Module Code
COM00106M

MEng, MMath, ACS Degree Examinations 2018-19

DEPARTMENT OF COMPUTER SCIENCE

Natural Language Processing (NLPR)

Open Individual Assessment

Issued: Spr/10/Mon (11th March 2019)
Submission due: Sum/1/Wed 12 Noon (17th April 2019)
Feedback and marks due: 22 May 2019

All students should submit their answers through the electronic submission system: http://www.cs.york.ac.uk/student/assessment/submit/ by Sum/1/Wed 12 Noon (17th April 2019). An assessment that has been submitted after this deadline will be marked initially as if it had been handed in on time, but the Board of Examiners will normally apply a lateness penalty.

Your attention is drawn to the section about Academic Misconduct in your Departmental Handbook: https://www.cs.york.ac.uk/student/handbook/.

Any queries on this assessment should be addressed by email to Suresh Manandhar at suresh.manandhar@york.ac.uk. Answers that apply to all students will be posted on the VLE.

Your exam number should be on the front cover of your assessment. You should not be otherwise identified anywhere on your submission.

Instructions

- You will be asked to choose a Natural Language Processing problem related to the contents of the course.
- You should prepare a report on the task and code to carry out the task. More details of these elements are given below.
- If necessary, you may be asked to give a presentation and demonstration of your code to a marking panel.
- The programs you write and the write-up should be submitted electronically through the web page http://www.cs.york.ac.uk/submit **as a single zip file**.
- Your code should be written in Python 3. If you plan to use any other language then this needs to be communicated to the module lecturer and agreed in advance.
- **Code submission:** All code necessary to run your implementation **must** be submitted.
- **Data submission:** The data necessary to run your code should be submitted. If the size of the dataset prevents it from being uploaded then you can submit a link to the dataset. **Preprocessing:** If you are preprocessing the data e.g. stemming and stopword removal, parsing, then you ONLY need to submit the data that has been preprocessed. You need not submit the raw unprocessed data.
- **The Report:** You need to submit a report that contains at least the following sections:
  - Introduction – describing precisely the general problem you are solving and the particular sub-problem that you are solving.
  - Literature review – describing the current work on the topic and its limitations. The literature review need not be extensive but has to be very focused and requires depth.
  - Problem modelling – that provides a (mathematical) model of the solution you have developed highlighting the key differences with existing models. The models that have already been covered in the NLPR module can be described briefly (without doing the full derivations). Focus needs to be on the new models you have derived.
  - Implementation – describing the data structures and high-level structure of the code. Provide command line instruction(s) to test the code.
  - Evaluation and results – describing the evaluation measures and datasets you have used and comparative analysis of your results when compared with the results of existing approaches.
- The maximum length of the report is 10,000 words EXCLUDING references. The title

page of the report must include a declaration of the word count and how that count was derived. For example:

> Number of words = 6000, as counted by the Word word count command.

- **Problem modelling** A key aspect of the assessment is the modelling of the problem. To receive full credit the report must describe 1. how the models were derived and 2. how the models are different from existing models.
- **Code:** It should be possible to test the code using standard libraries and approved machine learning libraries. Python scikit-learn, tensorflow and Keras are recommended libraries. If you plan to you use any special libraries, please discuss this with Suresh Manandhar in the first instance. To get further marks you need to write programs that are:
  1. efficient in both time and space;
  2. well documented; and
  3. use the most appropriate methods for the given problems.

Marking scheme

- Literature review: 20 Marks.
- Problem modelling: 20 Marks.
- Implementation write-up, documented code, high-level understandability and correct functioning of code: 40 Marks.
- Evaluation and comparison : 20 Marks.

The Tasks

Students are free to choose one of the tasks. When multiple students choose the same task, these will be allocated on a first-come first-served basis. Variants of the same task may be chosen by different students. The details of the allocation are available from the NLPR website. The list of tasks available are given below. Depending upon student feedback some additional tasks may be made available in the NLPR VLE module page. Slides describing the tasks will be available in the NLPR module page.

1. Deep learning for QA

The aim of this task is to take advantage of current developments in deep learning and apply this to a question-answering (QA) task. There are plenty of references in ACL and the web on utilising word embeddings for building QA systems. Many variations are possible such as, QA using background knowledge base, reading comprehension, factoid and non-factoid QA .
Key challenges:

- Understand current papers in question answering systems
- Understand key limitations of these methods
- Implement an existing method
- Demonstrate its effectiveness on one existing dataset. There are many around such as WebQuestions, SQuaD, TriviaQA, TREC QA, BaBI etc.
- Suggest an improvement
- Demonstrate it on the same dataset.

2. Aspect based Sentiment Analysis

Aspects are specific features of a product. For example lens, flash etc. can be seen as aspects of a camera. For aspect based sentiment analysis, one needs to extract both the sentiment and the aspect. The aim is to develop a deep learning based pipeline for aspect based sentiment analysis. Evaluation can be done using the SemEval 2016 and SemEval 2017 ABSA (Aspect based sentiment analysis) task dataset.
Key challenges:

- Develop a word/sentence embedding based method for aspect based sentiment analysis based on existing papers
- Extend the methods to cover additional subtasks as decribed in ABSA e.g. identifying aspect spans in the text, identifying sentiment words/phrases in the text etc.
- Demonstrate its effectiveness on the SemEval 2016/2017 Aspect Based Sentiment Analysis (ABSA) task.

3. Mixture models for Morphology learning

Implement a finite/infinite mixture model for morphology learning. Extend the approach developed for morphology learning in the lectures to cover multiple segmentations.

Key challenges

- Understand finite/infinite mixture models from a mathematical perspective.
- Understand current work in morphology learning.
- Develop a mixture model for morphology learning. You can build upon the approach developed in the lectures or develop your own approach.
- Extend the approach to cover multiple segmentations.
- Test it using the MorphoChallenge datasets for English and Turkish.

4. Relation extraction using distant supervision

Relation extraction using weak supervision in the form of distant supervision is a popular technique for aligning knowledge bases with text. It is also a popular method for relation extraction. The purpose of the project is to evaluate current developements in the field and suggest improvements.
Key challenges

- Understand distant supervision as a framework for weak supervision
- Understand current work in relation extraction using distant supervision
- Understand current work in knowledge base population
- Implement an existing method
- Suggest an improvement
- Test it using existing datasets.

5. Evaluating and extending existing methods for sentence representations

Current sentence representations such as BERT and Tree constrained relation networks generate sentence level representations that are better suited for textual entailment tasks such as FEVER and SQuaD datasets. The aim in this task is to critically evaluate these embeddings and use in a pipeline for a specific entailment task.
Key challenges

- Critically understand existing sentence representation methods
- Understand existing entailment datasets

- Develop a simple pipeline for textual entailment tasks using an existing sentence representations eg. BERT
- Evaluate on a dataset e.g. FEVER

6. Discrete embeddings for knowledge base completion

Discrete embeddings have the advantage that they are hashable and can be used as memory addresses. The aim of this task is to develop a simple knowledge base completion system that employs discrete embeddings for the entities and relations.
Key challenges

- Understand existing methods for learning discrete embeddings
- Understand existing knowledge base completion methods
- Implement a knowledge base completion method using a discrete embedding
- Test it on existing knowlege base completion tasks

7. Deep learning models for Word sense induction (WSI)

Combine clustering and word2vec type algorithms for learning word senses. Learn senses of multiple words simultaneously to learn shared senses. Evaluate on SemEval 2010 WSI dataset.
Key challenges

- Understand existing work on word embeddings.
- Understand existing work on deep neural networks.
- Use the SemEval 2010 WSI dataset for evaluating WSI using LDA.
- Provide comparison with simple Cosine similarity.
- Extend evaluation to include context based on dependency parse tree context.

End of examination paper