# AutoLibrary - A Personal Digital Library to Find Related Works via Text Analyzer

**Yichun Ren[1],¶ Jiayi Fan[2],¶ and Bingqi Zhou[3]¶**

UC San Diego, San Diego, California 92093, United States;

*Corresponding author email: [1]yir016@ucsd.edu, [2]jif087@ucsd.edu, [3]biz024@ucsd.edu

**Abstract**--When encountering scientific papers, it is challenging for readers themselves to find other related works. First of all, it is hard to identify keywords that summarize the papers to search for similar papers. This dilemma is most common if readers are not familiar with the domains of papers that they are reading. Meanwhile, traditional recommendation models based on user profile and collection data are not applicable for recommending similar works. Some existing digital libraries' recommender systems utilize phrase mining methods such as taxonomy construction and topic modeling, but such methods also fail to catch the specific topics of the paper. AutoLibrary is designed to address these difficulties, where users can input a scientific paper and get the most related papers. AutoLibrary solves the dilemma via a text analyzer method called AutoPhrase. AutoPhrase is a domain-independent phrase mining method developed by Jingbo Shang et al. (2018) that can automatically extract quality phrases from the input paper. After users upload the paper and select the fields of study of the paper, AutoLibrary utilizes AutoPhrase and our pre-trained domain datasets to return high-quality domain-specific keywords that could represent the paper. While AutoLibrary uses the top three keywords to search on Semantic Scholar for similar works at first, users could also customize the selection of the high-quality phrases or enter their own keywords to explore other related works. Based on the experiments and result analysis, AutoLibrary outperforms other similar text analyzer applications efficiently and effectively across different scientific fields. AutoLibrary is beneficial as it eases the pain point of manually extracting accurate, specific keywords from papers and provides a personalized user experience for finding related papers of various domains and subdomains.

**Keywords:** Automatic phrase mining, phrase mining, digital library, search engine, recommender system

## 1. INTRODUCTION

An existing dilemma when reading scientific papers is that it's difficult to identify the keywords that can summarize the paper. As a result, when one wants to read more about a certain topic by inputting possible keywords to search for similar papers on a search engine or digital library, he or she might not be able to find the papers they want. If the input keywords were too broad, they might find many unrelated papers in the search output. On the other hand, if the keywords were too specific, the output paper might contain the original paper and papers that have mentioned the same specific term but the broad domain or topics are different. This dilemma happens most often when one is not familiar with the domain or topic of the paper that he or she is reading. Nevertheless, computer science could assist people in extracting representative keywords from text via text analysis techniques.

Text analysis, also known as text data mining, aims at deriving information from the text (e.g., papers and magazines) by structuring raw data into structured data and interpreting the result. Phrase mining is a fundamental task of text analysis that extracts high-quality phrases from a text corpus. Instead

of traditional n-grams segmentation, phrase mining focuses on high-quality phrases that represent the input text expressly and thus improves the computational models of applications that require to find phrases with great interest and relevance in a specific domain, such as taxonomy construction and topic modeling [9] [10] [11]. Without an efficient phrase mining method, while nearly all the current digital libraries have recommendations for each document, they fail to catch the specific topics of the paper to recommend similar works. They usually use a traditional recommendation model which is built on user profile and collection data. The model is good at suggesting the papers that users would like to read but not the related works, which is what many researchers want [8].

To address the difficulty of manually extracting keywords from papers and poor recommender system for related work of other websites, we built a website called AutoLibrary where users could use it as their personal digital library to save their documents and could find similar papers for each input scientific paper. To achieve this goal, we decided to introduce AutoPhrase into this paper searching process [1]. While the user inputs a paper and specifies a domain, we first use AutoPhrase to extract quality phrases from the input paper. AutoPhrase is a phrase mining method created by Jingbo Shang. It minimizes the required human effort of other phrase mining methods and improves the result by using two new techniques. The first technique is Robust Positive-Only Distant Training and the second one is POS-Guided Phrasal Segmentation. Since it is hard to ensure the significance of quality phrases generated from a single paper to both the paper and its domain, we build a dataset that contains the quality phrases of different domains by running AutoPhrases on the corpora of each domain. After applying weight to the AutoPhrase results of a single document with our pre-obtained domain-specific phrases, we can rank the phrases again and filter out domain's unimportant phrases. Then by searching for keywords with the highest quality scores on Semantic Scholar, AutoLibrary scraps and displays the search result on its website. AutoLibrary also allows users to customize their search, such as manually adding keywords and changing the selection of keywords. It might also store users' searching history in their local machine so that they could quickly look back to papers that they read as well as their search results.

To summarize, the main benefit of AutoLibrary is that it eases the pain point of extracting keywords from papers of unfamiliar domains or topics and provides a customized user experience for scholarly paper searching. The usage of AutoPhrase for the text analyzer on scientific papers make our recommender systems outperform other recommender systems both in accuracy and in user-friendly. Our experiments on papers of different domains and result analysis support the conclusion.

## 2. RELATED WORK

As the knowledge base of human beings expands and the number of fields of study increases, there has been a growing need for more comprehensive and better organized digital libraries for academic papers. The unprecedented development of the Internet and computers made the realization of digital libraries possible and accessible by scholars and researchers all over the world.

In 1945, researcher Vannevar Bush proposed a conceptual digital device, called Memex[1], which could store huge bulks of books and documents and enable fast access to them. The device was inspired by the dilemma at that time when researchers were struggling to index printed documents. Though the device was not physically implemented, it was an influential concept for later hypertext systems development.

In 1956, Licklider, a pioneer of the Internet, started looking for ways to apply computer technology to improve libraries. Similar to Bush, he proposed to use the Internet to make library materials accessible to a wider range of people[2]. In the system that he called a procognitive system, Licklider listed the three components of it: a knowledge base, questions and answers. These components were the basis of modern digital libraries.

Though ideas and efforts had been aggregated in this domain, the first mature digital library was not invented until 1966. Education Resources Information Center (ERIC), sponsored by the Institute of Education Sciences of the United States Department of Education, contains the full text of a variety of publications, such as research papers, books and journals. Being an online collection, ERIC provides service to a large number of users and also accepts materials from them to enrich its database. Notably, it has a controlled vocabulary, the Thesaurus of ERIC Descriptors, which facilitates users to search for subjects that they are seeking.

As experiences and knowledge accumulate, the need for integrating and improving previous functionality and methodologies of digital libraries grew. In 2006, members of the DELOS Network of Excellence on Digital Libraries published the *Digital Library Manifesto*[3], which includes a

core idea of a digital library framework. The framework includes three types of systems: Digital Library, Digital Library System, and the Digital Library Management System. This framework brought intelligent discussions of the Digital Library Universe.

Nowadays, there are some very good digital library products in the world. Amongst them, one of the most widely used is Google Scholar, which is considered as hosting the largest scholar paper database in the world[4]. What really put it in the spotlight is its ranking algorithm. Google Scholar claims that its ranking algorithm is automatically based on multiple factors, instead of allowing users to select just one factor as most other libraries do. However, Jöran Bee and Bela Gipp studied its ranking algorithm and claimed that it gave an outstanding weight to the number of citations[5]. As a result, older papers with more citations are likely to be cited again and again, while leaving new academic papers out of searching results.

Like the recommender systems in other digital products, the recommendation models of current digital libraries use Library collection data, user profile, web server log file, etc. to recommend similar work, such as calculating user-to-user similarity and item-to-item similarity[8]. However, one major issue is that other recommender systems aim at providing user products they like, while the recommender system of digital libraries should provide related works for researchers to further study the specific domain. Such a recommendation model also does not employ the text information within each document.

Based on previous works on digital libraries, we position our website as a digital library system, since it allows users to interact with the Semantic Scholar collection, but doesn't preserve a database for scholar papers. In addition, we will mainly focus on improving the system by using state-of-the-art techniques to extract quality phrases from input papers and accurately recommend papers that they need.

# 3. DATASETS

The dataset we used to train our model is the arXiv dataset available on Kaggle. The dataset contains information about more than 1.7 million scholarly articles across STEM.

The way we incorporate AutoPhrase and the dataset is to use the latter as a positive-phrase pool for the former to train on. In order to get the positive pool, we extracted titles and abstracts from each paper to form our customized dataset to train our model.

The generated model will be saved and to make predictions on users' input files. On top of that, the prediction generated by AutoPhrase can vary according to the data that it was trained on. We decided to train AutoPhrase on three types of datasets, Table 1, generating three types of models: the model on all academic papers, models on specific domains and models on specific subdomains.

**Table 1:**
**Datasets**

| Dataset | Field of Studies | Example |
|---|---|---|
| All Papers | All fields | All |
| Domain | One field | Math |
| Subdomain | One subfield | Applied Mathematics |

# 4. METHODOLOGY

In this section, we explain how the recommender system of AutoLibrary works. First, it introduces the phrase mining method AutoPhrases with its two novel technologies created by Jingbo Shang. Second, it demonstrates how to utilize self-created domain datasets to apply weight to the results to improve the performance. Finally, it discusses how we overcome the anti-scraping technology of Semantic Scholar and display the information we collected on AutoLibrary. We use Django framework to build our website with those front-end and back-end methods.

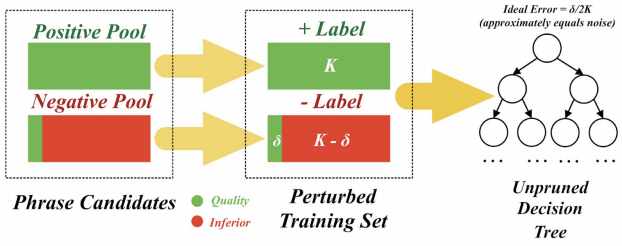
## 4.1 AutoPhrase

The AutoPhrase method [1] uses (1) robust positive-only distant training which generates positive labels samples from the domain's public general knowledge bases to replace the manual work of domain expert's work and (2) POS tags which utilizes basic language-specific knowledge in the process of phrasal segmentation so that AutoPhrase is applicable to different languages.

### *4.1.1 Robust Positive-Only Distant Training*

This technique uses general knowledge bases like Wikipedia to eliminate the need for domain experts to manually label candidate phrases with binary labels [1]. Candidate phrases from the corpus that match the high quality phrases from the general knowledge base are routed to the positive pool. The inferior phrases leftover are routed to the negative pool. However, because not all high quality phrases may be in the general knowledge base, some phrases could have been incorrectly routed to the negative pool, resulting in a noisy negative pool.

**Figure 1**
**Illustration of a Base Classifier**




Next we train an ensemble classifier that takes the average results of T independent base classifiers [1]. As shown in Figure 1, for each base classifier, K labels are sampled from each of the pools as a size-2K subset. In the sampled negative pool, there could be $\boldsymbol{\delta}$ wrongly labeled phrases that should have been positive, which is why it is a perturbed training set. To handle such a noisy negative pool, we train an unpruned decision tree because of its low training error. As long as a positive and negative phrase in the perturbed training set have different feature representations, accuracy will always be 100%. In this case the ideal error is $\frac{\delta}{2K}$ when there are $\boldsymbol{\delta}$ phrases out of the size-2K subset that were in the wrong pool, which is exactly the number of phrases that were incorrectly routed to the negative pool. We take the average from T independently trained ensemble classifiers as an additional measure for reducing noise. A phrase's quality score is defined as the fraction of these decision trees predicted the phrase to be a quality phrase [1]. Thus the ensemble error is defined as how likely more than half of the classifiers misclassify the phrase. As T becomes larger, the ensemble error will approach zero.

### *4.1.2 POS-Guided Phrasal Segmentation*

There are two parts to this process [1]. The first is to tag words with their POS (part-of-speech) and create a sequence of pairs out of the corpus, where a pair is $< w_i, t_i >$ and each pair is represented by a $\Omega$. Thus the corpus becomes a word sequence $\Omega = \Omega_1 \Omega_2 ... \Omega_n$ where each word is POS-tagged. The second is the phrasal segmentation process which builds on the pairs. This process creates $m$ segments out of the sequence, each segment separated using a boundary index sequence $C = \{c_1, c_2, ..., c_{m+1}\}$, where $1 = c_1 < c_2 < ... < c_{m+1} = n + 1$, and ith segment range from $\Omega_{ci} \Omega_{ci+1}$ *to* $\Omega_{ci+1-1}$ [1].

Unlike previous work [7] that punish the same-length phrase candidates in the same way, POS-guided phrasal segmentation measures the completeness of the phrases while considering the corpora's context by taking advantage of sentence structure since there are inherent divisions in sentences by POS where many phrases reside [1].

Once we have the segment's POS tag sequence $t$ and $ci$, which is the start index, we generate the end index $c_{i+1}$, the sequence of word $w[c_i, c_{i+1})$, and the indicator of if the word sequence forms a quality segment [1]. We then construct an algorithm that maximizes the following joint log-likelihood for a word sequence and boundary index sequence using the POS-Guided Phrasal Segmentation algorithm to return the ideal segmentation.

$$log\ p\ (\Omega,\ C) = \sum_{i=1}^{m} log\ p(c_{i+1},\ [w_{[c_i, c_{i+1})}] \mid c_g, g)$$

## 4.2 Weighted Quality Score

**Table 2:**
**Top 10 Quality Phrases of AutoPhrase Paper**

| | Before Weighting | | After Weighting | |
|---|---|---|---|---|
| **Rank** | **Phrase** | **Score** | **Phrase** | **Score** |
| 1 | information extraction | 0.9004 | knowledge base | 0.8734 |
| 2 | knowledge base | 0.8981 | information extraction | 0.8569 |
| 3 | domain specific | 0.8632 | domain specific | 0.8113 |
| 4 | text corpora | 0.8458 | text corpora | 0.7705 |

| | | | | |
|---|---|---|---|---|
| 5 | high quality phrases | 0.8303 | keyphrase extraction | 0.7199 |
| 6 | quality single word phrases | 0.8229 | pos tagger | 0.7127 |
| 7 | phrase mining | 0.7984 | natural language | 0.6910 |
| 8 | wikipedia article datasets | 0.7907 | massive text corpora | 0.6505 |
| 9 | dw ½ | 0.7896 | cn | 0.6310 |
| 10 | phrase quality | 0.7874 | auc | 0.6300 |

As mentioned above, AutoPhrase is built for extracting high-quality phrases from large text corpora and utilizes an ensemble classifier, a group of unpruned decision trees, to reduce the noise in the negative sample groups. The final quality score depends on the ratio of decision trees that mark the phrase as quality phrases. Thus its quality evaluations of phrases in a single document might be unreliable, since it could not generate enough decision trees. At the same time, the assessment based on POS tagging makes the entity names always have high quality scores, although some entities have low frequencies and are not domain-specific.

$$Quality_{weighted}(x) = Quality_{document\ i}(x) * Quality_{domain\ j}(x)$$

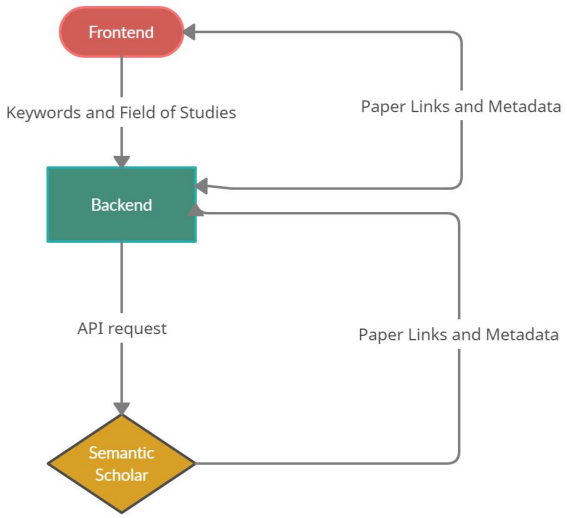
To handle the biased quality evaluation, we have to apply weight to the quality scores of the phrases again based on the phrases' corresponding quality scores in their domains. We have prepared a dataset that consists of AutoPhrase results of almost all the STEM (science, technology, engineering and mathematics) domains. Each AutoPhrase result is generated from corpora that contain thousands of abstracts and titles of scientific articles within the domain. Since the quality score is already standardized, ranging from 0 to 1, the most efficient way to apply weight is simply multiplying the quality scores of the same phrases for the input document x and its domain using the formula above. The weighted quality score not only could reassess the significance of the phrase according to both the input document and the domain, but also could filter the entities that are created by the document itself and would not contribute to finding other similar works.

### 4.3 Web Scraping

We decided to take advantage of the paper datasets and information of other well known scholar paper libraries, such as Google Scholar, arXiv and Semantic Scholar. We eventually chose Semantic Scholar because it provides good search results and contains a huge paper dataset, and it allows users to specify which field of studies to search from. Furthermore, we found a good API from the website which we can use to scrape data.

In our project, we set up a Flask backend that is responsible for accepting keywords and field of studies from the frontend, and use them to search for scholarly articles by using the Semantic Scholar API. The workflow can be seen in Figure 2.

**Figure 2**
**Workflow of the Application**




The data returned from Semantic Scholar is a json string, which consists of lots of metadata of papers. We extracted useful ones in our backend, such as dates, abstracts and titles, formulating them into another json string and returned to the frontend.

## 5. EXPERIMENTS

To test the performance of our digital library, we tested it with three other web-applications that have text analyzers. To formulate the comparison, we found eight papers from eight arXiv domain categories and applied AutoPhrase and the other three text analyzers to them.

## 5.1 Dataset

We selected eight papers, each of which corresponds to eight arXiv domains. In this way, we can test the performance of different analyzers across different domains. Table 3 shows their information.

**Table 3:**
**8 Papers Selected**

| Name | Domain |
|---|---|
| "Am I A Good Therapist?" Automated Evaluation Of Psychotherapy Skills Using Speech And Language Technologies [13] | Computer Science |
| AFFIRMATIVE ACTION IN INDIA VIA VERTICAL, HORIZONTAL, AND OVERLAPPING RESERVATIONS [14] | Economics |
| An End-To-End-Trainable Iterative Network Architecture for Accelerated Radial Multi-Coil 2D Cine MR Image Reconstruction [15] | Electrical Engineering and Systems Science |
| Softmax Policy Gradient Methods Can Take Exponential Time to Converge [16] | Mathematics |
| A roadmap for Feynman's adventures in the land of gravitation [17] | Physics |
| Fluid-solid interaction in the rate-dependent failure of brain tissue and biomimicking gels [18] | Quantitative Biology |
| Equilibrium Price Formation with a Major Player and its Mean Field Limit [19] | Quantitative Finance |
| Fairness in Credit Scoring: Assessment, Implementation and Profit Implications [20] | Statistics |

## 5.2 Compared Similar Applications

We found another three text analyzers to compare with AutoPhrase. They are the Jstor text analyzer, Webtools text analyzer, and MonkeyLearn text analyzer. Due to the characteristics unique to each analyzer, we used different measures to explore their results.

For AutoPhrase, we used models corresponding to papers' domains to get the weighted quality scores, as discussed in section 4.2. We then manually label the phrases as positive or negative quality-phrases with 1 or 0. The quality scores given by AutoPhrase and our manual labels were then used to draw the precision-recall curve. Note that for each domain, we only used the first 40 phrases to compare.

For the Jstor text analyzer, we viewed the extracted topics from the uploaded files as quality phrases extracted by the analyzer. Since it doesn't have a quality score, we tested the performance by computing the percentage of quality phrases amongst the extracted ones. We manually labelled them as positive or negative quality phrases, according to the relevance between the phrases and the topics of the papers. The percentage was then computed by taking the mean of the label vector.

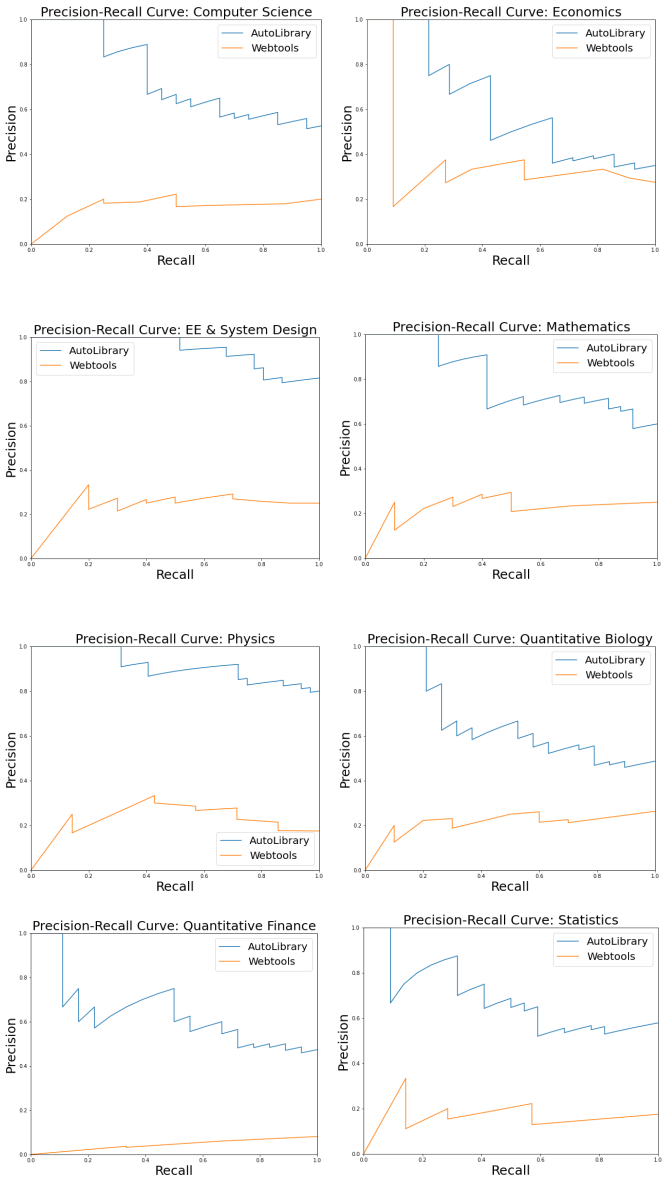
Webtools text analyzer ranks keywords and phrases according to their frequencies in the document. We only selected phrases (with at least two words) to avoid common single words such as "is", "the", and etc. We scaled the frequencies by standardizing them to the scale of $[0, 1]$. Then we manually labelled the phrases as positive or negative quality. Then scaled quality scores and the manual labels were then used to draw the precision-recall curve.

MonkeyLearn text analyzer is a pretrained model for keyword extraction. We don't know its implementation but it should be competent in the industry since the startup company that built the analyzer provides pricing plans for product teams and developers to use their text analysis APIs. The keywords results are similar to Jstor in the sense that they both didn't have quality scores. So we manually gave them positive-or-negative labels as we did to Jstor results. Then we computed the percentage of quality phrases by taking the mean of the label vector. Note that this analyzer only provides 10 quality phrases, as opposed to 40 in the case of Jstor quality phrases. So we compared them separately.

## 5.3 Overall Performance

Figure 3 shows the precision-recall curves of all eight papers under AutoLibrary and Webtools.

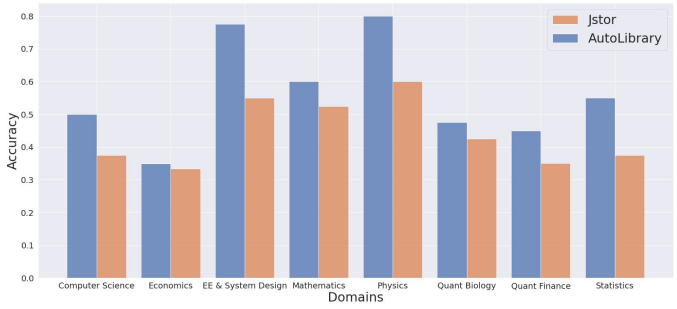
**Figure 3**
**Comparison between AutoLibrary and Webtools**




From Figure 3 we can see that the area under the precision-recall curve of AutoLibrary is bigger than that of Webtools in all eight domains. This means that AutoLibrary performs better than the Webtools analyzer across different domains. It makes sense intuitively since the latter one is very simple. Its results often contain non-quality phrases, such as "of the", "can be", and etc. Moreover, in the domain of Quantitative Finance, where there were lots of mathematical equations, the Webtools analyzer failed to filter out symbols like "$t_s$".

When comparing AutoPhrase and the Jstor analyzer, we used a histogram plot to compare their percentages of quality phrases in their extracted phrases.

**Figure 4**
**Percentage of Quality Phrases in Top 40 Extracted Phrases**



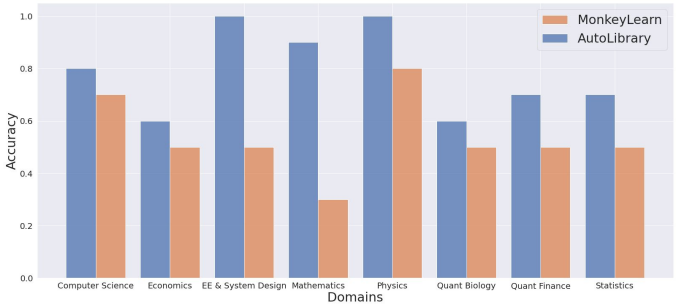

We can see that in all domains, AutoLibrary performs better than the Jstor analyzer. The main con of Jstor is that its recommendation is based on a fixed set of predefined topics. Therefore, it cannot make customized recommendations for specific papers. For example, when analyzing a statistics paper, it recommended "debt collection", which doesn't even exist in the original text. In contrast, AutoLibrary first extracts phrases from the input paper, which improves the contingency between the quality phrases and the original paper.

Another big shortcoming of Jstor is that the topics it recommends are sometimes too general. Although they make sense in English, they don't qualify as quality phrases. For example, when analyzing the Computer Science paper, one of the phrases extracted is "computer programming." It doesn't really help users since it's too general for them to search for related papers.

Last but not least, Jstor doesn't offer quality scores to recommended phrases, which means that users don't know which phrases can best represent their input papers. AutoLibrary, on the other hand, ranks the top 5 phrases in order, so that users can have a better overview of the papers.

Next we experimented with MonkeyLearn. Note that we only compared the top ten phrases from both AutoLibrary and MonkeyLearn, since the latter one only provides 10 keywords and phrases. The result comparison can be seen in Figure 5.

**Figure 5**
**Percentage of Quality Phrases in Top 10 Extracted Phrases**




From the figure, we can see that AutoPhrase outperformed MonkeyLearn in all domains. One big disadvantage of MonkeyLearn we found is that it probably relies heavily on the frequency of phrases. Although it can extract some really meaningful phrases amongst its top 5 phrases, it also recommends some phrases that make no sense. For example, when analyzing the Mathematics paper, it extracted "a1 a2 a1", "a2 a1 a2" and "a1 a2 a0" amongst the top 10 phrases. We believe that AutoLibrary defeats MonkeyLearn by weighting phrases against domain knowledge pools, which eliminates the reckless ones.

## 5.4 Result Analysis

We perform a result analysis of our model in two aspects:

1. Whether the model is able to differentiate similar papers published by the same author, while at the same time discovering their shared topics;

2. Whether the model is able to give precise results compared to manual labeling.

We select 5 papers published by Professor Shang:

**Table 4:**
**5 Papers Published by Professor Shang**

| Article | Publish Year | Domain |
|---|---|---|
| CrossWeigh | 2019 | Computer Science |
| AutoPhrase | 2018 | Computer Science |
| LM-LSTM-CRF | 2018 | Computer Science |
| AutoNER | 2018 | Computer Science |
| SetExpan | 2017 | Computer Science |

First, we get the AutoPhrase results for all 5 papers. The top 10 quality phrases are displayed in Table 5. There are some phrases, such as "maccabi tel aviv," "jiawei han," and "california," that ended up at the top of the ranked list while they are actually not domain-specific.

**Table 5:**
**Top 10 Quality Phrases from 5 Papers before Applying Weight**

| Rank | CrossWeigh | AutoPhrase | LM-LSTM-CRF | AutoNER | SetExpan |
|---|---|---|---|---|---|
| 1 | maccabi tel aviv | information extraction | neural networks | jiawei han | bipartite graph |
| 2 | hapoel jerusalem | knowledge base | conll03 ner | association for computational linguistics | data model |
| 3 | natural language processing | domain specific | highway layers | test f1 | entity intrusion |
| 4 | tel aviv org | text corpora | language model | natural language | semantic drift |
| 5 | lstm cnns crf | high quality phrases | pos tagging | xiang ren | california |

| | | | | | |
|---|---|---|---|---|---|
| 6 | natural language | quality single word phrases | bi lstm | domain specific | pubmed cvd |
| 7 | association for computational linguistics | phrase mining | highway units | modified iobes | texas |
| 8 | computational linguistics | wikipedia article datasets | sequence labeling | ablation experiments | grained types |
| 9 | cross validation | dw ½ | lstm crf | pre rec f1 | io n pte word2vec egoset seisa |
| 10 | entity disjoint filtering | phrase quality | conditional random | lstm crf | skip gram |

**Table 6:**
**Top 10 Quality Phrases from 5 Papers after Applying Weight**

| Rank | CrossWeigh | AutoPhrase | LM-LSTM-CRF | AutoNER | SetExpan |
|---|---|---|---|---|---|
| 1 | natural language processing | knowledge base | neural networks | natural language | bipartite graph |
| 2 | natural language | information extraction | pos tagging | domain specific | skip gram |
| 3 | computational linguistics | domain specific | bi lstm | named entity | ranked lists |
| 4 | cross validation | text corpora | sequence labeling | distant supervision | semantic drift |
| 5 | named entity recognition | keyphrase extraction | word embedding | lstm crf | text corpora |
| 6 | pos tagging | pos tagger | transfer learning | ablation experiments | texas |
| 7 | lstm crf | natural language | language model | distantly supervised | coarse grained |
| 8 | chicago | massive text corpora | word embeddings | ncbi | california |
| 9 | japan | cn | lstm crf | ner | skip grams |
| 10 | f1 | auc | conditional random | ram | ranked list |

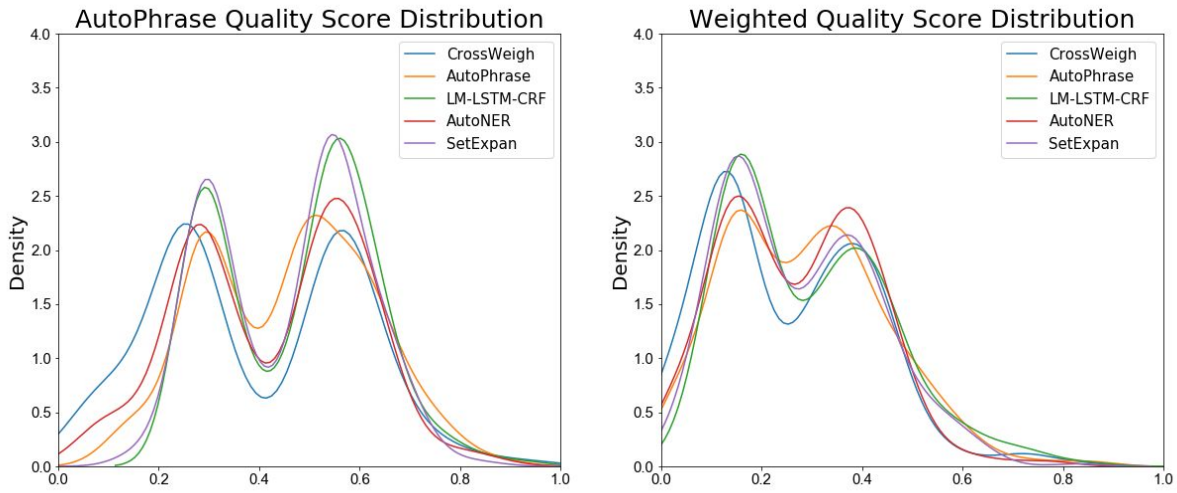



**Fig. 6.** Quality Score Distribution before & after Weighting: The quality score distribution shifts to the left after applying the weight. This is expected because scores of nonsignificant phrases are weighted down.

Thus to filter out these nonsignificant phrases, our model applies weight to the AutoPhrase result using the pre-processed arXiv dataset. For pre-processing, we have split the arXiv dataset into domains and run AutoPhrase on each of them to get domain specific phrases. For the 5 papers we are using, we select the domain to be "computer science" and the top 10 quality phrases are displayed in Table 6.

As we can see, there are some phrases, such as "natural language," "pos tagging," and "text corpora," shared across these 5 papers, as highlighted. At the same time, each of these papers has its own unique phrases, such as "cross validation" for CrossWeigh, "knowledge base" for AutoPhrase, "sequence labeling" for LM-LSTM-CRF, "distant supervision" for AutoNER, and "bipartite graph" for SetExpan. It proves that our model is able to differentiate similar papers published by the same author, while at the same time discovering their shared topics.

Next, we annotate the weighted results by manually checking and labeling whether the phrases can actually represent the paper. To visualize the performance of our model on the 5 selected papers, Fig. 7 presents the precision-recall curves. Then we compare the accuracy for phrases with a quality score > 0.5, > 0.6, and > 0.7, as shown in Table 7.

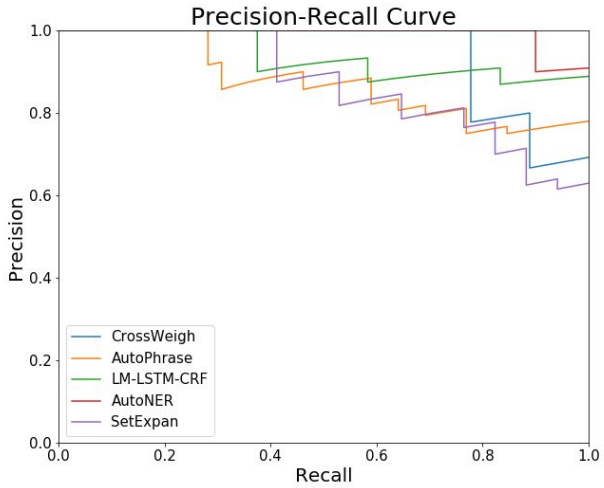



**Fig. 7.** Precision-Recall Curves for Weighted AutoPhrase Results of 5 Selected Papers: Looking at the area under the curve (AUC), a high AUC indicates that our model successfully extracts phrases that can actually represent the paper. In addition, as the size of the weighted results is small, especially for AutoNER and CrossWeigh, AUC is higher for these two papers.

**Table 7:**
**Accuracy of Weighted Results against Manual Labeling**

| Article | Accuracy | | |
|---|---|---|---|
| | Quality Score > 0.5 | Quality Score > 0.6 | Quality Score > 0.7 |
| CrossWeigh | 0.6429 | 1.0 | 1.0 |
| AutoPhrase | 0.7800 | 0.8571 | 1.0 |
| LM-LSTM-CRF | 0.8889 | 0.9231 | 1.0 |
| AutoNER | 0.8333 | 1.0 | 1.0 |
| SetExpan | 0.6296 | 0.8889 | 1.0 |

As we can see from the dataframe above, accuracy is higher for phrases with a higher quality score. Since our model is using the top 3 quality phrases, it is able to give precise results compared to manual labeling.

## 6. CONCLUSIONS

In conclusion, we used AutoPhrase as our keyword extractor and pretrained domain models to weight quality scores. In this way, we successfully combine the merit of high quality domain phrases and customization of specific papers. The accuracy score of our keyword extraction was proved to be higher than existing text analyzers, including other scholar paper search engines. This advantage is held across all domains, thus making it a domain-independent search engine.

Our web application easies the pain point in existing digital libraries that cannot recommend related papers according to specific topics and field of studies of given papers. It extracts accurate quality phrases from specific papers to make customized recommendations, which allows users to easily find keywords of a given paper and further explore fields that they are not familiar with.

However, our application has some limitations right now and we will improve it in the future. First of all, we can improve the runtime as it would take a much longer time to run if multiple users are using it at the same time. Second, as our application only supports STEM domains right now, we could add more scientific domains in the future. Third, we can have a user management system so that users could use different devices and browsers to access their uploaded documents.

## 7. REFERENCES


[1] Jingbo Shang, Jialu Liu, Meng Jiang, Xiang Ren, Clare R Voss, Jiawei Han, *"Automated Phrase Mining from Massive Text Corpora"*, accepted by IEEE Transactions on Knowledge and Data Engineering, Feb. 2018.

[2] Bush, Vannevar. "As We May Think." *The Atlantic*, 1, July 1945.

[3] Candela, Leonardo, et al. *The Digital Library Manifesto*. 2006.

[4] Licklider, J. C. R. *Libraries of the Future*. Massachusetts Institute of Technology, 1965.

[5] Orduña-Malea, E., Ayllón, J. M., Martín-Martín, A., & Delgado López-Cózar, E. (2015). Methods for estimating the size of Google Scholar. *Scientometrics*, *104*(3), 931–49.

[6] Beel, Jöran, and Bela Gipp. *Google Scholar's Ranking Algorithm: An Introductory Overview*. Proceedings of the 12th International Conference on Scientometrics and Informetrics (ISSI'09), 2009, pp. 230–41.

[7] J. Liu, J. Shang, C. Wang, X. Ren, and J. Han, *"Mining quality phrases from massive text corpora,"* in Proc. ACM SIGMOD Int. Conf. Manage. Data, 2015, pp. 1729–1744.

[8] Li, Hui and Gu, Yan and Koul, Saroj, Review of Digital Library Book Recommendation Models (November 25, 2009). Available at SSRN: https://ssrn.com/abstract=1513415 or http://dx.doi.org/10.2139/ssrn.1513415

[9] A. El-Kishky, Y. Song, C. Wang, C. R. Voss, and J. Han, "Scalable topical phrase mining from text corpora," Proc. VLDB Endow., vol. 8, no. 3, pp. 305–316, Nov. 2014.

[10] J. Leskovec, L. Backstrom, and J. Kleinberg, "Meme-tracking and the dynamics of the news cycle," in Proc. 15th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining, 2009, pp. 497–506.

[11] B. Li, B. Wang, R. Zhou, X. Yang, and C. Liu, "Citpm: A cluster- based iterative topical phrase mining framework," in Proc. Int. Conf. Database Syst. Adv. Appl., 2016, pp. 197–213.

[12] Tricot, Joe, devrishi, and Brian Maltzan. "ArXiv Dataset,"n.d.https://kaggle.com/Cornell-University/arxiv.

[13] Flemotomos, Nikolaos, Victor R. Martinez, Zhuohao Chen, Karan Singla, Victor Ardulov, Raghuveer Peri, Derek D. Caperton, et al. "'Am I A Good Therapist?' Automated Evaluation Of Psychotherapy Skills Using Speech And Language Technologies." ArXiv:2102.11265 [Cs, Eess], February 22, 2021. http://arxiv.org/abs/2102.11265.

[14] Sönmez, Tayfun, and M. Bumin Yenmez. "Affirmative Action in India via Vertical, Horizontal, and Overlapping Reservations." *ArXiv:2102.03186 [Econ]*, February 5, 2021. http://arxiv.org/abs/2102.03186.

[15] Kofler, Andreas, Markus Haltmeier, Tobias Schaeffter, and Christoph Kolbitsch. "An End-To-End-Trainable Iterative Network Architecture for Accelerated Radial Multi-Coil 2D Cine MR Image Reconstruction." *ArXiv:2102.00783 [Cs, Eess]*, February 1, 2021. http://arxiv.org/abs/2102.00783.

[16] Li, Gen, Yuting Wei, Yuejie Chi, Yuantao Gu, and Yuxin Chen. "Softmax Policy Gradient Methods Can Take Exponential Time to Converge." *ArXiv:2102.11270 [Cs, Eess, Math, Stat]*, February 22, 2021. http://arxiv.org/abs/2102.11270.

[17] Di Mauro, Marco, Salvatore Esposito, and Adele Naddeo. "A Roadmap for Feynman's Adventures in the Land of Gravitation." *ArXiv:2102.11220 [Gr-Qc, Physics:Physics, Physics:Quant-Ph]*, February 22, 2021. http://arxiv.org/abs/2102.11220.

[18] Terzano, Michele, Andrea Spagnoli, Daniele Dini, and Antonio Elia Forte. "Fluid-Solid Interaction in the Rate-Dependent Failure of Brain Tissue and Biomimicking Gels." *ArXiv:2102.11268 [Cond-Mat, Physics:Physics, q-Bio]*, February 15, 2021. http://arxiv.org/abs/2102.11268.

[19] Fujii, Masaaki, and Akihiko Takahashi. "Equilibrium Price Formation with a Major Player and Its Mean Field Limit." *ArXiv:2102.10756 [Econ, q-Fin]*, February 21, 2021. http://arxiv.org/abs/2102.10756.

[20] Kozodoi, Nikita, Johannes Jacob, and Stefan Lessmann. "Fairness in Credit Scoring: Assessment, Implementation and Profit Implications." *ArXiv:2103.01907 [Cs, q-Fin, Stat]*, March 2, 2021. http://arxiv.org/abs/2103.01907.