NLP course 2023

Homework 3

Relation Extraction

Prof. Roberto Navigli

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

SAPIENZA
UNIVERSITÀ DI ROMA

SAPIENZA NLP

Relation Extraction

An introduction

Sapienza NLP

What is a Relation?

- A **relation** can be defined as a binary or n-ary association between two or more entities.
- Relations are represented as tuples consisting of a relation type and the entities that participate in the relationship.
- For example, the relation "works at" might involve a person (subject of the relation) and a company (object of the relation).

Joe	works at	Apple
subject	relation type	object

Relation Extraction

- **Relation Extraction** is a Natural Language Processing task that involves identifying and extracting relationships between entities mentioned in text.

- **Goal:** extract structured information from unstructured data by identifying the type of relationship between entities, such as "born in", "works at", "married to", etc.

- We limit our case study only to **binary relations**.

Relation Extraction

Example

"This Must Be the Place" is a song by new wave band Talking Heads, released in November 1983 as the second single from its fifth album "Speaking in Tongues".

Identify entities that contain relations

"This Must Be the Place" is a song by new wave band Talking Heads, released in November 1983 as the second single from its fifth album "Speaking in Tongues".

Extract triplets made of (subject entity, relation type, object entity)

(This Must Be the Place, performer, Talking Heads)
(Talking Heads, genre, new wave)
(This Must Be the Place, part of, Speaking in Tongues)
(Speaking in Tongues, performer, Talking Heads)

Dataset

SAPIENZA NLP

The Dataset

- Each sample is a sentence annotated with **relation tuples** (subjects, relation types and objects) that appear in the sentence.
- There may be **multiple relation tuples** in a given sentence.
- Data splits:
  - **56196** training examples (`train.jsonl`)
  - **5000** development examples (`dev.jsonl`)
  - **2000** test examples (`test.jsonl`)

Dataset format

- JSONL, i.e. each line is a JSON object with two fields:
  - **tokens**: the pre-tokenized sentence, i.e. a list of tokens that make up the sentence.
  - **relations**: a list of relation tuples. Each element of the list is a relation tuple made of a dictionary with 3 keys:
    - subject: containing start and end token indexes, entity type and lexical representation.
    - relation: e.g. "/location/country/capital"
    - object: containing start and end token indexes, entity type and lexical representation.

Sapienza NLP

Dataset entry


```
{
	“tokens”: [“The”, “2023”, “chess”, “tournament”, “was”, “held”, "in", "Reykjavik", ",", "Iceland", "."],
	“relations”:
	[
	{
		"subject": {
			"start_idx": 9,
			"end_idx": 10,
			"entity_type": "LOCATION",
			"text": "Iceland"
		},
		"relation": "/location/country/capital",
		"object": {
			"start_idx": 7,
			"end_idx": 8,
			"entity_type": "LOCATION",
			"text": "Reykjavik"
		}
	}
	]
}
```

Modelling approaches

Two-phase pipeline

Simple approach, requires some data-handling skills.

Phase 1 - Entity recognition:

Predict entities - we don't need their category and they might not be named.

- **Training:** similar to NER notebook, dataset annotations could be converted to IOB labels, ...
- **Inference:** collect all predicted entities and build input for next phase, i.e. all possible combinations of entities.

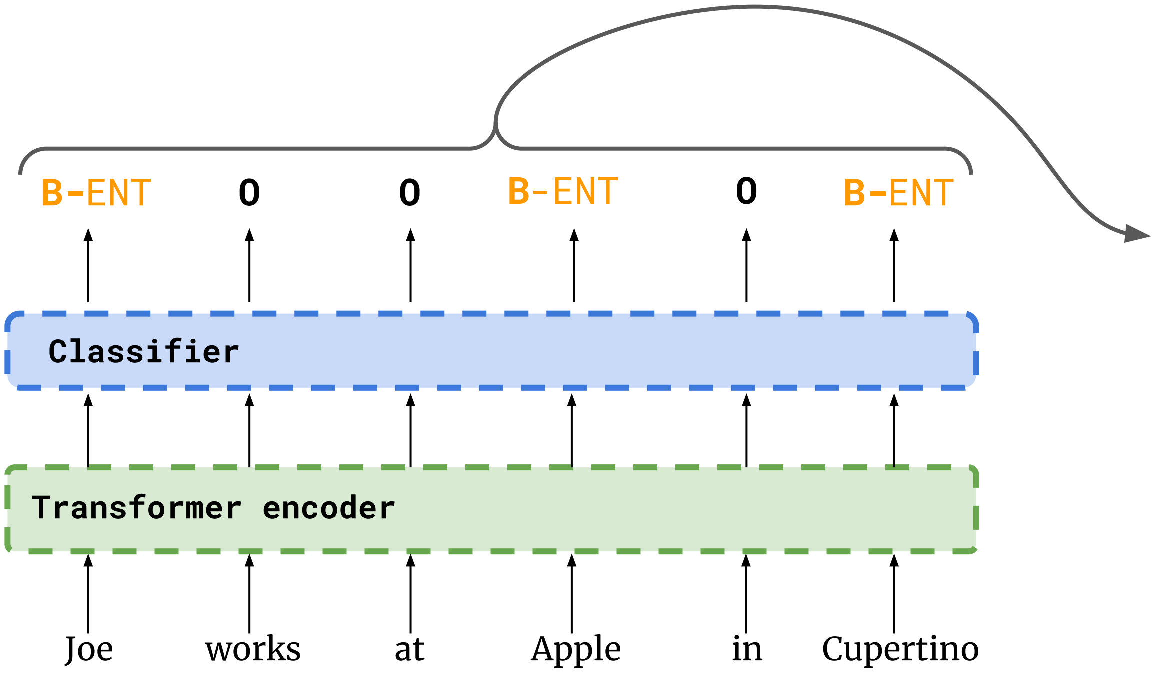
Phase 2 -Relation classification:

Given a sentence, a subject entity and an object entity, classify their relation.

You **should indicate** to the model **which tokens are the subject and the object**.

- **Training:** convert dataset to have one relation tuple per sample, find a way to identify subject and object
- **Inference:** simply predict the relation for a given sentence, subject and object.

Two-phase pipeline
Phase 1 - Entity recognition



Relation tuple candidates:

- Joe | Apple
- Joe | Cupertino
- Apple | Joe
- Apple | Cupertino
- Cupertino | Joe
- Cupertino | Apple

Two-phase pipeline
Phase 2 - Relation classification

Relation tuple candidates:
- Joe | Apple
- Joe | Cupertino
- Apple | Joe
- Apple | Cupertino
- Cupertino | Joe
- Cupertino | Apple

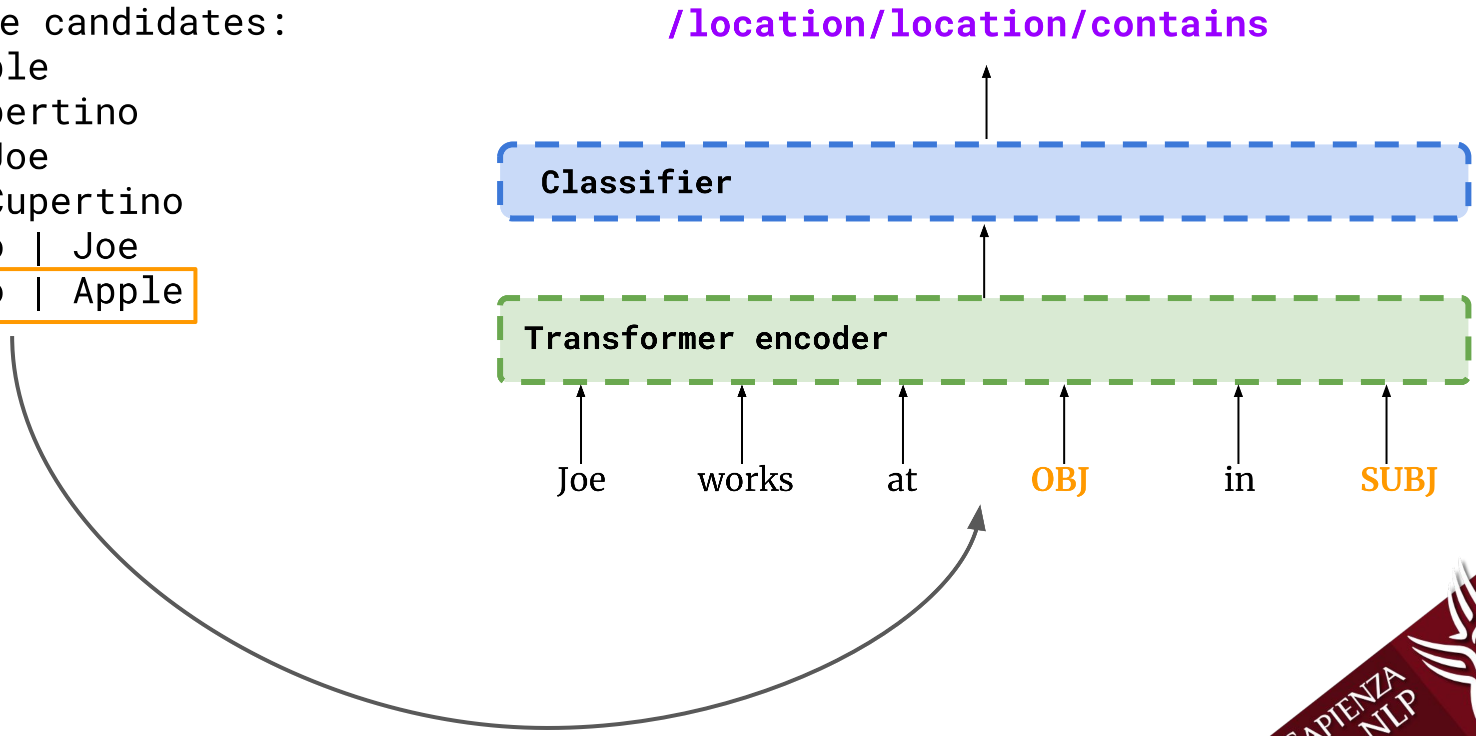


Table-filling end-to-end

More complicated approach, must manually implement some steps in the model.

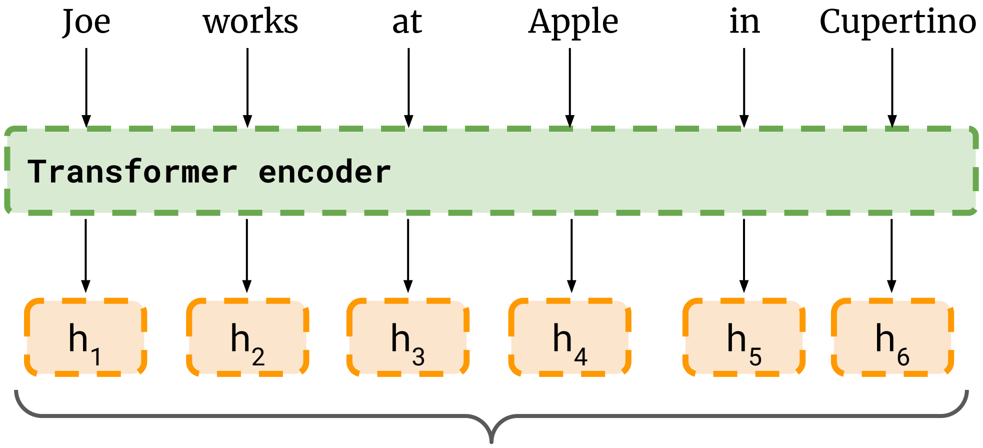
Main idea: build a matrix A having as rows and columns the sentence tokens.
For two given tokens i and j, $i \neq j$, $A[i, j]$ defines the relation with token i as subject and token j as object (can also be *no_relation*)

Steps to take:

1. Convert samples from the dataset into this matrix representation;
2. Find a way to derive the matrix through the model:
   a. element-wise product and projection layer;
   b. dot product and convolutional layer;
   c. using attention matrices computed by the transformer;
   d. ...
3. Measure the loss comparing the modelled matrix with the one you converted from the dataset

SAPIENZA NLP

Table-filling end-to-end
Schematic example



Build matrix **A** with dimensions
(6, 6, num_relations)

Table-filling end-to-end

Matrix example

Object

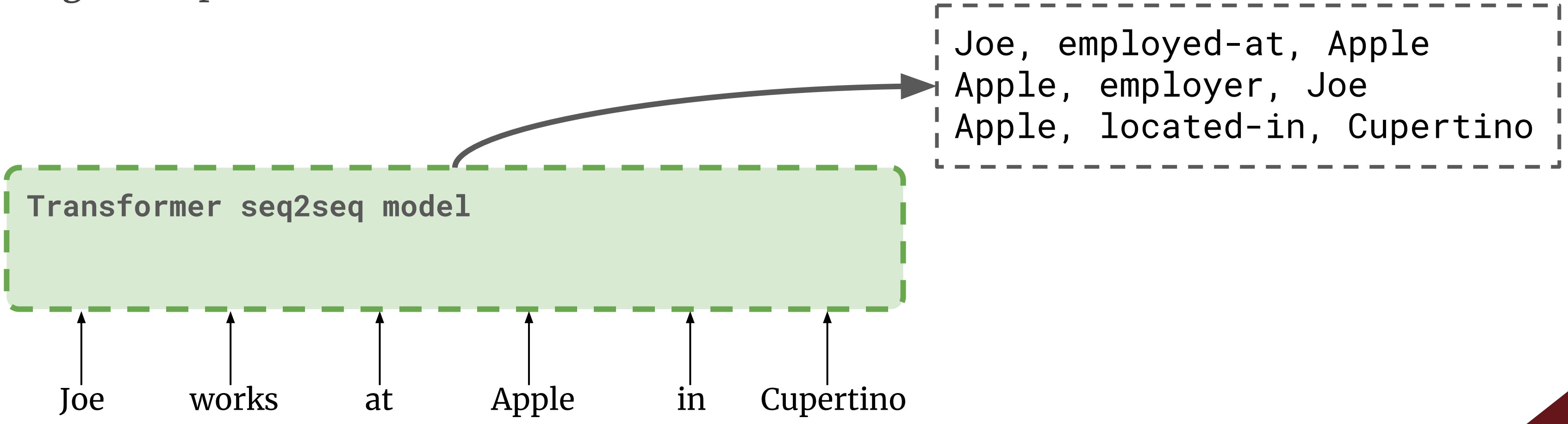
Subject

	h_1: Joe	...	h_4: Apple	h_5: in	h_6: Cupertino
h_1		...	**employed-at**	no_relation	no_relation
...	...		...	...	...
h_4	**employer**	...		no_relation	**located-in**
h_5	no_relation	...	no_relation		no_relation
h_6	no_relation	...	no_relation	no_relation	

Generative approach

Used in current state-of-the-art approaches, technically difficult.

Idea: train seq2seq model to generate one or more relation tuples (subject, relation, object) for a given input sentence.



Relevant Literature

- Pere-Lluís Huguet Cabot and Roberto Navigli. 2021.
  REBEL: Relation Extraction By End-to-end Language generation.
  In *Findings of the Association for Computational Linguistics: EMNLP 2021.*

- Jue Wang and Wei Lu. 2020.
  Two are Better than One: Joint Entity and Relation Extraction with Table-Sequence Encoders.
  In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP).*

SAPIENZA NLP

Submission

SAPIENZA NLP

What you will receive

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

    - `nlp2023-hw3/`
        - `data/`
        - `hw3/`
            - `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!

SAPIENZA NLP

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 **`hw3/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*)

SAPIENZA NLP

What we expect from you

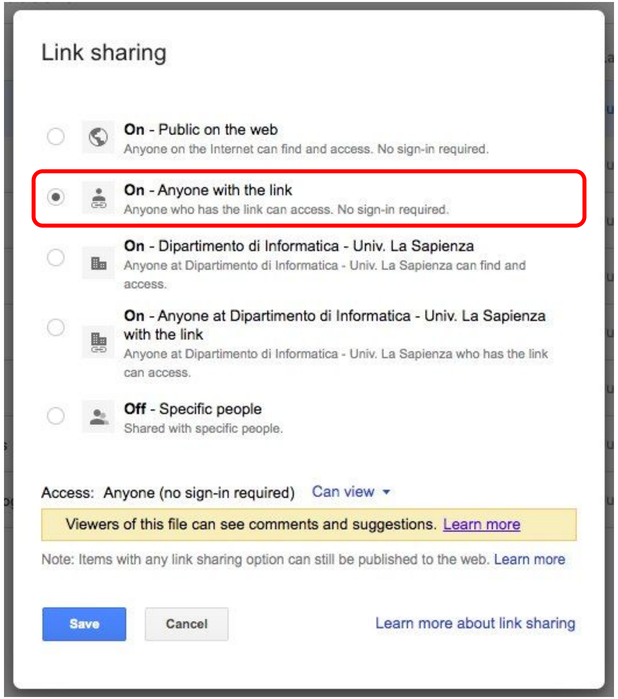
- Put **<u>everything</u>** your model needs (vocabulary, weights, ...) inside the **`model/`** folder, and **<u>be sure to properly load them</u>** in your model
- In **`hw3/stud/implementation.py`** implement the **`StudentModel`** class
  - Load your model and use it in the **`predict`** method
  - You **<u>must respect</u>** the signature of the predict method!
  - You can add other methods (i.e. the constructor)
- In **`hw3/stud/implementation.py`** implement the **`build_model`** function
  - It should initialize your **`StudentModel`** class.

SAPIENZA NLP

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_hw3.zip`**:
  - Ex: Luigi D'Andrea will submit a file named **`dandrea_1234567_hw3.zip`**
  - If you are unsure which name to put, use the one in your institutional email account

Submission Instructions



- 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.

SAPIENZA NLP

Evaluation

Sapienza NLP

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 **micro F1-score** obtained comparing your model's predictions with our golden labels
- **Strict evaluation**: a relation is considered correct only if the **subject and object entity spans are correctly extracted** and the **relation type is correctly classified** (i.e., fully overlap with the annotation).

SAPIENZA NLP

Evaluation

Micro-F1 explanation

The **micro F1-score** is defined as:

$$F_1 = 2 \cdot \frac{\text{recall} \times \text{precision}}{\text{recall} + \text{precision}}$$

$$\text{precision} = \frac{\text{“number of predictions that overlap with the gold annotations”}}{\text{“number of predictions”}}$$

$$\text{recall} = \frac{\text{“number of predictions that overlap with the gold annotations”}}{\text{“number of gold annotations”}}$$

Evaluation

Micro-F1 example

```
{
    “tokens”: [“Joe”, “works”, “at”, “Apple”, “in”, “Cupertino”],
    “relations”: [
    {"subject": {"start_idx": 0, "end_idx": 1, "entity_type": "PER", "text": "Joe"},
     "relation": "employed-at",
     “object": {"start_idx": 3, "end_idx": 4, "entity_type": "ORG", "text": "Apple"}},
    {"subject": {"start_idx": 3, "end_idx": 4, "entity_type": "ORG", "text": "Apple"},
     "relation": "employer",
     “object": {"start_idx": 0, "end_idx": 1, "entity_type": "PER", "text": "Joe"}},
    {"subject": {"start_idx": 3, "end_idx": 4, "entity_type": "ORG", "text": "Apple"},
     "relation": "located-in",
     “object": {"start_idx": 5, "end_idx": 6, "entity_type": "LOC", "text": "Cupertino"}}],
    “predictions”: [
    {"subject": {"start_idx": 0, "end_idx": 1, "entity_type": "PER", "text": "Joe"},
     "relation": "employed-at",
     “object": {"start_idx": 2, "end_idx": 4, "entity_type": "ORG", "text": "at Apple"}},
    {"subject": {"start_idx": 3, "end_idx": 4, "entity_type": "ORG", "text": "Apple"},
     "relation": "located-in",
     “object": {"start_idx": 5, "end_idx": 6, "entity_type": "LOC", "text": "Cupertino"}}],
}
```

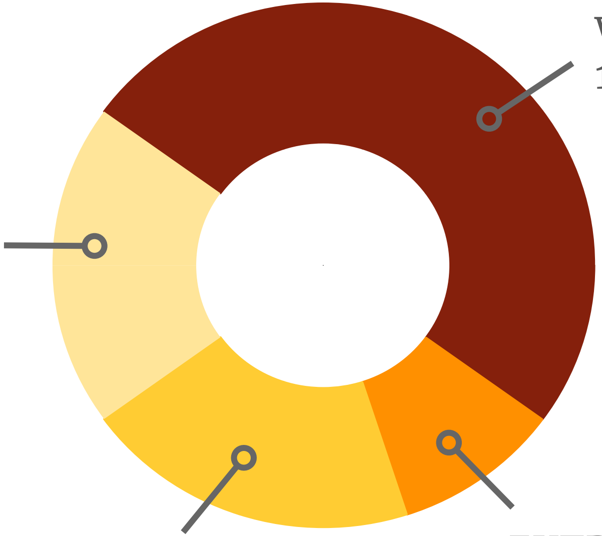
Number of predictions that overlap with gold annotations: 1

Precision: ½ = 0.5 Recall: ⅓ ≈ 0.33 F1: 0.40

SAPIENZA NLP

Evaluation

We will take into account the following criteria:



» **TOTAL: 35/30 POINTS**

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 2 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

SAPIENZA NLP

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...)

SAPIENZA NLP

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.

SAPIENZA NLP

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.
  - HuggingFace Transformers is allowed and suggested.

- 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 and framework **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 **performance of your model** on a SECRET test set.

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

- `F1 < 0.25             => FAIL`
- `0.25 < F1 < 0.40      => 0`
- `0.40 < F1 < T2        => 1`
- `T2 < F1 < T3          => 2`
- `T3 < F1 < T4          => 3`
- `T4 < F1 < T5          => 4`
- `T5 < F1 < T6          => 5`
- `F1 > T6               => 6`

Quantitative Results

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

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

- `F1 < 0.25          => FAIL`
- `0.25 < F1 < 0.40   => 0`
- `0.40 < F1 < T2     => 1`
- `T2 < F1 < T3       => 2`
- `T3 < F1 < T4       => 3`
- `T4 < F1 < T5       => 4`
- `T5 < F1 < T6       => 5`
- `F1 > T6            => 6`

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

Extras

You can achieve **up to 5 points with some extras!**

An “extra” is whatever you decide to add to your model to make it better. For instance:

- Comparative analysis of results from different approaches
- Informative plots in your report
- **Generative approaches** (e.g. using seq2seq models like BART)
- **New ideas** (including using other approaches in a clever way)

and more, according to internal baselines. 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`

SAPIENZA NLP

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 <u>ONLY</u> the provided 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

SAPIENZA NLP

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!)

SAPIENZA NLP

Deadline

When to deliver what

Sapienza NLP

Deadline

The students **who passed the first homework** may deliver the third one in one of the four available deadlines (2023):

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)

SAPIENZA NLP

Awards

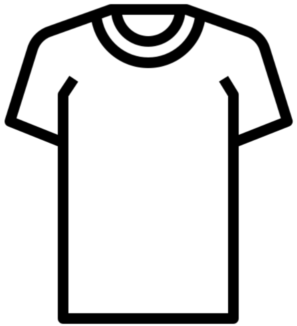
Get a **Sapienza NLP™** t-shirt

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

SAPIENZA NLP

Good Luck!!

Sapienza NLP