Clippy: smart PDF reader for better paper reading experience and knowledge mining

Sponsors

Xiaofei Xie - Singapore Management University, Singapore
Yuekang Li - Nanyang Technological University, Singapore
Yun Tang - Nanyang Technological University, Singapore

Project Abstract

Reading papers is an everyday task for researchers. This project aims to build a smart PDF reader to provide better reading experiences for researchers. This PDF reader should have two key features: First, it should be able to parse the PDF file to extract essential information (such as figures, tables, references, etc.) from the paper. Second, it should be able to mine knowledge from the extracted information to help with paper comprehension and inspire new research ideas.

Project Description

Each team should develop a PDF reader (based on PDF.js) according to the scope presented in the next section. The core features are:

1. resolving and presenting cross-references inside a paper;
2. building a knowledge graph for the references of a paper;
3. generating/mining summarizations for key components of the paper; and
4. (optionally) highlighting used, topic-specific phrasings.

Project Scope

This project aims to develop a PDF reader for a better paper reading experience. We suggest developing the PDF reader based on PDF.js (https://mozilla.github.io/pdf.js/) to avoid reimplementing the PDF parsing logic and basic UI components. In addition, it is relatively easy to add new UI components to PDF.js to fulfill the needs of the features.

The main functionalities of the paper reader are:

1. **resolving and presenting cross-references inside a paper**

   When we read a paper, we often encounter this issue: A paragraph refers to a figure, but they are on different pages. When we read the paper, we will need to switch between the pages, which is quite troublesome. Similar cases apply for tables, citations, etc. To solve this problem, we propose to resolve the cross-references and display the corresponding subject (figure, table, citation, etc.) in a pop-up window or sidebar when the user hovers his/her mouse cursor over the cross-reference item (or when the user clicks the item but this will override the default behavior of PDF.js, which is jumping to the page of the referred subject).

2. **building a knowledge graph for the references of a paper**

   A paper can cite many other papers, and the references often appear at the end of the paper as an independent section. The cited papers may also have citation/reference relations with each other. So this feature is to build a knowledge graph showing the citation relations of all the papers in the *references section* of the current paper. This can help to understand the relations between each work better.

3. **generating/mining summarizations for key components of the paper**

   A paper can have several key components: abstract, background, related works, an overview of technique, experiment baselines, answers to research questions, etc. The reader should be able to generate/mine summarizations for these key components and highlight the key sentences in the PDF. An example is the TLDR feature of semantic scholar (https://www.semanticscholar.org/product/tldr). The summarization can help to improve reading efficiency.

4. **(optionally) highlighting commonly used, topic-specific phrasings**

   A well-written paper should follow scientific writing styles. A paper can involve using many typical phrases/sentences for introduction, background description and experiment result analyses. Identifying and highlighting these sentences can help junior Ph.D. students or non-native English speakers to improve their writing skills.

Process Requirements

All development will happen in a public GitHub repository. Any agile-like process is accepted, as long as it is adequately explained and motivated. Furthermore, the team needs to employ practices of clean code.

Environmental Constraints

The application should be based on PDF.js (https://mozilla.github.io/pdf.js/). The analysis of paper references can involve the usage of the semantic scholar API (https://www.semanticscholar.org/product/api).

Project Restrictions

The code must be available on Github.

Project License

The software license must be the MIT License.

Level of Sponsor Involvement

Teams can contact sponsors via email:
Xiaofei Xie: xiaofei.xfxie@gmail.com
Yuekang Li: yuekang.li@ntu.edu.sg
Yun Tang: yun005@e.ntu.edu.sg

We can also arrange zoom meetings if needed.