Document Cards Implementation

Daniel Rossmann
01131558
daniel.rossmann@student.tugraz.at

David Seywald
00931000
d.seywald@student.tugraz.at

706.720 Visual Analytics VU SS 2018
Graz University of Technology

June 2018

Motivation and Goals

Due to the fact that large collections of research papers grow at a very high rate, it is very difficult for users to get an overview. Search machines for example display the title and some context. It is possible to browse through this search result, but they do not give the user a brief overview of the content.
Therefore a new approach has been invented, the so called Document Card. This representation helps to show the most important images and key terms on a single view. It amis a compact size, so that it is scalable to a large number of documents on different device sizes.

Figure: Concept of Document Cards

Process & Visualization

Our document cards implementation needs three extraction tasks to generate a .csv file out of an pdf-file input.

Key Term & Image Extraction/Transformation

For the extraction process we used common linux tools, which are freely available for most common distributions. The extraction process was done & tested on a Laptop running Linux Mint 18.3 (64Bit):

- Linux shell (bash)
- pdftotext - For text extraction from pdf files.
- pdfimages - For image extraction from pdf files
- pdfinfo - For metadata information extraction from pdf files
- node js (Version 8) - As Javascript runtime
- Natural - JavaScript library for natural text/language processing (https://github.com/NaturalNode/natural)

To start the extraction/transformation process, simple use our folder structure and tools from “preprocessing”:

- `\preprocessing`
  - `\in  - Put your pdf file(s) here`
  - `\out  - Output folder for temporary text & images`
  - `\node  - Nodejs Version 8 full`

Run the following command in your Linux shell:

```
.\clean.sh && .\extract.sh && transform.sh
```

After the process finishes, a file “visualizer.csv” will be created in the current folder, which can be loaded in the Visualizer Tool. Please do not change any data and choose all columns for visualization.

File Format

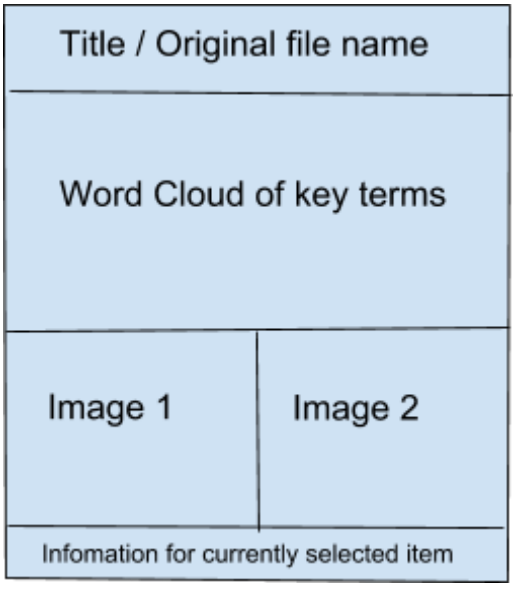
The main objective for the file format was to be simple and compatible with the Visualizer Webtool. Since we had to integrate images (png), we stored them as base64 encoded strings in one csv column.

File Column Name	Data Type	Description
document	string	original pdf file name
page	integer	pdf page of image
term	string	keyword / term
tf	integer	text frequency of the term
tfidf	number	text frequency / inverse document frequency measure
image	string	base64 encoded png image data
image_size	integer	image file size in bytes
key	string	pdf metadata entry name
value	string	pdf metadata value

Each entry results in its own row in the csv file. Not relevant column are set to *null*.

Layout

Our Layout was designed with simplicity and readability in mind. The main goal is to give the user a quick overview of the content/topic of the document.



Each element (term, title, image) is interactive and responsive. After clicking on an element, additional information is displayed in the Information area at the bottom.

The following Tools/Libraries where used in the visualization part:

- jQuery
- d3 - for simple selection & svg manipulation
- d3-cloud - A Word Cloud plugin (https://github.com/jasondavies/d3-cloud)
- d3-wordcloud - To make the Word Cloud simpler and easier to use (https://github.com/wvengen/d3-wordcloud)

Outlook

The following tasks can be done in future, to improve the current state of the visualization:

- Improve Word Cloud (scale text depending on tf-idf value)
- Improve Layout (dynamic number of images and key terms)
- Show multiple document cards in a single view
- Include full text snippets
- Highlight similar text/sections/images in other documents on interaction

References

1 "Document Cards: A Top Trumps Visualization for Documents”. https://ieeexplore.ieee.org/document/5290723 visited on 21 Mai. 2018.