NLP: Final Project

Project Summary

As a part of the curriculum of the Master 2 (M2) course entitled “Natural Language Processing”, you are required to finish this final project. To do so, they have to choose one for the 3 following tasks:

1. Creating a writing assistant.
2. Creating an Automated Fact Checking System.

More information about each task is provided below. Every task discussion is divided as follows:

- Description: what is the general idea behind the task domain as a whole.
- Requirements: what are the required steps by the students, and what is expected from the developed systems to be able to accomplish.
- Datasets: some online available datasets that could be used to train the developed models for the chosen task.

**Notes:

- You are to make teams of **3 people**, and each team can choose one of the tasks to solve.
- The final project presentation is due on January 29, 2025.
- This is an open-ended project; You are expected to do your oun research to fill in any gaps that may exist in your knowledge regarding solving the provided problems.

For any further detail, please contact the instructor: **Khodor Hammoud**

Project Presentation

Each team, composed of 3 members, will have 10-12 minutes to present their work. A powerpoint slideshow is **NOT** obligatory, you may add it if you choose to. In your presentation, you must describe all the following steps:

1. The methodology of thinking used in tackling the task. That is, what intuition resulted in the selected model.
2. A description of the required data preprocessing for the chosen task, and the method of implementing said preprocessing steps.
3. A full model description: what are the different layers constituting your neural network model, and the role of every layer, alongside the model architecture chosen. Each team must show a full understanding of the inner workings of their model.
4. A full description of the training phase and the model tuning (tensorboard might be helpful in this scenario).
5. Model results: accuracies, and find a state-of-the-art model for the same use case to compare your results to.

Have a notebook ready, with the model pre-loaded for live demonstration.

Processing Power

As this project requires GPU processing power, Google Colab (https://colab.research.google.com/) provides a decent free to use notebooks equipped with GPU units. Just keep in mind that sessions on google colab don't last very long. The session resets every 4 hours or so, so it would help you if you connect your google drive, and constantly backup your trained model during the training phase, then reload the final milestone when the session restarts.

1. AI-Powered Creative Writing Assistant

Description

The goal of this project is to develop an AI assistant that aids in writing. You can choose to implement an assistant that does one of 3 tasks:

1. The assistant will generate suggestions for continuing a story based on an input prompt
2. The assistant can improve existing sentences by rewriting them (fixing errors and such)
3. The assistant can summarise a given piece of text.

Requirements

- Use recurrent architectures (LSTMs/GRUs) or Transformers (built and trained from scratch) for text generation.
- Experiment with incorporating embeddings like BERT or FastText for sentence understanding and enhancement.
- Keep in mind hardware limits of Google Colab or your local machines

Datasets

Public domain datasets like

- BookCorpus: https://huggingface.co/datasets/bookcorpus/bookcorpus
- WikiText: https://huggingface.co/datasets/Salesforce/wikitext
- Wiki Summary: https://github.com/m3hrdadfi/wiki-summary
- You can add your own datasets as well

2. Automated Fact Checking System.

Description

Fact checking is the task of assessing whether claims made in written or spoken language are true. As the rest of the tasks in this project, this is a task that is normally performed by trained professionals: the fact checker must evaluate previous speeches, debates, legislation and published figures or known facts, and use these, combined with reasoning to reach a verdict.

For example, given a statement: “Paris is the capital of France”, the fact checker would have previous knowledge about France, Paris, and the relation that is “capital”, and whether it applies to Paris with respect to France.

Although decent approaches have been done in this field, fact-checking remains as an open problem, where there is not yet a dependable method to validate information’s correctness.

Automated fact checking is a field of major interest for a lot of industries, since its applications spread vast and wide. As a few examples:

- The ability to validate whether official news are spreading false/biast information or not.
- The ability to filter out social media posts based on correctness.
- Building dependable datalakes from open-web information by having a fact-checking “filter” that lets by only valid information from the web.

Requirements

Students who choose this task are required to build an automated fact-checking system that, provided a statement, can infer whether this statement is correct or not. For this to be feasible, a large factualy-correct database is required, for which DBpedia can be used (check Datasets).

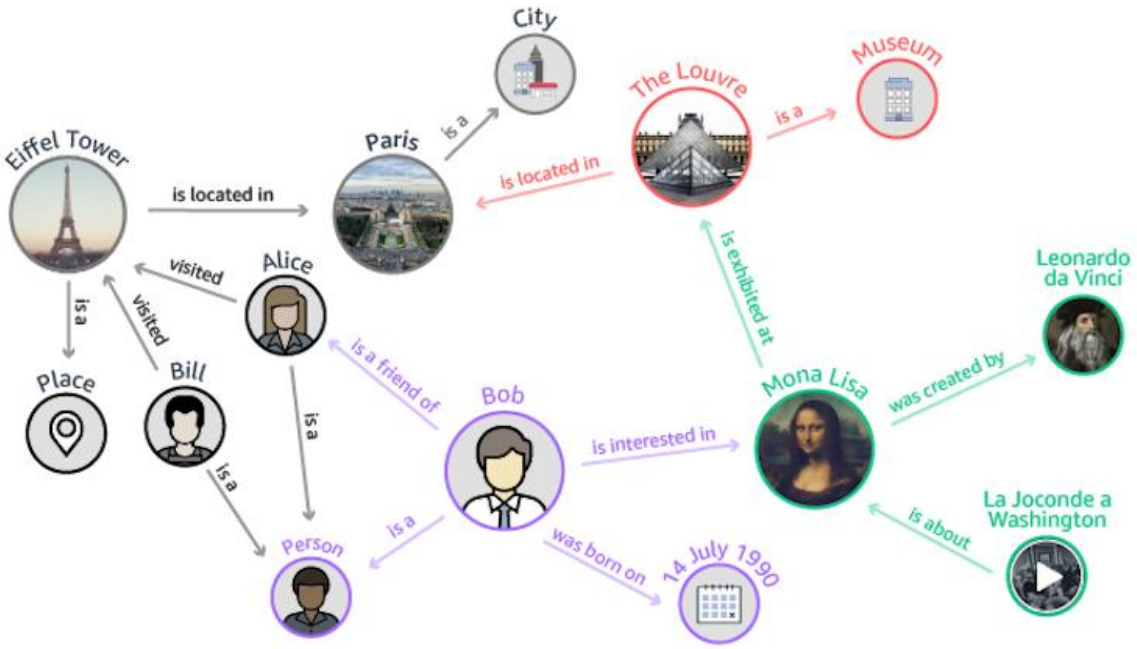
There are many approaches to tackle this task, below we describe a simplified version of one of them:

Given a statement to fact-check, say: “Paris is the capital of France”, we would extract the “triplets”

Paris → capital → France

And then we run matching between our extracted triplet and our knowledgebase, represented by DBpedia. Inside of DBpedia, we can run a query to check all the information available about Paris, and within we should find the relation “is_capital”, linking it to France. Hence, we can confirm that Paris is indeed the Capital of France.

Side note, this also falls within the field of knowledgebase building, since we can use the same approach to extract triplets from open text, and build a knowledge graph linking all the extracted entities. An example image is provided below;



Datasets

For the purposes of this task, students can use DBpedia. (https://www.dbpedia.org/)

DBpedia is a crowd-sourced community effort to extract structured content from the information created in various Wikimedia projects. This structured information resembles an open knowledge graph.

DBpedia provides multiple ways to query data. One can either download available dataset releases (https://wiki.dbpedia.org/develop/datasets), run SPARQL queries on the DBpedia public SPARQL endpoint (https://www.dbpedia.org/resources/sparql/), or use linked data access (for example, getting information for Paris, https://dbpedia.org/page/Paris), and linked data access is available in multiple machine-readable formats (JSON for example for Paris: https://dbpedia.org/data/Paris.json).

So you have a lot of options to choose from when it comes to queryng these data.

Query results are in the form of an RDF graph, where every element in the results has it's own entities linked to it through relations. For example, if we search "Paris", we'll find, amongst a lot others, the relation "is_capital", and linked to this relation, we can find "France".

is dbo:campus of	• dbr:Institut_supérieur_du_commerce_de_Paris • dbr:Center_for_Research_and_Interdisciplinarity • dbr:École_nationale_supérieure_de_création_industrielle • dbr:École_normale_supérieure_(Paris) • dbr:Mines_ParisTech • dbr:IONIS_Education_Group • dbr:Arts_et_Métiers_ParisTech • dbr:AgroParisTech • dbr:IAE_Paris • dbr:LISAA_School_of_Art_&_Design
is dbo:capital of	• dbr:Bourbon_Restoration • dbr:First_French_Empire • dbr:First_Restoration • dbr:Early_modern_France • dbr:Spanish_Republican_government_in_exile • dbr:France • dbr:France_in_the_Middle_Ages