Prof. Dr. Stefan Dietze (lecturer)
Katarina Boland (assistant)

Heinrich Heine Universität Düsseldorf

Data & Knowledge Engineering (DKE22)

Optional Home Project:
Fact-Checking

This home project exercise can be completed at home, using the expertise and skill sets acquired in previous lectures. All students are invited to present their solution for a grade improvement. In order to participate, please proceed as follows:

- Indicate your intention to submit and present via email to Katarina Boland (katarina.boland@hhu.de) by **04.12.2022**. No code or results required at this stage.
- Use the folder structure in `https://git.gesis.org/dke_2022_homeproject/home_project_students` to prepare your solution at home.
- Provide the source code and generated output via your Git repository by **11.01.2023 10:30 am** CEST time zone. To this end, it is sufficient to send a link to your public repository containing the source code and generated output via mail to katarina.boland@hhu.de.
- Present your solution (approach, results) to the class during the DKE 2022 lecture on **17.01.2023** (detailed arrangements about presentations will be announced later)

Submitting and presenting a working solution (on time) that fulfills all requirements will result in a 0.3 grade improvement in the final exam.

The home project deals with the classification of claims regarding their truth values, i.e. given a fact-checked claim, is it true, false or neither of these two categories? For this project, we will use the claims and their metadata provided by *ClaimsKG*[1]. *ClaimsKG* is a structured database which serves as a registry of claims. Basis of the database is a knowledge graph which provides data about claims, metadata (such as their publishing site), automatically annotated involved entities, links to fact-checking articles and normalized truth ratings. ClaimsKG is generated through a (semi-)automated pipeline which harvests claims and respective metadata from popular fact-checking sites on a regular basis, lifts data into an RDF/S model, which exploits established schema such as schema.org and NIF, and annotates claims with related entities from DBpedia.

Prerequisites

1. Basic knowledge of data & knowledge engineering principles (e.g. data cleaning, data exploration, RDF, SPARQL) and of supervised machine learning is assumed.
2. Clone/fork the provided Gitlab repository to work on the task on your local machine.
3. The ClaimsKG can be found at `https://data.gesis.org/claimskg`. Familiarize yourself with the model, i.e. the used schema for representing claim data.

[1] `https://data.gesis.org/claimskg/`

Detailed Task Description

Your task is to retrieve claims from ClaimsKG and classify them according to their truth values: true, false, or neither false nor true, based on their content and metadata. As Ground Truth, we will use the verdicts contained in the claim reviews in ClaimsKG for all claims published by the fact-checking portals before 2022. These you may use for training your algorithm. Note that for the test set, you will not have access to any information contained in the respective fact-checking articles but to all other kinds of (meta)data present in ClaimsKG.All claims in the test set will be in English language.

(a) Use the SPARQL endpoint of ClaimsKG to retrieve all claims along with all relevant metadata that you deem useful for the task according to the ClaimsKG data model. Your code should show how data has been fetched.

(b) You may use data from ClaimsKG as features exclusively or fetch related data from additional endpoints. For instance, claim instances in ClaimsKG are annotated with DBpedia entities that are mentioned in the claim. Additional data about entities (e.g. categories, types) can be obtained from DBpedia. You are also free to add external resources such as pre-trained Word Embeddings or resources like WordNet.

(c) Prepare a description of your algorithm: which features would you like to use for the classification? E.g. you may use annotated entities but also other metadata (e.g. the author of a claim) and textual features. Use a machine learning algorithm of your choice. Explain the rationale behind your choices.

(d) Prepare the feature set(s) of your algorithm.

(e) Train your algorithm on the Ground Truth data.

(f) Prepare a script that outputs your classification results when applying your model in a CSV file. It must contain the following fields in the given order:

- claim ID
- claim text
- predicted label (TRUE | FALSE | NEITHER)
- optional: confidence score

(g) On **04.01.2022**, we will add the test data to the Gitlab repository. It will consist of claim IDs for you to retrieve from ClaimsKG. Make sure you implement the retrieval of all claim data via their IDs (ignoring information contained in claim reviews!) beforehand to not run late when the test IDs are released. You will also find an evaluation script in the repository. Fetch and update your local repository.

(h) Let your classification algorithm classify the test data with the model you trained on the training data, save the output and run the evaluation script.

We expect the git repository you submit to contain all code and dependencies, links to all resources you used or the resources themselves, the output files produced on the test data and the output of the evaluation script. Also supply a readme file that explains how your training data can be generated (including your SPARQL queries) and how your model can be trained and applied. You may use Python or Java with the libraries of your choice.

Evaluation Metrics

Your approach will be evaluated using the Accuracy metric. Thus, all samples will be weighted equally. The evaluation script will also compute macro-average Precision and Recall scores for your

information.

Presentation Guidelines

We will circulate more detailed information about the presentation guidelines closer to the presentation slot, also considering the number of students who subscribe to the home project. For now, we plan to allocate approximately 15 minutes (10 minutes for presentation and 5 minutes for discussion) per presentation. A good presentation should cover the following aspects:

- The general idea of your approach
- An overview of the architecture and different modules you implemented
- Any preprocessing or data cleaning steps
- Unambiguous description of used features for the prediction task
- SPARQL queries involved in the solution
- Your results: the performance of your algorithm and some interesting insights, e.g. for which cases does it work well, for which doesn't it?
- Observations, lessons learned and possible future improvements

Assessment and Rewards

A grade improvement of 0.3 is given when the following criteria are met: You provided a working solution to the problem and your results are reproducible. You submitted the solution in time and in the specified format and presented it in the dedicated DKE22 lecture slot. Your classification algorithm performs better than simple baselines. You will find the precise threshold to beat in the readme file of the git project.