Automated similarity detection: identifying duplicated requirements *PEC1 - Work plan*

Quim Motger de la Encarnacin

Universitat Oberta de Catalunya
Universitat Politecnica de Catalunya

1 Introduction

This document is an exhaustive description of the scope and the work methodology plan for the master thesis titled *"Automated similarity detection: identifying duplicated requirements"*. The following sections aim to present a general depiction of the work plan, starting from the topic proposal and refinement until the elaboration of the final thesis and its defense.

To achieve this purpose, this deliverable is structured in two main chapters. First of all, the development of the thesis topic. This section provides a general description of the project proposal, which includes the main topic, the scope of the project, its motivations, the areas of interest, and the expected results from an academical point of view. Based on this description, it also presents a detailed list of both general and specific objectives, which will be used as the general guidelines for the tasks carried out during the thesis development.

Second of all, it is necessary to address the details about the work development methodology and the schedule plan, using the previous details - such as the specific objectives - to extract concrete tasks and evaluate its development from a management perspective. Taking this plan as a reference, it is also necessary to identify possible risks that may affect the success of some of the project goals, as well as to propose mitigation techniques for these risks.

This thesis is being developed as a final master thesis inside the *Computer Engineering Master's degree* program, and therefore it is planned and developed following its quality and evaluation guidelines and deadlines.

2 Thesis topic

In this section a general description of the thesis project and the goals of the project are introduced.

2.1 General description

Artificial Intelligence (AI) is a wide-known computer science area which has experienced an exponential growth both in the research field and in real use-case applicability. This computational representation of human cognitive knowledge can be used in many different areas of application, according to the features and the main goals of these fields. Two of the most known areas of AI are **Machine Learning** (ML) and **Natural Language Processing** (NLP).

In: A. Editor, B. Coeditor (eds.): Proceedings of the NLP4RE Workshop, Essen, Germany, 18-21 March, 2019, published at http://ceur-ws.org

On the one hand ML is a "general purpose technology"[1] which uses data and information to learn real-world related knowledge and to improve the reliability of a specific action. Its main application is to use this acquired knowledge to extract autonomous predictions about partial observations of this data. In essence this systems or algorithms differ from traditional programming schemes due to the results accuracy improvement based on its experience.

On the other hand NLP has a large potential in different applications involving automated, computational process of all kinds of documents and textual items. This technology applies to the task of developing partial representations of features and rules of natural language based on its textual information, which includes both syntactic and semantic knowledge [2]. The main purpose of this technology is to use this representative knowledge in order to apply automated analysis and generation of text units, such as comprehensive sentences or full documents.

One of the areas of application of AI - and to be specific, NLP and ML - is the **Requirements Engineering** (RE) field. RE is the set of activities and processes of software engineering focused on the development, analysis, communication and management of a set of requirements that describes the features of a system [3]. Software development experience in recent years proves that managing and maintaining large sets of requirements have become critical issues. This problem is even more challenging due to the large amount of data and the dimensions that these projects are dealing with nowadays [4]. Whether the analysis and the evaluation of requirements is a tedious, time-consuming task, it is critical that they are carried out with both accuracy and efficiency in any software development project.

Between the main problems of RE we can highlight the detection and **management of duplicated requirements** [5]. If ignored, these duplicated items may lead to redundancy in the textual information of a project and therefore this may lead to the duplicity of tasks, which are critical issues from the project management perspective. Moreover, the automation of this process and the standardized usage of specific, accurate tools are still at a state-of-the-art stage. It is difficult to find open source tools and frameworks providing generic, adaptive solutions for duplicate detection, and most of them are addressed to a very specific casuistic or use case [6]. Moreover, similarity detection algorithms are highly tightened to the quality of the data used for the detection process [7].

This is the starting point of this master thesis: **an analysis of the state-of-the-art proposal to apply automated requirements similarity detection**, using artificial intelligence techniques, for the detection of duplicates between project requirements. Based on a deep research of the state-of-the-art, this master thesis must be both a portfolio and a practical evaluation of real duplicate detection algorithms and scenarios in software engineering project requirements.

The details about the scope and the goals of the project are depicted in the following sections.

2.2 General objectives

Listed below are the global objectives (GO) of the project.

GO1. **To research the state-of-the-art of the requirements similarity detection field.** The final thesis document must include a summary of the collected information related to the application of similarity detection in the RE field. The achievement of this goal guarantees the necessary input knowledge to propose and develop the algorithmic tool to evaluate a real duplicated requirements detection scenario. Therefore it is required this literature review is thorough and exhaustive.

GO2. **To develop a requirement similarity-detection multi-algorithm tool.** The main core of the developed software must be the technical implementation of the algorithms selected to be validated. This development must satisfy usability and evaluation requirements, so that it is possible to apply an evaluation and extract conclusions from the achieved work.

GO3. **To define and to evaluate a real application use case for duplicated requirements detection.** As part of the Horizon 2020 OpenReq project, this project will be tested against a real

use case dataset from a company - a set of requirements including duplicated and not duplicated data. Therefore one of the main goals must be setting the experiment and its preparation, so the data can be tested with our tool. It is crucial that the experiment's execution provides tools and techniques to extract empirical results to evaluate at the end of the project, allowing to conclude the thesis with considerations and validated knowledge on the performance and accuracy of the algorithms.

2.2.1 Specific objectives

Using the list of global objectives as an input, the specific objectives (SO) are listed in the following list.

SO1. To study the current status of similarity detection in the RE field from a general point of view.

SO2. To review and to enumerate similarity detection techniques/algorithms, and to be specific the ML and NLP techniques that represent the state-of-the-art of the field.

SO3. To identify potential suitable algorithm candidates for the master thesis and the use case to be validated with.

SO4. To elaborate a development proposal for the implementation of PoC algorithms.

SO5. To integrate the algorithms with a unique tool to use and to test the different similarity detection scenarios.

SO6. To evaluate the requirements input data of the algorithms, in order to guarantee a comprehensive analysis of the results.

SO7. To optimize and to adapt the algorithms based on the use case requirements.

SO8. To analyze and to prepare a use case dataset for all scenarios (i.e. similarity detection algorithms).

SO9. To carry out a minimum, qualitatively acceptable number of experiments using the developed algorithms.

SO10. To perform a deep comparative analysis between algorithms.

SO11. To extract conclusions in terms of reliability of the results and performance of the algorithms.

2.3 Thesis structure

3 Work plan

In order to achieve the specific objectives of this thesis, this section introduces the details of the development methodology of work. It also includes the list of tasks to carry out and a schedule proposal for its achievement according to the deadlines of the project. For this purpose, a risk identification and mitigation proposal is also introduced.

3.1 Development methodology

This thesis will be developed following a **Kanban-based methodology** with some general influences from Scrum, which has been proven to be a good approach for projects of this kind of nature [9]. This decision can be justified with three main reasons.

First of all, given the nature of the project, it seems suitable to propose an agile software development methodology to achieve the goals depicted in this document. Although this project starts from a clear,

specific stage, and the objectives are detailed enough, the research of the state-of-the-art phase will deeply condition the specific tasks that will be done during the technical implementation and the evaluation process (i.e. the number and nature of algorithms). It is necessary to handle a certain flexibility in terms of requirements and tasks during the project's development.

Second of all, this methodology aims to provide results in short-term cycles by scheduling fine-grained tasks that guarantee the success of the project's objectives. Following one of the main guidelines of Kanban, which is the visibility and traceability of the tasks, it is intended to provide a dynamic framework that will allow to complete tasks in a short period of time by identifying, detailing and scheduling them according to the general schedule planning of the master thesis (see 3.2).

Finally, and following with this last criteria of cyclic, iterative results, one of the main goals of this methodology is to hold weekly retrospective and plan meetings, comparable to analog meetings from the Scrum methodology, with the thesis director. This meetings will allow not only a constant review of the work that has been done, but an iterative review of the remaining tasks and the different potential lines of work that might raise during the development of the project.

3.1.1 Project management activities

For achieving and applying the previous methodology plan, it is required to identify and describe the two main artifacts or activities that will guide the project's development.

- **Tasks backlog maintenance.** Based on the Scrum task backlog artifact and the task workflow suggestions from Kanban, a project tasks backlog will be used to identify, classify and organize the development of each of the tasks raised during the development of the project. The specific workflow implementation for this project will be as follows:

  To Do $\rightarrow$ **Analyzing** $\rightarrow$ **Doing** $\rightarrow$ **Reviewing** $\rightarrow$ **Done**

  This backlog must be maintained daily according to tasks progress.

- **Retrospective and plan meeting.** A regular weekly meeting is proposed to keep track of the general progress of the project. This meeting must include some of the general activities of both the retrospective meeting and the sprint plan meeting from SCRUM. Therefore, we set the length of each sprint to 1 week - which is a short period of time, but will be useful to guarantee the achievement of goals and deadlines as scheduled.

3.2 Stage description and tasks

A 4-stage project structure is proposed as follows.

- **Plan.** The main goal of this tasks are related to defining a project proposal and to define the main features of its development before starting to develop the master thesis itself.

  Tasks

  P1. To perform a first, generic research about the area of interest of the master thesis

  P2. To define the topic and scope of the project

  P3. To identify generic and specific goals

  P4. To describe and justify a work methodology

  P5. To elaborate a schedule planning, according to deadlines and available resources

  P6. To identify risks and to propose mitigation plans

P7. To structure the master thesis document and periodical deliverables

- **Research.** Includes all tasks related to the state-of-the-art review, as well as the study and the analysis of ML/NLP techniques, algorithms and technologies for the similarity detection field.

  Tasks

  R1. To research the state-of-the-art of similarity detection in RE field

  R2. To enumerate and to acquire a deep knowledge of the algorithms and techniques of interest for duplicated requirements detection

  R3. To review the state-of-the-art study and developed work from OpenReq project in similarity detection

  R4. To identify a set of potential similarity detection algorithms to be developed

  R5. To analyze and to compare the set of pre-selected algorithms

  R6. To select those similartiy detection algorithms or techniques for duplicated requirements to be implemented and evaluated

- **Development.** After the knowledge and the information extracted as a result of the *Research* phase, it is possible to start all tasks related to the technical development and implementation of the algorithms and the final software project.

  Tasks

  D1. To identify and to study specific NLP/ML technologies and frameworks used in the different algorithms

  D2. To implement PoC software for each algorithm and its techniques

  D3. To validate and to test the scalability of the algorithms for the general UC scenario

  D4. To implement stable, usable versions of the final algorithm candidates

  D5. To integrate the implementation of each algorithm with a generic tool for usage and evaluation purposes

- **Experimentation.** Finally the software project must be used to perform experiments based on a real use case scenario and to extract useful information about the performance of each algorithm.

  Tasks

  E1. To define a set of experiments (i.e. a set of requirements) to test the algorithms

  E2. To describe the measures and stats to use for comparison analysis, based on both an accuracy and a performance analysis

  E3. To run experiments and to collect results

  E4. To analyze collected data and to extract final conclusions

3.2.1 PEC submissions and content

In order to supervise and to audit the progress of the project, it is required to submit two follow-up reports prior to the final master thesis submission. These documents will be a guarantee of the achievement of the specific goals and tasks described in this document. The main goals of these reports are:

1. To detail the development of the project and its current state, identifying and justifying possible changes from the original goals.

2. To review the schedule planning and to justify possible deviation and mitigation activities, if any.

3. To deliver partial results from the developed worked.

The structure and the general goals of both reports are quite similar and therefore they do not require significant differentiations. However both reports must be an *As-Is* representation of the project status, and it is necessary to match the thesis plan with these deliverables. For this purpose, the content of each report is described below:

1. **Project development - phase #1.** *Deadline: November 18th, 2019*

   The first phase of the project development must include the following:

   - All tasks of the *Plan* stage finished
   - All tasks of the *Research* stage finished
   - Tasks D1-D2 from the *Development* stage finished or at a final phase

2. **Project development - phase #2.** *Deadline: December 16th, 2019*

   The second phase of the project development must include the following:

   - Tasks D3-D5 from the *Development* stage finished or at a final phase
   - All tasks from the *Experimentation* stage finished

If there are any goal changes or schedule deviations during the development of the project related to the tasks achieved for each report, these changes will be reflected and properly justified in the documents.

3.2.2 Gantt chart

Introduce Gantt chart for stage development.

3.3 Risk management

In this section we identify potential risks that might be raised during the development of the project and a proposal of mitigation activities for reducing the impact of these risks in the achievement of the goals described in 2.2 and 2.2.1.

R1. **Deviation of original schedule plan.** It is possible that the estimated time resources for each tasks may not always be accurate enough, leading to a delay in the completion of tasks and as a consequence a delay on finishing stages on time to keep the project on track.

Mitigation. As a preventive measure, to elaborate a stage plan assigning timing resources in a preventive way, dedicating a ratio of extra hour per task and a time period at the end of each phase to correct possible delays with respect to the original plan. If this is not enough, to propose a prioritization of specific tasks that will not compromise the achievement of the main goals of the project.

R2. **Unexpected obstacles in developing a similarity detection algorithm.** The most critical part of the project is all problems related to the implementation and development of similarity detection algorithms identified during the *Research* phase. It is possible that some of these algorithms are difficult to develop due to different causes, like a lack of knowledge of a specific technology, or the raise of issues or requirements not identified during the state-of-the-art analysis.

Mitigation. Analyze in iterative cycles (i.e. during retrospective meetings) the algorithms development viability and evaluate if necessary the possibility to discard or to change the requirements of the project.

R3. **Unexpected results or difficulties in UC evaluation.** It is possible that an algorithm's execution leads to unexpected problems for evaluation purposes, such as a bad estimation of the required resources for its execution. These resources can be conceived as technical resources (i.e. enough RAM) or human/time resources (i.e. its development is more complex than expected, or a single execution requires too much time to be evaluated.

Mitigation. As a preventive measure, dedicate specific effort during the state-of-the-art analysis to understand the nature of the algorithms and to identify the requirements for its execution. If this is not enough and the risk is materialized, an alternate approach is to reduce the size of experimentation and try to apply a scalar analysis for the real UC application.

References

[1] Brynjolfsson, Erik, and Tom Mitchell. What Can Machine Learning Do? Workforce Implications. Science. American Association for the Advancement of Science. https://science.sciencemag.org/content/358/6370/1530.

[2] Collobert, Ronan, Jason Weston, Len Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa, Natural Language Processing (Almost) from Scratch, Journal of Machine Learning Research, http://www.jmlr.org/papers/volume12/collobert11a/collobert11a.pdf.

[3] Jeremy Dick, Elizabeth Hull, Ken Jackson, Requirements Engineering, Springer, pp. 79, ISBN 978-3-319-61073-3

[4] Kasauli, Rashidah, Grischa Liebel, Eric Knauss, Swathi Gopakumar, and Benjamin Kanagwa. Requirements Engineering Challenges in Large-Scale Agile System Development - IEEE Conference Publication. https://ieeexplore.ieee.org/document/8049141.

[5] Natt och Dag, J., Regnell, B., Carlshamre, P. et al, A Feasibility Study of Automated Natural Language Requirements Analysis in Market-Driven Development - Requirements Eng (2002) 7: 20, https://doi.org/10.1007/s007660200002

[6] Tung Khuat, Nguyen Hung, Le Thi My Hanh, A Comparison of Algorithms used to measure the Similarity between two documents. International Journal of Advanced Research in Computer Engineering & Technology (IJARCET), https://pdfs.semanticscholar.org/43f8/027780d2694331ca373c57f9a2ace509a7b6.pdf

[7] Yu Huang, Fei Chiang, Refining Duplicate Detection for Improved Data, http://ceur-ws.org/Vol-2038/paper3.pdf Quality

[8] Requirements Engineering - Tools and Solutions Offered by OpenReq, OpenReq, https://openreq.eu/.

[9] Kniberg, Henrik, Mattias Skarin, and David Anderson. "Kanban y Scrumobteniendo lo mejor de ambos." Prlogo de Mary Poppendieck & David Anderson. ESTADOS UNIDOS DE AMRICA: C4Media Inc (2010).