FACULTY OF INFORMATION TECHNOLOGY CTU IN PRAGUE

Review report of a final thesis

Reviewer:	Ing. David Šenkýř
Student:	Vigneshwar Manoharan
Thesis title:	Using Neo4j DB system to store and query linguistic pattern
Branch / specialization:	Web and Software Engineering
Created on:	January 31, 2022

Evaluation criteria

1. Fulfillment of the assignment

▸ [1] **assignment fulfilled**
[2] assignment fulfilled with minor objections
[3] assignment fulfilled with major objections
[4] assignment not fulfilled

2. Main written part

70 / 100 (C)

The scope of the thesis is sufficient, and all parts are informationally valuable. However, the State-of-the-art section describes various disciplines of natural language processing that are out of the scope of the thesis. I positively evaluate Section 4 about the required configuration of used tools. On the other hand, there is no need to mention the IDE used.

Additionally, I have the following remarks.

Factual issues/inaccuracies
— Section 2.1.3: Stanford CoreNLP is Java-based, not Python-based, as the author stated. However, there is a Python portation called Stanza.
— Section 2.1.3: How does the author know that Stanford CoreNLP provides 'tremendous scalability'? There is no citation or explanatory comment.
— Section 2.1.3: 'Spacy is the NLTK tool's next stage.' – they are two standalone frameworks.
— One of the mentioned NLP frameworks is called 'spaCy'. The author should use the official name and, most importantly, be consistent in using the name. The same goes for Neo4j, Postman, and Bolt.
— Section 2.3.1: The first version of TEMOS was written in Java and used Stanford CoreNLP. The current version is written in Python and uses spaCy. The author mixes these two statements.
— Section 3.1: 'Thus, if I query Neo4j from Python through an interface, I will have access to a TEMOS tool and will be able to receive a Query...' – confusing, the opposite is true.

TEMOS accesses the created tool of this thesis via a defined interface.
— Fig. 5.3 and Fig. 5.4: A screenshot of one class from an IDE is not a class diagram that I expect in a master's thesis.

Typographic and language aspects
— Basic formatting should be definitely improved. Repeatedly, spaces before the parenthesis are missing, spaces around commas and dashes are missing (e.g. p. 13, p. 21). There are issues with capital letters of general words in the middle of sentences (e.g. Section 2.1.2). Moreover, this is inconsistent even in the same sections.
— There is an issue with subsections numbering containing zero: 3.2.0.1, 3.2.0.2, and 3.2.0.3.
— Regarding typographic aspects: some lines are overflowing, the correct dash symbol is not used, and inconsistent styles of listings are present.
— Names of sections are inconsistent: title-case vs. sentence-case vs. upper-case. The section name should not contain the colon symbol (Section 2.3.1.2).
— The programming language keywords should be formatted differently than the standard text.
— Code samples in the text are not numbered; therefore, they are difficult to reference.

Citation
— The names of cited authors should be cited precisely.
— As in the main text part, the basic formatting (spaces) can be improved, and the style should be consistent.
— Not all citations are complete (missing ISBN of [10], etc.).
— The placement of sentence citation sometimes does not follow the citation practice (e.g. p. 31).
— The source of Fig. 3.7 is not cited.

Further comments and recommendations follow.
— In Section 1.2, there are not yet defined terms ‘text’, ‘pos’, ‘dep’, and ‘head’.
— In a technically oriented work, I expect to work with the vector graphic format of figures, where possible. Moreover, when the original figure uses the vector graphic format.
— The custom hand-drawn shapes in Fig. 3.8, Fig. 5.10, Fig. 5.17, Fig. 5.19, Fig. 5.21, Fig. 5.22, and Fig. 5.24 do not look professional.
— The content of Fig. 2.8, Fig 3.3, Fig 3.4, Fig. 5.2, Fig. 5.8, Fig. 5.9, Fig. 5.11, Fig. 5.12, Fig. 5.13, Fig. 5.25, and 5.27 is not readable. The images are too small.
— Fig. 3.4 is not referenced in the text.
— Tables should not be presented as bitmap images (Fig. 3.9, Fig. 5.6, Fig. 5.7, Fig. 5.14, and Fig. 5.16).

3. Non-written part, attachments 87 /100 (B)

The implementation consists of two Python classes and one supporting class for API. I positively evaluate a class with tests. However, the names of test methods should be more informative (not ‘test__sentence1’).

4. Evaluation of results, publication outputs and awards 75 /100 (C)

The main objective of storing and querying patterns was fulfilled. The student implemented a solution to a pattern matching problem using Neo4j. No other aspects of

querying (e.g. mapping parts of patterns in response) are discussed. As the student mentioned in the Future Work section, the current solution is not designed for concurrent access typical for databases.

The overall evaluation 75 /100 (C)

Generally, the presented approach is interesting. I believe the main written part should be written more carefully. The problem of concurrent access should be solved before the solution can be deployed in practice.

Questions for the defense

Do you have an idea how to extend your solution with concurrent database access?

Instructions

Fulfillment of the assignment

Assess whether the submitted FT defines the objectives sufficiently and in line with the assignment; whether the objectives are formulated correctly and fulfilled sufficiently. In the comment, specify the points of the assignment that have not been met, assess the severity, impact, and, if appropriate, also the cause of the deficiencies. If the assignment differs substantially from the standards for the FT or if the student has developed the FT beyond the assignment, describe the way it got reflected on the quality of the assignment's fulfilment and the way it affected your final evaluation.

Main written part

Evaluate whether the extent of the FT is adequate to its content and scope: are all the parts of the FT contentful and necessary? Next, consider whether the submitted FT is actually correct – are there factual errors or inaccuracies?

Evaluate the logical structure of the FT, the thematic flow between chapters and whether the text is comprehensible to the reader. Assess whether the formal notations in the FT are used correctly. Assess the typographic and language aspects of the FT, follow the Dean's Directive No. 52/2021, Art. 3.

Evaluate whether the relevant sources are properly used, quoted and cited. Verify that all quotes are properly distinguished from the results achieved in the FT, thus, that the citation ethics has not been violated and that the citations are complete and in accordance with citation practices and standards. Finally, evaluate whether the software and other copyrighted works have been used in accordance with their license terms.

Non-written part, attachments

Depending on the nature of the FT, comment on the non-written part of the thesis. For example: SW work – the overall quality of the program. Is the technology used (from the development to deployment) suitable and adequate? HW – functional sample. Evaluate the technology and tools used. Research and experimental work – repeatability of the experiment.

Evaluation of results, publication outputs and awards

Depending on the nature of the thesis, estimate whether the thesis results could be deployed in practice; alternatively, evaluate whether the results of the FT extend the already published/known results or whether they bring in completely new findings.

The overall evaluation

Summarize which of the aspects of the FT affected your grading process the most. The overall grade does not need to be an arithmetic mean (or other value) calculated from the evaluation in the previous criteria. Generally, a well-fulfilled assignment is assessed by grade A.