Toward Human-Like Summaries Generated from Heterogeneous Software Artefacts

Mahfouth Alghamdi, Christoph Treude, and Markus Wagner
• News headlines.
• Abstract of research papers.
Automatic Text Summarisation.

- Goal of the automatic text summarisation is “to take an information source, extract content from it, and present the most important content to the user in a condensed form” [1]

- Applications
  - Social networks
  - Software engineering data

Approaches for Automatic Text Summarization.

**Input**
- Single-document
- Multi-document

**Output**
- Extractive
- Abstractive
Facts about GitHub’s statistics (from Oct 1st, 2017 to Sept 30th 2018)

https://octoverse.github.com/
Why Should We Summarise Software Engineering Data?

• To gain a comprehensive understanding of the contribution and productivity of an individual developer [2].
  • Help the new developer to integrate quickly into existing software project development.
  • Help the manager to improve the productivity of the developer.

Generating Summaries From Software Artefacts: A Challenge Task.

• Summaries produced from multi-document which contains heterogeneous software artefacts.

• First step toward our ultimate goal:
  • Understanding the characteristics of the students’ summaries.
    • can help us to solve a subset selection problems

• The characterization of these summaries will guide the search in at least two ways:
  • In a single-objective formulation.
  • In a many-objective optimisation problem.
Characteristics of Students’ Summaries.

• We collected a total of 545 human-written summaries produced on a weekly basis by 53 students from 15 GitHub projects.

• All the summaries were analysed using 27 features related to readability metrics, lexical features, and information theoretic entropy.

• The students’ summaries were grouped by courses, weeks, and teams.
For inspecting and visualizing our 27-dimensional characterisation, t-distributed Stochastic Neighbour Embedding (t-SNE) [3] was used.

To facilitate the interpretation, we added (before employing t-SNE) to each grouping the respective Euclidean average as each group’s centre.

Results: characteristics of the student summaries based on text features grouped by courses.

- Course #2 taught to graduate students (Non-industrial projects)
- Course #1 and course #3 taught to undergraduate students (Industrial projects).
Results: characteristics of the student summaries based on text features grouped by teams.
Results: characteristics of the student summaries based on text features grouped by weeks.
Our approach of utilising t-SNE to interpret the students’ summaries data at different grouping levels using the 27 features allows us to identify summaries that can serve as “gold standard” summaries.

We will use these to evaluate our future work on extractive summarisation techniques.
Next steps.

• Sentences in software artefacts that are close to the students’ summaries.

• Cosine similarity as a similarity measure
  • Texts similarity (bag of words with term frequency)
  • Features similarity (in 27-dimensional space).

• Optimisation methods:
  • Brute-force search
  • Heuristic search
    • Greedy search
    • Local search (restricted, unrestricted, and unrestricted subset)
Preliminary Results: Issue title’s artefacts.
Preliminary Results: Issue title’s artefacts

Summaries produced by all algorithms for particular team in weeks 1

ALGO#1: Brute-force, ALGO#2: Greedy, ALGO#3: RLS restricted, ALGO#4: RLS unrestricted, and ALGO#5: RLS unrestricted subset
Next steps.

• There are 15 types of software artefacts we are interested in.

• Measure the similarity between the text in issue title and the students’ summaries for a given time window.
  • Students’ summaries grouped at different levels (weeks, teams, and courses)

Contact us:
Mahfouth Alghamdi, Christoph Treude, and Markus Wagner
givenname.familyname@adelaide.edu.au