COMP34711 Natural Language Processing
Coursework specification (v1)

Due: 6pm on Friday November 24th, 2023

You are asked to design, implement and test solutions for two NLP tasks. For each task, you will submit to Blackboard:

- source code (a single Google Colab notebook for each task), named using your student ID and task ID;
- answers to an online questionnaire that describes your solutions;
- files with the results for the test cases, strictly prepared and named in the format requested.

You will be provided with the questionnaires and test cases 24 hours before the deadline for each of the tasks. Before that, you will be provided with training data that you will use for the development.

This is an individual assignment, so you should solve all the tasks on your own. You are not permitted to collaborate with other students on this coursework. You must not use any generative agents (such as ChatGPT) to provide the answers to the test cases – this is not the learning objective of this coursework and will be considered as cheating. You can, however, re-use publicly available existing code or libraries from ***NLTK***, ***gensim***, ***Hugging Face, PyTorch, Tensorflow/Keras*** and ***scikit-learn*** as long as you clearly and transparently report their use both in your code *and* in the questionnaire for each of the tasks. In any case, your solution should be sound and robust, following best practice in software engineering, including providing sensible comments, developing efficient, re-usable and modular code, avoiding unnecessary processing and/or data storage. Re-using existing bad code cannot be used as an excuse for not providing a professional-level solution – so be careful what code you reuse.

The majority of topics for this coursework have been covered in class, the homework exercises and the textbook (in particular Chapter 6 for Task 1 and Chapters 9 and 10 for Task 2). In the lab support sessions (in weeks 8 and 9, on Tuesdays at 10 am), you can seek advice on how to use a particular framework or machine learning library. However, GTAs will not be able to help with the solution design or the choices you are making, or to check the "correctness" of your solutions.

Marking

You have a number of options for your coursework solutions: three for each of the tasks. The mark will depend on how much effort you have put in the coursework and the quality of the outcomes (test cases; code; questionnaire).

Marking of the test cases will be done *automatically*, by comparing your results with a "gold" standard reference data. Note that this coursework is not intended to have a single correct answer in terms of what is developed or how. We should all acknowledge that, and we will make sure that the marking takes that into account too.

For a pass mark, you would be expected to implement one approach for each of the tasks. There would be potentially some issues with the implementation, and your output on the test dataset or the questionnaire may not be complete. The results would be ranked low in comparison to other solutions.

For a 2.2 mark, you would implement one approach for each of the tasks, and the outcomes would be of a reasonable quality. There may be some issues with the code and the outcomes, but nothing major. The results would be ranked lower in comparison to other solutions.

For a 2.1 mark, you would implement one approach for one of the tasks and two for the other. Your implementation would be efficient, and your outcomes would be of a good quality. You would have followed best practice in software engineering. Your questionnaires would be complete, and the results would be ranked higher in comparison to other solutions.

For a first-class mark, you would implement two approaches for each of the tasks. Your implementation would be robust and efficient, following best practice in software engineering. Your outcomes would be of a very good quality. Your questionnaires would be complete and comprehensive, and the results would be ranked high in comparison to other solutions. For a high first-class mark, your solutions would take care of difficult cases and your results would be at the top of the ranking.

All suspected plagiarism cases will be taken seriously and passed to the Department for further action.

Submission deadline

The deadline specified above is a hard one: extensions can only be granted by the Department as a result of DASS or formally processed Mitigating Circumstances (rather than by the lecturers or GTAs). If you have an extension, please get in touch during the lab support session to make appropriate arrangements. Marks for late submissions will be reduced in line with the university's policy, again by the Department and not by the lecturers. See details at:

http://documents.manchester.ac.uk/DocuInfo.aspx?DocID=4271
http://documents.manchester.ac.uk/display.aspx?DocID=24561

You should start submitting your work well before the deadline – remember that you will need to fill in a questionnaire as part of the submission, and that may require 30 minutes or so.

Changes to this document

If a need arises for any changes or clarification in this document, a new version will be uploaded to Blackboard and an announcement made via the Discussion forum (please make sure you are subscribed to it).

Data specification

For this coursework, we will release a relatively large corpus of movie plot synopses. A plot synopsis is a narrative summary of the plot of a movie. Some of the plot synopses in our dataset are human-made and reflect real-world movies, while some have been generated automatically by a generative language model. While the teaching team has taken all reasonable steps to review the automatically generated documents, not all of them have been checked for content, and – although unlikely – some of them might have inappropriate references or disturbing scenarios. All documents provided here are intended for automated model training and evaluation, rather than for human consumption, so you are not expected to read and 'manually' review any of the documents. Both types of synopsis have been randomly distributed in all subsets of the corpus that will be shared with you.

Each movie plot has its ID, and is split into a movie title (*title*) and the narrative (*plot_synopsis*). Each plot is also tagged with one or more tags (labels) that reflect the genre i.e. characteristics of movie plots (e.g. *comedy, murder, romantic, sci-fi*, etc.).

The corpus is split into three datasets.

Training dataset (to be used in both Task 1 and Task 2) consist of over 8,000 documents merged in a single CSV file. The first three columns are *ID*, *title* and *plot_synopsis*, and then there is one column for each of the 9 labels (*comedy, cult, flashback, historical, murder, revenge, romantic, scifi, violence*): "1" in these columns mean that the plot is labelled with that column label, whereas "0" means it is not.

ID	title	plot_synopsis	comedy	cult	flashback	historical	murder	revenge	romantic	scifi	violence
6416fe15-6f	Shattered Vengeance	In the crime-ridden city of Tremont, renowned investigative journalist, Amelia Williams, wages war against an influential criminal syndicate. The Syndicate, led by the ruthless mafia boss, Victor Russo, smuggles drugs and extorts businesses, instilling fear throughout the entire city. Amidst the chaos, Amelia's investigative reporting catches the attention of Jared Bennett, a former Special Forces operative haunted by his personal demons. Amelia, driven by her desire for justice, covertly gathers vital evidence against the Syndicate, documenting their illegal activities in a clandestine exposV©. However, her secret investigation is abruptly cut short when Amelia is found brutally murdered in her apartment. The loss devastates her loyal intern, Julia Evans, who becomes determined to avenge her mentor's death and complete the mission Amelia started. Upon hearing the tragic news, Jared, who had developed feelings for Amelia, is consumed by a thirst for revenge. Haunted by his past failures, Jared pledges to help Julia expose the Syndicate and bring justice to Amelia's killers. With their hearts set on vengeance, they form an unlikely alliance, setting in motion a thrilling tale of violence, romance, and redemption. As Jared and Julia delve deeper into the treacherous underworld of Tremont, they discover an unexpected ally in the form of Anthony Marino, an undercover cop who has infiltrated the Syndicate. Together, the trio infiltrates a high-profile underground casino run by the Syndicate, hoping to	0	0	0	0	1	1	1	0	1

For each of the tasks, you will use the *title* and/or *plot_synopsis* columns that represents the move plot titles and synopses (i.e. documents). For training in Task 2, you will need the labels attached to each synopsis.

Development dataset (for Task 2 only) consists of over 1,000 documents in the same format as the training dataset. You will use this dataset only to optimise your classifier for Task 2 – do not use this dataset for Task 1. There will be a separate dataset of examples for Task 1 that you can use to optimise your work.

Test datasets are described in the specification below.

COMP34711 Natural Language Processing – Task 1: Distributional semantics

The task is to use different vector representations to estimate the similarity between two terms. Applying what you have learnt on lexical processing and distributional semantics, you should obtain (up to) **two representations** from the following three options:

a) a sparse representation (BoW with either tf*idf or PPMI);
b) a dense static representation (either word2vec or GloVe);
c) a pre-trained contextual representation (either BERT or RoBERTa).

The first two representations should be learnt from the training corpus provided, and the third should make use of a pre-trained model.

To evaluate your solutions, you will be given a set of term pairs and you will return their **cosine** similarity based on each of the representations you have developed. For a high mark, you should consider building a representation for **multi-word** test terms (e.g. '*big desk*') too.

You should experiment with the different settings – e.g. with the size and type of context, with pre-processing, etc. You should naturally aim to report the solutions that look as best in capturing the similarity.

Training data

- Please only use the training corpus (not the development corpus). The text of the documents is in the *plot_synopsis* column, which you could use along with or without the *title* column. (The documents will also have class labels, which you are welcome to ignore for this task.)
- We will provide a CSV dataset with ~150 example test pairs and their gold standard similarities so that you can experiment with your solutions. The similarities provided in the examples are the average of human ratings of the similarity between given two terms on scale 0 to 10 (0 meaning that they are not similar at all or even opposite, and 10 meaning that they are synonymous). Several hundred human annotators rated each of these pairs; their ratings were then averaged, and these numbers are the one provided in the examples. Your similarity calculation is **not** expected to provide such values (as you are relying in cosine similarity), but you can use these as an indication as whether the similarity you have provided is consistent with what the human ratings suggested: for example, two terms for which you have returned a high similarity of 0.9 should have a higher gold standard rating than two terms for which you have provided a lower similarity of 0.5. We will provide a scoring script for you to help with that.

Test data

The test data will contain a set with term pairs in the same format as the example test pairs (apart from similarities), for which you should provide separate sets of results: one for each representation you have developed. Note that some of terms may be multi-word terms, and some of them may be ambiguous or rare.

Results submission

You should submit a separate CSV file for each method implemented, named *your_student_ID-Task1-method-x* (where x is a, b or c). Do not submit the results from more than two approaches.

When submitting your results for the test cases, you should submit similarity for each test term pair in the following format:

term_pair_id, similarity

Please use the term pair IDs as provided in the test dataset.

Ranking

The ranking of your results will be based on how well your output corresponds to the gold standard explained above, that contains "manually" evaluated similarity of each term pair. The ranking will be based on the consistency between the gold standard pair ranking and the one predicted by your methods.

Questionnaire

When submitting your coursework, you will be asked a series of questions to describe your solutions in a structured way, including the links to any public code and libraries you have used. For example,

- for the sparse and static dense representation: you will be asked as what model you have implemented, what kind of tokenisation and pre-processing you have applied (if any); what type and size of context you have used; how long it took to calculate the similarities for the test dataset (run-time), etc.

- for the pre-trained models: you will be asked what model you have used; some specifics of the model (e.g. the number of parameters, the vocabulary size, the input size, etc); how long it took for the test dataset to be processed for this representation (run-time), etc.

The questionnaire will be released together with the test dataset, and may take 15 minutes to fill in.

Code

When submitting your coursework, you will be also asked to submit a single Google Colab notebook with all the implementations for this task. Please name it using your student ID (*your_student_ID-Task1*).

COMP34711 Natural Language Processing – Task 2: Text classification

The task is to implement, train and evaluate a multi-label text classifier that assigns document-level labels to each document in a corpus. As the documents, you will use the title and/or plot synopses from the movie dataset. There are nine labels in total (*comedy, cult, flashback, historical, murder, revenge, romantic, scifi, violence*) and each document has at least one label assigned to it. You are expected to develop (up to) **two solutions** out of the following three:

a) Developing a "traditional" classification method (either SVM or Naïve Bayes classifier);
b) Developing a "traditional" deep learning method (either LSTM or bi-directional LSTM);
c) Fine-tuning a pre-trained model (either T5, BART, BERT or RoBERTa) for the classification task.

You should experiment with the different hyperparameters (e.g. the size (or partition) of input you will pass to your model). You should naturally aim to report the solutions that look as best on the validation dataset.

Training and development data

- Please use the training corpus provided for training your classifiers.
- Use the development dataset only for validation of your classifiers – do not train the final classifier on the merged training and development datasets.

Test data

The test data will contain a large number of documents that you will be expected to process. The test documents will be in the same format and with a similar distribution as the training and development data.

Results submission

You should submit a separate CSV file for each method implemented, named *your_student_ID-Task2-method-x* (where x is *a, b* or *c*). Do not submit the results from more than two approaches. When submitting your results for the test cases for a given method, you should submit a label vector of length 9 for each test document along with its unique ID. The i^{th} component in the label vector should be "1" if the document is classified as belonging to $class_i$, or "0" if not. Specifically, the format should be:

doc_id, $label_1$, …, $label_9$

Note: please do not change the order of classes and associated labels ($class_1$ = *comedy*; $class_2$ = *cult*; $class_3$ = *flashback*; $class_4$ = *historical*; $class_5$ = *murder*; $class_6$ = *revenge*; $class_7$ = *romantic*; $class_8$ = *scifi*; $class_9$ = *violence*) as that will obviously affect your results.

Ranking

Your results will be compared to the gold standard labels that have been produced for the test dataset by human annotators. The ranking of your results will be based on two aspects: how well your output predicts labels for each document (the document level) and how well it predicts labels for specific classes (the class level). We will use averaged weighted **F-measure** in both cases. For the document level, if too many labels are provided for a given document, the F-score will be adjusted to focus more on precision; otherwise, we will give more weight to recall. Therefore, you may want to control the number of labels returned for each document. For the class level, we will give more weight to the classes that are likely more difficult to predict (e.g. smaller classes). We will provide a validation script that will return your results at the document-level as well as for each class.

Questionnaire

When submitting your coursework, you will be asked a series of questions to describe your solutions in a structured way, including the links to any public code and libraries you have used. For example,

- for the traditional classification methods: you will be asked to report on any pre-processing you have applied; what features and weights you have used; how large the vocabulary was; how long it took to train the classifier and how long it took to apply the model to the development and test datasets (on Google Colab). You will be also asked to report the results on the development dataset (average document-level precision and recall; precision and recall for each of the classes – all by applying the script that will be provided);

- for the deep learning models: you will be asked to report any pre-processing you have applied; what representation you have used; how many hidden layers your architecture has, etc. You will be also asked to report how long it took to train the classifier and how long it took to apply the model to the development and test datasets (on Google Colab), along with the results on the development dataset (average document-level precision and recall; precision and recall for each of the classes – all by applying the script that will be provided).

The questionnaire will be released together with the test dataset, and may take 15 minutes to fill in.

Code

When submitting your coursework, you will be also asked to submit a single Google Colab notebook with all the implementations for this task. Please name it using your student ID (*your_student_ID-Task2*).