TP 2: Deep Neural Network Models for Advanced NLP Problems

Project Summary

As a part of the curriculum of the Master 1 (M1) course entitled "Advanced Machine Learning and Text Mining", the students are required to finish this final year project. To do so, they have to choose one for the 3 following tasks:

1. Creating a Machine Translation System.
2. Creating a Question Answering System.
3. 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:

- student 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 March 19, 2021.
- These projects are ordered in terms of their level of challenge, with machine translation being the least challenging task, and the fact-checking system being the most challenging.
- This is an open-ended project; The students are expected to do their oun research to fill in any gaps that may exist in their knowledge regarding solving the provided problems. We will have the needed building blocks to develop and train neural network models for NLP purposes throughout the course, but the procedural information needed to accomplish the tasks do require some research around every field's domain.

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

Project Presentation

Each team, composed of 3 members, will have 15 minutes to present their work. The presentation must include a powerpoint slideshow describing 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 algorithms 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 comparison to the state-of-the-art,

In addition to the slides, 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. Machine Translation System.

Description

Machine translation refers to translations that are automatically created with a computer, with no human involvement. It uses Sequence-to-sequence learning (Seq2Seq), which is about training models to convert sequences from one domain (e.g. sentences in English) to sequences in another domain (e.g. the same sentences translated to French).

The most basic function of Machine Translation is the word-by-word translation where a word in source language is replaced with the corresponding word in another language. More advanced models use data-driven approaches to distinguish the different meanings each word holds depending on the context in which it is used.

There are mainly 3 approaches for machine translation, **Rule-Based**, **Statistical**, and **Neural**. Out of the aforementioned approaches, we are interested in the latter; Neural machine translation. This is the use of deep neural network models combined with large amounts of data to prtform machine translation.

Requirements

Students who select this task are required to develop a system that can translate between at least 2 languages, in both directions; for example: English to French and vice versa. Translations are to be comprehendible, and the languages chosen should include English (English ← → Another_Language).

The developed system should include a deep neural network model, adequately trained on the appropriate datasets.

In terms of which models/approaches to use, there are multiple ways to handle this task, one of the more popular ones is using **LSTM RNNs**.

Research has proven that using LSTM models produces great results for machine translation, although it might suffer in psrformance when it comes to long text sequences.

Datasets

There exists a lot of datasets available online for machine translation tasks, and here are a few:

1- European Parliament Proceedings Parallel Corpus 1996-2011 http://www.statmt.org/europarl/ : provides datasets fot translating from 20 different languages to English.
2- Tab-delimited Bilingual Sentence Pairs http://www.manythings.org/anki/ : provides datasets fot translating from English to 81 different languages.
3- There exists more, you are free to look them up.

2. Question Answering System.

Description

Question answering a field of information retrieval and natural language processing (NLP) concerned with building automated systems which answer questions posed by humans in a natural language. There are 2 types of systems; Closed-Domain QA and Open-Domain QA.

- *Closed-Domain QA* is about building systems that answer questions from a specific domain. So, given a piece of text, we hope to develop systems capable of answering questions from that specific text.
- *Open-Domain QA* deals with questions about nearly anything, and can rely on general ontologies and world knowledge. These systems usually have much more data available from which to extract the answer.

In this task, we are concerned with closed-domain QA.

Types of Questions in modern systems:

- Factoid questions: Where is the Louvre Museum located?
- Complex questions: What do scholars think about Jefferson's posi4on on dealing with pirates?

We are interested in Factoid questions in this task.

Requirements

Students who select this task are required to create an automated closed-domain QA system capable of providing correct and inteligeable answers to human questions regarding a provided piece of text.

Your final model will be given a paragraph, and a question about that paragraph, as input. The goal is to answer the question correctly

The developed system should include a deep neural network model, adequately trained on the appropriate datasets.

You can look into plenty of available techniques in the literature to assist you at creating your approach. You're not required to implement something original, but projects seeking originaliy will be rated higher. Originality doesn't necessarily have to be a completely new approach – small but well-motivated changes to existing models are very valuable.

Datasets

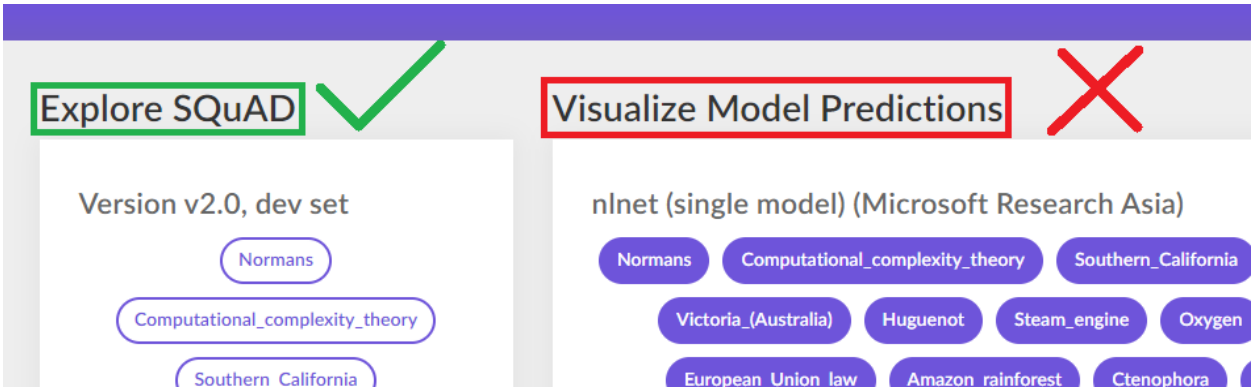
Students choosing this task are to use the Stanford Question Answering Dataset SQuAD 2.0.

The SQuAD dataset is developed by Stanford Univarsity, and is available to download from:

https://rajpurkar.github.io/SQuAD-explorer/

It provide a training set, and a development (testing) set. The SQuAD website offers 2 columns when exploring the dataset (https://rajpurkar.github.io/SQuAD-explorer/explore/v2.0/dev/) the SQuAD dataset on the left column, and the predictions made on the dataset by multiple state of the art models (nlnet, BiDAF, BERT). For the purposes of this project, we are not interested in the predictions made by

these models, but feel free to explore them and check the performance of some of the best models in the question-answering nlp task.



Dataset Visual Structure

Similar to the literature exams we used to do in elementary and middle school, this dataset is in the form of a set of paragraphs, with multiple questions on each. The dataset comes with multiple titles (Normans, Computational_complexity_theory,…). Each title comes in the following structure:

1. A piece of text from which the questions and answers will be extracted (**Context**).
2. A series of **questions** extracted from said text.
3. A set of accepted **answers** for every question.

** Note that some questions have multiple accepted answers.

Dataset JSON Structure

The dataset can be downloaded in JSON format. SQuAD offers training and Development datasets. The JSON structure is as follows:

```
{"version": "v2.0",
 "data": [
        {"title": "TITLE_1",
        "paragraphs": [
                {"qas": [
                        {"question": "QUESTION_1",
                        "id": ID_1,
                        "answers": [
                                {"text": "ANSWER_1_1",
                                "answer_start": INDEX_1_1},
                                {"text": "ANSWER_1_2",
                                "answer_start": INDEX_1_2}
                        ],
                        "is_impossible": (false/true)},
                        {"plausible_answers": [
                                {"text": "P_ANSWER_1_1", "answer_start": P_INDEX_1_1},
                                {"text": "P_ANSWER_1_2", "answer_start": P_INDEX_1_2}
                        ]},
                        {"question": "QUESTION_2",
                        "id": ID_2,
                        "answers": [
                                ...
                        ],
                        "is_impossible": (false/true) },
                        {"plausible_answers": [
                                ...
                        ]}
                ],
                "context": "CONTEXT_1"
                },
                {"qas": [
                        ...
                ],
                "context": "CONTEXT_2"
                },
                ...
        ]}
]}
```


Thus:

- Every title has multiple contexts, each of which has multiple questions.
- Every question might have solutions (is_imposible == false) or not (is_imposible == true).
- Disregard questions without solutions.
- Plausible answers are only to be considered when “answers” is empty.
- For every given answer, “answer_start” indicates the position from the context where the answer starts.

3. 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).

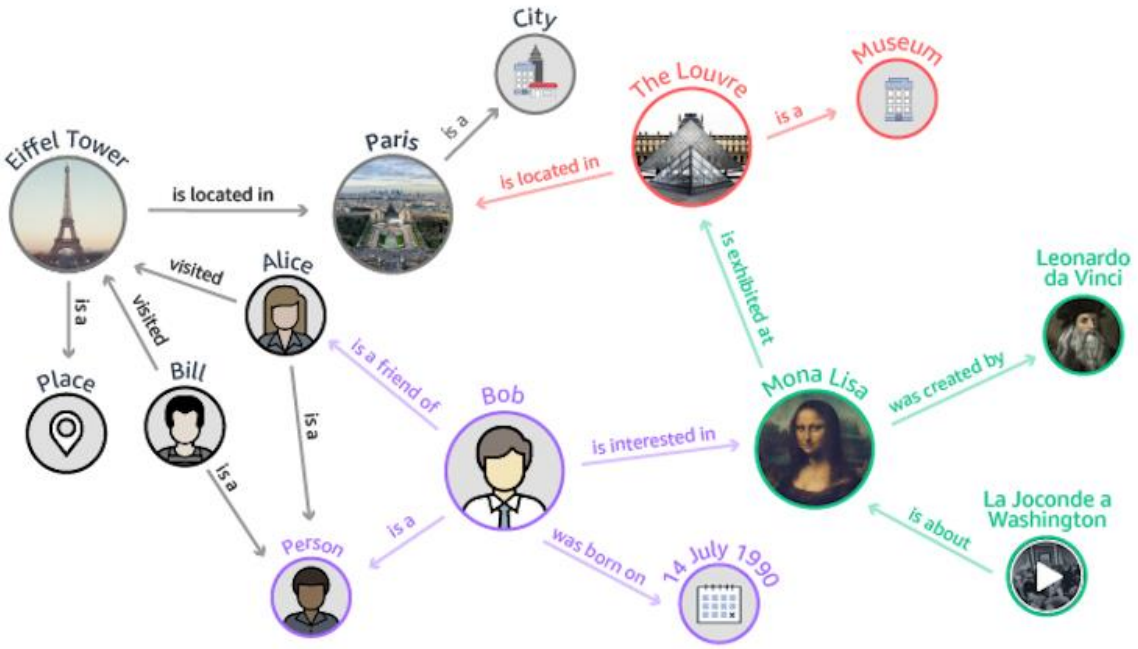
There are many approaches to tackle this task, below we describe a simplified version of one of them:

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 extractd 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. And 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