Analyzing Future Communities in Growing Citation Networks

Sukhwan Jung Aviv Segev
Department of Knowledge Service Engineering
KAIST - Korea Advanced Institute of Science and Technology
291 Daehak-ro, Yuseong-gu, Daejeon, Korea
+82-042-350-1694 +82-042-350-1614
raphael@kaist.ac.kr aviv@kaist.edu

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 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. 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.

Categories and Subject Descriptors
H.2.8 [Database Management]: Database Applications— Data mining; H.3.3 [Information Storage and Retrieval]: Information Search and Retrieval—Information filtering.

General Terms
Algorithms, Measurement, Design, Theory.

Keywords
Community; Prediction; Topic detection; Link prediction; Citation Network; Community Detection.

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) [7], 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 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 only 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. 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 contribution to the performance of the prediction. To analyze the models, two citation networks from the Stanford Large from High Energy Physics Theory (18479 papers, 136428 citations) and High Energy Physics (30566 papers, 347414 citations).

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 and results on citation networks. Finally, Section 5 discusses the results and model limitations.

2. RELATED WORK

2.1 Topic Detection and Prediction
Topic Detection and Prediction has been studied in many research fields. Topic Detection and Tracking (TDT) [6] is a multi-site research project aiming to predict novel topics. Their goal is to find a new topic in news systems by effectively identifying the first article or report mentioning the new topic. There have been many studies using 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 [25]. Topic-conditioned First Story Detection (FSD) method in conjunction

CIKM'13, Oct. 27–Nov. 1, 2013, San Francisco, CA, USA.

DOI string from ACM form confirmation

with a supervised learning algorithm [26] and Document Clustering [27] are used to identify the earliest report to 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 [15].

2.2 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 [16]. The shortest path between two nodes in a graph is a simple measure of link prediction. Some methods, such as Common Neighbors [18], Jaccard's coefficient [22], Preferential Attachment [18], 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 [10], Simrank [9], Rooted PageRank [16], and so on. 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.

2.3 Community Detection

Community detection searches structural information of a given graph to partition it into sub-graphs called communities or modules [12]. Agglomerative methods and divisive methods are commonly used in community detection. Newman's community detection algorithm [19] is a widely used agglomerative method that uses modularity as the quality function. The recently developed Louvain method [3] 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. Evolutionary clustering [4,5] takes 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 [28]. The current work proposes a set of models based on the temporal graphs and the community prediction techniques.

3. COMMUNITY PREDICTION METHOD IN CITATION NETWORKS

Citation networks are directed social networks [20] between research papers, with nodes as papers and edges as citations between them. It is a form of network where link prediction can be applied without extra considerations about deleted edges or nodes; nodes and edges never disappear from citation networks. Community detection algorithms have been proven to detect communities in such existing networks [21] 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 $t\text{-}1 < t < t+n$ are chosen; G_{t-1} and G_t are the test set and G_{t+n} is the ground truth n time steps later. The node prediction algorithm explained in 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 predicted graph that 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 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 run) to predict a graph $t+n$ time steps in the future. Figure 1 illustrates how the graph grows with node and edge prediction.

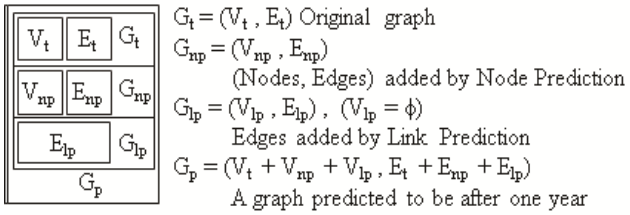

Figure 1. Illustration of predicted graph

Community detection algorithms explained in 3.1.3 are then run, 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 a 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 [14] can capture more graphical features of a given graph, but the number of nodes is increased exponentially [23]; a citation network does not grow in such a way. Hence the method is not used in this paper.

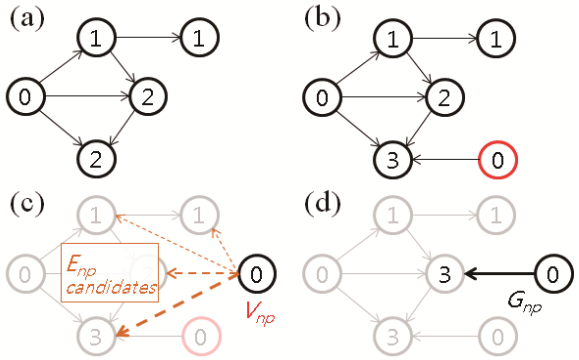

Figure 2. Example of the node prediction process (the incoming node has to change to 3)

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

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 cited before – nodes with no inbound edges – have $s_p = 0$ and hence are never selected. In Figure 2.c, four candidates for E_{np} shown with brown 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 (Figure 2.d). In one of the alternative methods explained in 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, six link prediction algorithms are run on the resulting network. The Preferential Attachment [18] method follows preferential attachment theory where the rich get richer. Common Neighbor [18] simply measures the number of common neighbors between two nodes to calculate the similarity score. Jaccard's Coefficient [22] uses a more complex method where the relative fractions of the number of common neighbors are considered. A similar approach is presented by Adamic/Adar [1], but they weight rarer features more heavily. Simrank [9] and Rooted PageRank [16], on the other hand, do not use neighborhood information and use the random walk with restart (RWR) approach. Katz [10] also ignores neighborhoods but utilizes path distance instead of RWR. Figure 3 shows the detailed algorithms.

3.1.3 Community Detection

Two implementations of the Louvain method are used in this paper: Louvain-smallest algorithm and Louvain-best algorithm. 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 run twice, with 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}.

Adamic/Adar	$\sum_{z \in \Gamma(x) \cap \Gamma(y)} \frac{1}{\log \lvert\Gamma(z)\rvert}$
Common Neighbors	$\lvert\Gamma(x) \cap \Gamma(y)\rvert$
Jaccard's Coefficient	$\frac{\lvert\Gamma(x) \cap \Gamma(y)\rvert}{\lvert\Gamma(x) \cup \Gamma(y)\rvert}$
Katz$_\beta$	$\sum_{\ell=1}^{\infty} \beta^\ell \cdot \lvert \mathrm{paths}_{x,y}^{\langle \ell \rangle} \rvert$ where $\mathrm{paths}_{x,y}^{\langle \ell \rangle}$:= {paths of length exactly ℓ from x to y} weighted: $\mathrm{paths}_{x,y}^{\langle 1 \rangle}$:= number of collaborations between x, y. unweighted: $\mathrm{paths}_{x,y}^{\langle 1 \rangle}$:= 1 iff x and y collaborate.
Preferential Attachment	$\lvert\Gamma(x)\rvert \cdot \lvert\Gamma(y)\rvert$
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$_\gamma$	$\begin{cases} 1 & \text{if } x = y \\ \gamma \cdot \frac{\sum_{a \in \Gamma(x)} \sum_{b \in \Gamma(y)} \mathrm{score}(a,b)}{\lvert\Gamma(x)\rvert \cdot \lvert\Gamma(y)\rvert} & \text{otherwise} \end{cases}$

Figure 3. Link prediction algorithm [16]

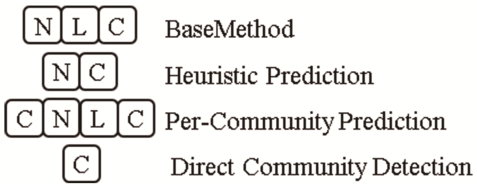

Figure 4. Outline of methods

3.2 Alternative Methods

Three alternative methods of the given model, referred as the *nlc-method*, are also proposed in this paper. Figure 4 outlines how the modules explained above are used. '**N**'ode prediction (3.1.1), '**L**'ink prediction (3.1.2), and '**C**'ommunity Detection (3.1.3) are visualized as a block, and each method consists of a series of modules run in left-to-right sequential order. For example, the base method in Figure 4 requires the node prediction module to run first, the link prediction module second, and then the community detection module.

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 run, 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[28]. 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 run first on G_t (replaced by G_p after the first run) 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 $|c_i|$ ($c_i \in C_t$). $|V_{np}| * (|c_i| / |V_t|)$ new nodes are assigned to each community c_i. The node prediction algorithm is run per communities to connect the assigned number of new nodes to the community. After the algorithm is run 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 run with G_t to produce C_t, a set of communities in time step t. It is used to predict communities in time step $t+n$.

4. EXPERIMENTS

4.1 Data

Two citation networks are taken from the Stanford Large Network Dataset Collection[1], 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 1 shows detailed information. The selected research fields have a tendency to have heavy citations, and networks are dense with number of edges exceeding the number of nodes by factor of 7 to 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.

Table 1. Details of dataset used

Name	No. of Papers	No. of Citations
hepTh	18479	136428
hepPh	30566	347414

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 [17] is used to run the link prediction algorithms for the experiment. LPmade is a link prediction software package with a total of 21 link predictors implemented, including all the algorithms used in this paper.

For community detection, the Python community detection library *Community detection for NetworkX*[2] is used in this experiment. This library uses the Louvain method to detect and cluster communities in *NetworkX*[3] 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.

[1] http://snap.stanford.edu/data/
[2] http://perso.crans.org/aynaud/communities/
[3] http://networkx.lanl.gov/

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 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 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 $F = 2 * (p*r) / (p/r)$ where $p = |c_{matched}|/|c_{t+n}|$ and $r = |c_{matched}|/|c_{ground\ truth}|$.

4.3 Results

4.3.1 Node Prediction

Figure 5 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 also show promising results. Figure 6 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 Figure 5 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 Figure 6. 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

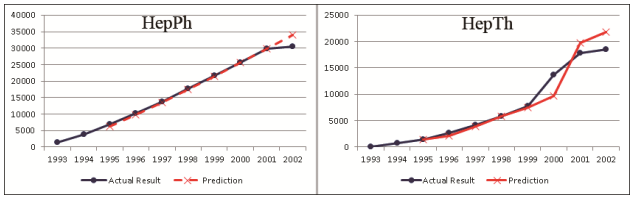

Figure 5. Node count $|V_t|$ and $|V_{np}|$ per year in hepPh/hepTh dataset

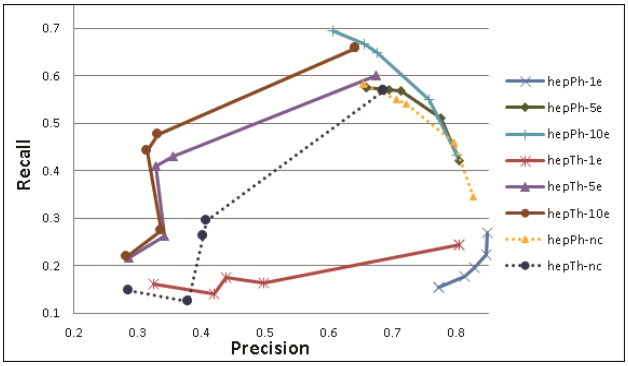

Figure 6. Precision versus recall of edges (E_{np}) in node prediction module.

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 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 2. Rooted PageRank is run with random walk restart parameter α = {0.01, 0.05, 0.25, 0.50} and Katz is run with damping parameter β = {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.

Table 2. Precision and recall of E_{lp}

	hepTh		***hepPh***	
Predictor	**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

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 run on graphs generated by node and link prediction modules, and the evaluation method presented in 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 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 4.2, Figure 7 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 with *threshold* = 0.5, 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 noticing 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.

Figure 8 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)*. Figure 8 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 Figure 8 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 Figure 7 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 grows and as a graph is more fine-grained into more communities each with smaller sizes.

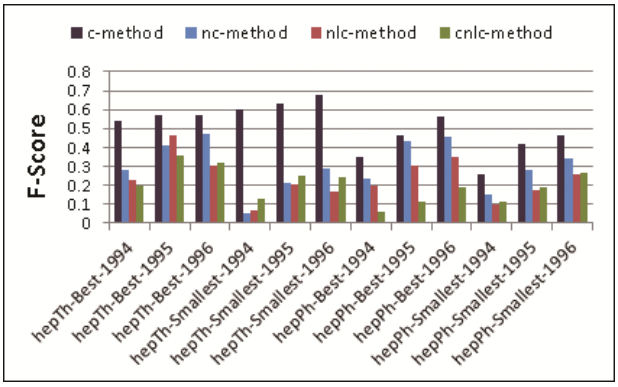

Figure 7. F-score of 4 methods with different community detectors in hepTh/hepPh dataset for 1 year prediction.

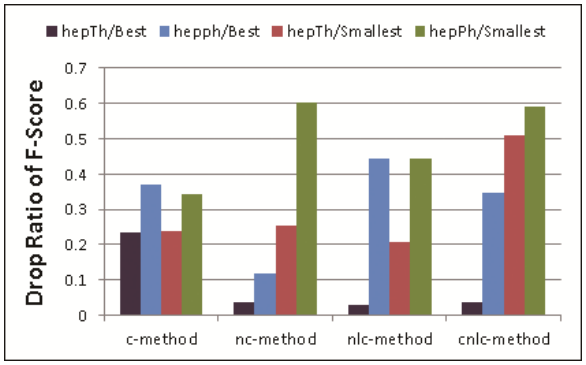

Figure 8. F-score ratio when prediction at t+5 is compared against prediction at t+1, with threshold = 0.5

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 varies for emerging / disbanding communities in the citation networks is also tested. The *nc-method* and the *nlc-method* are run 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 hep-th dataset are used as time steps. A community matching algorithm proposed in 4.2 is replaced by more advanced version.

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 3, $sim_{old}(c_1, c_2)$ 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 $w_{old} * sim_{old}(c_1,c_2) + w_{new} * sim_{new}(c_1,c_2) = sim(c_1,c_2)$, where $w_{new} + w_{old} = 1$ so that $0 \leq sim(c_1,c_2) \leq 1$.

Table 3. Community matching scheme

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

$sim_{old}(c_1, c2)$ is calculated in the same way explained in 4.2, replacing c_1 and c_2 with $c_{1,old}$ and $c_{2,old}$. $sim_{new}(c_1, c2)$ uses a different approach, since it is dealing with 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.

Figure 9 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. Figure 10 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 run. 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, distance, and so on.

Figure 10 also shows the F-score of communities found to be formed and disbanded. The disbanded result is very high in either method, 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 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.

4.4 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 show 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

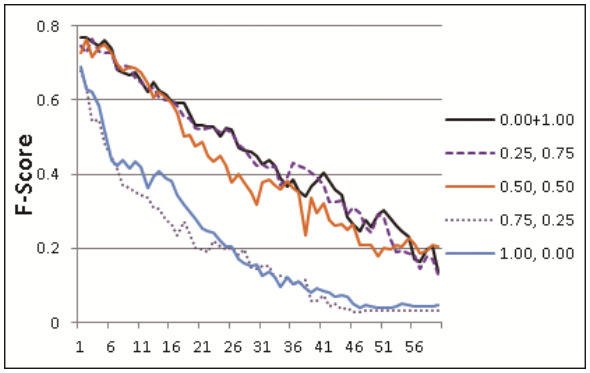

Figure 9. F-score of the *nlc-method* with different w_{old} & w_{new} values.

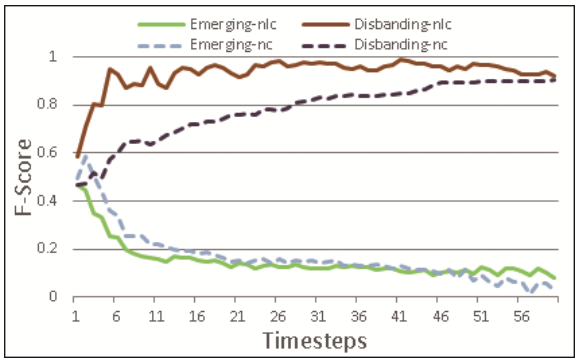

Figure 10. F-score of newly emerged and disbanded communities in the nc/nlc-method with w1=0.0 and w2=1.0

link prediction and community detection algorithms could increase the overall performance of community prediction. One possible solution for community detection is evolutionary clustering [5,13], which takes temporal consistency of the communities into account.

Analysis of Figure 8 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 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. Although the labeling of the community was not analyzed in the current work, there are some limitations that should be considered. 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. 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 thus 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 minutes.

5. CONCLUSION

The paper presents a model for analyzing future communities in a citation network. The model includes a heuristic method to predict nodes, link prediction methods, and community detection methods, 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.

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, consistency, and so on. Another possible direction is to investigate the break-even point for community size versus different combinations of analysis methods.

6. ACKOWLEDGEMENTS

This work was supported by the Korean Government IT R&D program of MKE/KEIT. [Project No. 10035166, Development of Intelligent Tutoring System for Nursing Creative HR]

7. REFERENCES

[1] Adamic, L. and Adar, E. 2003. Friends and neighbors on the web. *Social Networks*. 25, 3 (Jul. 2003), 211–230.

[2] Adams, J. 2005. Early citation counts correlate with accumulated impact. *Scientometrics*. 63, 3 (Jun. 2005), 567–581.

[3] Blondel, V.D. et al. 2008. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*. 2008, 10 (Oct. 2008), P10008.1– P10008.12.

[4] Chakrabarti, D. et al. 2006. Evolutionary clustering. *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2006), 554–560.

[5] Chi, Y. et al. 2007. Evolutionary spectral clustering by incorporating temporal smoothness. *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2007), 153–162.

[6] Fiscus, J.G. and Doddington, G.R. 2002. Topic detection and tracking evaluation overview. *Topic detection and Tracking*. 17–31.

[7] Garfield, E. et al. 1964. *The use of citation data in writing the history of science*. Institute for Scientific Information.

[8] Gibson, D. et al. 1998. Inferring web communities from link topology. *Proceedings of the 9th ACM Conference on Hypertext and Hypermedia* (1998), 225–234.

[9] Jeh, G. and Widom, J. 2002. SimRank: a measure of structural-context similarity. *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2002), 538–543.

[10] Katz, L. 1953. A new status index derived from sociometric analysis. *Psychometrika*. 18, 1 (1953), 39–43.

[11] Kwak, H. et al. 2009. Mining communities in networks: a solution for consistency and its evaluation. *Proceedings of the 9th ACM SIGCOMM conference on Internet measurement conference* (2009), 301–314.

[12] Lancichinetti, A. et al. 2008. Benchmark graphs for testing community detection algorithms. *Physical Review E*. 78, 4 (Oct. 2008), 046110.1-046110.5.

[13] Leskovec, J. et al. 2008. Microscopic evolution of social networks. *Proceeding of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2008), 462–470.

[14] Leskovec, J. and Faloutsos, C. 2007. Scalable modeling of real graphs using Kronecker multiplication. *Proceedings of the 24th International Conference on Machine Learning* (2007), 497–504.

[15] Lewis-Beck, M.S. and Tien, C. 1999. Voters as forecasters: a micromodel of election prediction. *International Journal of Forecasting*. 15, 2 (Apr. 1999), 175–184.

[16] Liben-Nowell, D. and Kleinberg, J. 2007. The link-prediction problem for social networks. *Journal of the American Society for Information Science and Technology*. 58, 7 (2007), 1019–1031.

[17] Lichtenwalter, R.N. and Chawla, N.V. 2011. LPmade: Link prediction made easy. *Journal of Machine Learning Research*. 12, 1 (2011), 2489–2492.

[18] Newman, M.E.J. 2001. Clustering and preferential attachment in growing networks. *Physical Review E*. 64, 2 (2001), 25–102.

[19] Newman, M.E.J. 2006. Modularity and community structure in networks. *Proceedings of the National Academy of Sciences*. 103, 23 (Jun. 2006), 8577–8582.

[20] Otte, E. and Rousseau, R. 2002. Social network analysis: a powerful strategy, also for the information sciences. *Journal of Information Science*. 28, 6 (Dec. 2002), 441–453.

[21] Rosvall, M. and Bergstrom, C.T. 2008. Maps of random walks on complex networks reveal community structure. *Proceedings of the National Academy of Sciences of the United States of America*. 105, 4 (Jan. 2008), 1118–1123.

[22] Salton, G. and McGill, M.J. 1986. *Introduction to modern information retrieval*. McGraw-Hill, Inc.

[23] Seshadhri, C. et al. 2011. An In-depth Study of Stochastic Kronecker Graphs. *2011 IEEE 11th International Conference on Data Mining* (Dec. 2011), 587–596.

[24] Sun, J. et al. 2010. Community evolution and change point detection in time-evolving graphs. *Link Mining: Models, Algorithms and Applications*. 73–104.

[25] Tong, H. et al. 2008. Internet users' psychosocial attention prediction: web hot topic prediction based on adaptive AR Model. *International Conference on Computer Science and Information Technology* (Aug. 2008), 458–462.

[26] Yang, Y. et al. 2002. Topic-conditioned novelty detection. *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2002), 688–693.

[27] Zhang, J. et al. 2005. 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), 1617–1624.

[28] Zheleva, E. et al. 2010. Using friendship ties and family circles for link prediction. *Advances in Social Network Mining and Analysis* (2010), 97–113.