NLP 2025
Homework 2

Slides provided by:
Roberto Navigli
Luca Gioffré
Lu Xu
Luca Moroni
Alberte Fernandez Castro
Bruno Gatti

Homework 2

You can choose your next homework(s)* among these tasks:

- **Automatic Translation**: Transform non-modern Italian into modern Italian
- **Text Cleaning:** Transform OCRed text into clean text
- **Culture-oriented Evaluation:** Culture dataset synthesis and LLM evaluation on culture specificity
- **Sentence Splitting:** State-of-the-art sentence splitting with encoders vs. decoders

* Attending students (i.e. those who delivered and passed homework 1). Note that non-attending students (i.e. those who didn't deliver homework 1) will have to deliver 3 homeworks, so they will have to choose two homeworks from the above list. **The only pair that is not allowed is Automatic Translation and Text Cleaning.**

Exam dates

Exam (3 Homeworks) | 2 Homeworks:

- **June 3rd** (only for students who need to finalize their grades by the July graduation session (i.e. June 20th), submission by **May 27th**)
- **June 17-19th** (submission by **June 8th**)
- **July 4th** (submission by **June 23th**)
- **September 18th** (submission by **September 8th**)

Exam only:

- **January 2026**
- **February 2026**
- **+ extraordinary sessions**

Background

To carry out most of the following homeworks, you will have to know some additional concepts, which we will present in these slides.

Every homework will have listed the **minimal requirements** and the suggested approaches, but you can be creative and come up with new methods as well.

Background: LLM-as-a-Judge

LLM-as-a-Judge is a relatively recent paradigm in which LLMs are prompted to evaluate a task's completion instead of executing the task

- from "*Reply to this answer*" to "*How would you rate the correctness of this answer?*"

You can either use general purpose closed models (ChatGPT, Claude, Gemini, ...) or smaller open models (Llama-3.1) or even models fine tuned specifically for this task (Prometheus)

Gemini

Claude

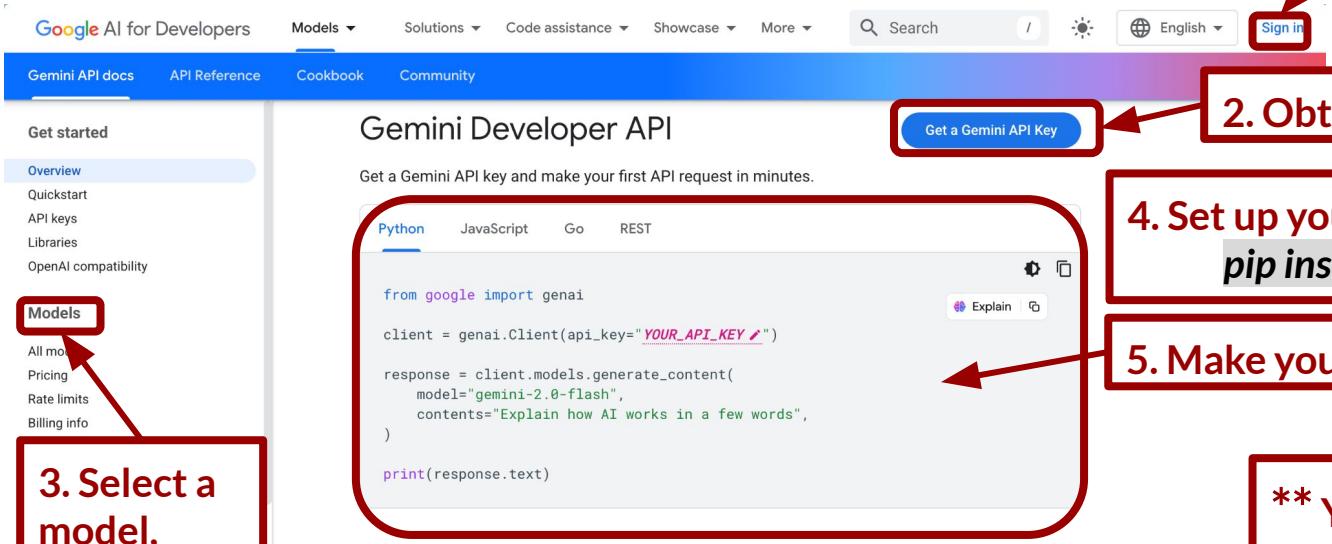
Using closed-source LLMs through API

- **Create an account on the platform you want to use**
- **Obtain an API key and *store it safely***
- **Select a model**
  - Be careful about price and rate limits
  - Remember: Newer or more expensive models aren't always necessary for your task
- **Set up your development environment**
  - see the documentation of the chosen API; each platform provides you minimal code ready to be run
- **Write your first API call and get responses!!!**

Gemini

Claude

E.g. Gemini



1. Sign in

2. Obtain an API key

3. Select a model, check price and limits

4. Set up your environment
pip install -q -U google-genai

5. Make your first API call

** You can find all the information you need in here https://ai.google.dev/gemini-api/docs

* There is FREE quota of Germini (*with rate limits*)

Free Tier | Tier 1 | Tier 2 | Tier 3

Model	RPM	TPM	RPD
Gemini 2.5 Flash Preview 05-20	10	250,000	500
Gemini 2.5 Flash Preview TTS	3	10,000	15

SAPIENZA NLP

Where to find information of APIs?

You can find all the information of the API from its official document

- Gemini
  - https://ai.google.dev/gemini-api/docs
- GPT
  - https://platform.openai.com/docs/overview
- Claude
  - https://docs.anthropic.com/en/home
- Qwen
  - https://www.alibabacloud.com/help/en/model-studio/first-api-call-to-qwen
- ...

We recommend you to use one of the Gemini versions (free daily requests)

Using locally deployed open-source LLMs

To run an LLM-as-a-Judge locally, you can either choose a general purpose model or a fine tuned one (of course, the latter should be better)

All Instruction fine tuned models can be potentially used for this task (e.g., Llama-3.1-xB-Instruct).

You may also use models specifically fine tuned for the task of **LLM-as-a-Judge.** Prometheus is a very flexible one; you can give it your personal criteria and it will score the answers using those.

How will you use the LLMs?

The Prompt of the model

###Task Description:
An instruction (might include an Input inside it), two responses to evaluate (denoted as Response A and Response B), a reference answer, and an evaluation criteria are given.
1. Write a detailed feedback that assess the quality of the two responses strictly based on the given evaluation criteria, not evaluating in general.
2. Make comparisons between Response A, Response B, and the Reference Answer. Instead of examining Response A and Response B separately, go straight to the point and mention about the commonalities and differences between them.
3. After writing the feedback, indicate the better response, either "A" or "B".
4. The output format should look as follows: "Feedback: (write a feedback for criteria) [RESULT] (Either "A" or "B")"
5. Please do not generate any other opening, closing, and explanations.

###The instruction to evaluate:
{orig_instruction}

###Response to evaluate:
{orig_response}

###Reference Answer (Score 5):
{orig_reference_answer}

###Score Rubrics:
[{orig_criteria}]
Score 1: {orig_score1_description}
Score 2: {orig_score2_description}
Score 3: {orig_score3_description}
Score 4: {orig_score4_description}
Score 5: {orig_score5_description}

###Feedback:

How will you use the LLMs?

Script and example data

The model is trained on annotated question-answer pairs, where each question is associated with handwritten **rubric scoring.**

Rubic Scoring is a list of descriptions one for each Score 1 to 5.

Look at the data https://github.com/prometheus-eval/prometheus-eval:

- prometheus-eval/Feedback-Collection

The model can be simply prompted through a Pyhton package and can run also on CPU (we recommend to use COLAB).

There is also a multilingual version of Prometheus which includes Italian
→ Unbabel/M-Prometheus-7B

SAPIENZA NLP

How will you use the LLMs?

Code Example

```
# Absolute Grading: Outputs score of 1 to 5

from prometheus_eval.vllm import VLLM
from prometheus_eval import PrometheusEval
from prometheus_eval.prompts import ABSOLUTE_PROMPT, SCORE_RUBRIC_TEMPLATE

model = vLLM(model="prometheus-eval/prometheus-7b-v2.0")
judge = PrometheusEval(model=model, absolute_grade_template=ABSOLUTE_PROMPT)

instruction = "Struggling with a recent break-up, a person opens up about the intense feelings of
loneliness and sadness. They ask for advice on how to cope with the heartbreak and move forward
in life.",

response = "I'm genuinely sorry to hear about…", # Model Response
reference_answer = "I can only imagine how” # Gold reference answer

rubric_data = {
  "criteria":"Is the model proficient in applying empathy and emotional intelligence to its responses
when the user conveys emotions or faces challenging circumstances?",
  "score1_description":"The model neglects to …",
  "score2_description":"The model intermittently ...",
  "score3_description":"The model typically identifies…",
  "score4_description":"The model consistently …",
  "score5_description":"The model excels in …."
} # Rubric Descriptions

score_rubric = SCORE_RUBRIC_TEMPLATE.format(**rubric_data)
….
```

SAPIENZA NLP

Background: Human-Metrics Correlation

To assess whether your automatic metric *agrees* with the human judgement you need to compute the correlation between the two scores.

You will need a subset of annotated samples (i.e., samples annotated* by you) AND the scores of the metric on that annotated samples.

You can do this with binary metrics (i.e., binary classification) but also with Likert-style judgments from an LLM.

Obviously, if you use LLM-as-a-Judge as the metric, the human annotator shall use the same criteria and guidelines given to the LLM.

*annotated = classified for classification or scored for generation

Background: Human-Metrics Correlation

The easiest score to compute is the accuracy of the metrics score against your scores (i.e., you use your scores as the true labels and the metrics score as the pred labels):

- 1: the sample is evaluated in the same way by your manual annotation and by the automated metric
- 0: otherwise

```
from sklearn.metrics import cohen_kappa_score
y1 = ["negative", "positive", "negative", "neutral", "positive"]
y2 = ["negative", "positive", "negative", "neutral", "negative"]
cohen_kappa_score(y1, y2)
```

However, accuracy is ill-suited, especially for imbalanced datasets

The most used pairwise correlation metric is Cohen's Kappa Coefficient.
This is a better approximation as it accounts for the agreement by chance and the dataset imbalance

- `sklearn` contains an implementation of it and many others

Background: In-Context Learning

In-Context Learning is the ability of LLMs to learn from examples provided in the prompt, *without updating model weights*.

- The model is given a few-shot prompt (e.g., input-output pairs).
- It generalizes patterns from these examples to generate appropriate responses.

As there is no fine-tuning required, this prompting technique is extremely flexible across tasks (translation, classification, etc.).

However, it is also sensitive to prompt format and order (can be a hit or miss) and may be less effective for complex or domain-specific tasks with respect to fine tuning

Background: In-Context Learning

1. **Clearly define what you want the model to do**
2. **Gather Example Pairs**: prepare 3–5 clear examples relevant to your task.
3. **Format the Prompt Consistently**

   → use a pattern like: `"Input: [text]\nOutput: [label]"`
4. **Repeat for all examples, then add your final test input**.

   End with a new input to get the model's prediction:

   `"Input: I hate waiting.\nOutput: "`
5. **Evaluate and Iterate**: Adjust wording, number, and examples to improve results.

You can put all these information in the (user) prompt, but it is better to use the different prompts available (system, user and assistant)

Background: In-Context Learning

You can do this using **Conversation Turns**

- **System prompt:** sets the tone and role of the LLM
- **User prompt**: Contains the task instruction, example inputs, and the query you want answered.
- **Assistant prompt:** simulates past model outputs (like in a dialogue), you can include assistant responses after each user input to help the model understand the task

```
[
  {
    "role": "system",
    "content": "You are an AI that classifies the sentiment of text as Positive, Negative
  },
  {
    "role": "user",
    "content": "Input: I love this place!"
  },
  {
    "role": "assistant",
    "content": "Output: Positive"
  },
  {
    "role": "user",
    "content": "Input: This is awful."
  },
  {
    "role": "assistant",
    "content": "Output: Negative"
  },
  {
    "role": "user",
    "content": "Input: I'm not sure how I feel about this."
  }
]
```

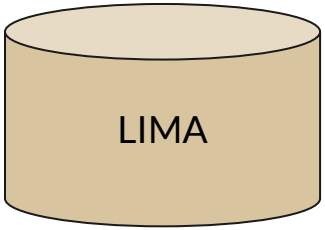
Background: using CINECA

CINECA

We'll give this opportunity to the groups that we think submitted the best works. **That does not necessarily mean that these groups are the ones with the best scores on the test set.**

Make a meaningful fine tuning of a large language model in a real large computing environment

You can use CINECA for any of the homeworks!



Lima ("Less is more for Alignment") is a dataset of 1k instruction-response pairs used for Instruction Tuning.

You can fine tune Minerva on this dataset.

SAPIENZA NLP

Background: using CINECA

Groups having access to CINECA will be required to perform a fine tuning step on it in their homework – this is a good opportunity to get your hands dirty!

- You will have more information on accessing it in the very next days.

However, you can fine tune small models even without CINECA (e.g., use quantization)

- Is it possible to fine tune Minerva 350M on COLAB and run inference on other models
  Kaggle offers free weekly TPU hours after registration

Track 1: Ancient to Modern Italian Automatic Translation

Sapienza NLP

From Ancient to modern Italian

You will need to automatically translate sentences from archaic to modern Italian

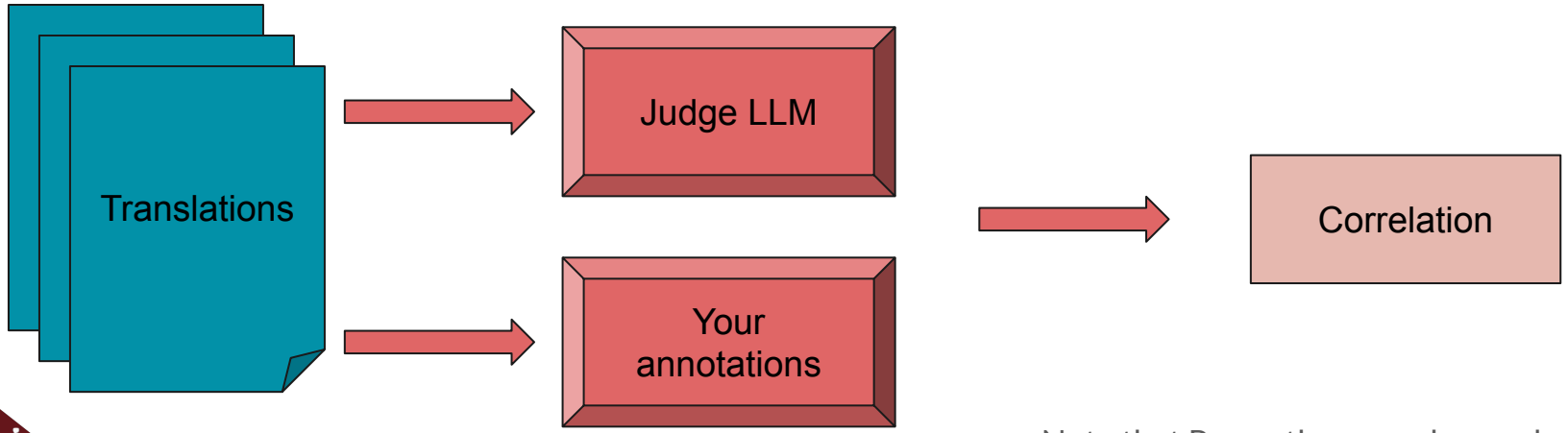
- Try **at least 2** solutions such as LLMs and Transformer-based Machine Translation Systems (see next week's lecture)
- For LLMs, you can try also In-Context Learning
    - As always, be creative!

To evaluate the quality of the translation, you will need to

- Use an LLM in the LLM-as-a-Judge paradigm
- Carry out a qualitative (and comparative) analysis of the different translation systems studying the correlation between the LLM and human judgments

How will you evaluate the translation?

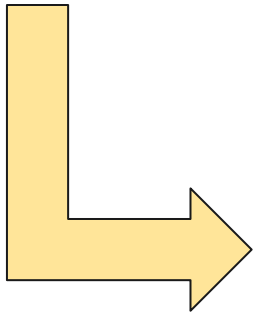
You can prompt an LLM to make a judgment in several ways; in this homework, you will focus on a Likert-scale prompting style (similar to questionnaires)



Note that Prometheus can be used only when you have a gold reference (which is missing for this dataset)
You may still investigate how to use it as an extra work

Machine translation from old Italian

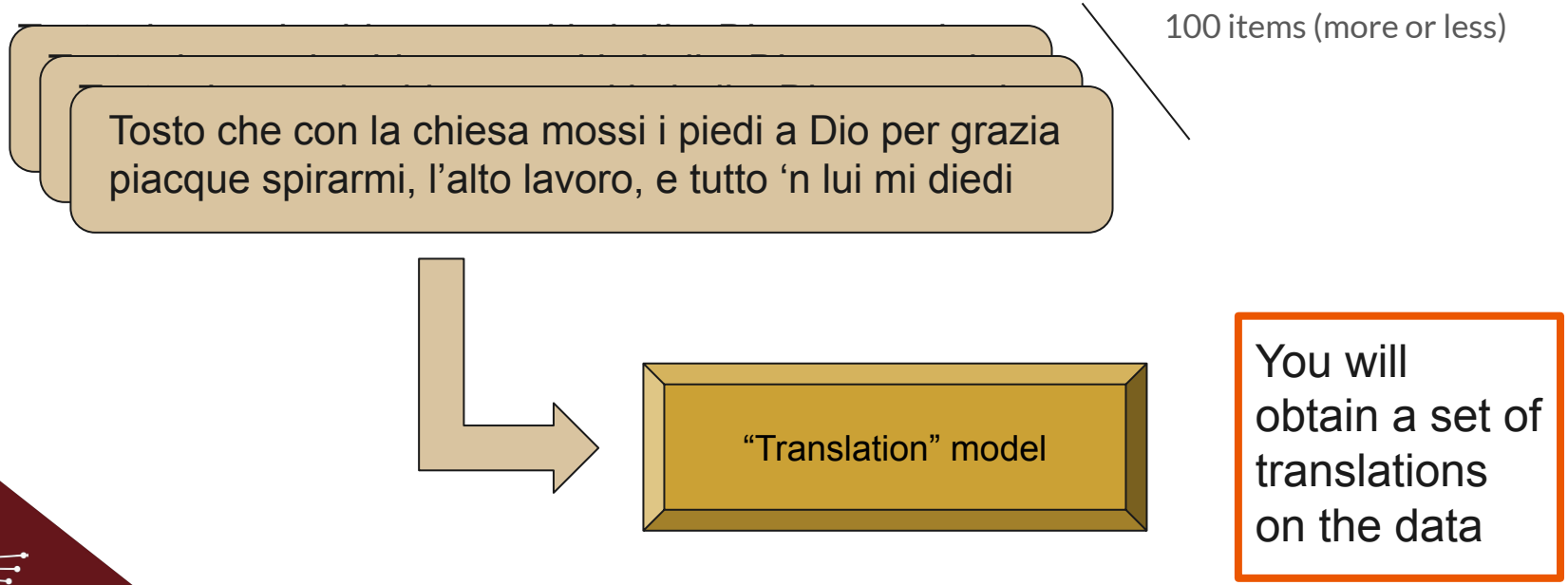
“Acciottolata come la strada, questa corte è tutta in pendío.”



Lastricata come la strada, questa corte è completamente in pendenza.

Test data

You will receive the test data containing the archaic sentences, without the gold references (i.e., *the modern translations*)



LLM as a judge evaluation guidelines:

This is an example of the LLM-as-a-judge evaluation grid (Rubric Scores):

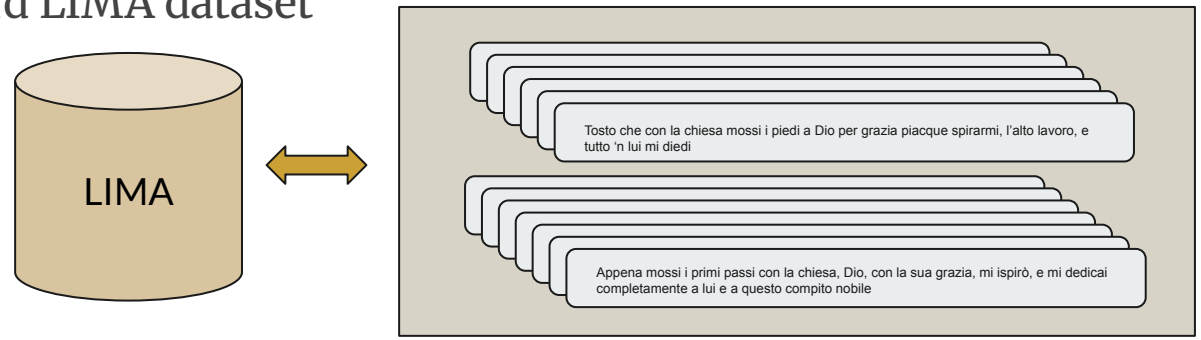
1. **Completely unacceptable translation**: the translation has no pertinence with the original meaning, the generated sentence is either gibberish or something that makes no sense
2. **Severe semantic errors, omissions or substantial add ons** on the original sentence. The errors are of semantic and syntactic nature. It's still something no human would ever write
3. **Partially wrong translation**, the translation is lackluster, it contains errors, but are mostly minor errors, like typos, or small semantic errors.
4. **Good translation**. The translation is mostly right, substantially faithful to the original text, but the style does not perfectly match the original sentence, still fluent and comprehensible, and could semantically acceptable
5. **Perfect translation**. The translation is accurate, fluent, complete and coherent. It retained the original meaning as much as it could.

Don't just copy-paste but experiment with different rubrics tackling different aspects at a time to have a holistic view of a model performance!

Fine tuning option

You can also experiment with fine tuning a model for generating a better translation
In order to have a good instruction fine tuning step, you will need to add examples of the task you want it to perform.

To analyse this, you can create a small sample of parallel data (*same sentences in two languages*), add it to LIMA and test whether the result improve compared to fine tuning on the standard LIMA dataset

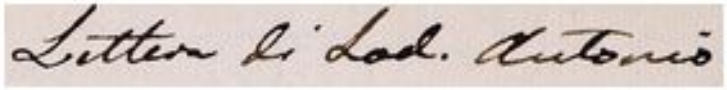


SAPIENZA NLP

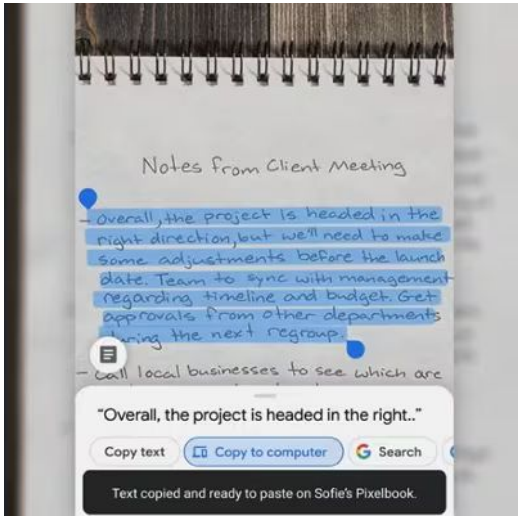
Track 2: Text Cleaning

Text Cleaning - OCR-derived text to clean text

Optical Character Recognition (OCR) is a set of methods used to convert printed or handwritten text within images or scanned documents into machine-readable text.




```
{
    "decade_id": 0,
    "img": "002_02_00.jpg",
    "nameset": "train",
    "text": "Lettera di Lod. Antonio",
    "width": 130,
    "height": 1120
}
```

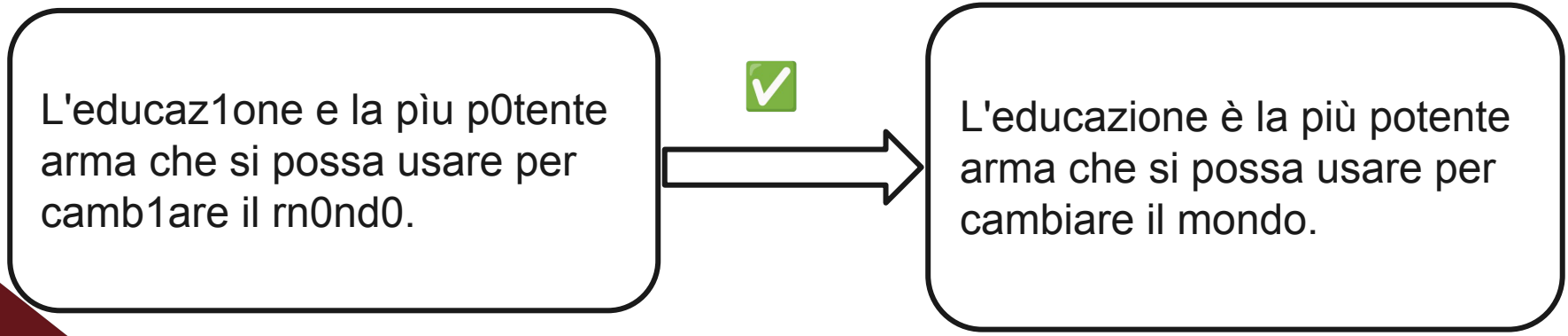



SAPIENZA NLP

OCRed to clean text

Problem? OCR systems often output noisy output text due to font ambiguity, scan quality, or complex layouts.

Your task: Use different small LLMs to fix this and compare their performance.



OCRed to clean text

Problem? OCR s mbiguity, scan quality, or comp

Your task: Use erformance.

Common OCR mistakes in Italian

`1` for `i` → *educaz1one*, *camb1are*

`ì` for `ù` → *pìu*

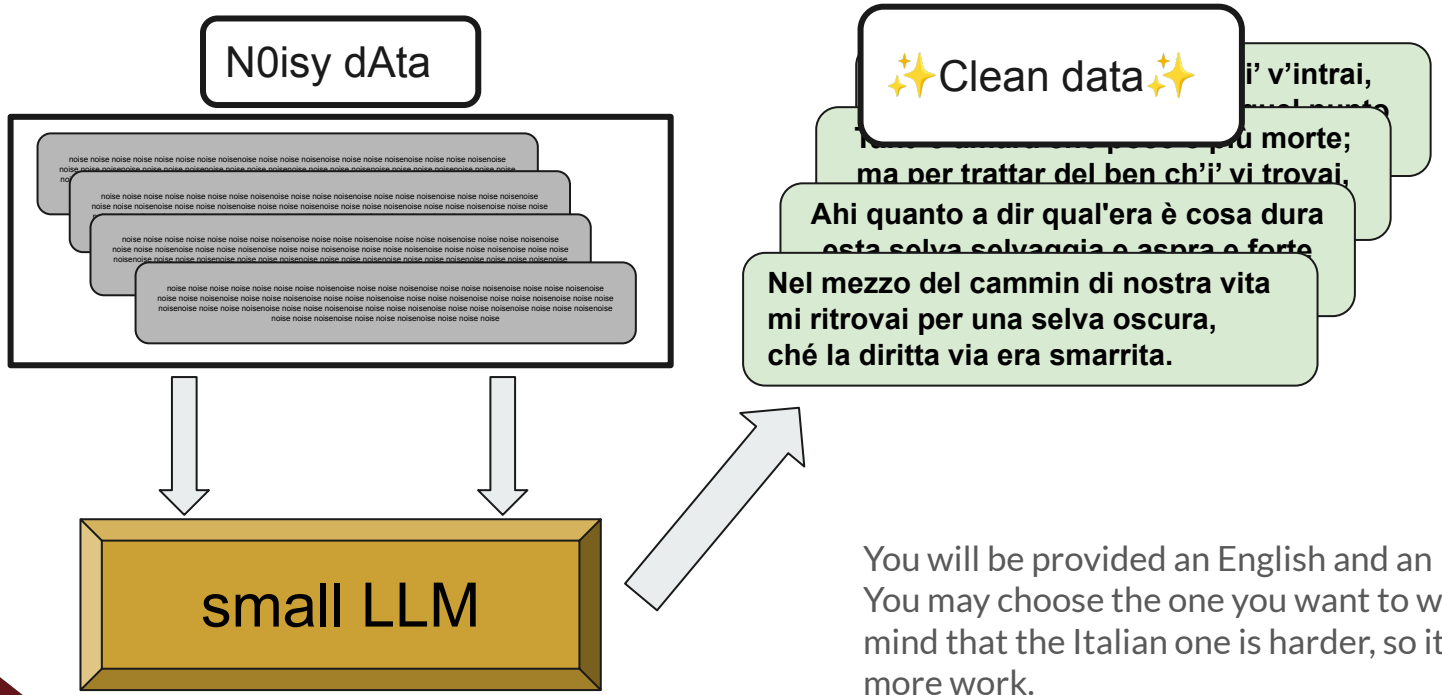
`e` for `è` (accent loss) → *e la pìu*

`rn` for `m` → *rn0nd0*

`0` (zero) for `o` → *p0tente*, *rn0nd0*

L'educaz one è la più potente arma che si possa usare per camb1are il re il mondo.

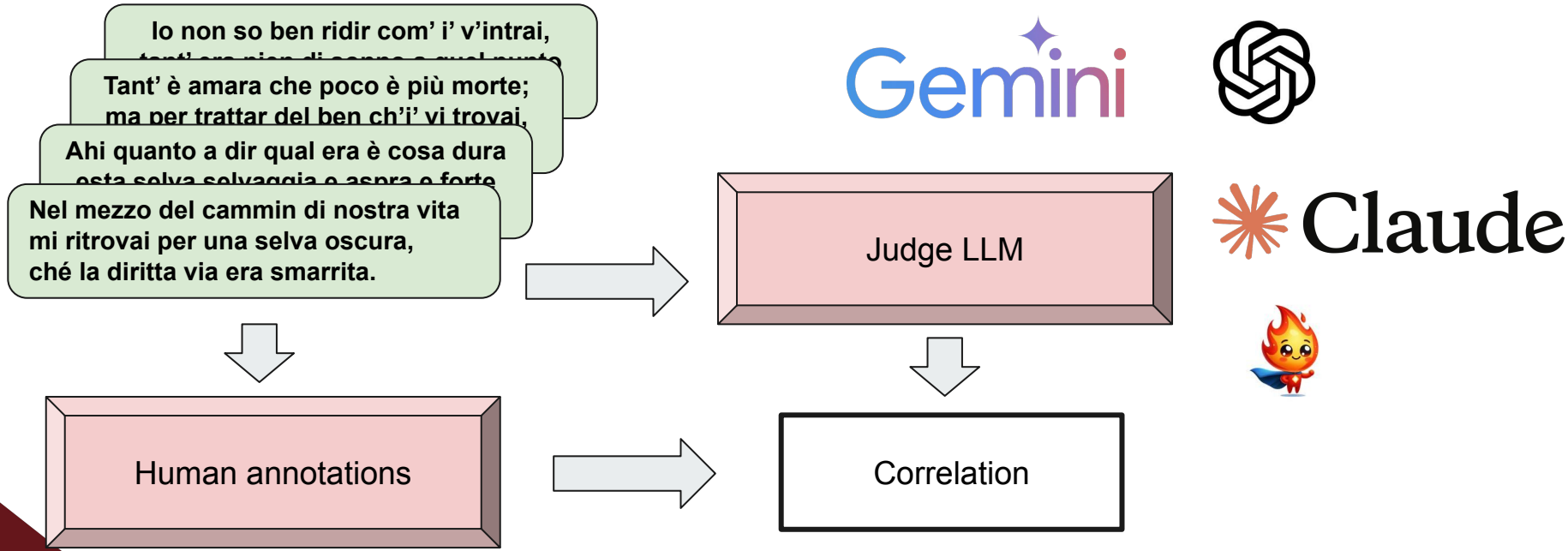
We'll give you the noisy data (ITA and ENG):



You will be provided an English and an Italian dataset.
You may choose the one you want to work on but keep in mind that the Italian one is harder, so it will count as more work.
You will have a parallel corpus (corrupted and clean paragraphs) for both dataset.

SAPIENZA NLP

How will you evaluate it?



Track 3: CulturaLLM

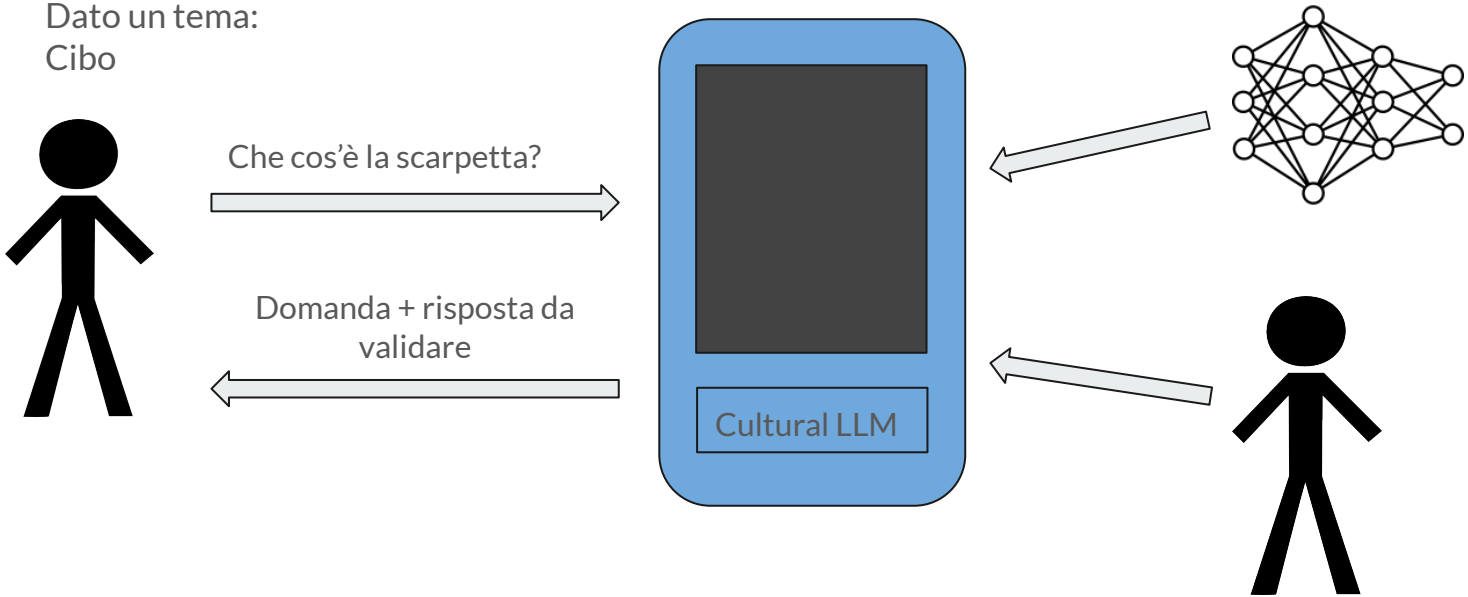
CulturaLLM: Gamification & Large Language Model

- **Obiettivi:**
  - **valutare la qualità** dei Large Language Model tramite **gamification**, in particolare la specificità culturale di un LLM
  - **creare un benchmark innovativo** di valutazione della specificità culturale dei modelli per la lingua italiana
- Presentazione alla **Rome Future Week**, 15-21 settembre e in collaborazione con l'**Università di Hannover**
- Quali domande chiedere per capire se un LLM ha (maggiore) conoscenza della cultura italiana (o di un'altra nazionalità/lingua)?

Forschungszentrum · Research Center
L3S
SAPIENZA
UNIVERSITÀ DI ROMA

Progetto personalizzato (1)

CulturaLLM: Gamification e Large Language Model



Progetto personalizzato (1)

CulturaLLM: Gamification e Large Language Model

- Tre stadi:
  - **Question:** dato un tema (cibo, comportamenti, cinema, personaggi televisivi, sport, ecc.), l'utente deve porre una domanda
  - **Answer:** data una domanda, l'utente deve rispondere (può anche rispondere che la domanda non ha "specificità culturale")
  - **Validate:** date due o più risposte, l'utente ne effettua un ranking (oppure: corretta / sbagliata)
- Sistema di punteggi, leaderboard
- Login
- Piccolo tutorial interattivo
- Mobile friendly

What data is available?

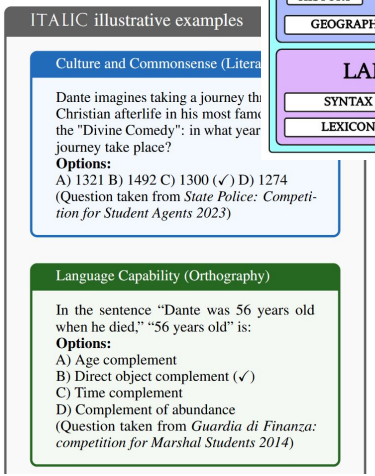
ITALIC:
An Italian Culture-Aware Natural Language Benchmark

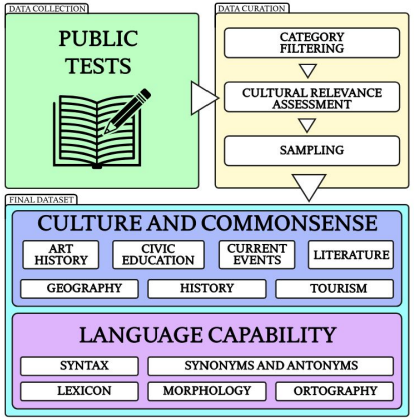
Andrea Seveso[1,3], Daniele Potertì[2], Edoardo Federici, Mario Mezzanzanica[1,3], Fabio Mercorio[1,3]

[1]Dept of Statistics and Quantitative Methods, University of Milano-Bicocca
[2]Dept of Economics, Management and Statistics, University of Milano-Bicoc
[3]CRISP Research Centre crispresearch.eu, University of Milano-Bicocc

Abstract

We present ITALIC[1], a large-scale benchmark dataset of 10,000 multiple-choice questions designed to evaluate the natural language understanding of the Italian language and culture. ITALIC spans 12 domains, exploiting public tests to score domain experts in real-world scenarios. We detail our data collection process, stratification techniques, and selection strategies. ITALIC provides a comprehensive assessment suite that captures commonsense reasoning and linguistic proficiency in a morphologically rich language. We establish baseline performances using 17 state-of-the-art LLMs, revealing current limitations in Italian language understanding and highlighting significant linguistic complexity and cultural specificity challenges. ITALIC serves as a benchmark for evaluating existing models and as a roadmap for future research, encouraging the development of more sophisticated and culturally aware natural language systems.





NORMAD: A Framework for Measuring the Cultural Adaptability of Large Language Models

Abhinav Rao*† Akhila Yerukola*† Vishwa Shah†
Katharina Reinecke‡ Maarten Sap†

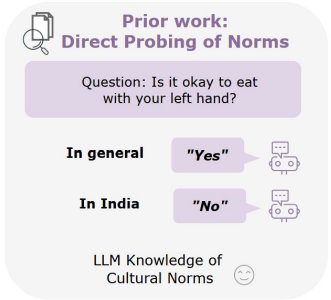
†Language Technologies Institute, Carnegie Mellon University
‡ Paul G. Allen School of Computer Science & Engineering, University of Washington
†{abhinavr,ayerukol,vishwavs,msap2}@cs.cmu.edu, ‡reinecke@cs.washington.edu

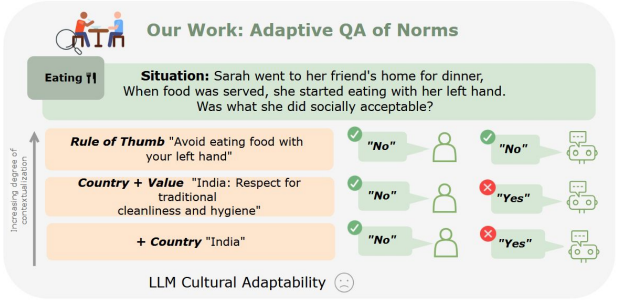
Abstract

be effectively and safely deployed to bal user populations, large language models Ms) may need to *adapt* outputs to user val- and cultures, not just know about them. We roduce NORMAD, an evaluation framework assess LLMs' cultural *adaptability*, specif- lly measuring their ability to judge social eptability across varying levels of cultural m specificity, from abstract values to explicit social norms. As an instantiation of our framework, we create NORMAD-ETI, a benchmark of 2.6k situational descriptions representing social-etiquette related cultural norms from 75 countries. Through comprehensive experi-

larly concerning the representation of various demographics (Bender et al., 2021), human values, and cultures (Masoud et al., 2023). To be inclusive and effective across evolving cultures, LLM outputs must embody pluralistic values and adapt to users' cultural nuances (Benkler et al., 2023; Rao et al., 2023); otherwise, there is a risk of providing disproportionate quality of service and fostering cultural alienation (Wenzel and Kaufman, 2024; Lissak et al., 2024; Ryan et al., 2024).

Previous work has largely focused on assessing knowledge and biases by probing LLMs with curated socio-cultural knowledge databases (Nguyen et al., 2023; Dwivedi et al., 2023; Fung et al., 2024;



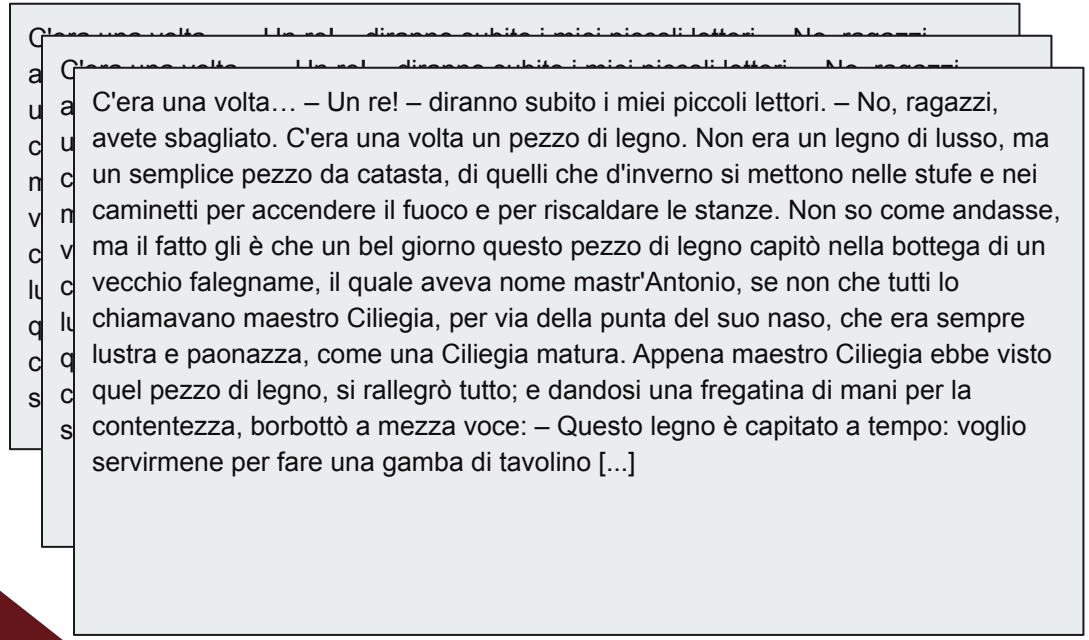


Your assignment for cultural NLP

- Find ways to generate a dataset of cultural questions and appropriate answers in new directions compared to the existing papers
- Provide an evaluation of at least two different models in their ability to provide cultural-specific answers
- OR just integrate your module into the B.Sc. students' application

Track 4: Sentence Splitting

Introduction to sentence splitting

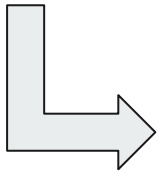


Whether it is Machine translation, summarization, Named entity recognition or basically any NLP task, dealing with long documents is not easy

SAPIENZA NLP

Introduction to sentence splitting

Pinocchio era un burattino. Parlava e camminava come un bambino



Pinocchio era un burattino

Parlava e camminava come un bambino.

spaCy

Stanza

It's the first step in many NLP pipelines (parsing, translation, summarization)

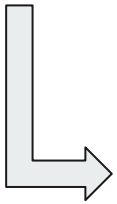
Many Sentence "out of the box" sentence splitters are available but…

Sapienza NLP

Introduction to sentence splitting

TEST GOLD

1) «Al sagrestano gli crede?»
2) «Perché?»



UDPipe 2 -VIT model

1) » «Al sagrestano gli crede?» «Perché?»

Ersatz

1) » «Al sagrestano gli crede?
2) » «Perché?

These models often don't work Out-of-Domain (ODD) sentences!

Building a sentence splitter that actually works OOD is a non-trivial task in NLP

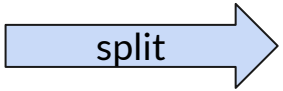
SAPIENZA NLP

Sentence Splitter

Your task is to build a Sentence Splitter using a given dataset and test its performance on OOD data

A sentence splitter can be rule-based, an LLM or a trained classifier

C'era una volta… – Un re! – diranno subito i miei piccoli lettori. – No, ragazzi, avete sbagliato. C'era una volta un pezzo di legno. Non era un legno di lusso, ma un semplice pezzo da catasta, di quelli che d'inverno si mettono nelle stufe e nei caminetti per accendere il fuoco e per riscaldare le stanze. …



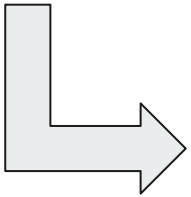
1. C'era una volta…
2. – Un re! – diranno subito i miei piccoli lettori.
3. – No, ragazzi, avete sbagliato.
4. C'era una volta un pezzo di legno.
5. Non era un legno di lusso, ma un semplice pezzo da catasta, di quelli che d'inverno si mettono nelle stufe e nei caminetti per accendere il fuoco e per riscaldare le stanze.
6. …

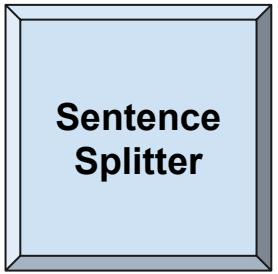
Sentence Splitter: data

You will be given "I promessi sposi" by Alessandro Manzoni, annotated for sentences splitting, divided in *training* and *development* sets.

We will also provide you with a OOD test set, which you can use to test your model.

You will use this dataset to train a sentence splitter





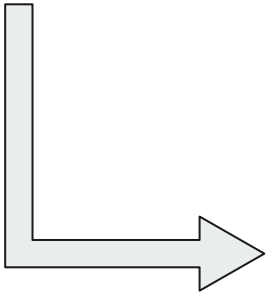
Your Sentence splitter will be tested out of domain

SAPIENZA NLP

Different approaches to tackle this task:

Era sul finire, quand'ecco un nuovo flagello. Molte cose importanti, di quelle a cui più specialmente si dà titolo di storiche. -Altolà- disse Renzo; una fioca voce rispose..

Rule based system: break at each point



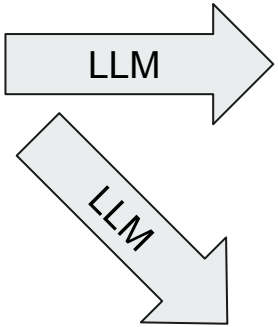
Molte cose importanti, di quelle a cui più specialmente si dà titolo di storiche.

Era sul finire, quand'ecco un nuovo flagello.

-Altolà- disse Renzo; una fioca voce rispose

Different approaches to tackle this task:

Era sul finire, quand'ecco un nuovo flagello. Molte cose importanti, di quelle a cui più specialmente si dà titolo di storiche. -Altolà- disse Renzo; una fioca voce rispose..



Era sul finire, quand'ecco un nuovo flagello.<EOS> Molte cose importanti, di quelle a cui più specialmente si dà titolo di storiche.<EOS> -Altolà- disse Renzo; una fioca voce rispose..

This is what you would expect… But in reality… Hallucinations happen!

Era sul finire quand'ecco che un nuovo cose importante fatto, <EOS> una fioca <EOS>-Altolà-<EOS>

EOS = End of Sentence

Different approaches to tackle this task:

Era sul finire, quand'ecco un nuovo flagello. Molte cose importanti, di quelle a cui più specialmente si dà titolo di storiche. -Altolà- disse Renzo; una fioca voce rispose..

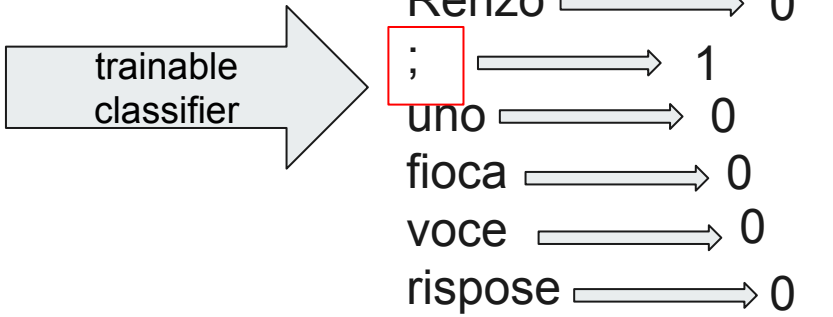
Era sul finire, quand'ecco un nuovo flagello.<EOS> Molte cose importanti, di quelle a cui più specialmente si dà titolo di storiche.<EOS> -Altolà- disse Renzo; una fioca voce rispose..

But even when hallucinations don't happen:

1. Very costly
2. Very time consuming
3. Extremely slow

Different approaches to tackle this task:

Era sul finire, quand'ecco un nuovo flagello. Molte cose importanti, di quelle a cui più specialmente si dà titolo di storiche. -Altolà- disse Renzo; una fioca voce rispose..



You can also see this problem as a binary classification task on tokens.

0

1

SAPIENZA NLP

Token level classification with an encoder

You can also see this problem as a binary classification task on tokens.

0

1

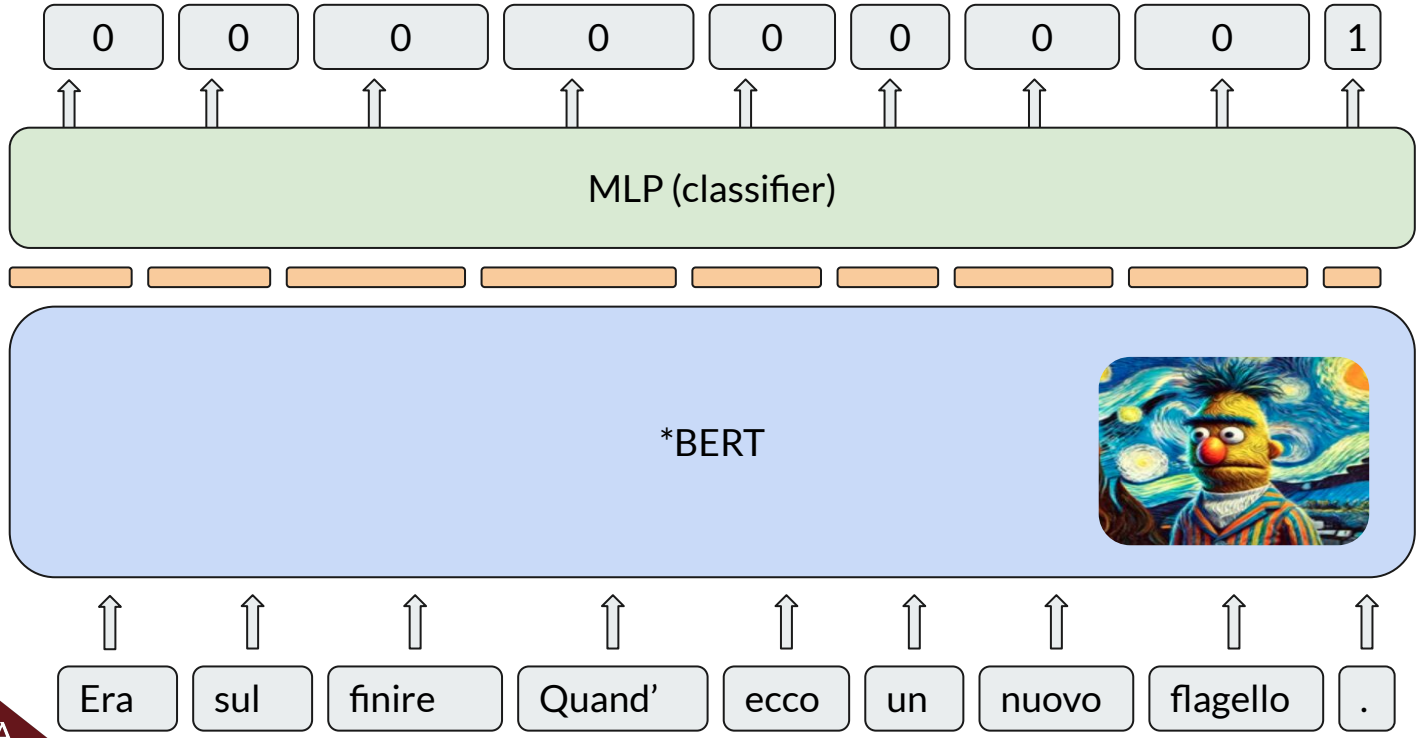
Each token needs to be classified.

- "1" if EOS
- "0" otherwise

This means that we first need to give these tokens a **vector representation with an Encoder** (BERT, ModernBERT, EuroBERT, DeBERTa, ...)

*BERT

Token level classification with an encoder



Evaluation Grids

Translate Non-modern to Modern Italian

16 pt - Generate—*be creative*—with **at least two Open** LLMs (e.g., Minerva-350M and Llama-3.1-1B). You are encouraged to use whatever tools or systems to enhance the quality of the generated content (e.g., explicit semantics)

10 pt - Evaluate the generated output with an LLM-as-a-Judge (locally or via API), defining proper guidelines

4 pt - Do qualitative analysis of the output of the translation models, computing the Agreement between your scores and the LLM-as-a-judge scores on a small sample (e.g., 20 sentences)

EXTRA (mandatory for 3-people groups) **2 points:**

- try to use **Prometheus as a Judge.**

SAPIENZA NLP

Submission Guidelines Translation

1. Report: you can find the specific guidelines for the report in slides below
2. Code: A colab notebook with runnable code. Well formatted, well commented, with the same guidelines as HW1.
3. Output files: you need to provide the output of every model that you used. The output needs to be in json lines format (`.jsonl`).
    a. Your generated translation contents in "`groupname-hw2_transl-modelname.jsonl`" file
    b. Your LLM-as-judge outputs in "`groupname-hw2_transl-judge.jsonl`" file

- Remember to correctly format the .JSONL file (it's not the same as a .JSON!)

    - If you test more than one criteria in the LLM-as-a-Judge part, you can submit different files, one for each criteria named "`groupname-hw2_transl-judge_criteria.jsonl`"

OCRed Text Cleaning

16 pt - Generate—*be creative*- with at least two Open LLMS (e.g., Minerva-350M and Llama-3.1-1B). You are encouraged to use whatever tools or systems to enhance the quality of the generated content (e.g., explicit semantics)

10 pt - Evaluate the generated output with an LLM-as-a-Judge (locally or via API), defining proper guidelines

4 pt - Do a comparative analysis by computing the agreement between the ROUGE-1, -2, -L and the LLM-as-a-Judge with human annotations on the output of the translation models (use a small sample, e.g., 20 sentences).

EXTRA (mandatory for 3-people groups) **2 points:**

- try to use **Prometheus as a Judge.**

SAPIENZA NLP

OCRed Text Cleaning

English dataset

The Vampyre by Polidori, containing ~10k tokens – you are required to submit at least the first 5k tokens. To do so, split the text on newlines and take from index 0 to 24

```
In [57]: vampyre = vampyre['0'].split('\n')

In [58]: mandatory = vampyre[:24]

In [59]: sum([len(line.split()) for line in mandatory])
Out[59]: 5033
```

Italian dataset

Pinocchio by Collodi, containing ~44k tokens – you are required to submit at least the first 5k tokens – from index 1 to 7

Note: you can use either (or both) dataset, independently of the language you speak – the Italian one is harder as it is *real* OCR data

Submission Guidelines OCR

1. Report: you can find the specific guidelines for the report in slides below
2. Code: A colab notebook with runnable code. Well formatted, well commented, with the same guidelines as HW1.
3. Output files: you need to provide the output of every model that you used. The output files MUST mirror the same format of the input file (`.json`) and use the same ID.
    a. Your "clean" text goes into a "`groupname-hw2_ocr-modelname.json`" file
    b. Your LLM-as-judge outputs go into a "`groupname-hw2_ocr-judge.json`" file

- If you test more than one criteria in the LLM-as-a-Judge part, you can submit different files, one for each criteria named "`groupname-hw2_ocr-judge_criteria.json`"

SAPIENZA NLP

Sentence Splitter

15 pt Train a token-level Sentence Splitter using an encoder model (*BERT), evaluate with classification metrics.

- 3-people groups must train at least 2 models.

11 pt Test the generative approach with small models (e.g. Llama-3.1-1B, Llama-3.1-3B)

- 3-people groups must use at least 2 small models.

4 pt Comparative analysis of the two approaches on the validation set – which one is better? Whic

EXTRA – 2 points:

Test the model(s) on the OOD test set and do a second analysis as the last point – Is the model committing recurrent errors? How would you suggest mitigating them?

SAPIENZA NLP

Submission guidelines Sentence Splitter

1. **Report**: you can find the specific guidelines for the report in slides below
2. **Code**: A colab notebook with runnable code. Well formatted, well commented, with the same guidelines as HW1.
3. **Output files**: You need to submit the **CSV (comma separated) files** containing the output of your model on the unlabeled test set.
    a. One output file per model
    b. Follow this naming convention: "`groupname-hw2_split-modelname.csv`"
4. The output files **must** contain two columns, `token` and `label` (refer to the train and dev set files)

Avoiding Frequent Errors

Memorandum

1. Output Files

- All output files must be placed in the designated shared folder.
- Ensure that access permissions are properly set so that we can view and download all necessary files.
- The output files must be exactly match the specified format (see slides above)

2. Code

- The code must be in colab notebooks, must be runnable, and well formatted
- If your code does not run, we will deduct points
- The code should be commented and clear, as for HW1

Memorandum

MAX 2 Pages for main part of the report
+
No limit Appendix with Images and Table

3. Report Clarity

- The report must be written in ACL format and properly structured.
- The report should include the following clearly separated sections:
  - **Introduction** (***VERY*** brief)
  - **Methodology**: explain your starting hypothesis and the methods you will use to test them
  - **Experiments:** explain your experimental setup; here you give technical details on models, resources, etc. and how you implemented what you described before
  - **Results**: present the results referencing tables and ***commenting*** them, i.e., why you accept or refuse your starting hypothesis and how you explain the (most notable/unexpected) results.

Use the Appendix for figures, tables, pseudo code and additional, specific explanations

SAPIENZA NLP

Memorandum

MAX 2 Pages for main part of the report
+
No limit Appendix with Images and Table

3. Report Clarity

- Mixing content between sections or presenting information in an unclear or disorganized manner will result in a lower grade
- All images, tables, and results must be clearly referenced and discussed in the Results
    - tables and figures **should not** be screenshots - import images in pdf for better rendering
- The Methodology should be detailed and clear enough to be understood by a reader with minimal prior knowledge of the topic
    - do not waste space by explaining to us the course core concepts (tokenizers, Transformers, ...)
    - write a concise description if what you're using is not covered in the course and is something relatively new/not commonly used
- **Appendix:** additional details/describe all your experiments, but the main body of the report shall contain ***ONLY*** the final work you're presenting (be **merciless** with what to include, what to put in the appendix, and what to NOT include)

Submission guidelines

Submission procedure – VERY IMPORTANT!!

⚠️

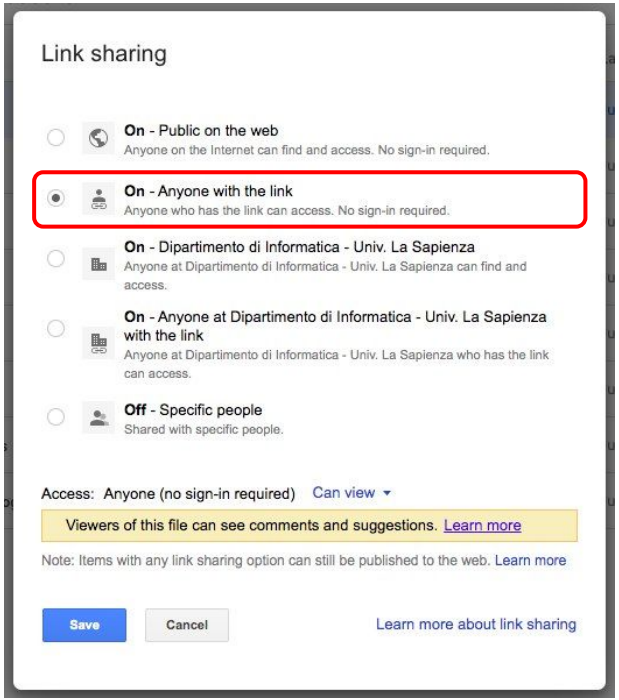
1. Complete the first form (by **Wednesday 28 May!!!!**) uploaded in Classroom so we know how many of you are submitting and which project did you choose.
2. Do the homework.
3. Submit it using this submission form (**NOT the first form**). You can submit a Drive folder or a Github repository. For the Drive folders **they should also be shared with us in Google Drive**.

   ONLY ASSIGNMENTS SUBMITTED THROUGH THE SUBMISSION FORM WILL BE ACCEPTED!!!!!!!!! Therefore, if you share with us the folder in Drive in any other way but the submission form, **we will not accept it**.

Groups

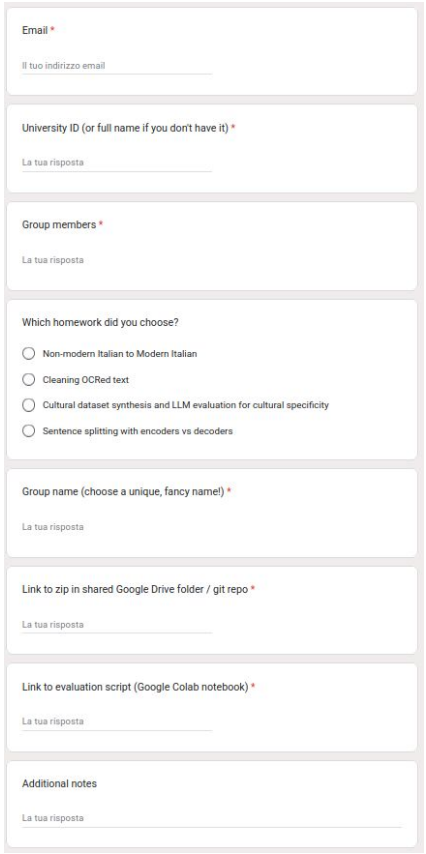
- Groups of 2 or 3 students
- Groups of 3 have to deliver additional work (see slides above detailing each task for reference)
- Singletons are exceptionally possible: for graduation reasons or any other reason that have to be approved by prof. Navigli

Submission Instructions



- If submitting a zip file, upload the zip on your **institutional** Drive and make it **link-shareable** and **public** to anyone.
- Make sure it is accessible via an incognito (unauthenticated) page of your browser!
- You have to submit the homework through this submission form Google Classroom. You will be asked to fill this already mentioned form with the requested information and the **link** to the zip you uploaded on Drive **OR** the link to the public repo, plus a **link** to the Google Colab notebook.

Submission instructions



When submitting using the form:

- Use your institutional email if you have one
- Select the task you are submitting.
- Please write your group members with the same spelling
- Do not abuse the additional notes field (one short line if needed, otherwise send us an email)
- **Double check** that the files you are sharing are accessible from anyone!

SAPIENZA NLP

Plagiarism

We will check for plagiarism both manually and automatically.

It is **not allowed** to:

- Copy code or reports from other students/groups;
- Share your code with other students outside your group;

Projects violating any of the above conditions will be **desk-rejected**