Disambiguation

## An Introduction

- **Word sense disambiguation (WSD)** is the “computational identification of the meaning for words in context” (Navigli, 2009)

A mouse takes much more space than a trackball.



*Any of numerous  
small rodents*



*A hand-operated  
electronic device*

*Person who is quiet  
or timid*



# Word Sense Disambiguation

## An Introduction

- WSD is usually framed as a **multiclass classification problem** where the classes belong to a specific Sense Inventory
- **WordNet**: a lexical-semantic database containing structured knowledge for the English Language

### Mouse:



- *mouse%1:05:00:: Any of numerous small rodents typically resembling diminutive rats*
- *mouse%1:18:00:: Person who is quiet or timid*
- *mouse%1:06:00:: A hand-operated electronic device that controls the coordinates of a cursor on your computer screen*

# Word Sense Disambiguation

## Task Challenges



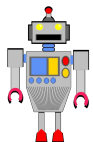
Sense distributions follow the **Zipf distribution** making classes highly unbalanced



The number of possible **senses is in the hundreds of thousands!**



The **expertise to annotate sense tagged training data** makes it difficult to scale on different domains and languages



Polysemous words have senses with **similar meaning**, making the model struggle to classify them correctly

# Word Sense Disambiguation

Even for a person, it is sometimes hard to choose between definitions that **have similar meanings**.

Let us consider the possible senses that the polysemous word *Italic* can assume:

*I prefer italic to highlight important words*

WordNet  
(Miller, 1995)

- **Italic.n.01:** a style of handwriting with the letters slanting to the right.
- **Italic.n.02:** a branch of the Indo-European languages of which Latin is the chief representative.
- **Italic.n.03:** a typeface with letters slanting upward to the right.

# Polysemy vs Homonymy

Idea: cluster the similar senses of a polysemous word to obtain a list of **highly distinguishable, coarse grained candidates**.

Two or more words are **homonyms** when they have the same lexical form but different, **unrelated** meanings.

- **Italic.n.01:** a style of handwriting with the letters slanting to the right.
- **Italic.n.03:** a typeface with letters slanting upward to the right.

Homonymy

- **Italic.n.02:** a branch of the Indo-European languages of which Latin is the chief representative.

# Coarse-Grained WSD

WSD with coarse-grained senses

Instead of using all the candidates, now we have classes that are highly distinguishable: we have clustered **candidates with similar meaning** in *homonymy clusters*.

WordNet  
(Miller, 1995)

- **Italic.n.h.01:** { a style of handwriting with the letters slanting to the right;  
a typeface with letters slanting upward to the right }.
- **Italic.n.h.02:** {a branch of the Indo-European languages of which Latin is the chief representative.}

In this way we have to disambiguate only between homonymy clusters, which is easier due to their **distant meanings**.



# Coarse-Grained WSD

WSD with coarse-grained senses

WordNet  
(Miller, 1995)

- **race.n.01:** any competition.
- **race.n.02:** a contest of speed.
- **race.n.03:** people who are believed to belong to the same genetic stock.
- **subspecies.n.01:** (biology) a taxonomic group that is a division of a species.
- **slipstream.n.01:** the flow of air that is driven backwards by an aircraft propeller.
- **raceway.n.01:** a canal for a current of water.

fine-to-coarse mapping to Homonymy Clusters

**race.n.h.01**

- **race.n.01:** any competition.
- **race.n.02:** a contest of speed.
- **slipstream.n.01:** the flow of air that is driven backwards by an aircraft propeller.
- **raceway.n.01:** a canal for a current of water.

**race.n.h.02**

- **race.n.03:** people who are believed to belong to the same genetic stock.
- **subspecies.n.01:** (biology) a taxonomic group that is a division of a species.

# Dataset

# The Dataset

## Coarse-Grained WSD

- We will use WordNet (*Miller, 1995*), the standard English source of word senses.
- In each dataset file, each sample is a tokenized sentence with information about the instances to disambiguate, such as lemmatization and part of speech tag.
- You will receive 2 dataset files:
  - A **coarse-grained** dataset, containing candidates and correct gold homonymy clusters. This is the official dataset for this homework and the submission will be evaluated on the performances on this data.
  - A **fine-grained** dataset, containing candidates and correct gold WordNet senses. This is a key resource for you to obtain **bonus** points by doing novel comparative analysis.
- You will receive an **additional file**, "*coarse\_to\_fine.json*" containing a mapping between each coarse grained candidate and its fine-grained sub-senses along with their definitions.

# The Dataset

Coarse-grained dataset (mandatory usage)

- Each sample of the coarse-grained dataset is a sentence with annotations about words and their senses:
  - **idx**: document id and sentence id
  - **instance\_ids**: mapping between token based offsets of each instance and its id
  - **words**: list of tokenized words
  - **lemmas**: list of tokenized and lemmatized words
  - **pos\_tags**: list of part of speech tags
  - **senses**: mapping between token based offsets and gold homonymy clusters
  - **candidates**: list of possible homonymy clusters for each instance

# The Dataset

Data Format: coarse-grained dataset

```
{
  "d000.s002": {
    "instance_ids": {
      "1": "d000.s002.t000"
      "5": "d000.s002.t001"
    }
    "lemmas": ["the", "race", "will", "take", "place", "today"],
    "words": ["The", "races", "will", "take", "place", "today"]
    "pos_tags": ["DT", "NOUN", "VB", "VB", "NOUN", "ADP"]
    "senses": {
      "1": "race.n.h.01"
      "5": "today.r.h.01"
    }
    "candidates": [
      "1": ["race.n.h.01", "race.n.h.02"]
      "5": ["today.r.h.01"]
    ]
  },

```

# The Dataset

Fine-grained dataset (recommended for bonus points)

- Each sample of the fine-grained dataset is a sentence with annotations about words and their senses:
  - **idx**: document id and sentence id
  - **instance\_ids**: mapping between token based offsets of each instance and its id
  - **words**: list of tokenized words
  - **lemmas**: list of tokenized and lemmatized words
  - **pos\_tags**: list of part of speech tags
  - **senses**: mapping between token based offsets and gold WordNet synsets
  - **candidates**: list of possible WordNet synsets for each instance

# The Dataset

Data Format: fine-grained dataset

```
{
  "d000.s002": {
    "instance_ids": {
      "1": "d000.s002.t000"
      "5": "d000.s002.t001"
    }
    "lemmas": ["the", "race", "will", "take", "place", "today"],
    "words": ["The", "races", "will", "take", "place", "today"]
    "pos_tags": ["DT", "NOUN", "VB", "VB", "NOUN", "ADP"]
    "senses": {
      "1": "race.n.02"
      "5": "today.r.01"
    }
    "candidates": [
      "1": ["race.n.01", "race.n.02", "race.n.03",
"subspecies.n.01", "slipstream.n.01", "raceway.n.01"]
      "5": ["today.r.01", "today.r.02"]
    ]
  },

```

# Additional data

## Coarse-to-fine mapping

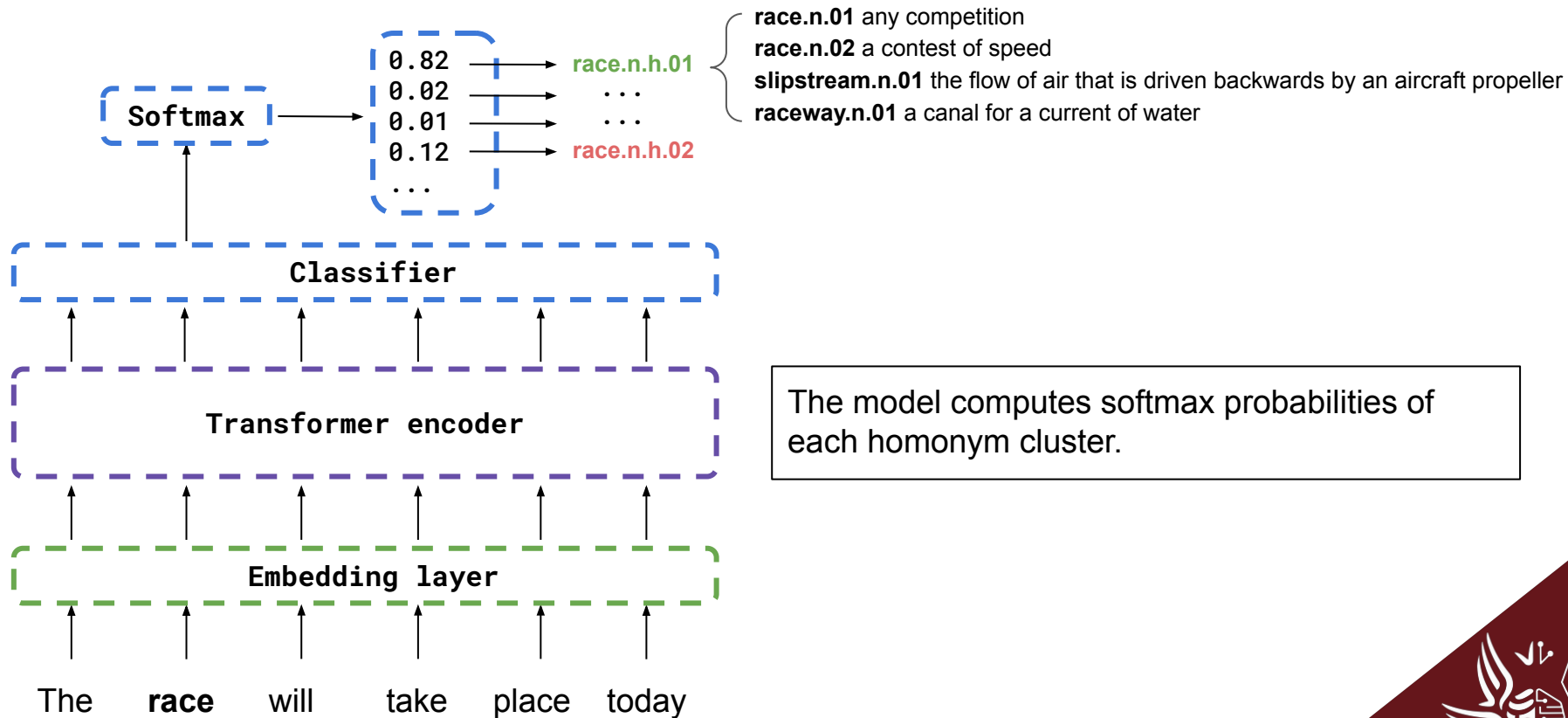
```
{ "race.n.h.01": [  
    "race.n.01" : "any competition",  
    "race.n.02" : "a contest of speed",  
    "slipstream.n.01" : "the flow of air that is driven backwards by an aircraft  
propeller",  
    "raceway.n.01" : "a canal for a current of water"  
],  
  "race.n.h.02": [  
    "race.n.03" : "people who are believed to belong to the same genetic stock",  
    "subspecies.n.01" : "(biology) a taxonomic group that is a division of a  
species"  
],  
  "today.r.h.01": [  
    "today.r.01" : "in these times"  
    ...  
  ],  
}
```



# Possible Approach



# Coarse-Grained WSD as Multiclass Token Classification



# Homework Extras

# Increase the complexity of your model!

- Take inspiration from recent papers:
  - **GlossBERT**: BERT for word sense disambiguation with gloss knowledge ( [ACL 2019](#) )
  - **EWISER**, Breaking Through the 80% Glass Ceiling: Raising the State of the Art in Word Sense Disambiguation by Incorporating Knowledge Graph Information( [ACL 2020](#) )
  - **BEM**, Moving Down the Long Tail of Word Sense Disambiguation with Gloss Informed Bi-encoders ( [ACL 2020](#) )
  - **ESC**: Redesigning WSD with Extractive Sense Comprehension( [NAACL 2021](#) )
  - **ConSeC**: Word Sense Disambiguation as Continuous Sense Comprehension ( [EMNLP2021](#) )

# Fine- vs. coarse-grained WSD

- Using the **fine-grained** version of the dataset you can train a standard WSD model:
  - **Compare the results** of your architectures on the two tasks.
  - **Apply the coarse-to-fine mapping to the output** of the fine-grained wsd model to obtain coarse-grained disambiguations.
    - Is it better than the model trained on coarse-grained data? (If motivated on your report, you can submit this model!)
  - **Use both systems:** you can use the coarse grained system to filter out senses for the fine-grained wsd.
  - Use **latent homonymy cluster embedding** to add useful information.
  - WSD coarse-grained by training a fine grained system that is rewarded positively for every synset in the correct homonym
  - Train a multiclass multilabel classifier (multilabel for each sense in a given homonymy cluster)
  - **Analyze qualitatively your results.**

# Other extras

- Use sense definitions:
  - Find a way to employ senses definitions in your pipeline (it will improve your results!)
- Find new homonyms:
  - Find ways to detect new homonyms and validate their contribution
- Test on Multilingual Dataset:
  - Building on a multilingual homonym detector, you can train a Multilingual Model on coarse grained data, and test if the model is able to generalize well in other languages.
  - Possible Multilingual resources:
    - [XL-WSD](#): An Extra-Large and Cross-Lingual Evaluation Framework for Word Sense Disambiguation.
- Extend coarse granularity to Entities:
  - Coarse Grained named entity recognition
  - Possible start of a thesis project with SapienzaNLP!

# Submission

# What you will receive

- We will provide you with a folder organized as follows (some files are omitted):

- nlp2023-hw2/
  - data/
  - hw1/
    - model.py
    - **stud/**
  - **model/**
  - **requirements.txt**
  - test.sh

- You are allowed to edit only the items in bold!



# What you will receive

- We will evaluate your work using Docker
  - You should be fine even if you don't know anything about it
- If **test.sh** runs on your side, it will run on ours as well
  - Just keep in mind: do not change any file but those we marked in bold as editable in the previous slide
- Additionally, we wrote a **README.md** to get you everything up and running
- You can find the code repository [here](#)!

# What we expect from you

- The zip folder we gave you (but populated :))
- Put your training code (if you used Colab, download the notebook .ipynb and place it) in **hw2/stud/**
- If you use any additional library, modify the **requirements.txt** file as needed (click [here](#) for info)
- Use the data (train, dev and test) in the data folder
  - use each file as defined in the **standard ML conventions** (*train for training, dev for model selection and test for final testing of the model*)

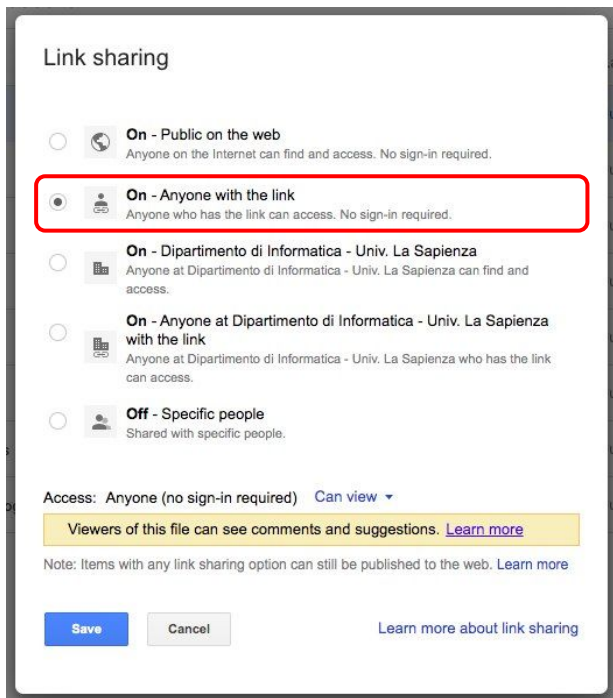
# What we expect from you

- Put everything your model needs (vocabulary, weights, ...) inside the **model/** folder, and be sure to properly load them in your model
- In **hw2/stud/implementation.py** implement the **StudentModel** class
  - Load your model and use it in the **predict** method
  - You must respect the signature of the predict method!
  - You can add other methods (i.e. the constructor)
- In **hw2/stud/implementation.py** implement the **build\_model** function
  - It should initialize your **StudentModel** class.


# What we expect from you


- Use **test.sh** to check that everything works
- Add your **report.pdf** to the folder (yes, export it in PDF even if you are using Word!)
- Name the zip folder **lastname\_studentid\_hw2.zip**:
  - Ex: Luigi D'Andrea will submit a file named **dandrea\_1234567\_hw2.zip**
  - If you are unsure which name to put, use the one in your institutional email account


# Submission Instructions





Link sharing

☐  **On - Public on the web**  
Anyone on the Internet can find and access. No sign-in required.

☒  **On - Anyone with the link**  
Anyone who has the link can access. No sign-in required.

☐  **On - Dipartimento di Informatica - Univ. La Sapienza**  
Anyone at Dipartimento di Informatica - Univ. La Sapienza can find and access.

☐  **On - Anyone at Dipartimento di Informatica - Univ. La Sapienza with the link**  
Anyone at Dipartimento di Informatica - Univ. La Sapienza who has the link can access.

☐  **Off - Specific people**  
Shared with specific people.

Access: Anyone (no sign-in required) [Can view](#) ▼

Viewers of this file can see comments and suggestions. [Learn more](#)

Note: Items with any link sharing option can still be published to the web. [Learn more](#)

[Save](#) [Cancel](#) [Learn more about link sharing](#)

- Upload the zip on your **institutional** Drive and make it **link-shareable** and **public** to anyone (an automatic script will download it).
- Make sure it is accessible via an incognito page of your browser!
- Do **NOT modify** the folder structure
- You have to submit the homework through the [submission form](#) on Google Classroom. You will be asked to fill a form with the requested information and the **link** to the zip you uploaded on Drive.

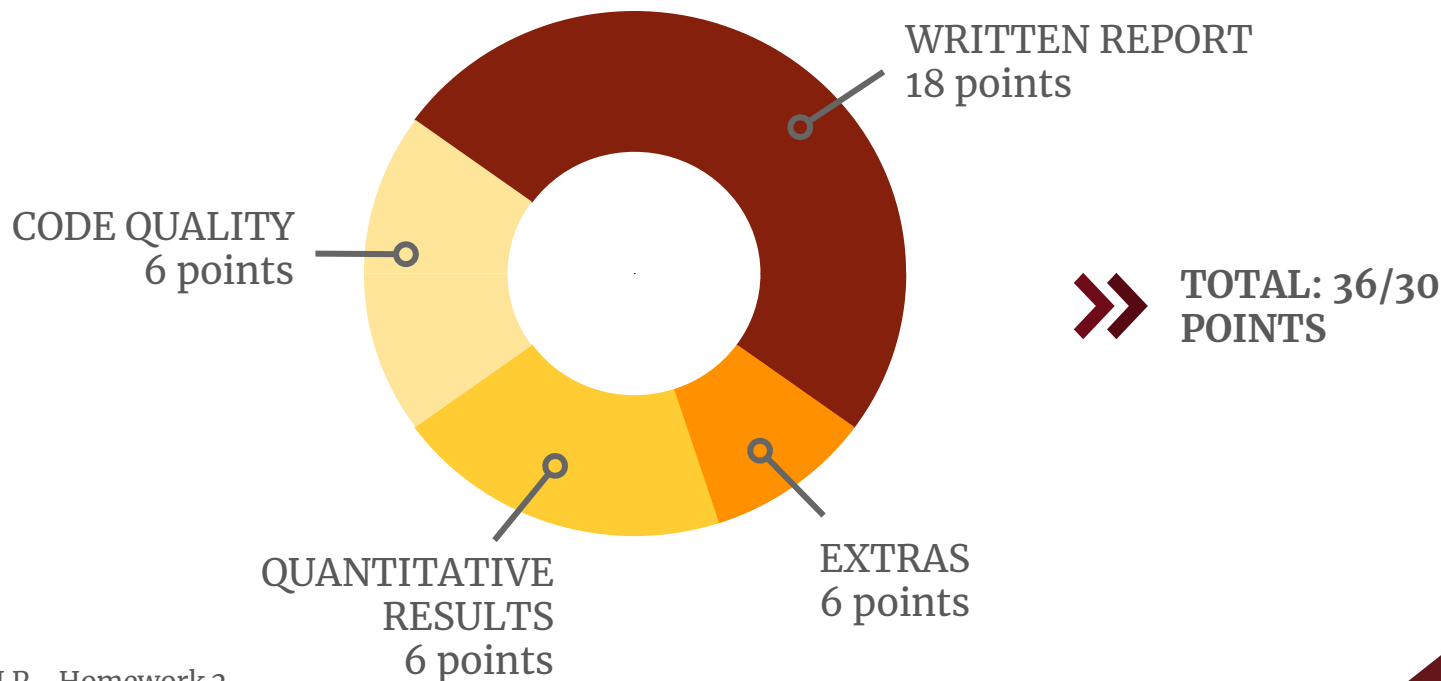
# Evaluation

# Evaluation

- Use the **validation split** to select the **best model/hyperparameters** configuration
- Use the **test split** to evaluate your model and **estimate its performance**
- The final evaluation will be conducted on a **SECRET** test set
- The evaluation metric will be the **F1-score** obtained comparing your model's predictions with our golden labels.

# Evaluation

We will take into account the following criteria:





# Report: dos and don'ts

- **ACL 2023 paper template**
  - Freely available: [LaTeX](#), [Word](#) or [Overleaf](#)
  - You can use either the LaTeX or the Word template, your choice
  - **DO NOT MODIFY** the template (margins, spacing, font size)
  - Use the non-anonymous flag, so you can enter your name
- **Max 3 pages**
  - For the report, including title, subtitles, etc.
  - This is a **STRICT RULE!**
- **Unlimited extra pages for images, tables and references**
  - Every image and table must have a caption (don't abuse them please :) )
  - Tables and images must be referenced in the report

# Report: what you are expected to do



We expect a good report to be:

- **Readable and understandable**
  - We will not give penalties for English errors, but we expect the report to follow a clear flow. We don't want to read just a sequence of statements on what you did without showing the reasoning behind your choices
- **Well-structured and organized**
  - Take inspiration from the many papers available online and organize your report in well-defined sections (e.g. method, setup, experiments, results...)

# Report: what you are not expected to do



We expect a good report **NOT** to include:

- Unnecessary **task** or **dataset descriptions**
  - just focus on your solution to the problem
- **Code** copy-paste
  - Your code should be self-explanatory, so no need to show it in the report. You can add **pseudocode** to show some particular algorithm, but **no code or screenshots**, please!

# Report: what you are not expected to do



We expect a good report **NOT** to include:

- **Unnecessary low-level implementation details**
  - Avoid any **low-level implementation/technical details** like “I used a dictionary to store these values”, “I had to use configuration X to solve this exception”, “I could not use Y because there was a dependency issue with Z”, etc.
  - Instead, **we are interested in high-level abstractions/strategies** you decide to use to tackle the homework, as well as the **intuitions behind your choices**.  
E.g. use and description of a particular model, explanation of how and why an architecture works, etc.

# Application: what you are expected to do



Your project should conform to the following rules:

- You **MUST** use PyTorch.
  - TensorFlow and other deep learning frameworks are **NOT** allowed.
  - PyTorch Lightning **is allowed** and suggested
- **Frameworks** that use PyTorch (e.g. AllenNLP, torchtext...) are allowed.
- Libraries (such as tqdm, sklearn, NLTK) are fine, but since the line between a framework and a library is sometimes blurred, please ask in the Google Classroom group before using any external library: **any other library MUST be agreed with the TAs.**

# Application: what you are not expected to do



Your project should conform to the following rules:

- **You are not allowed** to use tools/architectures that have not been explained yet in the course, in particular:
  - word embeddings (Word2Vec, GloVe, etc.) **are allowed**,
  - contextualized word embeddings (ELMo, etc.) **are allowed**,
  - Transformer-based models (BERT, BART, RoBERTa, etc.) **are allowed and suggested**.
- For any doubt, please ask the TAs on Google Classroom.
- **Comment** your code, please!

# Quantitative Results

We will evaluate the **accuracy of your model** on a SECRET test set.

You can get **from 0 to 6** points according to the following **thresholds**:

- $A < 0.70$   $\Rightarrow$  FAIL
- $0.70 < A < 0.75$   $\Rightarrow$  0
- $0.75 < A < T2$   $\Rightarrow$  1
- $T2 < A < T3$   $\Rightarrow$  2
- $T3 < A < T4$   $\Rightarrow$  3
- $T4 < A < T5$   $\Rightarrow$  4
- $T5 < A < T6$   $\Rightarrow$  5
- $A > T6$   $\Rightarrow$  6

# Quantitative Results

We will evaluate the **accuracy of your model** on a SECRET test set.

You can get **from 0 to 6** points according to the following **thresholds**:

- $A < 0.70$   $\Rightarrow$  FAIL
- $0.70 < A < 0.75$   $\Rightarrow$  0
- $0.75 < A < T2$   $\Rightarrow$  1
- $T2 < A < T3$   $\Rightarrow$  2
- $T3 < A < T4$   $\Rightarrow$  3
- $T4 < A < T5$   $\Rightarrow$  4
- $T5 < A < T6$   $\Rightarrow$  5
- $A > T6$   $\Rightarrow$  6

Thresholds will be defined  
based on an internal reference  
model and the **normalized  
distribution of YOUR scores!**



# Extras

You can achieve **up to 6 points with some extras!**

See Homework Extras section for some suggestions about what we consider an extra.

Don't forget to **explain your choices** in the report! Extras that are not explained in the report will not be considered for evaluation.

# Evaluation

- `test.sh` is identical to what we will be using
- **If it does not run on your side, we will not correct your homework**
- Note that, if you use **any kind of hard-coded paths**, this script won't work
- Use [paths relative](#) to the project root folder, e.g.:
  - **NO:** `/home/pincopallino/my_folder/model/weights.pt`
  - **OK:** `model/weights.pt`

# Warnings

Things you should be aware of



SAPIENZA  
NLP



# Please be aware that

This is an **individual exercise**! Collaboration among the students is **not** allowed.

We will check for **plagiarism** both manually and automatically.

It is **not allowed** to:

- Copy from other students.
- Share your code with other students.
- Use ChatGPT or similar systems for report writing.
- Copy from online resources (StackOverflow, GitHub, Medium, Kaggle and so on).

You are also allowed to use the **SOME** parts of the presented class notebooks. However, you **MUST** explicitly specify these parts in your code comments.

# Data policy

- For your experiments, use **ONLY** the provided and suggested data (train, dev and test) in the data folder; use each file as defined in the standard ML conventions (train for training, dev for model selection and test for testing).
- If you train it on dev or test set, it will be a **FAIL**.

# Tips



# A few tips to organize your work:

- **Start as soon as possible!**
  - Training a neural network requires time, possibly hours, depending on your hardware
- **Start small!**
  - If you don't get decent results with a very simple neural network, there is a good chance that adding other things won't make your model perform better
- **Leave some time for hyperparameter tuning!**
  - Sometimes good hyperparameter combinations can do wonders for your neural network
- **Use Google [Colab](#) (free GPUs!)**

# Deadline

When to deliver what





# Deadline

The students **who passed the first homework** may deliver the second one in one of the four available deadlines (2022):

1. Early submission: May 31st (23:59 AoE) → only this date allows late submission!  
**Late submission: June 2th (23:59 CEST)**  
Presentation: 5th June, 8.30
2. Submission: June 28th (23:59 AoE)  
Presentation: July 5th, 8.30
3. If particularly well deserved (e.g. bonus and/or involvement),  
**secret submission deadline: July 17-ish (23:59 AoE)**  
Presentation: July 24-ish, 9.00
4. Submission: September 5th (23:59 AoE)  
Presentation: September 13th, 8.30

# Deadline

The students **who passed the first homework** may deliver the second one in one of the four available deadlines (2022):

1. Early submission: May 31st (23:59 AoE) → only this date allows late submission!  
**Late submission: June 2th (23:59 CEST)**  
Presentation: 5th June, 8.30 (up to 12 minutes)
2. Submission: June 28th (23:59 AoE)  
Presentation: July 5th, 8.30(up to 12 minutes)
3. If particularly well deserved (e.g. bonus and/or involvement),  
**secret submission deadline: July 17-ish (23:59 AoE)**  
Presentation: July 24-ish, 9.00(up to 12 minutes)
4. Submission: September 5th (23:59 AoE)  
Presentation: September 13th, 8.30(up to 12 minutes)

# Awards

Get a Sapienza NLP™ t-shirt



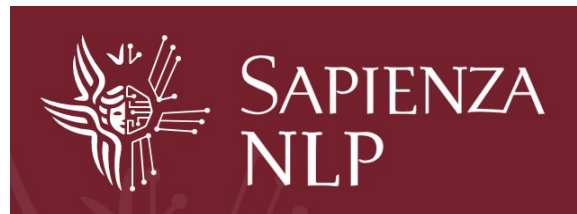
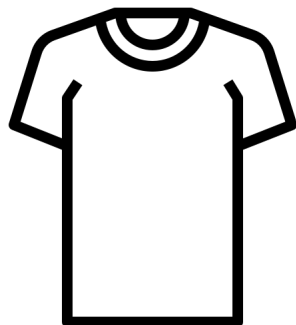
SAPIENZA  
NLP



# Win a Sapienza NLP t-shirt!

We will hand out amazing Sapienza NLP t-shirts to the **overall top-5** students!

The final ranking will be computed according to the scores on our **secret** test set.



# That's not all

If your work is novel, interesting and original, we will gladly invite you to work together with us to extend on a fully-fledged paper for **TOP-TIER INTERNATIONAL CONFERENCE!**

Just over the last 12 months, the Sapienza NLP group published more than a dozen of papers!

# Questions?

If you have a question that may interest your colleagues, **please ask it on Google Classroom.**

Otherwise, for personal or other questions, email **ALL** of us (but please, only reach for things that can't be asked on the Google Classroom).

Our emails are:

{bonomo, ghonim, martinelli, molfese, perrella, lproietti}@diag.uniroma1.it

# Bonus 1: Language Recognition in Low-Resource Languages

- Low-resource languages:
  - Kazakh (Cyrillic script)
  - Khmer
  - Georgian
  - **Burmese** (Unicode encoding)
  - Albanian
  - Azerbaijani (Latin script)
  - Mongolian (Cyrillic script)
  - Armenian
  - Lao
  - Uzbek (Latin script)
  - Sinhala
  - Tibetan
  - Sorani Kurdish (Arabic script)
  - Swahili
  - Belarusian (Cyrillic script)
  - Uzbek (Cyrillic script)
  - Malayalam
  - Nepali
  - Kurmanji Kurdish (Latin script)
  - **Welsh**
  - Pashto (Arabic script)
  - Kyrgyz (Cyrillic script)
  - **Kashmiri (Arabic script)**
  - **Māori**
  - **Tajik (Cyrillic script)**
  - **Maltese**
  - **Faroese**
  - **Dhivehi (Thaana script)**
  - **Turkmen (Latin script)**
  - **Malay (Arabic script)**
- Babelscape tshirt + 500€ prize from Babelscape

# How to work on the Bonus Exercise

- **Step 1: collect text corpora; ideas:**
  - Wikipedia ([dumps.wikipedia.org](https://dumps.wikipedia.org/))
  - Universal Dependencies: <https://universaldependencies.org/>
  - CommonCrawl? <https://commoncrawl.org/>
  - Oscar corpus
  - Other online corpora
  - Ask ChatGPT (“What corpora are available in Welsh?”), You.com, Bing AI, Duckduckgo.com, etc.
  - See if any of those languages are translated / recognized by Google Translate or any other API
  - From manual collection to fully automatic crawling (but declare it)
  - Provide information about the sources and the format
- **Step 2: build a language recognition model as you like**



# Evaluation

- Bolded languages mandatory
- How to participate:
  - Write an email with subject “Bonus 1 NLP 2023” to [passarelli@babelscape.com](mailto:passarelli@babelscape.com), [cecconi@babelscape.com](mailto:cecconi@babelscape.com) and [navigli@diag.uniroma1.it](mailto:navigli@diag.uniroma1.it)
  - We will send instructions on how to test and submit
- How to obtain the 500€ money prize from Babelscape:
  - At least 15 low-resource languages (see list before) included (10 bold + 5 extra of your choice)
  - + all the 21 EU languages in Europarl (<https://www.statmt.org/europarl/>)
  - Language recognition  $\geq 95\%$  at sentence level on your test set and on our secret test set
  - First come first served (only one prize will be awarded):
    - First window from today till the September exam date (same submission date)
    - After that, the first who surpasses all thresholds
- Teamwork is allowed (but if 2 are involved, 10 bold+15 extra languages)