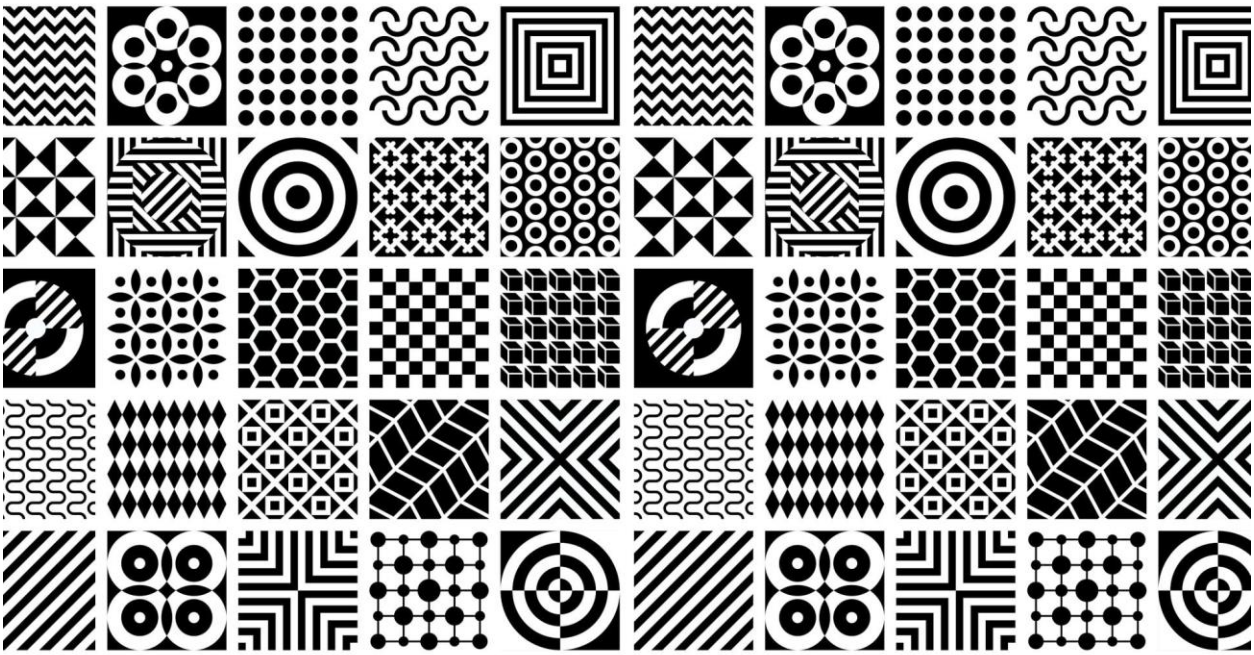
CS 410 Project Report

Reproduction of Paper "Generating Semantic Annotations for Frequent Patterns with Context Analysis"



Team TR Squirrels: Ye Xu, Weidi Ouyang, Raj Datta

INTRODUCTION

Our project focused on reproducing the paper "Generating Semantic Annotations for Frequent Patterns with Context Analysis" by ChengXiang Zhai et al (see reference 1.) We were interested in this paper since it has broad applicability covering a variety of data types. We were also enthused by the fact that the paper used multiple data mining techniques such as frequent patterns, clustering, and similarity functions. We believed that it would allow us to solidify our knowledge based on class materials while learning and exploring new areas as well (e.g. frequent patterns.)

The paper uses frequent pattern mining as a fundamental building block. The frequent patterns that occur in datasets, however, have to be interpreted to understand their relevance and semantic applicability. The goal of the paper is to create a way to attach (annotate) meaningful information to frequent patterns that lets us understand the frequent pattern better. The analogy of a dictionary is taken whereby words that are looked up, have a description, but also have examples and other related words presented to fully understand the looked-up word. The equivalent in the frequent pattern universe would then be to identify a) the definition of the frequent pattern by its context indicators, b) representative transactions with the frequent pattern, and c) semantically similar patterns. These specifics are extracted through a series of data mining steps and algorithms which are detailed in the paper.

We undertook the effort by first understanding the specifics of the paper, finalizing project scope, selecting appropriate tools and libraries, and implementing the desired result.

PROJECT SCOPE

When we evaluated the specifics of what the paper had accomplished, we realized that the full scope of the paper would require much more effort than what was expected for the team project and endeavored to refine the scope more clearly and realistically. In consultation and agreement with the lead TA, Bhavya, we decided on the following:

- To use only one dataset, not three as in the paper
- To implement only one of the two clustering algorithms for removal of title pattern redundancy.
- To implement the entire sequence of steps/algorithms necessary to extract the context indicators and define the given frequent pattern in the context units space (section 4.1 of paper).
- To leave extraction of representative transactions (section 4.2 of paper) and semantically similar patterns (section 4.3 of paper) as optional, to be implemented only if time allowed
- We would then match our results against the paper's results for the appropriate database we decide to use (either section 5.1, 5.2, or 5.3 of the paper.)

IMPLEMENTATION

We decided to choose the DBLP dataset for our implementation. Specifically, we started with DBLP50000 (see reference 2.) As in the paper, we decided to focus our efforts on the author data attributes for itemset frequent pattern mining, and on title data attributes for sequential frequent pattern mining. We also decided to implement the agglomerative hierarchical micro-clustering algorithm (mentioned as Algorithm 1 in the paper.)

We wrote our scripts in Python and chose NLTK for stemming and stop-words removal for titles, MLXend library for frequent pattern mining of authors and PySpark library for sequential frequent pattern mining of titles, as they are more modern Python-based libraries with good adoption in the industry. These are not the same libraries used when the paper was written (which may be somewhat outdated), but using the same algorithms. For the author pattern mining, we used the FPGrowth algorithm with some code sample adjustment for finding closed frequent patterns (see reference 3). For title pattern mining we used the PrefixSpan algorithm. For stemming, we used the Porter stemmer (as opposed to Krovertz stemmer used in the paper) due to our familiarity with NLTK and possibility of higher false negative rates in Krovertz stemmer. We used the default stopword removal (for English and German languages) in NLTK (the paper only mentions the fact that 12 words were removed without mentioning which words.)

Data acquisition and pre-processing

After we acquired the DBLP50000 raw dataset (with 50000 transactions), we parsed it and removed transactions that had no author names, which resulted in 49233 rows. We found that some of our computations (e.g. the hierarchical clustering algorithm is not scaled up well for large data such as the gigantic distance matrix) were taking too long due to the size of the dataset. We decided to take a smaller slice of raw data only from the year 2000 (based on recommendation from Bhavya, lead TA.) This reduced our dataset size to a manageable 4004 rows of transactions.

Obtaining Closed Frequent Patterns

After obtaining the clean dataset, the next step is to get the closed frequent patterns for authors and titles which become the fundamental building blocks of the rest of the paper's approach. Given that we had a smaller dataset, we decided to lower the support of author count to 4 (as opposed to 10 as in the paper) and obtained 14 such closed frequent patterns. And using the same support count (4) as in paper for title frequent patterns, we obtained 1912 frequent patterns (each title has multiple word sequences and hence a higher likelihood for patterns.)

As part of defining the building blocks, we also wrote scripts to find transactions related to author and title frequent patterns by building a reverse index of the transactions, which in turn helps downstream

computations related to clustering and building weight vectors of context units.

Clustering

We implemented the agglomerative hierarchical clustering algorithm with complete linkage, outlined as Algorithm 1 in the paper. We implemented the Jaccard distance measure as defined in the paper (Definition 9) for the purpose of clustering. After the visualization of the resulting dendrogram (Figure 1) and the elbow analysis of clustering iterations, we chose the threshold/cutoff of maximum depth at 0.01 to give 166 clusters. We tested a range of cutoff thresholds that gave different numbers of clusters around the elbow of the velocity curve where the clustering of branches slows down and examined the clustered words at these cutoff points. We felt that anything more than 166 clusters would have some redundancy, while less than 166 clusters would lack specificity and lose some information. Therefore, we chose 166 as our heuristic best.

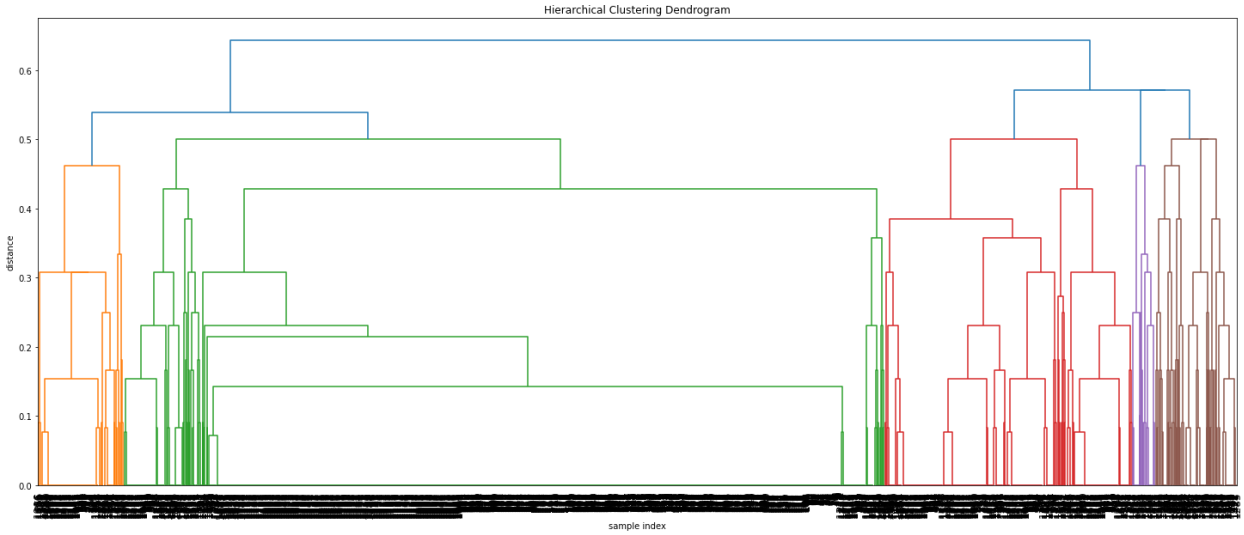


Figure 1: Dendrogram of Hierarchical Clustering of the 1912 sequential patterns of titles.

Weighting and Context Indicators

Weighting and the choice of context indicators is core to the context modeling part of the paper. For this, we chose the full set of our closed frequent patterns by combining both author and title frequent patterns into one pool of context indicators of 180 patterns. We then computed the mutual information between pairwise patterns as defined by paper and generated a 180 x 180 weight matrix of context indicators.

Optional Features

Having achieved suitable implementation covering 4.1 of the paper, we decided to implement the optional parts of identifying example transactions (section 4.2 of paper), as well as finding semantically similar patterns (section 4.3 of paper.) After implementation, we created one example of context annotation for a given author pattern, and one example of context annotation for a given title pattern. In each example, the given pattern had top 5 weighted context indicators as its definition, 5 most representative titles from transactions and 5 most semantically similar titles or authors as synonyms to verify the soundness of the implementation.

RESULTS ANALYSIS & CONCLUSION

Since we had chosen the DBLP dataset for processing, our target for results comparison became what is presented in section 5.1 of the paper, which simply presents examples of the patterns and its contextual definition, representative transactions, and some semantically similar patterns (SSP's). Since we had taken a data slice from only one year (the year 2000), we knew the results wouldn't be exactly the same as in the paper, but should be indicative of the power of the approach mentioned in the paper.

The following are the two examples generated from our project. As shown below, they are defining what the paper intended as contextual definition, representative examples, and similarly related patterns quite well. Having accomplished all 3 aspects (sections 4.1, 4.2, 4.3 of the paper), makes this more than what we had targeted in the scope of the project (since 4.2 and 4.3 were decided as optional.)

Results Example1: Context Annotation of An Author "Ralf Steinmetz".

Author	Definition	Representative Titles (top 5)	Synonym Authors (top5)	Synonym Titles (top5)
Ralf Steinmetz	Ralf Steinmetz	Domain Name Based Visualization of Web Histories in a Zoomable User Interface.	Sanjay Kumar Madria	user interfac
	virtuellen	Intelligent graphical user interface design utilizing multiple fuzzy agents.	Roberto Gorrieri	process descript
	Sanjay Kumar Madria	Realistic Force Feedback for Virtual Reality Based Diagnostic Surgery Simulators.	Thomas S. Huang	virtuellen
	workbench	Alignment and Correspondence Using Singular Value Decomposition.	Edwin R. Hancock	summari
	Edwin R. Hancock	Symbolic Graph Matching Using the EM Algorithm and Singular Value Decomposition.	Gerald Sommer	high speed

Results Example2: Context Annotation of A Title

Title	Definition	Representative Titles (top5)	Synonym Titles (top5)	Synonym Authors (top5)
virtual realiti	Sanjay Kumar Madria	Realistic Force Feedback for Virtual Reality Based Diagnostic Surgery Simulators.	receiv	Roberto Gorrieri
	Edwin R. Hancock	Editorial.	diagnost	Sanjay Kumar Madria
	Roberto Gorrieri	Multi-level transaction model for semantic concurrency control in linear hash structures.	debug program	Thomas S. Huang
	analysi access	An adaptable constrained locking protocol for high data contention environments: correctness and performance.	versu,Bharat K	Bhargava
	Thomas S. Huang	Blackboard Segmentation Using Video Image of Lecture and Its Applications.	analysi access	Bill Hancock

Overall, we feel that the results are very useful and are convinced that this can help in building intelligence in applications reliant on various types of data. What we have built can easily be scaled up for larger datasets (by providing more computing power), and tuned to the specifics of the application (by modifying a variety of parameters in the algorithms implemented.) This was a useful exercise for the team to understand the importance of the mining approaches we learnt in the course.

REFERENCES

1. Q. Mei, D. Xin, H. Cheng, J. Han, and C. Zhai. 2006. Generating semantic annotations for frequent patterns with context analysis. In *Proc. of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '06, ACM, pp. 337–346.

2. Source of DBLP50000: https://hpi.de/naumann/projects/repeatability/datasets/dblp-dataset.html
   Citation**: Frequency-aware Similarity Measures.**Lange, Dustin; Naumann, Felix (2011). 243—248.

3. How to Find Closed and Maximal Frequent Itemsets from FP-Growth | by Andrewngai | Towards Data Science

4. Project Github site: https://github.com/bearnomore/CourseProject

5. Presentation developed as Tutorial of Paper available on Github site: https://github.com/bearnomore/CourseProject/blob/main/Guidance%20of%20Reproducing%20the%20paper.pptx

6. Video of paper tutorial presentation and walkthrough of implementation details: https://mediaspace.illinois.edu/media/t/1_0x0ykbbk

APPENDIX: Setup Instructions

Instruction for Reproducing paper "Generating Semantic Annotations for Frequent Patterns with Context Analysis"

(Instructions are also available as a README file in the project github site (see reference 4.)

1. Overview of the github CourseProject

There are three folders, Datasets, PythonCodes and JupyterNoteBookDemo, in the repository. The PythonCodes and the JupyterNotebookDemo contain the same scripts but different file formats (py file and ipynb file). Within each of these two script folders, there are 6 folders giving the execution order for reproducing the paper using the DBLP dataset (paper section 5.1). All datasets imported and generated using the scripts are located in the Datasets Folder. The input and output paths of these Datasets files need to be changed if downloaded to your local computer. Except for the raw dataset "dblp50000.xml", all other datasets are generated by the scripts.

In addition, the final report, the link (https://mediaspace.illinois.edu/media/t/1_0x0ykbbk) to the video demo and the powerpoint slides of paper review and project introduction are also in the CourseProject repository.

2. Python Library and Packages

numpy, scipy, pandas, nltk, csv, os, mlxend and pyspark are libraries needed for running the srcipts. Except for pyspark, all libraries can be downloaded and installed through pip or conda, depending on your preference and execution environment. The installation of pyspark (and Spark) is a bit complicated and requires some environmental configuration and functional java of version 8.0 or above. Here is the link of the tutorial how to install pyspark/Spark on Windows system:

https://www.datacamp.com/community/tutorials/installation-of-pyspark

3. Script running instruction

3.1. Parse the raw data (Folder 1. RawDataParsing)

Download "dblp50000.xml" and run script "DBLP_raw_data_parsing.py" or "DBLP_raw_data_parsing.ipynb". This generates the dataset "DBLP2000.csv".

3.2. Build the Context Units Space (Folder 2. ContextModeling)

3.2.1. Find closed Frequent Pattern (FP) for Authors using FPgrowth algorithm in MLXtend Lib.

Run script "Author_FP_mining.py" or "Author_FP_mining.ipynb" to import "DBLP2000.csv" and generate the dataset "authorsFP2000_with_index.csv", which contains 14 closed FPs of authors and their transaction index in "DBLP2000.csv" (e.g. author "Edwin R. Hancock" is a closed FP and it showed in the

839th, 1119th, 1127th, 1204th and 1576th row of DBLP2000, its transaction index list is [839, 1119, 1127, 1204, 1576]).

3.2.2. Preprocess DBLP titles

Run script "DBPL_preprocessing_titles.py" or "DBPL_preprocessing_titles.ipynb" to import "DBLP2000.csv" and generate the dataset "DBLP2000_preprocessed_titles.txt". In this step, stop words are removed and the titles are stemmed.

3.2.3. Find Title sequential Pattern using PrefixSpan algorithm in PySpark

Run "titles_seqPattern_mining.py" to import "DBLP2000_preprocessed_titles.txt" and to find closed sequential frequent patterns from titles of DBLP2000. I had issue with configuration of Spark in Jupyter Notebook environment therefore no corresponding script in "ipynb" format was put in the "JupyterNoteBookDemo\ContextModeling" directory . However, the python script was executed successfully in the windows cmd of my laptop. The script "titles_seqPattern_mining.py" generates an output folder containing the pattern file "part-00000". Set the "part-00000" file to txt format.

Run "Title_sequentialFP_processing.py" or "Title_sequentialFP_processing.ipynb" to import "part-00000.txt" and generate the cleaned dataset "titlesFP2000.csv".

3.2.4. Find transaction index of title sequential patterns

Run "Find_transaction_index_of_title_FPs.py" or "Find_transaction_index_of_title_FPs.ipynb" to import "titlesFP2000.csv"and generate "titlesFP2000_with_index.csv", which adds the list of transaction index to each title pattern.

3.2.5. Reduce title FP redundancy by microclustering (hierarchical clustering)

Run "Hierarchical_clustering_titleFPs2000.py" or "Hierarchical_clustering_titleFPs2000.ipynb" to import "titlesFP2000_with_index.csv" and generate "titlesFP2000_final.csv". This script applies the hierarchical clustering with Jaccard Distance defined per paper and clusters 1912 title sequential patterns into 166 clusters. It chooses the most frequent pattern in each cluster as the "centroid" pattern to further build the context unit space.

3.2.6. Combine author FPs and title FPs to build the context units space

Run "DBLP2000_context_units_with_transaction_index.py" or "DBLP2000_context_units_with_transaction_index.ipynb" to import 'authorsFP2000_with_index.csv' and 'titlesFP2000_final.csv' and to generate the final context units dataset "DBLP2000_context_units.csv".

3.3. Define given frequent patterns using context units defined above (Folder 3. PatternDefinition)

3.3.1. Build weight vectors of FPs in the context unit space

Run "Weighting_function.py" or "Weighting_function.ipynb" to import ""DBPL2000_context_units.csv" and "DBLP2000.csv" and generate "Context_units_weights.csv". This script generates context vectors for all context units defined in 2.2 and builds a weight matrix between the pairwised context FPs. Each element of the matrix is the Mutual Information score between the context unit pair per definition in

the paper.

3.3.2. Annotate the given FP (e.g. an author) by context units with highest weights

Run "Defining_pattern_with_context_units.py" or "Defining_pattern_with_context_units.ipynb" to import "Context_units_weights.csv". In this step, we first pick an author from the author FPs and rank the weights of its context vector. The context units with top 5 weights are selected as the definition of this author and are saved as "author_annotation_example1.csv".

Similarly, we pick a title from the title FPs and rank the weights of its context vector, and save the context units with top 5 weights as "title_annotation_example1.csv".

3.4. Find representative titles of the given pattern (Folder 4. RepresentativeTitles2Pattern)

Run script "Find_representative_titles_to_pattern.py" or "Find_representative_titles_to_pattern.ipynb" to import "DBLP2000_context_units.csv", "DBLP2000.csv" and "Context_units_weights.csv". This script first generates the weight matrix of transactions (titles) in the context units space as the dataset "transaction_weights.csv", and then computes the cosine similarity between the transaction weight vectors and the given pattern weight vector (e.g. the same author and title chosen in 2.3.2). The similarity matrix of transaction to author FPs is saved as "similarity_scores_of_transaction_to_author.csv", and the similarity matrix of transaction to title FPs is saved as "similarity_scores_of_transaction_to_title.csv".

This script then generates the top 5 representative titles with highest similarity scores to the given author and to the given title pattern as dataset "rep_titles_author_example1.csv" and "rep_titles_title_example1.csv", respectively.

3.5. Find synonyms of the given pattern (Folder 5. Synonyms2Pattern)

Run "Find_synonyms_of_pattern.py" or "Find_synonyms_of_pattern.ipynb" to import "Context_units_weights.csv" and to compute the cosine similarity between the candidate patterns of similarity (e.g. all closed frequent patterns of authors) and the given pattern (e.g. the same author and title chosen in 2.3.2). Select the authors with the highest 5 similarity scores as the synonyms of the given author or title other than the author or title itself.

This script generates 2 datasets for synonyms of author pattern: "coauthor_to_author_example1.csv", "syn_titles_to_author_example1.csv", and 2 datasets for synonyms of title pattern: "syn_titles_to_title_example1.csv" and "syn_authors_to_title_example1.csv".

3.6. A final display of the context annotation of the given pattern (Folder 6. ContextAnnotation)

Finally, Run "Author_context_annotation_example1.py" (or "Author_context_annotation_example1.ipynb") and "Title_context_annotation_example1.py" (or "Title_context_annotation_example1.ipnb") to combine the output datasets generated in step 2.4, 2.5 and 2.6. This script builds the two examples of context annotation for the given author pattern and the given title pattern respectivley and fullfills the two experiments in paper section 5.1.