ECS8052 Knowledge Engineering Module Assessment

Assignment Details

- The date for submission isas published on canvas.
- You will receive feedback within 10 working days of submission.
- **This assessment contributes 100% of the module mark.**
- This is an **individual assessment**. When submitting your assignment, you are agreeing to the following statement:

  I certify that the submission is my own work, all sources are correctly attributed, and the contribution of any AI technologies is fully acknowledged.

- Plagiarism is a serious offense. Please ensure you have read and understood the university policy on plagiarism. Lack of awareness of what is and is not acceptable will not be accepted as an excuse. Ensure that you use your own words even when referencing a source. Make sure libraries, tutorials, and code that you use are properly acknowledged. Indicate clearly where you have used AI assistants (including, but not limited to ChatGPT, Claude, Gemini, Copilot).

- Keep a copy of all submitted coursework, including all code and data files used during the submission.

- Remember to back your work up regularly using the School's Gitlab service or Microsoft OneDrive (available to you for free as a QUB student). Loss of work due to computer failure will not be accepted as a valid reason for late or non-submission.

Assignment Brief

In this assignment you will complete a project in which you will apply the techniques studied in the course to the **PrimeKG** dataset.

PrimeKG is the *Precision Medicine Oriented Knowledge Graph*. It contains information about thousands of diseases, their effects, their genetic associations, and the drugs that can be used to treat them. PrimeKB is fully described in a recent paper [Chandak 2023] and also at the PrimeKB website and you should familiarise yourself with the main concepts described in the paper and on the website. *You are not expected to understand all of the biological details*.

Further information about the deliverables and assessment criteria are found at the end of this document.

Getting started

- Create a single folder for your project (let's call this `ke-project` here).
- Inside `ke-project` create two folders `code` and `data`

- Download the following four files into the `data` folder
  - README.txt (3.1KB)
  - kg.csv (936.3MB). Describes the nodes and edges in the graph, together with some annotations.
  - disease_features.tab (108.3MB). Annotations for nodes in the graph corresponding to diseases.
  - drug_features.tab (9.6MB). Annotations for nodes in the graph corresponding to drugs.
- Create a Python Virtual Environment for this assignment. You can do this by running the following command in `ke-project`:

```
python -m venv ./env
```

This which will create a folder `env` in `ke-project` in which a local, fresh, python installation will be created. In this environment, you can maintain correct versions of the necessary libraries for your submission. You will be required to submit a list of the necessary dependencies. Since this will be a clean environment, you will need to install all necessary libraries into the new environment (including `jupyter` etc).

- Activate the environment (`source env/bin/activate` on Mac/Linux; `env\bin\activate.bat` on windows).
- Install Jupyter into the new environment (`pip install jupyter`)
- Create a single Jupyter notebook in `ke-project/code` which all of your code should be written.
- If you use a procedure that requires a random seed, please use your student number as the random seed. Your code should give the same results each time it runs.

Part 1: Data Exploration

In this section, you will explore some basic properties of the dataset.

The file `kg.csv` fully describes the structure of the graph. Each row describes one edge ('relation') in the graph. The column headings are as follows:

Column Heading	Description
`relation`	Type of relation.
`display_relation`	Short display name for relation
`x_index`	Unique ID of first node.
`x_id`	ID of first node in original source.
`x_type`	Type of first node.
`x_name`	Name of first node.
`x_source`	Original database in which first node

Column Heading	Description
	was found.
`y_index`	Unique ID of second node.
`y_id`	ID of second node in original source.
`y_type`	Type of second node.
`y_name`	Name of second node.
`y_source`	Original database in which second node was found.

Load `kg.csv` and answer the following questions.

1. How many different types of relation, and how many of each type of relation are there?
2. How many different types of nodes, and how many of each type are there?
3. Are the values obtained for the previous two parts consistent with the values given in Chandak 2023? If not, can you suggest why, and can you verify whether your suggestion is correct?
4. For each type of node, sort the nodes of that type alphabetically by name, and list the index and name of the first three nodes in the sorted list.
5. Produce a table that shows how often each type of node is in each type of relation.

Part 2: Exploring the Knowledge Graph

In this section, you will construct the annotated knowledge graph and explore a sub-part of the graph in depth.

1. Construct and visualise an ontology for the graph that shows the different *types* of nodes and the relations between them. All nodes and edges should be clearly labelled.
2. Using the example from the lectures as a template, construct the knowledge graph using the data in `kg.csv`.
3. Select two nodes in the graph that correspond to *diseases*. The nodes should not be directly connected, but should be separated by no more than three relations/edges. Extract the subgraph containing all paths between the two nodes that traverse up to six edges. Visualise the graph. *You may find it helpful to select a starting node with a small number of outgoing relations. For visualisation purposes, you may limit your visualisation to show only 100 paths.*

Part 3: Deriving a Knowledge Base and Inferring New Relations

1. Convert the subgraph extracted in part 2.3 into a rule based system such that each directed relation in the graph from a node x to a node y is represented as a rule `if the node is x then the node is connected to y`. You do not need to list the rules explicitly in the report.

2. Use forward chaining and breadth first search to infer new relations that are not explictly present in the subgraph. Select three of these relations and for each one, map out the chain of reasoning that enables the inference to be made.

Part 4: A Bayesian View of the Data

1. Design a Bayesian network to model the joint distribution of anatomical region, protein, disease, and drug. Your network should use conditional distributions that can be derived directly from the data. If you need to make any assumptions, please state these clearly.
2. Select a drug and use your network to infer the probability that it is associated with each anatomical region. Are your findings consistent with what is known about the drug from its node attributes (or other sources)?

Deliverables

You should submit the following files packaged in a single `zip` file:

1. A report, in PDF format, of no more than 3000 words.
2. A single Jupyter notebook containing all of the code needed to reproduce your results.
3. A `requirements.txt` file that can be used to recreate your environment.
4. A ten-minute video recording in which you briefly present your work.

These files should be named, respectively:

1. `{fname}_{sname}_{student_id}_report.pdf`
2. `{fname}_{sname}_{student_id}_code.ipynb`
3. `{fname}_{sname}_{student_id}_requirements.txt`
4. `{fname}_{sname}_{student_id}_video.mp4`

where `{fname}` is replaced with your first name (e.g. `Iain`), `{sname}` is replaced with your surname (e.g. `Styles`) and `{student_nr}` is replaced with your student ID number (e.g. `40123456`). The first name and surname should match exactly the name used on your Canvas account.

In accordance with university policy, assessed work submitted after the deadline will be penalised at the rate of 5% of the total marks available for each calendar day late up to a maximum of five calendar days, after which a mark of zero shall be awarded.

The Report

You must submit a project report of **no more than 3000 words**. This comprehensive report should cover all aspects of the project. It should contain four sections corresponding to the four parts of the assignments.

In each section, you should demonstrate your understanding of the task, of the methods used, and of the technical details of the solution. You should present your results using graphical visualisations and/or tables where appropriate, and provide your interpretation of the results. Your report should contain references to any external sources you have used to support your work.

Your report should: * Demonstrate your understanding of the task and of the methods you have used to solve it. * Explain any decisions or choices you have made, justifying these with reference to external sources if appropriate. * Provide your interpretation of the results and finding. * Describe any difficulties you encountered and explain how you overcame them.

Your report should include a cover sheet that includes your name, student ID, and the word count of your report.

Penalties for exceeding the word count follow the University Guidance and are as follows:

Length	Penalty
+10%	no penalty
+>10% - 20%	10% penalty
+>20% - 30%	20% penalty
+>30% - 40%	30% penalty
+>40% - 50%	40% penalty
+>50%	maximum mark of 50%.

Text contained in figures, tables, titles, references, code snippets, and cover page is not included in the word count.

These deductions are applied to the mark for the assignment.

Deductions will not lower the assignment grade below the passing threshold.

The Notebooks

You should submit **one Jupyter notebook**. The notebook should contain all code used during the assignment. The code should be structured according to the four parts of the assignment presented **in order** with clear headings in markdown cells used to indicate the purpose of each block of code. The notebook does not need to include detailed commentary on results and findings, but it should contain sufficient comments to enable the purpose of each clock of code to be easily understood.

Your code should assume that the data files will be in `../data/` relative to the notebook.

The Requirements

You must include a `requirements.txt` file (noting the naming requirement listed above) that can be used to recreate your environment to ensure that your code can be run. You can create this within your project virtual environment using the command

```
pip freeze > requirements.txt
```

To test and evaluate your code, the assessors will create a new Python virtual environment for each submission and install the required dependencies using the command `pip install -r requirements.txt`. Please ensure that this works before submission.

The Video

You should submit a 5-minute video presentation in which you summarise the methods you have used and the main findings of your project. Your presentation should include one introductory slide that lists your name and student ID, one slide for each of the four sections of the assignment, and one concluding slide (six slides in total). The presentation does not need to cover everything you have done; instead it should highlight the most important aspects of the problem, your approach, and your findings.

The production quality of the video is not important and it is sufficient to record a Teams session in which you share your screen.

Mark Scheme

Marks will be allocated as follows

- Part 1: maximum of 25 marks
- Part 2: maximum of 25 marks
- Part 3: maximum of 25 marks
- Part 4: maximum of 25 marks

For each part, the following marking criteria will be used:

Criterion	Marks available
Demonstrating a sound understanding of the problem	4 marks
Description and justification of your solution	8 marks
Evaluation and analysis of results and findings	7 marks
Quality, clarity, and efficiency of code	3 marks
Coverage in Video Presentation	3 marks

Submissions that include non-running code (after installing necessary dependencies from `requirements.txt`) will receive a 20 mark deduction.