TopicTrend: Discover Emerging and Novel Research Topics

Jovian Lin (A0026542M)
School of Computing
National University of Singapore
jovian.lin@gmail.com

ABSTRACT

The amount of information in digital libraries today is staggering. For instance, ScienceDirect[1] contains nearly 10 million articles, and this number will only continue to increase over time. Although the integration of search engines in digital libraries have greatly supported information seeking and retrieval, the amount of results or articles presented is still sizable — ranging from a few hundreds to tens of thousands. Therefore, in order to tackle this problem of information overload, we introduce TopicTrend — a practical ScienceDirect web application that helps users, particularly *junior researchers*, discover emerging and novel research topics, patterns, and relationships *efficiently*. TopicTrend takes away the chore of sifting through an extensive list of search results in a digital library. Instead, based on a user's search query and the search engine's results, it provides a condensed *list of research topics*, as well as a *visualization framework* that reveals a set of "hot" research topics and their interconnectedness via a graph[2] visualization. Whenever there is a need to dive into the details, articles pertaining to individual research topics can also be retrieved within the visualization framework. This paper describes the design and implementation of TopicTrend using the SciVerse Application Framework [10]. We also describe how we evaluated the effectiveness of the system with the help from four researchers, and identified key areas for improvement. Finally, we explain our future work and how it can help refine the shortcomings of our system.

[1]ScienceDirect is one of the largest online collections of published scientific research in the world. It is operated by the publisher Elsevier and contains nearly 10 million articles from over 2,500 journals and over 6,000 e-books, reference works, book series and handbooks issued by Elsevier.

[2]A graph is an abstract representation of a set of objects where some pairs of the objects are connected by links. The interconnected objects are represented by vertices/nodes, and the links that connect some pairs of vertices are called edges.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval; H.3.7 [**Information Storage and Retrieval**]: Digital Libraries

General Terms

CS6242, Digital Libraries, Project, Topic, Trend

Keywords

digital libraries, query, search, topics, trend, visualization

1. INTRODUCTION

Formulating a worthy research idea is the first step necessary for success in academia. Above all else, a worthy research idea must be original and innovative, since the goal of research is to advance the state-of-the-art in knowledge and technology. In addition, it must be tackling a problem that the academic community deems to be important and relevant at the moment — a requirement that is perhaps the hardest for most junior researchers to accept. Therefore, in order to come up with innovative research ideas, researchers have to read a lot of published articles, which in no doubt is a time-consuming task.

Fortunately, the web allows scientific articles to be easily accessible. Online scientific literature and digital libraries such as the ACM Digital Library (DL) [1], IEEE Xplore [4], CiteSeer [9], and ScienceDirect [8] provide broad bibliographical and full-text access to journals and conference proceedings. Users can search for a specific article, typically by title or author. Some of the digital libraries may also be browsed based on the type of publication such as journals, transactions, and proceedings. Most of the existing systems, however, are not designed to help users understand research trends [6]. A few digital libraries provide some simple, statistical facts. For example, the ACM DL shows download counts and CiteSeer shows the most frequently cited papers. However, simple analysis often requires extensive navigation and effort from the user's side [6].

It is even more difficult to understand how topics interact and influence research activity in general. Therefore, Smeaton et al. [11] performed a content analysis of all papers published in the SIGIR proceedings to understand research trends and to identify emerging and "hot" areas. Their focus was to determine what topic areas appear in the papers at the SIGIR conferences but not to visualize the results. They represented the clustering results in a table, whose rows and columns represent topics and years respectively. Each cell

in the table contains the number of papers. The topics are sorted approximately in order of a combination of the year of their first appearance and the number of papers published. Since they color coded the cells by the number of papers, it is easy to recognize "hot" topics. However, users have to read through and compile the numbers to see the trends of topics, which makes it especially difficult to compare trends of several topics [6].

1.1 A Researcher's Workflow

As a prelude to our system, we give a brief description of a researcher's workflow. Normally, a researcher will submit a research topic as a query in a digital library's search engine, such as ScienceDirect [8]. The search engine processes the query and returns a set of results, which is usually a list of titles and their associated journals. Next, the user will select a few articles for more in-depth study, which is typically based on whether the title catches the user's attention. As an example, Figure 1 shows a screen shot of the results that a search engine returns. Based on the query "information retrieval", a list of titles is displayed to the user, whereby the most relevant ones (with respect to the search query) are shown at the top. Furthermore, there were about 147,800 articles related to the query, and the list spans over 2957 pages, in which each page exhibits a maximum of 50 results.

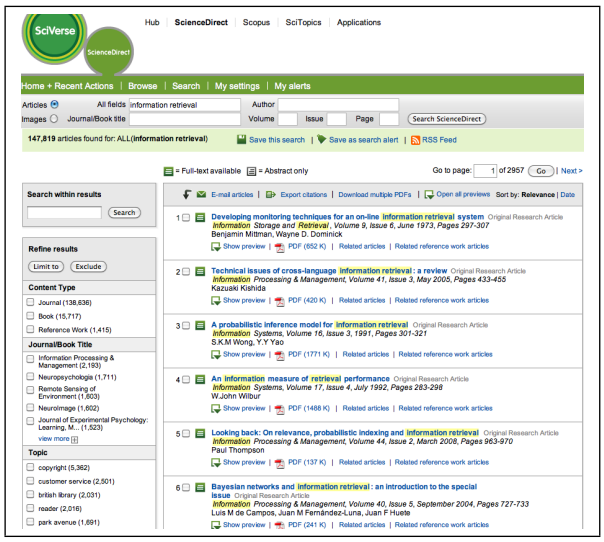

Figure 1: Screen shot of Elsevier's ScienceDirect displaying a list of results based on the query "information retrieval".

In general, depending on the scope of the user's search query, the size of the ranked list ranges from a few hundreds to tens of thousands. This sheer amount of one-dimensional information does not necessary mean that the researcher's information needs are satisfied. In fact, it would be more helpful if the system could return an intelligent summary from this deluge of data, as allowing the researcher to select articles based on how interesting their titles look or sound is an inefficient task. Firstly, the researcher will likely miss out on potentially good articles as the titles do not necessary express the entire article's context sufficiently. Even if the researcher goes through every article that the search engine recommends, it is nevertheless a daunting and time-consuming task. Secondly, the ranked list does not provide any informative visualization regarding neighboring research topics, as well as the strength of their relation. Thirdly, the displayed set of results cannot assist the researcher in identifying emerging and "hot" research topics that have not been explored in depth.

A solution to these problems is to develop a visualization framework that displays research topics (retrieved from the search engine's result set) that are related to the search query of the researcher. These research topics will be selected based on their level of "freshness"; in other words, topics that have emerged recently will be given higher priority. The information will be represented as a graph, in which the nodes represent research topics while the edges represent the relational strength between the topics.

1.2 TopicTrend

In this paper, we propose a ScienceDirect web application called TopicTrend. It is able to harness a user's search query and the search engine's results to provide a condensed *list of research topics*, as well as a *visualization framework* that reveals a set of "hot" research topics and their interconnectedness via a graph. That is, the system takes into account the recency of the topics and their related articles. For example, a research topic that appeared in recent years will be deemed more important than a research topic that has been on-going for several years. In addition, the system filters articles based on the research topic — a management on the deluge of data that the search engine returns. As novelty of a research paper is the key in getting acceptance, users of the system will be able to efficiently identify emerging topics of unprecedented value.

2. INTERFACE

TopicTrend works within the environment of ScienceDirect. When a user submits his query in ScienceDirect's search box, TopicTrend inspects the top 1000 documents that ScienceDirect returns and displays a list of potential research topics to the user (shown in Figure 2). The topics are ranked according to the frequency in which they appeared in recent years, and this temporal measurement is indicated by a score (also in Figure 2). In addition, a *green* upward-pointing arrow states that the topic has been used in more recent years, whereas a *red* downward-pointing arrow states that the topic has been consistently used for more than two years.

When the user activates the visualization framework of TopicTrend, a graph consisting of nodes (the research topics) and edges (the strength between topics) is displayed. This is shown in Figure 3. Here, the user is able to get a clear picture of the relationships between research topics. In addition, the articles have also been organized into their respective topics. This is illustrated in the right black box in Figure 3. Categorizing the articles based on their topics reduces information overload. Also, the user is able to directly go to the article itself by clicking on the article's link under the heading "related articles".

3. IMPLEMENTATION

The system is required to take in the metadata of 1000 research papers, process them, and output a ranked list of research topics. The metadata includes: (i) age of the article (with respect to the current time), (ii) title, (iii) abstract

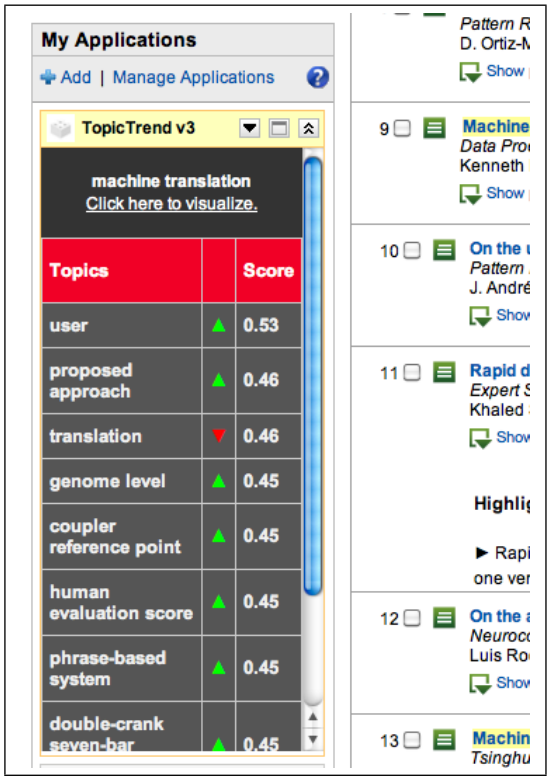

Figure 2: A display of a list of topics that are related to the user's search query. A *green* upward-pointing arrow states that the topic has been increasingly used in the recent two years, whereas a *red* downward-pointing arrow states that the topic has been consistently used for more than two years. The numerical score represents the "freshness" of the topic, which ranges from 0 to 1.

(this is the most important piece of metadata), and (iv) the personally identifiable information (PII). The reason for using the abstract is that it contains the key topic terms while at the same time, it is not as lengthy as the whole article itself. That is, an abstract is a compact piece of metadata that contains the contextual information needed. This allows us to perform computation in real-time. In addition, due to the need to provide results in real time, advanced topic modeling algorithms such as Latent Dirichlet Allocation (LDA) [2] were not used because it took too long to compute. Also, we did not take the article citations into consideration, as doing so causes a tenfold increase in the run-time, making the system impractical for real use.

In addition, as this project requires a thorough analysis of the textual abstracts, OpenNLP [7] was heavily relied on. OpenNLP is an open source, java-based natural language processing (NLP) package which is able to perform sentence detection, tokenization, and part-of-speech tagging. The application server side processing was implemented on a Tomcat 6.0 server using JavaServer Pages (JSP). On the client side, as much as possible, we chose technologies that were browser independent and eliminated the need for browser plug-ins. All client side scripting was done in Javascript using only features of the HTML DOM and javascript methods, properties, and classes that were available in all of the most widely used browsers.

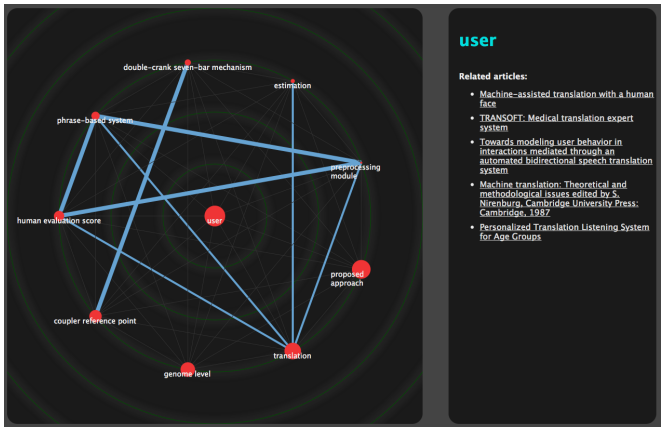

Figure 3: A visualization of the topics that are related to the user's search query. The red nodes represent the research topics; the bigger they are, the "fresher" they are. In addition, the blue edges vary in thickness; the thicker they are, the stronger is the affiliation between the connected nodes.

In this section, we give a detailed description of the sequence involved in converting the results from a user's query into a ranked list of research topics (Figure 2), as well as its respective visualization (Figure 3).

3.1 Extracting Noun Phrases (NPs)

Noun phrases (NPs) have to be extracted from the 1000 abstracts to form a list of NPs. This list of NPs will be treated as potential research topics. Here, we describe how we used OpenNLP in order to churn out this list of NPs.

3.1.1 Sentence Detection

OpenNLP has a sentence detection tool for splitting up raw text into sentences. In addition, it uses a maximum entropy model to evaluate the characters ".","!", and "?" in a string to determine if they signify the end of a sentence. Given the raw text:

> Pierre Vinken, 61 years old, will join the board as a nonexecutive director Nov. 29. Mr. Vinken is chairman of Elsevier N.V., the Dutch publishing group. Rudolph Agnew, 55 years old and former chairman of Consolidated Gold Fields PLC, was named a director of this British industrial conglomerate. Those contraction-less sentences don't have boundary/odd cases...this one does.

The sentence detection tool tries to detect the sentences and splits the paragraph into the following four sentences:

- Pierre Vinken, 61 years old, will join the board as a nonexecutive director Nov. 29.
- Mr. Vinken is chairman of Elsevier N.V., the Dutch publishing group.
- Rudolph Agnew, 55 years old and former chairman of Consolidated Gold Fields PLC, was named a director of this British industrial conglomerate.
- Those contraction-less sentences don't have boundary/odd cases...this one does.

3.1.2 Tokenization

Tokenization is the process of breaking a stream of text up into words, phrases, symbols, or other meaningful elements called tokens. Tokens are usually words, punctuation, numbers, etc. The list of tokens becomes input for further processing. OpenNLP has a tool for tokenization, which segments an input character sequence into tokens. After performing tokenization, we get the following:

- [Pierre] [Vinken] [,] [61] [years] [old] [,] [will] [join] [the] [board] [as] [a] [nonexecutive] [director] [Nov.] [29] [.]
- [Mr.] [Vinken] [is] [chairman] [of] [Elsevier] [N.V.] [,] [the] [Dutch] [publishing] [group] [.]
- [Rudolph] [Agnew] [,] [55] [years] [old] [and] [former] [chairman] [of] [Consolidated] [Gold] [Fields] [PLC] [,] [was] [named] [a] [director] [of] [this] [British] [industrial] [conglomerate] [.]
- [Those] [contraction-less] [sentences] [do] [n't] [have] [boundary/odd] [cases] [...this] [one] [does] [.]

3.1.3 Part-of-Speech Tagging

Part-of-Speech Tagging is the process of marking up a word in a text (corpus) as corresponding to a particular part of speech, based on both its definition, as well as its context — i.e. relationship with adjacent and related words in a phrase, sentence, or paragraph. A simplified form of this is commonly taught to school-age children, in the identification of words as nouns, verbs, adjectives, adverbs, etc.

We use OpenNLP's part-of-speech tagger tool, which uses maximum entropy that tries to predict whether words are nouns, verbs, or any of 70 other part-of-speech tags depending on their surrounding context. Thus, given the tokens:

- [Pierre] [Vinken] [,] [61] [years] [old] [,] [will] [join] [the] [board] [as] [a] [nonexecutive] [director] [Nov.] [29] [.]
- [Mr.] [Vinken] [is] [chairman] [of] [Elsevier] [N.V.] [,] [the] [Dutch] [publishing] [group] [.]

After performing Part-of-Speech Tagging with OpenNLP, we get the following:

- [NNP] [NNP] [,] [CD] [NNS] [JJ] [,] [MD] [VB] [DT] [NN] [IN] [DT] [JJ] [NN] [NNP] [CD] [.]
- [NNP] [NNP] [VBZ] [NN] [IN] [NNP] [NNP] [,] [DT] [JJ] [NN] [NN] [.]

3.1.4 Chunking and Extracting NPs

Text chunking consists of dividing a text in syntactically correlated parts of words, like noun groups, verb groups, but does not specify their internal structure, nor their role in the main sentence. Here, we perform chunking on the tokenized sentences and the part-of-speech tags. To be more accurate, we use OpenNLP's Treebank Chunker, which goes a little further in showing sentence structure by breaking the sentence into simple chunks. Eventually, we retrieve only the NPs using a set of hard-coded rules.

3.2 Scoring the NPs

3.2.1 Categorizing an Article's Age

The search results in ScienceDirect are ranked based on closeness to the query, *not* their age. In order to take into account the temporal component of the articles, we separate the age into three types:

1. Those that are at most two years old
2. Those that are more than two years old, but less than four years old.
3. Those that are more than four years old.

The three different age groups are used to filter and prioritize newer articles. This is done by assigning the most weight to articles that fall into the first age group (i.e., those that are at most two years old), whereas articles in the second group will receive less weight than the first. The third group will receive the least weight of all.

We note that separating into these three types is a simple and naive approach, as different disciplines experience different rates of obsolescence, or "atrophy". Therefore, this approach may introduce problems in the later stages. However, due to time constraints, we chose this method for its simplicity and to come up with a working system for demonstration. In our future update, we will allow users to vary this time range.

3.2.2 Filtering Research Topics from NPs

First, stop words are removed in order to improve the system's performance. This includes words like "proposed", "survey", "evaluation", "algorithm", etc. Next, based on the collection of NPs of individual articles, we gather all the unique NPs into a hashtable. Upon noticing a duplicated NP (which indicates a recurring research term), the title, PII and age of that particular article are noted and stored. Therefore, an individual NP can have multiple articles associated with it. After iterating through all the 1000 articles, we will have a list of NPs as well as metadata (of articles) associated with them. We calculated the level of "freshness" of an NP based on the following algorithm:

Let:

$t_{a,x}$ = no. of articles aged between 0 and 2 years for an NP "x".
$t_{b,x}$ = no. of articles aged between 2 and 4 years for an NP "x".
$t_{c,x}$ = no. of articles aged 4 years and above for an NP "x".

$$\text{Score} = \frac{t_{a,x} + \alpha}{t_{a,x} + t_{b,x} + t_{c,x} + \beta} \quad \text{, where } \alpha=1,\ \beta=20$$

The algorithm gives higher scores for NPs which have articles appearing in the most recent two years. In other words, we score each NP based on the time in which their related papers were published. This score is reflected by the size of the node in the visualization framework — the bigger the node, the "fresher" the NP. The system is then able to produce a list of NPs that are predominantly ranked based on temporal data.

Participant	Participant 1	Participant 2	Participant 3	Participant 4
School	University of Cambridge	University of Auckland	National University of Singapore	National University of Singapore
Course/Year	PhD / Year 2	PhD / Year 3	PhD / Year 2	Undergraduate / Year 4
Research Field	**Chemistry**	**Transportation Research**	**Artificial Intelligence**	**Engineering**
Query 1	molecular dynamics	vehicle scheduling	latent dirichlet	lithography techniques
Score	7/10	8/10	5/10	9/10
Query 2	drug designs	vehicle routing	statistical regression	electron beam lithography
Score	6/10	8/10	8/10	8/10
Query 3	protein interactions	crew scheduling	sequence memorizer	reactor geometry
Score	6/10	5/10	6/10	7/10
Query 4	peptides	demand forecasting	bayesian modelling	film deposition
Score	7/10	4/10	8/10	7/10
Average Score	**6.5/10**	**6.25/10**	**6.75/10**	**7.75/10**

Figure 4: Four participants tested TopicTrend using 4 different queries from their respective domains. For each query, TopicTrend returns 10 of the closest research topics (i.e., outputs). A participant then rates each of the 10 outputs that was generated based on his query — giving a score of "1" for a relevant output, and a score of "0" for a non-relevant one. For every query, the scores were summed up and divided by the number of outputs (i.e., 10). Eventually, the average of all 4 queries (per participant) was calculated.

3.3 Displaying the Results

We used the ranked list of NPs to produce two visual outputs — a ranked list of research topics (Figure 2) and a graph visualization (Figure 3).

3.3.1 Displaying the List of Research Topics

We retrieved the top 10 NPs from the ranked list of NPs. The score of each NP is displayed, which allows the user to view its numerical strength that ranges between 0 and 1. In addition, a colored arrow that was either green and upward-pointing, or red and downward-pointing is shown. A green upward-pointing arrow states that the topic has been increasingly used in the recent two years. That is, for an NP, the number of articles (that contain the NP) pertaining to the recent 2 years are more than the articles pertaining to 2 years and above. Whereas a red downward-pointing arrow states that the topic has not been increasingly used in the recent two years. This means that either the NP has been used constantly throughout the years, or it has been used less often as the years go by.

3.3.2 Visualizing the Research Topics via a Graph

The JavaScript InfoVis Toolkit [5] was used to visualize the information on a weighted graph animation. With its adequate tools, it allows us to create interactive data visualizations for the Web. The nodes, edges, names, PIIs, and other information are converted into a JSON structure, and passed to the toolkit, and the resulting visualization is displayed in ScienceDirect through a HTML iframe (Figure 3).

The nodes in the graph contain the same top 10 NPs, whereas the edges reflect the strength between the nodes — something that is not present in the "list view" (Figure 2). As each NP contains a set of PIIs that uniquely identifies a research paper, we calculate the relationship strength between the top NPs by considering the common PIIs that they have. That is, if two different terms, *Term A* and *Term B*, exist in a paper with *PII X*, then the edge (or strength) between *Term A* and *Term B* will be increased by one unit. Therefore, for the graph visualization, the thickness of the edge reflects the number of articles that the two connecting nodes (i.e., the topics) have in common.

In addition, users are able to view the articles that are related to the NP. This is because each of the NP contains a list of PIIs that uniquely identifies an article, allowing us to create a hyperlink that points directly to the article in ScienceDirect.

3.4 Limitations in Elsevier's API

Problems were encountered with Elsevier's API at the start, which affected the progress and more importantly, the implementation of the application. Previously, for the version of TopicTrend that was presented in SGCodeJam24, the method *gadgets.sciverse.getAllResults* (or *getAllResults*) was used to directly retrieve all the "keywords" from every article without any additional processing on the developer's side. In addition, the *getAllResults* method is only able to retrieve data from a maximum of 200 articles. Here, "keywords" is one of the metadata that *getAllResults* is able to return, and contains words and phrases that the authors picked to best describe their articles. However, the *getAllResults* method is no longer functioning — it merely returns empty JSON objects when called. The staff at Elsevier have also admitted that this method call has been broken since their September 2011 updates, and will only be fixed in their next major release in December 2011. Therefore, we have since used a different but functioning method from their API. However, this requires us to perform additional computation on the texts using the OpenNLP toolkit that runs on our own servers.

Unfortunately, that is not the only problem with Elsevier's API. A crucial method (i.e., *makeContentApiRequest*) that fetches the metadata of the articles works irregularly. For

The system was easy to use.	4.75
I felt comfortable using this system.	4.75
Overall, I am satisfied with this system.	4
I was able to perform searches and view the visualization results efficiently.	4.5
The "look" of this system was pleasant.	4.75
The system gave me interesting results.	4
The topics produced by the system were relevant.	3
The topics produced by the system were related to the search query.	3
The topics produced by the system were related to one another.	3
I was able to get a better understanding of the topics using the visualization view.	4
I was able to discover trends of the topics using the visualization view.	4
I was able to discover relationships between topics using the visualization view.	4
I was able to discover potential, novel topics by noticing the edges of the graph.	4

Table 1: Average ratings for TopicTrend, using the scale of 1=Disagree, 5=Agree.

example, the method is able to return a JSON object for the query "information retrieval", but returns an empty object for the query "information retrieval SVM". To put it more accurately, depending on the user's query, the JSON text that is fetched may be corrupted, and cannot be converted into a proper JSON object.

4. EVALUATION

In order to understand how useful TopicTrend is, three graduates and an undergraduate were invited to evaluate the system. All four of the participants came from different fields of research — *Chemistry*, *Transportation Engineering*, *Computer Science*, and *Electrical Engineering*. The ages of the participants ranged from 25 to 29. The participants were given a brief tutorial of the system right before the start of the evaluation, spending no longer than 20 minutes reading about and interacting with features of the system. In addition, each of the four sessions lasted no more than an hour.

4.1 Accuracy of the System

A *quantitative evaluation* was first performed on the system. The four participants assessed TopicTrend using 4 different queries from their respective domains. For each query, TopicTrend returns 10 of the closest research topics (i.e., outputs). A participant then rated each of the 10 outputs that was generated based on his query — giving a score of "1" for a relevant output, and a score of "0" for a non-relevant one. For every query, the scores were summed up and divided by the number of outputs (i.e., 10). Eventually, the average of all 4 queries (per participant) was calculated. The results of this study is shown in Figure 4, in which we observed that the more "junior" the researcher is, the higher is his average score for the system. Conversely, the more "senior" the research is, the lower is his average score for the system.

4.2 User Satisfaction Ratings

Next, we performed a *qualitative evaluation* on the system, in which the participants were asked to fill up a satisfaction questionnaire that employed a five-point Likert scale; that is, all ratings were on a scale of 1 to 5. The questions and their average ratings are shown in Table 1.

From the results of the evaluation, it is clear that the system, with its simple and vibrant interface, excels visually. However, the outputs produced by the system were considered "mediocre" by the participants, which coincided with the quantitative results in Figure 4. The reason for this could be due to the quality of the NPs. That is, the NPs that were derived from the part-of-speech tagging process may not be a good representation of the research terms due to the existence of stop words. Stop words such as "purpose", "survey", "experiments", and "results" have to be manually keyed into a filter called the "stop word list" in order to improve the quality of the output terms.

Further analysis has shown that despite the setback, compared to the deluge of results from traditional search methods, users of TopicTrend were still able to get a better understanding of the topics and trends that were related to their search query. In addition, by focusing on the nodes that do not have any edges between them, users are able to discover potential, untapped topics.

5. FUTURE WORK

In order to improve the quality of the results, a term frequency-inverse document frequency (tf-idf) weighting system could be employed to identify important words. In tf-idf, a word's importance increases to the number of times it appears in a document, but is offset by the frequency of the very same word in the corpus. We believe that the tf-idf weighting scheme will be able to remove the stop phrases the plague our system.

It would also be interesting to implement topic models onto the data set. A topic model is a type of statistical model for discovering the abstract "topics" that occur in a collection of documents. The Latent Dirichlet Allocation (LDA) [2] is perhaps the most common topic model currently in use. However, as mentioned in Section 3, traditional LDA is time-consuming and is unsuitable for real-time computation. Fortunately, we could test out an "online" version of LDA [3], which can handily analyze massive document collections, including those arriving in a stream. More importantly, the online LDA is able to find topic models as good as those found using the traditional LDA, but at a fraction of the time [3]. In addition, as abstracts are usually made up of short pieces of text, we do not risk increasing the computational time. With topic models, each abstract is seen as a mixture of a small number of topics. Using this mixture of topics, we can perform more in-depth and advance analysis which could improve the results of TopicTrend.

In addition, instead of having the user type in a textual search query, we could perform a search using an exemplar — by allowing the user to input an existing article. The system will first retrieve a set of textual terms from the exemplar. Based on this set of terms, the system will then perform a search on each of the terms. The results will be displayed in a similar fashion as the current TopicTrend. However, the good thing about this implementation is that the user does not need to explicitly state what he wants.

This is useful in situations whereby the user is unable describe or express his query completely.

6. CONCLUSION

TopicTrend is a visualization tool that alleviates the problem of information overload in search engines of digital libraries. It allows users, particularly *junior researchers*, to discover emerging and novel research topics, patterns, and relationships *efficiently* via a graph visualization — effectively creating a summary from their search query. Therefore, instead of sifting through an extensive list of search results, users will be able to attain a visual summary of their search query which improves the efficiency of their discovery process. In this paper, we demonstrated TopicTrend and gave a thorough description of its implementation. Our evaluation showed that TopicTrend excels visually. That is, the visualization scheme was beneficial in aiding the process of topic discovery. However, the quality of the outputs produced by the system was mediocre, which was due to the existence of stop words. Subsequently, we described how we would improve the system in our future work. This included the use of the "term frequency-inverse document frequency" (tf-idf) weights to identify important words from the stop words, as well as the statistical nature of topic modeling, particularly LDA and its variants. We also described another method of searching — by using an article as an exemplar — which would be useful in situations whereby the user is unable to express his query.

7. REFERENCES

[1] ACM-Digital-Library. http://portal.acm.org.

[2] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022, Mar. 2003.

[3] M. Hoffman, D. M. Blei, and F. Bach. Online learning for latent dirichlet allocation. In *NIPS*, 2010.

[4] IEEE-Xplore. http://ieeexplore.ieee.org.

[5] JavaScript-InfoVis-Toolkit. http://thejit.org//.

[6] B. Lee, M. Czerwinski, G. Robertson, and B. B. Bederson. Understanding research trends in conferences using paperlens. In *CHI '05 extended abstracts on Human factors in computing systems*, CHI EA '05, pages 1969–1972, New York, NY, USA, 2005. ACM.

[7] OpenNLP. http://incubator.apache.org/opennlp/.

[8] Science-Direct. http://sciencedirect.com.

[9] Scientific-Literature-Digital-Library. http://citeseer.ist.psu.edu/cs.

[10] SciVerse-Developers. http://developers.sciverse.com/api.

[11] A. F. Smeaton, G. Keogh, C. Gurrin, K. McDonald, and T. Sødring. Analysis of papers from twenty-five years of sigir conferences: what have we been doing for the last quarter of a century? *SIGIR Forum*, 36:39–43, September 2002.