Knowledge-Based Systems 69 (2014) 34–44

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

Analyzing future communities in growing citation networks

Sukhwan Jung *, Aviv Segev

Department of Knowledge Service Engineering, KAIST, Daejeon 305-701, South Korea

ARTICLE INFO

Article history:
Received 25 October 2013
Received in revised form 22 April 2014
Accepted 23 April 2014
Available online 5 May 2014

Keywords:
Community
Topic detection
Link prediction
Citation network
Community detection

ABSTRACT

Citation networks contain temporal information about what researchers are interested in at a certain time. A community in such a network is built around either a renowned researcher or a common research field; either way, analyzing how the community will change in the future will give insight into the research trend in the future. The paper views the research community as a Social Web where the communication is through academic papers. The paper proposes methods to analyze how communities change over time in the citation network graph without additional external information and based on node and link prediction and community detection. Different combinations of the proposed methods are also analyzed. The identified communities are classified using key term labeling. Experiments show that the proposed methods can identify the changes in citation communities multiple years in the future with performance differing according to the analyzed time span. Furthermore, the method is shown to produce higher performance when analyzing communities to be disbanded and to be formed in the future.

1. Introduction

Citation networks represent a picture of the current situation of research information in a specific field. The network therefore represents communities centered on a specific researcher or on a shared research field. Analyzing how the community will change in the future will give insight into the research trend in the future and how a field will evolve.

Citation network analysis originated with the paper of Garfield et al. (1964) [17], which showed that the analysis indicated a high degree of coincidence between a historian's account of events and the citational relationship between these events. The present work, however, takes the opposite approach and looks to the future: it examines whether the prediction of citation networks can assist in the analysis of future events.

The paper presents a new perspective of viewing the research community as a Social Web where the communication is through academic papers. The goal of the paper is to assist a community member in making a decision as to what "products" (research topics) are good and what topics are less trendy. The idea of using a social analysis approach on the academic community is the main contribution of this work.

The paper presents several methods to analyze how communities change over time in the citation network. The methods are based on a graph representation of the citation community at given time stamps with nodes representing papers and edges representing citations. External information such as author names, institutions, and existing keyword classifications is not used. The prediction methods are composed of different combinations of proposed building block algorithms for node prediction, edge prediction, and community detection. The node prediction analyzes the change in previous years in the number of citations and gives higher probability to highly cited papers. After the node prediction, six link prediction algorithms are compared to analyze the performance. The analysis showed that the link prediction methods can be classified into two categories that contribute to the performance of the community detection. The Louvain method is used as the basic community detection method. Three topic detection methods are used to label the detected communities. The TF/IDF (Term Frequency/Inverse Document Frequency) method is a widely accepted method in IR (Information Retrieval) and is used to extract representative terms, in this case labels, from documents. The Keyword Extraction method uses a statistical algorithm and natural language process technology to analyze the text and identify terms of importance. The Concept Tagging method gives a high-level abstraction of a given text by utilizing natural language process techniques with external databases such as DBpedia or OpenCyc and returns concepts which were not directly mentioned in the text itself. The basic community analysis building blocks are organized in four different methods to provide an analysis of the order in which the methods can be used and of their individual

* Corresponding author. Tel.: +82 1042273922.
E-mail addresses: raphael@kaist.ac.kr (S. Jung), aviv@kaist.edu (A. Segev).

http://dx.doi.org/10.1016/j.knosys.2014.04.036
0950-7051/

contribution to the performance of the prediction. To analyze the models, two citation networks from the Stanford Large Network Dataset Collection from High Energy Physics Theory (18,479 papers, 136,428 citations) and High Energy Physics (30,566 papers, 347,414 citations) are used.

The rest of the paper is organized as follows. The next section reviews the related work. Section 3 describes the methods used for analyzing future communities in citation networks. Section 4 presents the experiments on citation networks, and Section 5 provides a further discussion of the results. Finally, Section 6 provides some concluding remarks.

2. Related work

2.1. Topic Detection and Prediction

Topic Detection and Prediction has been studied in many research fields, to identify newly emerging topics and to capture the possible topics of given documents respectively. Topic Detection and Tracking (TDT) [15] is a multi-site research project aiming to predict novel topics. Its goal is to find a new topic in news systems by effectively identifying the first article or report mentioning the new topic [3]. There have been many studies using Natural Language Processing (NLP) topic detection approaches. The Adaptive Auto Regression (AR) model based on the Recursive Weighted Least Square (RWLS) method is presented to capture the Internet users' psychosocial attention behavior on how 'hot' topics such as 'Olympic Games' grow on the Internet [42]. The topic-conditioned First Story Detection (FSD) method in conjunction with a supervised learning algorithm [44,45] and Document Clustering [46] are used to identify the earliest report on a certain event in news articles. Other methods are also used in topic predictions. Survey analysis has been used to predict the result of a presidential election [26].

2.2. Topic modeling

Latent Dirichlet Allocation (LDA) [5] is a generative probabilistic model using a three-level hierarchical Bayesian model to model multiple topics from collections of text corpora. Its expandable nature has enabled many researchers to build models on top of it. Hierarchical LDA [6] creates a hierarchy of topics using a random partition process called the Chinese restaurant process. LDA-dual model [38] is an extension of the LDA model introduced to simultaneously deal with two types of text to solve the author disambiguation problem. Labeled LDA (L-LDA) [34] is an extension of multinomial Naïve Bayes supervised LDA where topics are constrained to those that directly correspond to the labels of a given document. Spatial LDA (SLDA) [43] is used to graphically categorize and identify images by treating images as documents and partial sections as words.

Commonsense knowledge, or *human knowledge*, is introduced in opinion mining [9] to catch topics incomprehensible by statistical textual models, such as poetry. LDA with WordNet (LDAWN) [8] incorporated the word sense into LDA by using the WordNet lexicon. Commonsense-based Topic Modeling [33] uses human commonsense data instead of common a *bag-of-words* model. While LDA solved some of the issues such as the overfitting problem, its performance still depends on the volume of the given text corpus with which it is trained. The model proposed here does not require training and can capture the meaning of phrases such as *"getting fired"* as opposed to *bag-of-words* based models such as LDA.

2.3. Topic identification

Topic detection focuses on finding a new topic, provided by either the experimenter [19] or the NLP method. Generative models are used to generate documents by selecting a distribution over topics and then selecting each word in the document from a topic chosen according to this distribution [19]. Generative models are used to analyze research paper abstracts from Proceedings of the National Academy of Sciences (PNAS) in order to generate a number of topics which successfully resemble the data structure. Identifying communities in web pages revealed that the communities exhibit hierarchical topic generalization characteristics, showing that the communities in a general setting are shown to reveal common properties of their members such as a common viewpoint or related topics [18]. Dynamic Community Identification [4] can therefore have a large role in topic identification. The conventional definition of communities as "*unusually densely knit subsets of a social network*" is argued as misleading in dynamic social communities in [41], which proposed an optimization-based approach for modeling dynamic community structure; it is shown to accurately track the dynamic community structure of social networks.

2.4. Link prediction

Link prediction models the evolution of a network using its topological characteristics and primarily deals with the prediction of edges between existing nodes. There are a number of different approaches to link prediction [27]. The shortest path between two nodes in a graph is a simple measure of link prediction. Some methods, such as Common Neighbors [30], Jaccard's coefficient [36], Preferential Attachment [30], and Adamic/Adar [1], use the node neighborhood information. The whole path within the network can also be used in link prediction, for example Katz [21], Simrank [20], and Rooted PageRank [27]. Common to those algorithms is that they do not deal with addition of nodes and deletion of edges. Their purpose is to generate a ranked list of predictive edges between existing nodes in a given network. The Community Prediction Method in the Citation Networks section outlines the differences between these methods and the contribution of each of these methods to the prediction.

2.5. Community detection

Community detection searches structural information of a given graph to partition it into sub-graphs called communities or modules [23]. Agglomerative methods and divisive methods are commonly used in community detection. Newman's community detection algorithm [31] is a widely used agglomerative method that uses modularity as the quality function. The recently developed Louvain method [7] is an agglomerative method and is commonly used because of its low computational complexity and high performance. When merging communities, this method considers not only the modularity but also the consolidation ratio. These algorithms, however, do not consider temporal information and disregard important factors such as consistency. Community Evolution [40] and Evolutionary Clustering [10,12,13] take the temporal changes in networks into consideration. There have also been studies about utilizing communities with link predictions. Family and friendship ties can be regarded as known community structures and are shown to help in predicting links in social networks [47]. The current work proposes a set of models based on temporal graphs and community prediction techniques.

3. Community prediction method in citation networks

Citation networks are directed social networks [32] between research papers, with nodes as papers and edges as citations between them. It is a form of a network where link prediction can be applied without extra considerations about deleted edges

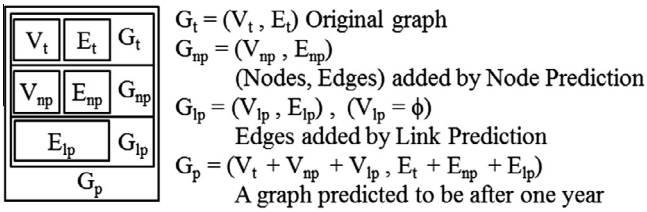

Fig. 1. Illustration of a predicted graph.

or nodes; nodes and edges never disappear from citation networks. Community detection algorithms have been proven to detect communities in existing social networks [35] as well. The goal of this research is to predict changes in community structures of citation networks by utilizing link prediction and community detection algorithms. The proposed model is explained next.

A graph G at certain time step t is represented by $G_t = (V_t, E_t)$ and is composed of a set of nodes V_t and a set of edges E_t. Three time steps $t-1 < t < t+n$ are chosen; G_{t-1} and G_t are the test set and G_{t+n} is the ground truth n time steps later. Fig. 1 illustrates how the graph grows with node and edge prediction. The node prediction algorithm explained in Section 3.1.1 is first run on G_{t-1} and G_t. The set of predicted nodes V_{np} is connected to existing nodes V_t by corresponding edges E_{np} where $G_{np} = (V_{np}, E_{np})$, a component of the predicted graph. G_{np} combined with G_t represents the citation network after node prediction is made. It is fed to a set of link prediction algorithms explained in Section 3.1.2. A list of predicted edges is filtered so that only edges $e_{source,target} \in E_{lp}$ with start point $source \in V_{np}$ and endpoint $target \in V_t$ remain. This filtering is necessary to mimic the characteristics of citation networks where new edges can only form from a new node to existing ones. Only edges are added; hence a graph component added in link prediction G_{lp} can be represented as $G_{lp} = (\phi, E_{lp})$. Merging G_{lp} with previous graph G_t and G_{np} forms G_p, a graph predicted to be at time step $t+1$. This process is repeated n times (with G_p instead of G_t after the first iteration) to predict a graph $t+n$ time steps in the future. Community detection algorithms explained in Section 3.1.3 are then used, in turn returning a predicted community structure C_p. Each community c_i in C_p is a subset of nodes $V_p = \sum c_i$ in a given graph G_p, with i ranging from 1 to the number of communities in C_p.

3.1. Base method

3.1.1. Node prediction

In this paper, a new method is presented to predict the list of nodes in a citation network predicted to appear in a future timestamp. The cumulative nature of citation networks suggests that an edge $e(i, j) \in E_t$ in each citation network $G_t = (V_t, E_t)$ represents a citation from paper i to j created up to time step t. As a paper cannot cite another paper after its publication, all edges $e(i, j)$ created in time step t must have a node created in the same time step as its start point i. The link prediction method does not deal with creation of nodes; hence additional consideration is necessary to deal with such nodes.

A simple heuristic is used to predict the number of nodes to be added in the next time step, since the graph continues to grow and the number of new publications (nodes) per time step stays about the same. The number of nodes to create (ΔV for future reference) can be predicted as below;

$$\Delta V = |V_t| - |V_{t-1}|$$

where $|V_t|$ represents the number of nodes in V_t.

Node-specific information is not required since it is impossible to predict the labels of new nodes. Added nodes V_{np} are given new unique ids to identify themselves from existing nodes V_t. Then the set of edges E_{np} is created to connect V_{np} to V_t. This step is necessary as nodes unconnected to the main graph contain no structural information and hence will not receive any attention during link prediction. At least one edge should be added for each predicted node to connect them to the given graph. Based on well-known Preferential Attachment, the "rich-gets-richer" phenomenon in research society, initial citation counts have impact on future citations [2]; hence a highly cited paper has a greater chance to be cited again. Nodes with higher in-degree count are considered to have higher probability of having an in-bound edge from a new node. The Kronecker Graph generation method [25] can capture more graphical features of a given graph, but the number of nodes is increased exponentially [37]; a citation network does not grow in such a way. Hence the method is not used in this paper.

Fig. 2 shows an example of node prediction based on a graph at $t-1$ (Fig. 2a) and t (Fig. 2b). Numbers written in the nodes show their in-bound edge count. In this example, one node (indicated by a red circle) is added in Fig. 2b; hence one node is predicted to appear in G_{t+1} (Fig. 2c).

For every node created, an outgoing edge is also created. When $\Gamma(v)$ consists of the inbound neighbors of the node v, each node v in V_t has a select factor s_p proportional to the number of inbound neighbors $|\Gamma(v)|$ to become an endpoint for such edges; papers that have never been cited before – nodes with no inbound edges – have $s_p = 0$ and hence are never selected. In Fig. 2c, four candidates for E_{np} shown with dotted lines have varying width, and one is chosen for G_{np} to connect the new node to the original network with probability proportional to the in-degree count of endpoint nodes (Fig. 2d). In one of the alternative methods explained in Section 3.2.1, this module is modified so multiple edges are added instead of one. The performance changes are analyzed in the experiments.

The result is a set $G_{np} = (V_{np}, E_{np})$ with nodes V_{np}, each connected to one of the original nodes by edge set E_{np}. G_{np} with G_t forms a predicted graph after node prediction is completed. Link prediction is then run on the network G_t and G_{np} combined in order to predict new outgoing edges from such new nodes.

3.1.2. Link prediction

After the node prediction is completed, we analyze the resulting network using the six link prediction algorithms summarized in Table 1. We chose these methods to provide different perspectives for our analysis of the link prediction.

Link prediction models the evolution of a network using its topological characteristics and primarily deals with the prediction of edges between existing nodes. There are a number of different approaches to link prediction [27]; with the shortest path between two nodes in a graph as a simplest measure of link prediction. Common to these algorithms is that they do not deal with addition of nodes and deletion of edges. Their purpose is to generate a ranked list of predictive edges between existing nodes in a given

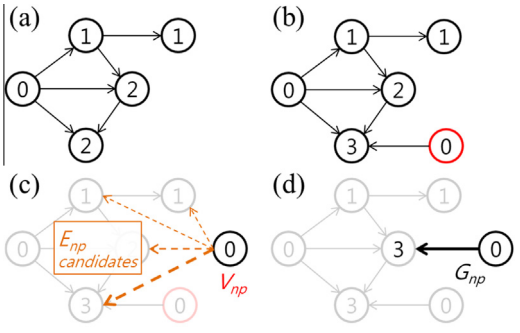

Fig. 2. Example of the node prediction process (the incoming node has to change to 3).

Table 1
List of link prediction algorithms.

Common Neighbors	$\lvert\Gamma(x) \cap \Gamma(y)\rvert$
Jaccard's Coefficient	$\lvert\Gamma(x) \cap \Gamma(y)\rvert / \lvert\Gamma(x) \cup \Gamma(y)\rvert$
Adamic/Adar	$\sum_{z \in \Gamma(x) \cap \Gamma(y)} (1/\log\lvert\Gamma(z)\rvert)$
Preferential Attachment	$\lvert\Gamma(x)\rvert \cdot \lvert\Gamma(y)\rvert$
Katz$_\beta$	$\sum_{\ell=1 \text{ to } \infty} \beta^\ell \cdot \lvert\text{paths}_{x,y}^{(\ell)}\rvert$ where $\text{paths}_{x,y}^{(\ell)}$:= {paths of length exactly ℓ from x to y} weighted: $\text{paths}_{x,y}^{(\ell)}$:= number of collaborations between x and y unweighted: $\text{paths}_{x,y}^{(\ell)}$:= 1 iff x and y collaborate
Rooted PageRank$_\alpha$	Stationary distribution weight of y under the following random walk: with probability α, jump to x with probability $1-\alpha$, go to a random neighbor of current node
SimRank$_\gamma$	1 if $x = y$ $\gamma \cdot \sum_{a \in \Gamma(x)} \sum_{b \in \Gamma(y)} \text{score}(a,b) / (\lvert\Gamma(x)\rvert \cdot \lvert\Gamma(y)\rvert)$ otherwise

network. The Community Prediction Method in the Citation Networks chapter outlines the differences between these methods and the contribution of each of these methods to the prediction.

3.1.2.1. Methods utilizing node neighborhood. Some methods use the node neighborhood information in order to perform link prediction. The Common Neighbors method [30] is born from an idea that any two nodes will likely be connected by a link in the future if they have a large number of Common Neighbors. With $\Gamma(x)$ representing the set of neighbors of the node x, Common Neighbors calculates the connection weight of $\langle x, y\rangle$ node pairs by calculating

$$\text{score}(x,y) := |\Gamma(x) \cap \Gamma(y)|$$

A more evolved idea of Common Neighbors is Jaccard's coefficient [35]. The Common Neighbors method cannot distinguish node pairs $\langle x, y\rangle$ and $\langle v, w\rangle$ when two pairs have the same ten Common Neighbors when the nodes in the former pair have ten neighbors each – in which case every one of their neighbors is common – while the nodes in the latter pair have hundreds of neighbors each. Jaccard's coefficient calculates the relative fraction of Common Neighbors by considering the number of total neighbors in both nodes; the formula is

$$\text{score}(x,y) := |\Gamma(x) \cap \Gamma(y)| / |\Gamma(x) \cup \Gamma(y)|$$

Adamic/Adar is born from a similarity measure for websites and is similar to Jaccard's coefficient in principle. The Adamic/Adar method refines the measurement by weighting rarer features – nodes with smaller neighbor counts more. Initial measurement introduced $\sum_{z:\text{feature shared by } x,y} (1/\log(\text{frequency}(z)))$ is altered to form the formula

$$\text{score}(x,y) := \sum_{z \in \Gamma(x) \cap \Gamma(y)} (1/\log(\Gamma(z)))$$

Preferential Attachment [30] is a network growth model used for graph generation as well as link prediction. The basic assumption of this measure is the simple heuristic thought of the "rich-gets-richer" phenomenon found in various areas such as Physics [14] and Ecology [39]. In Preferential Attachment, a node x is more likely to have a new link pointing towards it as its neighbor count $\Gamma(x)$ grows. Further proposed by Newman [30], the measure is defined as

$$\text{score}(x,y) := |\Gamma(x)| \cdot |\Gamma(y)|$$

3.1.2.2. Methods utilizing the whole path within the network. Katz [21] more or less has a similar notion to the Common Neighbor with one difference; Common Neighbor considers Common Neighbors, which can be translated as paths with length 2, but Katz considers paths with multiple lengths where $\text{paths}_{x,y}^{l}$ represents the set of all paths from x to y with length l. To count the shorter paths heavier, the score is exponentially damped by path length with damping factor $\beta > 0$; when β becomes small enough, paths with length more than 2 will mostly be damped out and the result will be much like Common Neighbors.

$$\text{score}(x,y) := \sum_l \beta^l \cdot |\text{paths}_{x,y}^{l}|$$

Rooted PageRank [27] and Simrank [20] use the notion of random walk. Rooted PageRank is a stationary distribution propagated by random walk with reset. Simrank measures when two random walks with different starting nodes first meet. Two nodes are considered to be similar based on how much they are connected to similar neighbors. The similarity of two nodes are calculated as follows; one node has a similarity of 1 with itself, and the similarity of two different nodes is positively correlated with the sum of all the similarity scores of their neighbor pairs and is negatively correlated with the product of their neighbor counts.

$$\begin{aligned}\text{score}(x,y) &:= \text{similarity}(x,y) : \\ &= r \cdot \sum_{a \in \Gamma(x)} \sum_{b \in \Gamma(y)} \text{similarity}(a,b) / (|\Gamma(x)| \cdot |\Gamma(y)|)\end{aligned}$$

3.1.3. Community detection

This paper uses two algorithms of the Louvain method [22], Louvain-smallest algorithm and Louvain-best algorithm, to analyze the community detection. The Louvain-smallest algorithm returns a smallest partition of the graph and the Louvain-best algorithm returns a more coarse-grained partition where the graph is partitioned into fewer (and larger) communities.

A list of communities C is produced from graph G with community detection algorithms. Each community $c_i \in C$ consists of a subset of nodes V in a given graph G. In this model, community detection is implemented with both the predicted network and with the true result. The lists of communities are then compared to evaluate the result; G_p is fed to community detection algorithms to produce C_p while G_{t+n} is used to create C_{t+n}.

3.2. Methods based on combinations of building blocks

Three alternative methods of the given model, referred as the *nlc-method*, are also proposed in this paper. Fig. 3 outlines how the modules explained above are used as building blocks. '**N**'ode prediction (Section 3.1.1), '**L**'ink prediction (Section 3.1.2), and '**C**'ommunity detection (Section 3.1.3) are each visualized as a block, and each method consists of a series of modules used in left-to-right sequential order. For example, the base method in Fig. 3 requires the node prediction module to be used first, the link prediction module second, and then the community detection module.

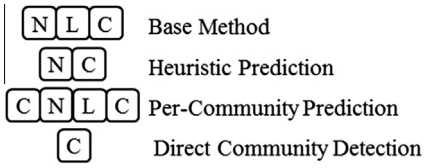

Fig. 3. Outline of methods.

3.2.1. Heuristic prediction

The *nlc-method* adds an edge per node predicted by the node prediction module. The purpose of these edges is to act as catalysts in the link prediction module, allowing newly added nodes to have a chance of getting a non-zero similarity score with other nodes. This enables the link prediction module to predict more edges connected to the new nodes.

In the heuristic prediction method (the *nc-method*), it is assumed that the preferential-attachment approach used in the node prediction module is able to predict edges to some extent. To test this hypothesis, the link prediction module is omitted and the node prediction module is modified in this method; more edges are added in the node prediction module to compensate for the loss of predicted edges. The number of edges to add per predicted node in time step t is calculated as $m = \Delta E/\Delta V$ where $\Delta E = |E_t| - |E_{t-1}|$. After the node prediction module is executed, adding m edges instead of one, G_{np} is set to be G_p, a predicted graph at time step $t + 1$. The same method is repeated on G_p n times to get a predicted graph at time step $t + 1$. The number of edges added per node is data dependent. If a fixed number is used, the method risks either not predicting enough edges in a complex graph or predicting too many edges in a simple graph, resulting in poor prediction performance.

3.2.2. Per-Community Prediction

A better result is obtained when link predictions are done on top of known communities in a network [47]. The Per-Community Prediction method (the *cnlc-method*) is proposed to evaluate whether communities found by community detection methods can also be used to improve the accuracy of edges added in the link prediction module.

The *cnlc-method* is the same as the *nlc-method*, except that this method has an additional step before any predictions are made. Community detection algorithms are used first on G_t (replaced by G_p after the first iteration) to detect the community structure C_t of a given network. Then V_{np} is distributed to each community according to the number of its membership nodes. The total number of new nodes $|V_{np}|$ is multiplied by the relative size of each community $(|c_i|/|V_t|)$ when the number of nodes in a community is represented by $|c_i|$ $(c_i \in C_t)$. $|V_{np}| \cdot (|c_i|/|V_t|)$ new nodes are assigned to each community c_i. The node prediction algorithm is used per communities to connect the assigned number of new nodes to the community. After the algorithm is used for all communities, the nodes and edges added per communities are joined to form G_{np}. The method is identical to the *nlc-method* afterwards.

3.2.3. Direct community detection

Direct community detection uses only the community detection algorithms to test whether the community detection alone would be able to predict future communities in a network. This method is named the *c-method*.

Without producing G_{np} and G_{lp}, the community detection module is executed with G_t to produce C_t, a set of communities in time step t. It is used to predict communities in time step $t + n$.

3.3. Community labeling

In citation networks, keywords of the papers can be useful in labeling the detected communities. Many publications include *keywords* representing the main concepts in each paper for indexing purposes. With labels treated as topics in topic detection methods, a community can be distinguished by a set of common keywords appearing among its papers. A similar approach is adopted in this paper. The labeling methods used in this paper utilize the title and abstract of the research papers in order to find a list of terms – keywords – using natural language processing. For each community C, $text_c$ on which NLP is performed is created by concatenating the title and abstract of C's membership papers c.

Preprocessing is performed on the terms list by first tokenizing each document with the space character as the separator and then removing any token with length less than 3 in order to remove obvious stop words. By using a large enough corpus of documents, irrelevant terms are more distinct and can be thrown away with a higher confidence. To extract a list of representative keywords, or *labels*, for each community the following methods were used: TF/IDF (Term Frequency/Inverse Document Frequency), Concept Tagging, and Keyword Extraction.

TF/IDF is widely used in IR to extract a list of representative keywords from a corpus of documents and is well known for its robust performance on a large enough dataset. The inverse document frequency is calculated as the ratio between the total number of documents and the number of documents that contain the term. TF/IDF is obtained by dividing each term's term frequency by its inverse document frequency, with $text_c$ of each community as a document.

The Concept Tagging method creates lists of the main concepts to mimic human-based tags. The Keyword Extraction method extracts all the topic keywords to index the content. The Keyword Extraction is based on words in the text, while the Concept Tagging can include concepts represented as words that do not necessarily appear in the text. AlchemyAPI (http://www.alchemyapi.com/) is a popular natural language processing service via API, providing users with a rich suite of content analysis and metadata annotation tools such as semantic metadata extraction about people, places, and topics. Keyword Extraction and Concept Tagging are performed based on AlchemyAPI. Each method takes a text or URL as an input and returns a set of keywords or concepts respectively, with their relevance score varying from 0 to 1. In addition, we use the Keyword Extraction method to analyze the given text to find the sentiment score. The Concept Tagging method utilizes multiple algorithms such as semantic tagging, text mining, and machine learning techniques [29] and shows the related information about the found concept in the form of links to an external website such as DBpedia. Each method is run with $text_c$ of each community as an input text. Opinion mining tools such as Sentilo [16] or SenticNet [10] can also be used to enhance the outcome with a more sentiment-sensitive dataset. Sentiment classification can further be improved by incorporating domain-specific features and sample selection [44].

4. Experiments

4.1. Data

Two citation networks are taken from the Stanford Large Network Dataset Collection (http://snap.stanford.edu/data/), using the citation list from the High Energy Physics (hepPh) and High Energy Physics Theory (hepTh) sections of physics in e-Print arXiv archive. Table 2 shows detailed information. The selected research fields have a tendency to have heavy citations, and networks are

dense with the number of edges exceeding the number of nodes by factor of 7–10. This is a common characteristic of citation networks, as many papers cite multiple papers. Dense graphs increase the likelihood of formation of new communities.

The dataset contains a list of citation records, having a numeric id for each paper and the date of publication. Additional attributes are provided with hepTh dataset. This additional information can be used to label the communities found in hepTh.

4.2. Evaluation method

The random predictor method is implemented as a baseline against which the results of this model are compared. In the random predictor method, the number of nodes to be added is the same as in the other algorithms, but the link prediction module is skipped and the network is randomly divided into n clusters, where n is the number of communities found in the ground truth set from the given dataset. The link prediction module is skipped because it is not needed for the random predictor method to work; randomly partitioning a set of nodes does not need edge structure information. The baseline predictor is compared against community prediction methods in this paper. LPmade [28] is used to perform the link prediction algorithms for the experiment.

For community detection, the *Community detection for NetworkX* (http://perso.crans.org/aynaud/communities/) is used in this experiment, which uses the Louvain method to detect and cluster communities in *NetworkX* (http://networkx.lanl.gov/) format graphs. The Louvain method is an iterative two-module method; it first maximizes modularity by finding small communities, then coarsens the network and repeats the process until maximum modularity is achieved.

The comparison of the predicted communities against the true result is not straightforward. Identification of membership nodes is required to identify the same community in two graphs, but predicted nodes do not have the same label as their actual counterparts. Any predicted node in this model has a new id attached to them, and it is impossible to match predicted nodes to new nodes in the true result, even if they are structurally identical.

The Jaccard's coefficient-like method is introduced in this paper to counter this problem. A similarity score $\mathrm{sim}(c_i, c_j)$ is calculated as $|c_i \cap c_j|/|c_i \cup c_j|$ where $|c_i \cap c_j|$ is the number of nodes in both communities and $|c_i \cup c_j|$ is the number of nodes in either of two communities. Two communities are considered to have the same predecessor if they have $\mathrm{sim}(c_i, c_j)$ above a threshold $0 < threshold < 1$. It is set to 0.5 in this experiment. Communities detected from the ground truth are compared against the experiment result to produce F-score $F = 2 \cdot (p \cdot r)/(p + r)$ where $p = |c_{\mathrm{matched}}|/|c_{t+n}|$ and $r = |c_{\mathrm{matched}}|/|c_{\mathrm{ground\ truth}}|$.

4.3. Results

4.3.1. Node prediction

Fig. 4 illustrates the result of the heuristic node prediction algorithm on each dataset used in this paper. The X-axis represents the year, and the Y-axis represents the number of nodes. The heuristic method predicted the number of nodes with correlation coefficient $r = 0.98$ and 7.5% margin of error. Prediction performance is high in hepTh, but the module failed to capture a sudden change of actual node count in 2000 and 2002. The margin of error is 12% in hepTh. The drop of actual node count in 2002 can be explained in that the recording could have stopped before 2002 ended. Performance increases in the hepPh dataset. Discarding 2002 where the same drop occurs, the margin of error is 2.4% in hepPh with correlation coefficient $r = 0.99$.

The edges added at the node prediction stage, representing new citations, also show promising results. Fig. 5 shows the precision $p = E_{np} \cap E_{t+n}/E_{np}$ and recall $r = E_{np} \cap E_{t+n}/E_{t+n}$ value of edges added in the node prediction module E_{np} with precision value as the X-axis and recall value as the Y-axis. Dotted lines show the performance of node prediction in the *nc-method* where the number of edges added per node varies with given data, while solid lines show the performance of node prediction when 1, 5, and 10 edges are predicted per node. The large gap found in Fig. 4 suggests that the hepTh dataset in 2000 and both datasets in 2002 are incomplete. Precision and recall values of E_{np} indeed show inconsistent results when 2000 and 2002 data are tested; hence they are removed from Fig. 5. hepTh is found to have an outlier in 1998; hence year 1998 is also removed. In both datasets, performance improves as more edges are added per node. This is true with up to 10 edges added per node. hepTh starts with low precision and low recall in 1995, and both precision and recall increase as the years go by. With the exception of hepPh-1e (one edge added per predicted nodes), hepPh in 1995 shows high precision and relatively low recall, and precision decreases with recall increasing relatively more. This pattern is limited by the number of average citations per paper, which is up to 16 in year 2002. The F-score (with beta = 1) peaks when 10 edges are added, and both precision and recall drop when 15 edges are added. When more edges are added, the node prediction module starts to over-predict, causing the F-score to drop. The graph shows that the node prediction works better on more complex datasets and when the dataset grows in size. The *nlc-method* and the *cnlc-method*, however, add one edge per node in the node prediction module; this is intentionally done so the prediction of edges is performed in the link prediction module instead of the node prediction module.

4.3.2. Link prediction

The model further predicts edges in the graph by using the link predictors mentioned in Section 3.1.2. Mean precision $p = E_{lp} \cap E_{t+n}/E_{lp}$ and recall $r = E_{lp} \cap E_{t+n}/E_{t+n}$ of edges predicted in the edge prediction module are shown in Table 3. Rooted PageRank is used with random walk restart parameter $\alpha = \{0.01, 0.05, 0.25, 0.50\}$, and Katz is used with damping parameter $\beta = \{0.5, 0.05, 0.005\}$. Changes in parameters have no visible effect on either algorithm, and results with different parameters for each method are merged together.

Simrank returns 0 for both precision and recall in both datasets, because the predicted nodes are weakly connected to the network with only one neighbor in any method with the link prediction module. The number of predictors used in this experiment utilizes neighborhood information and thus tends to predict edges between existing nodes V_t that have more neighbors. These edges, as explained before, are filtered out to mirror the characteristics of citation networks. As a result, Adamic/Adar, Common Neighbor, and Jaccard's Coefficient failed to predict any new edges in 6 of the 16 test sets used. Simrank failed to predict any edge at all.

4.3.3. Community detection

Community detection algorithms are used on graphs generated by node and link prediction modules, and the evaluation method presented in Section 4.2 is used to evaluate the result. The performance of the Louvain-smallest algorithm is higher with hepTh but is lower with hepPh. This suggests that the community detectors

Table 2
Details of dataset used.

Name	No. of papers	No. of citations
hepTh	18,479	136,428
hepPh	30,566	347,414

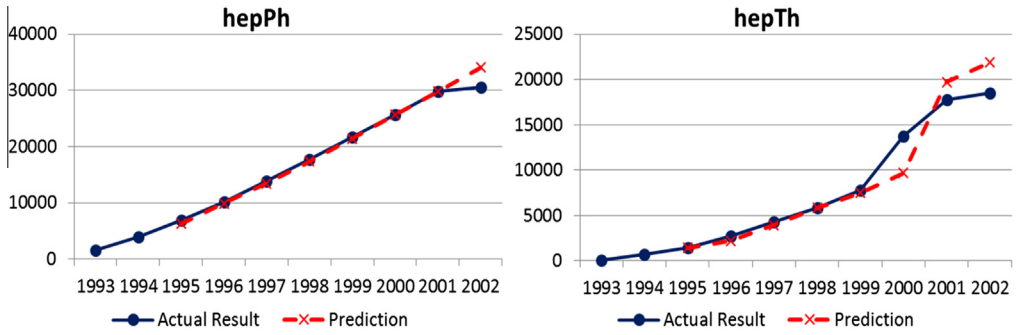

Fig. 4. Node count $|V_t|$ and $|V_{np}|$ per year in hepPh/hepTh dataset.

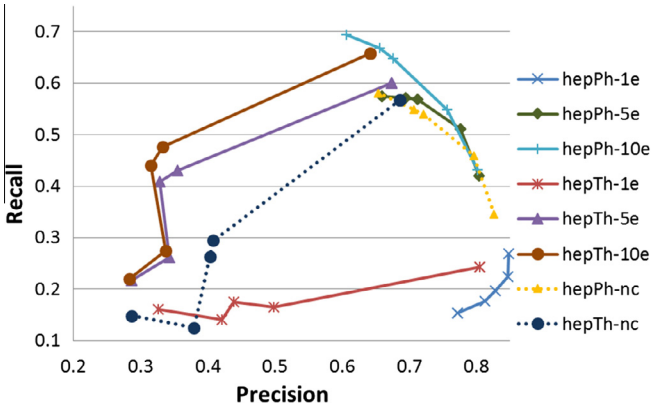

Fig. 5. Precision versus recall of edges (E_{np}) in node prediction module.

Table 3
Precision and recall of E_{lp}.

Predictor	hepTh		hepPh	
	Precision	Recall	Precision	Recall
Adamic/Adar	0.3443	0.0026	0.8584	0.0038
Common Neighbors	0.3443	0.0026	0.8584	0.0038
Jaccard's Coefficient	0.3443	0.0026	0.8584	0.0038
Katz	0.5721	0.1959	0.7148	0.3624
Preferential Attachment	0.5721	0.1959	0.7148	0.3624
Rooted PageRank	0.5721	0.1959	0.7148	0.3624
Simrank	0.0000	0.0000	0.0000	0.0000

have little effect on the overall performance of the model compared to the specific structures of the given network.

Using the community matching algorithm introduced in Section 4.2, Fig. 6 compares the *F*-score of following-year predictions made by the *nlc-method*, the *nc-method*, the *cnlc-method*, and the *c-method* with different combinations of community detection methods, datasets, and years as an *X*-axis and *F*-score as a *Y*-axis. The results of four methods are grouped at each column. The random predictor is not able to detect any communities in any of the test sets and hence is not presented.

The *c-method* outperforms other methods in every test. The *c-method* uses the current community structure to predict future communities, and this result proves that the communities do not change much in one year. The *nc-method* is the second best predictor in most of the cases; methods with more modules resulted in worse performance. It is also worth noting that the performance increases as the graph grows in each dataset, while the smaller hepTh dataset returns a higher performance compared to the larger hepPh dataset.

Fig. 7 illustrates how much the *F*-score decreases when predictions are made for graphs five years in the future, calculated by $Fscore(t+5)/Fscore(t+1)$. Fig. 7 shows that the performance drop ratio of the *c-method* over five years depends on which dataset is used; this supports an earlier statement that the community predictors have relatively less effect on the performance change. The *X*-axis shows the combination of community detection algorithms and datasets grouped by four methods presented in this paper. The *Y*-axis represents the *F*-score ratio of predictions 5 years in the future against predictions for the following year.

The *c-method* assumes that the community structure does not change over time. Analyzing the *c-method* in Fig. 7 suggests that the community structure changes more on the hepTh dataset. The Louvain-best algorithm in the hepTh dataset returned less than 5% of the initial prediction with all methods but the *c-method*, which returned over 20%. This suggests that the growth in hepTh dataset is more random compared to hepPh and the community structure found by the Louvain-best algorithm in hepTh dataset is prone to random alteration. Methods other than the *c-method* change the community structure by adding nodes and edges and are unable to effectively mimic the growth of such graphs without introducing random factors large enough to alter the community structure of the graph.

While the absolute *F*-score on the Louvain-smallest algorithm in Fig. 6 was generally lower than that of Louvain-best algorithm, it is shown to have a lower performance drop over the years. The *nc-method* and the *cnlc-method* are able to retain 60% of original prediction performance after 5 years in hepPh dataset with the Louvain-smallest algorithm. This result is opposite that of the Louvain-best algorithm with hepTh; the *c-method* shows the largest performance drop in this combination. This shows that the *nc-method* and the *cnlc-method* work better as the size of a graph

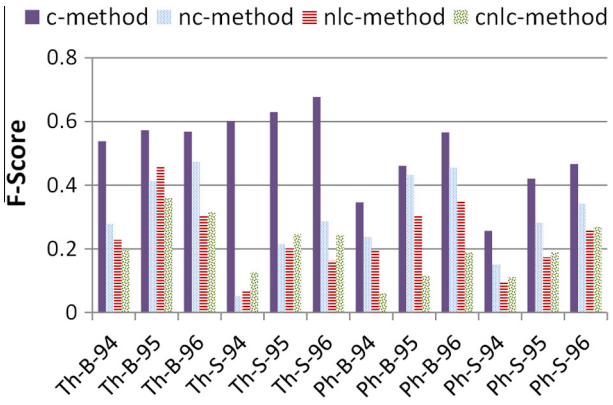

Fig. 6. *F*-score of 4 methods with different community detectors in hepTh/hepPh dataset for 1 year prediction.

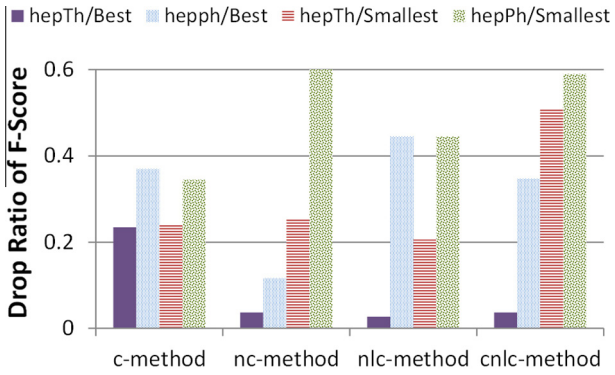

Fig. 7. *F*-score ratio when prediction at $t + 5$ is compared against prediction at $t + 1$, with threshold = 0.5

grows and as a graph is more fine-grained into more communities, each with smaller sizes.

In short, the *c-method* can be used to predict the community structure in the near future. In the larger graph with fine-grained communities, the *nc-method* (lower computational complexity) or the *cnlc-method* (better result with wider range of input) can be used to predict further into the future.

4.3.4. Identifying emerging and disbanding communities

The variance of the resulting performance for emerging/disbanding communities in the citation networks is also tested. The *nc-method* and the *nlc-method* are analyzed with preferential-attachment, which is used in the link prediction module. The Louvain-best algorithm is used in the community detection module, and months starting March 1996 from the hepTh dataset are used as time steps.

Membership nodes in two compared communities c_1 and c_2 are first divided into two subsets, each containing (1) a series of nodes that were present before any prediction is made c_{old} and (2) the newly added nodes c_{new}. As shown in Table 4, $sim_{old}(c_1, c_2)$ and $sim_{new}(c_1, c_2)$ are calculated from respective node subsets from which $sim(c_1, c_2)$ is derived. $sim_{old}(c_1, c_2)$ and $sim_{new}(c_1, c_2)$ are each weighted with weight constant w_{old} and w_{new} respectively. The resulting formula is $w_{old} \cdot sim_{old}(c_1, c_2) + w_{new} \cdot sim_{new}(c_1, c_2) = sim(c_1, c_2)$, where $w_{new} + w_{old} = 1$ so that $0 \leqslant sim(c_1, c_2) \leqslant 1$.

$sim_{old}(c_1, c_2)$ is calculated in the same way explained in Section 4.2, replacing c_1 and c_2 with $c_{1,old}$ and $c_{2,old}$. $sim_{new}(c_1, c_2)$ uses a different approach, since it is dealing with a set of new nodes with random identifiers; comparing nodes with their identifiers is unfavorable. Only the number of nodes in the community is considered, and $sim_{new}(c_1, c_2)$ is therefore calculated as $\min(c_{1,new}, c_{2,new})/\max(c_{1,new}, c_{2,new})$ with exceptional case where $sim_{new}(c_1, c_2)$ is set to 1 if $\max(c_{1,new}, c_{2,new})$ is 0.

Fig. 8 shows the result of the *nlc-method* with the different weight combination of the new community detection algorithm having $w_{old} = 0$ and $w_{new} = 1$ returning the best result.

At each timestep, any newly emerged and disbanded communities are identified. Fig. 9 shows that both methods in question (the *nc-method* and the *nlc-method*) show similar performance in detecting emerging communities, with the *nlc-method* starting to outperform the *nc-method* after about the 45th iteration. This result suggests that the link prediction module used in the *nlc-method* is better than the modified node prediction module used in the *nc-method*; extrapolating the given citation network with more accurate network properties such as average node degrees and distance.

Fig. 9 also shows the *F*-score of communities found to be formed and disbanded. The disbanded result is very high in either method,

Table 4
Community matching scheme.

	c_1	c_2	output
Existing nodes	$c_{1,old}$	$c_{2,old}$	$sim_{old}(c_1, c_2)$
New nodes	$c_{1,new}$	$c_{2,new}$	$sim_{new}(c_1, c_2)$

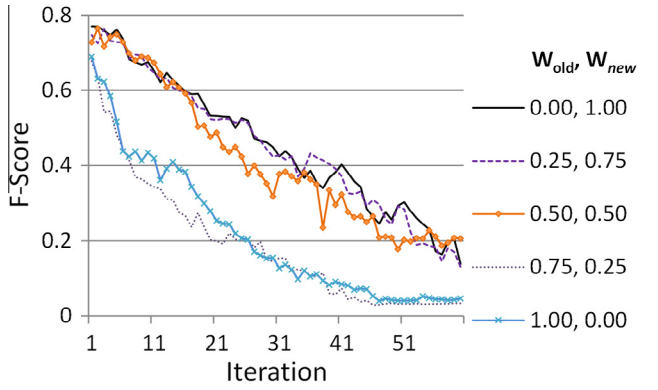

Fig. 8. *F*-score of the *nlc-method* with different w_{old} and w_{new} values.

increasing as the detection goes further. This shows that it is easier to detect communities that will be disbanded in the future than to detect communities that will be formed in the future. At the same time, this result also points to a limitation of this experiment; each community is not tracked throughout the experiment but rather is individually identified at each timestep. With network structure continuously distorting at each timestep, it is possible that the communities identified do not reflect the past changes in their structure, influencing the output result.

4.3.5. Community labeling

A sample community labeling test is done to determine the possibility of community labeling with low computational cost. Three methods are compared: Keyword Extraction, Concept Tagging, and TF/IDF as the basic baseline. Table 5 presents the top four relevant labels representing five randomly picked communities extracted by each method. Each line in a group is related to the same community. TF/IDF treats each keyword the same, and hence most of the returned keywords do not hold specific meaning in the community ('Note', 'Their', 'Master' and 'Form' are such examples). On the other hand, Keyword Extraction and Concept Tagging show better results, returning more domain-specific terms such as

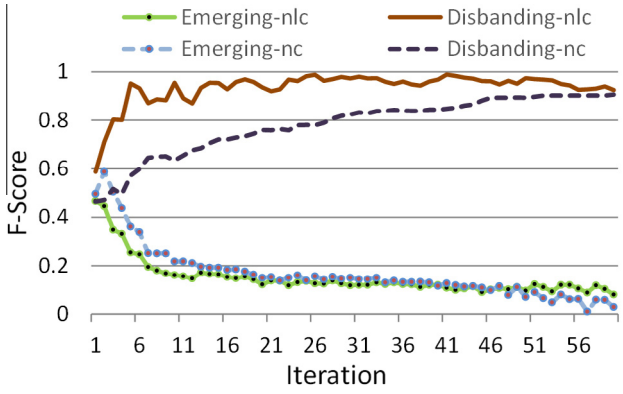

Fig. 9. *F*-score of newly emerged and disbanded communities in the nc/nlc-method with $w1 = 0.0$ and $w2 = 1.0$.

Table 5
Sample labels extracted using TF/IDF, Concept Tagging, and Keyword Extraction.

TF/IDF	Uncertainties	Dependence	Note	Background
	R-Matrix	Cal	Hierarchy	Their
	Higher	Correlators	Genus	Matrix
	Equation	Master	Phases	Unitarity
	Nonlinear	General	Scalar–Tensor	Form
Keyword Extraction	Black Hole	Theory	String Theory	Non-Fluctuating Modes
	KP Hierarchy	Hamiltonian Structures	System R-Matrix Formulation	KP Hierarchies
	Model	Matrix Model	External Field	Complex Matrix Model
	Theory	String	Matrix Model	String Theory
	Scalar–Tensor Quantum Gravity	Quantum	Gauge Theory	Nonlinear Gauge Theory
Concept Tagging	String Theory	Statistical Mechanics	Fundamental Physics Concepts	Entropy
	Structure	Group	Standard Model	Quantum Field Theory
	Mathematics	Derivative	Generating Function	Complex Number
	Algebraic Structure	Geometry	Representation Theory	Lie Algebra
	String Theory	General Relativity	Quantum Field Theory	Vector Space

'Fundamental Physics Concepts', 'Representation Theory', 'KP Hierarchies', and 'Scalar–Tensor Quantum Gravity'.

Representing a series of labeled communities in such format becomes very text intensive as the number of communities increases, and a number of techniques have been tried to visualize the labeled communities. 15 false positive communities found at the 10th timestep (December 1992) are used as a sample data. Figs. 10 and 11 show the bar representation of labels extracted by the Keyword Extraction and Concept Tagging methods. Each figure shows every label found among 15 communities in a single row, and labels relevant to the community are colored according to their relevance. Fig. 11 with 53 columns (concepts) shows that few concepts appear through many communities. Considering the data as the false-positive result, such concepts are likely to be the common reason why they were falsely identified. Fig. 10 shows less of such a pattern but is more difficult for a person to comprehend, with about double the number of columns (103 keywords).

A star diagram visualization method is shown in Fig. 12, using top five labels from each community. Because the communities do not share the same labels to represent themselves, each 'star' is visible at 5 different slots of a total of 36 slots. It is possible to differentiate communities with different coloring, but more comprehensive techniques such as changing the label ordering can further improve the outcome. This is left for future research.

5. Discussion

The experiments showed that citation networks can be used to successfully predict communities up to 5 years into the future. The contribution of the prediction methods to the success of the results can be analyzed according to each building block. The node prediction method shows promising results.

The experiments showed that the performance of the methods differs considerably based on the prediction time span. Short term predictions for a single year should use clustering (the *c-method*). The advantage of this method can be attributed to the communities' slow pace of change in research that is represented by the change in the citation networks. However, as the prediction span increases, the performance of the per-community method (the *clnc-method*) increases much faster than that of the clustering method. Since the pace of new research topics varies between research fields, the assumption is that for slow changing fields with short prediction spans the *c-method* would be more useful while for faster changing fields and longer prediction spans the per-community method would be better.

The results indicated a limitation of the method: when there is a sudden extreme change in the number of nodes appearing in one year, then there is a gap between the predicted results and the true results. The experiments show that after one year the gap is minimized. One possible way to minimize the error in such cases is to consider the average change in multiple years in the past.

Although the node prediction module achieves high performance, the link prediction achieves lower performance. Use of different link prediction and community detection algorithms could increase the overall performance of community prediction. One possible solution for community detection is evolutionary clustering [13,24], which takes temporal consistency of the communities into account.

Analysis of Fig. 7 implies that all the methods work better on a more stable network. Methods introduced in this paper should be tested on more active citation networks where community

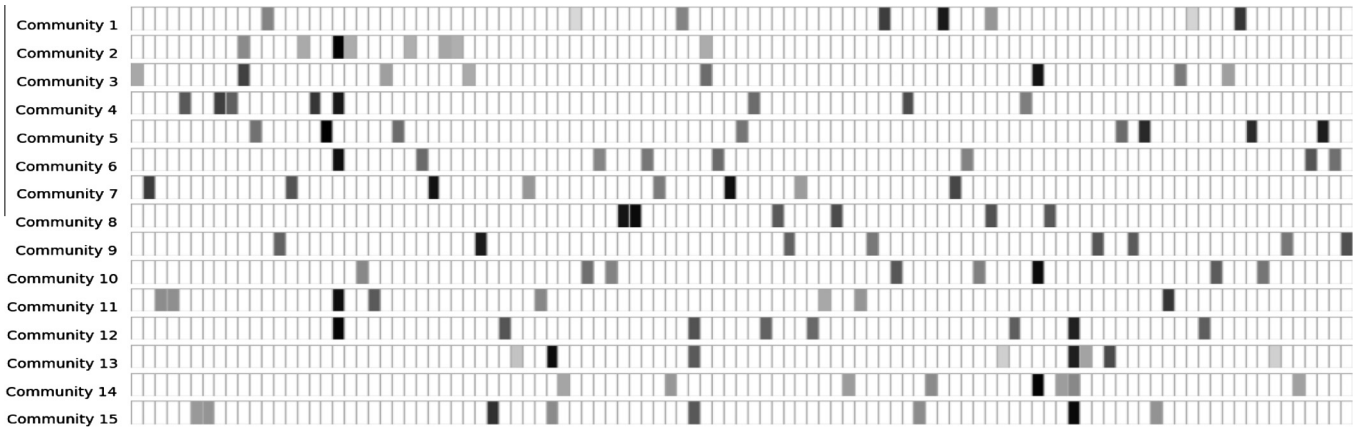

Fig. 10. Bar representation of labels of false-positive results with the Keyword Extraction method.

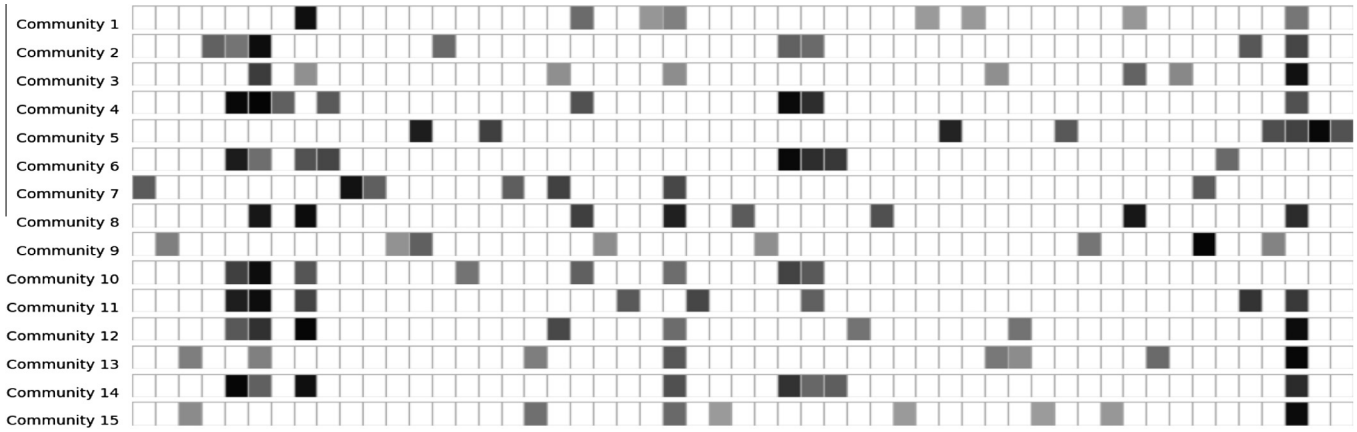

Fig. 11. Bar representation of labels of false-positive results with the Concept Tagging method.

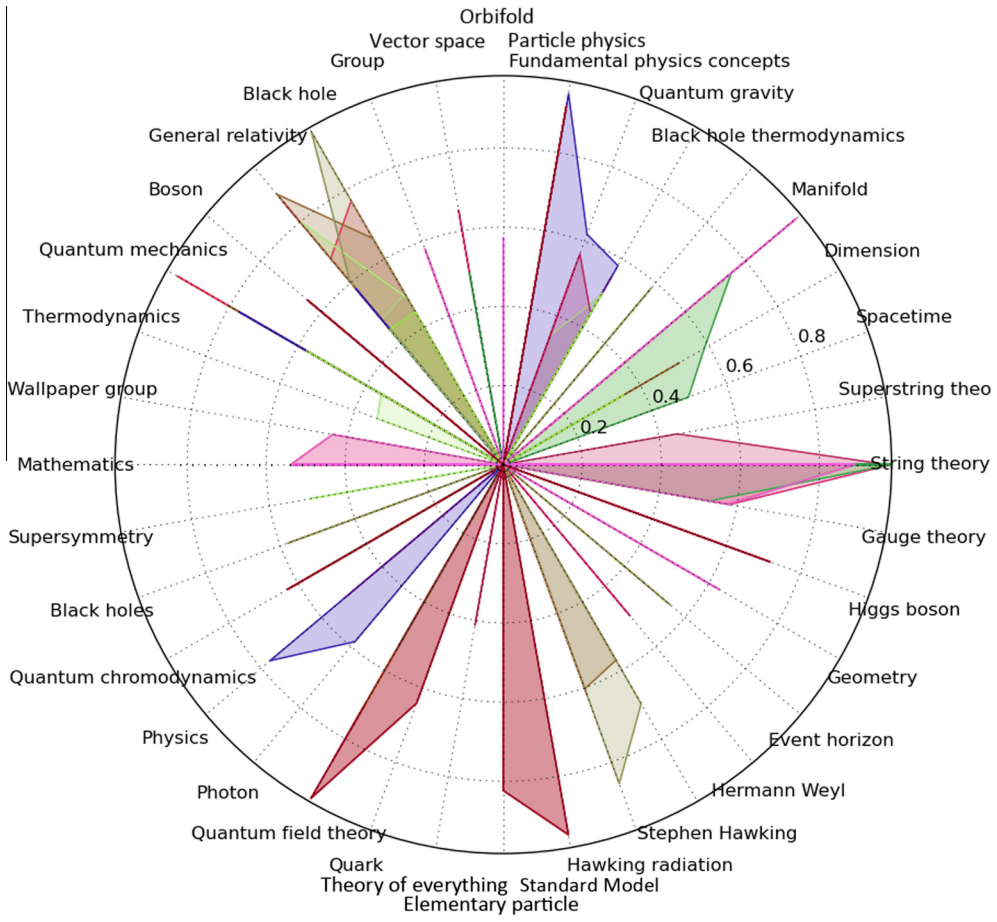

Fig. 12. Radar graph of labels of false-positive results.

structure frequently changes. Suggestions for additional work include the analysis of whether the break-even point for the best community detection method depends directly on the community size. The labeling is based on Keyword Extraction from existing papers. However, some of the nodes represent futuristic papers that have no attributed keywords. The labeling should take into consideration the number of existing paper nodes versus predicted nodes. It is possible that this problem can be alleviated by utilizing narrative-based NLP instead of the semantic NLP techniques [11]. The processing time of the algorithm is dependent on the number of citations and links. Of the two datasets, one was approximately double the size of the other, and therefore the performance time was approximately twice as long in all methods, thus indicating a linear complexity. The time required to analyze the biggest data set using the method that requires the longest processing time was approximately 10 min.

6. Conclusion

The paper presents a model for analyzing future communities in a citation network. The model includes a heuristic node prediction method, link prediction methods, and community detection meth-

ods, which are combined in various ways. Four community analysis methods are proposed. The analysis methods in the model show promising results in analyzing the possible communities in the future. The analysis time span was found to be a considerable factor in the performance of the community analysis methods. The tracking of community propagation shows good results, and the method can be used to predict the topic propagation in research fields with added labeling features, which have practical applications such as letting businesses invest funds in promising research areas.

Directions of future research include addressing datasets in different fields and creating a measurement for the performance of each method derived from the characteristics of the network such as size, centrality, and consistency. Another possible direction is to investigate the break-even point for community size versus different combinations of analysis methods.

Acknowledgements

This work was supported by the IT R&D program of MSIP/KEIT [10044494, WiseKB: Big data based self-evolving knowledge base and reasoning platform].

References

[1] L. Adamic, E. Adar, Friends and neighbors on the web, Soc. Netw. 25 (3) (2003) 211–230.
[2] J. Adams, Early citation counts correlate with accumulated impact, Scientometrics 63 (3) (2005) 567–581.
[3] J. Allan, J. Carbonell, G. Doddington, J. Yamron, Y. Yang, Topic detection and tracking pilot study final report, Evaluation (1998) 194–218.
[4] T. Berger-wolf, C. Tantipathananandh, D. Kempe, Dynamic community identification, in: P.S. Yu, J. Han, C. Faloutsos (Eds.), Link Mining: Models, Algorithms and Applications, Springer, New York, 2010, pp. 307–336.
[5] D.M. Blei, A.Y. Ng, M.I. Jordan, Latent Dirichlet allocation, J. Mach. Learn. Res. 3 (March) (2003) 993–1022.
[6] D.M. Blei, T.L. Griffiths, M.I. Jordan, J.B. Tenenbaum, Hierarchical topic models and the nested chinese restaurant process, Adv. Neural Inform. Process. Syst. (2003).
[7] V.D. Blondel, J.L. Guillaume, R. Lambiotte, E. Lefebvre, Fast unfolding of communities in large networks, J. Stat. Mech.: Theory Exp. 2008 (10) (2008) P10008.1–P10008.12.
[8] J.L. Boyd-Graber, D.M. Blei, X. Zhu, A topic model for word sense disambiguation, in: EMNLP-CoNLL, vol. 17, 2007.
[9] E. Cambria, B. Schuller, Y. Xia, C. Havasi, New avenues in opinion mining and sentiment analysis, IEEE Intell. Syst. 28 (2) (2013) 15–21.
[10] E. Cambria, D. Olsher, D. Rajagopal, SenticNet 3: A Common and Common-Sense Knowledge Base for Cognition-Driven Sentiment Analysis, AAAI, Quebec City, 2014.
[11] E. Cambria, B. White, Jumping NLP curves: a review of natural language processing research, IEEE Comput. Intell. Mag. 9 (2) (2014) 48–57.
[12] D. Chakrabarti, R. Kumar, A. Tomkins, Evolutionary clustering, in: Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2006, pp. 554–560.
[13] Y. Chi, X. Song, D. Zhou, K. Hino, Evolutionary spectral clustering by incorporating temporal smoothness, in: Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2007, pp. 153–162.
[14] A.P.S. De Moura, Biased growth processes and the rich-get-richer principle, Phys. Rev. E – Stat., Nonlinear Soft Matter Phys. 69 (5 Pt 2) (2004) 056116.
[15] J.G. Fiscus, G.R. Doddington, Topic detection and tracking evaluation overview, Top. Detect. Track. (2002) 17–31.
[16] A. Gangemi, V. Presutti, D. Reforgiato Recupero, Frame-based detection of opinion holders and topics: a model and a tool, IEEE Comput. Intell. Mag. 9 (1) (2014) 20–30.
[17] E. Garfield, I.H. Sher, R.J. Torpie, The Use of Citation Data in Writing the History of Science, Institute for Scientific Information, 1964.
[18] D. Gibson, J. Kleinberg, P. Raghavan, Inferring web communities from link topology, in: Proceedings of the 9th ACM Conference on Hypertext and Hypermedia, 1998, pp. 225–234.
[19] T.L. Griffiths, M. Steyvers, Finding scientific topics, Proc. Nat. Acad. Sci. USA 101 (Suppl) (2004) 5228–5235.
[20] G. Jeh, J. Widom, SimRank: a measure of structural-context similarity, in: Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2002, pp. 538–543.
[21] L. Katz, A new status index derived from sociometric analysis, Psychometrika 18 (1) (1953) 39–43.
[22] H. Kwak, Y. Choi, Y. Eom, H. Jeong, S. Moon, Mining communities in networks: a solution for consistency and its evaluation, in: Proceedings of the 9th ACM SIGCOMM Conference on Internet Measurement Conference, 2009, pp. 301–314.
[23] A. Lancichinetti, S. Fortunato, F. Radicchi, Benchmark graphs for testing community detection algorithms, Phys. Rev. E 78 (4) (2008) 046110.1–046110.5.
[24] J. Leskovec, L. Backstrom, R. Kumar, A. Tomkins, Microscopic evolution of social networks, in: Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2008, pp. 462–470.
[25] J. Leskovec, C. Faloutsos, Scalable modeling of real graphs using Kronecker multiplication, in: Proceedings of the 24th International Conference on Machine Learning, 2007, pp. 497–504.
[26] M.S. Lewis-Beck, C. Tien, Voters as forecasters: a micromodel of election prediction, Int. J. Forecast. 15 (2) (1999) 175–184.
[27] D. Liben-Nowell, J. Kleinberg, The link-prediction problem for social networks, J. Am. Soc. Inform. Sci. Technol. 58 (7) (2007) 1019–1031.
[28] R.N. Lichtenwalter, N.V. Chawla, LPmade: link prediction made easy, J. Mach. Learn. Res. 12 (1) (2011) 2489–2492.
[29] Y. Ma, Y. Zeng, X. Ren, User interests modeling based on multi-source personal information fusion and semantic reasoning, Act. Media Technol. 6890 (2011) 195–205.
[30] M.E.J. Newman, Clustering and preferential attachment in growing networks, Phys. Rev. E 64 (2) (2001) 25–102.
[31] M.E.J. Newman, Modularity and community structure in networks, Proc. Nat. Acad. Sci. 103 (23) (2006) 8577–8582.
[32] E. Otte, R. Rousseau, Social network analysis: a powerful strategy, also for the information sciences, J. Inform. Sci. 28 (6) (2002) 441–453.
[33] D. Rajagopal, D. Olsher, E. Cambria, K. Kwok, Commonsense-based topic modeling, in: Proceedings of the Second International Workshop on Issues of Sentiment Discovery and Opinion Mining, New York, NY, USA, 2013, pp. 6:1–6:8.
[34] D. Ramage, D. Hall, R. Nallapati, C.D. Manning, Labeled LDA: a supervised topic model for credit attribution in multi-labeled corpora, in: Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1, Stroudsburg, PA, USA, 2009, pp. 248–256.
[35] M. Rosvall, C.T. Bergstrom, Maps of random walks on complex networks reveal community structure, Proc. Nat. Acad. Sci. USA 105 (4) (2008) 1118–1123.
[36] G. Salton, M.J. McGill, Introduction to Modern Information Retrieval, McGraw-Hill, Inc., 1986.
[37] C. Seshadhri, A. Pinar, T.G. Kolda, An in-depth study of stochastic Kronecker graphs, in: 2011 IEEE 11th International Conference on Data Mining, December 2011, pp. 587–596.
[38] L. Shu, B. Long, W. Meng, A latent topic model for complete entity resolution, in: IEEE 25th International Conference on Data Engineering, 2009. ICDE '09, March 2009, pp. 880–891.
[39] T.J. Stohlgren, D.T. Barnett, J.T. Kartesz, The rich get richer: patterns of plant invasions in the United States, Front. Ecol. Environ. 1 (1) (2003) 11–14.
[40] J. Sun, S. Papadimitriou, P. Yu, Community evolution and change point detection in time-evolving graphs, Link Min.: Models, Algorithms Appl. (2010) 73–104.
[41] C. Tantipathananandh, T. Berger-Wolf, D. Kempe, A framework for community identification in dynamic social networks, in: Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining – KDD '07 2007, New York, New York, USA, pp. 717–726.
[42] H. Tong, Y. Liu, H. Peng, J. Tang, Internet users' psychosocial attention prediction: web hot topic prediction based on adaptive AR model, in: International Conference on Computer Science and Information Technology, August 2008, pp. 458–462.
[43] X. Wang, E. Grimson, Spatial Latent Dirichlet Allocation, NIPS, 2007.
[44] R. Xia, C. Zong, X. Hu, E. Cambria, Feature ensemble plus sample selection: domain adaptation for sentiment classification, IEEE Intell. Syst. 28 (3) (2013) 10–18.
[45] Y. Yang, J. Zhang, J. Carbonell, C. Jin, Topic-conditioned novelty detection, in: Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2002, pp. 688–693.
[46] J. Zhang, Z. Ghahramani, Y. Yang, A probabilistic model for online document clustering with application to novelty detection, in: Proceedings of the 18th Annual Conference on Neural Information Processing Systems, 2005, pp. 1617–1624.
[47] E. Zheleva, L. Getoor, J. Golbeck, U. Kuter, Using friendship ties and family circles for link prediction, Adv. Soc. Netw. Min. Anal. 2010 (2010) 97–113.