ACM Community Topic Modeling: Community Detection and Topic Drift with Word Embedding

Chenyue Meng CHENYUE@STANFORD.EDU
Chenjie Yang YANGCJ@STANFORD.EDU
Yixin Wang WYIXIN@STANFORD.EDU
Stanford University, Stanford, CA 94035

Abstract

Network structures together with attributes of nodes have provided us with precious information to reveal organizational principles in networks. In this report, we exploit the clustering property in networks from structural level to linguistic attribute level. We applied BigCLAM algorithm to detect overlapping communities in ACM academic network. Next we aggregate paper titles within each community and extract tf-idf features to filter out keywords as description of community topic, and verify the effectiveness of this approach. To further improve the precision of community keywords, we trained a word embedding representation on paper titles and develop an algorithm to distill better keywords. As an extension, we utilized our framework to study topic drift in ACM citation network and found convincing results.

1. Introduction

The quantitative study of citations has a long history in bibliometrics (Egghe & Rousseau, 1990). The recent availability of large citation datasets has made it possible to study properties of citation networks that were inaccessible one decade ago. The basic graph statistics facts (static or dynamic) and the underlying community structures of citation networks can help us understand academic subfields better: their importance, their interconnections, and their growth or ebb over time.

Studies based on citation networks group papers based on their citation relationship, rather than the content of the publications themselves. However, it's also natural to consider using clustering techniques on document content to detect and analyze academic communities. An interesting question is whether the two are relevant. The intuition is that the communities detected in citation networks will expose similarities between papers, which may arise from the shared subfields these papers focus on. Such similarities can also be revealed by the textual features of papers. In this project, we aim to compare the community detection results attained from the citation networks with the clustering results achieved from the textual contents of papers. The project consists of two parts: (1) after detecting communities in citation networks, it's hard to measure the detection quality. We need to verify that the nodes within a community are indeed related to the same topic. We can extract keywords from textual contents of papers to label the fields/topics these papers lie in, and then use these labeled topics to verify whether papers within the same community share similar topics. (2) we can cluster papers directly using textual contents and then compare them with the communities we get from citation networks. We believe exploring the similarities between the two can give us more insights on both parts.

The rest of this proposal is structured as follows. In section 2, we discuss about related work and their inspiration on our project. In section 3, we describe the dataset and the project overview. In section 4, we introduce the methodology of community detection and document representation algorithms which are used in our project. Section 5 describes experiment results and analysis. We plan our next steps in section 6.

2. Related Work

2.1. Community Detection

A community is typically thought of as a group of nodes with more connections amongst its members than the remainder of the network. For citation networks, nodes are papers and there will be a edge between 2 nodes if one of them referenced the other. A paper/author can belong to different sub-fields simultaneously, especially for those

Proceedings of the 33^{rd} International Conference on Machine Learning, New York, NY, USA, 2016. JMLR: W&CP volume 48.

coming from interdisciplinary research. Thus, the communities we want to detect are overlapped.

While there have been a lot of work in community detection area, mainstream methods like Graph partitioning (Schaeffer, 2007), modularity optimization (Newman, 2006) as well as betweenness centrality (Girvan & Newman, 2002) based techniques all conceptualize networks as consisting of dense clusters that are linked by a small number of weak ties. Therefore, these algorithms tend to cut edges in a way that the network will be separated into a set of non-overlapped networks. On the other hand, early research on overlapping community detection methods (Palla et al., 2005) all assume that community overlaps are less densely connected than the non-overlapping parts of communities, which is not the case in practical (Yang & Leskovec, 2012b).

Paper (Yang & Leskovec, 2013) proposed BigCLAM, a non-negative matrix factorization approach that can detect densely overlapping or non-overlapping communities in massive networks under a unified framework. Empirical results show that this method can bring high accuracy as well as scalability. We will start from BigCLAM for community detection in citation networks.

2.2. Text Features

Text features are frequently used to compare the similarity of two documents. A widely used scheme is to extract tf-idf features, then using cosine similarity to measure the distance between two document (Hofmann, 1999). However, using tf-idf features is potentially challenging for visualization because tf-idf features are high dimensional. In high-dimensional space, the L_k norm distance function is susceptible to the dimensionality curse for many classes of data distributions (Beyer et al., 1999), i.e. everything have similar distance to each other.

Word-Embedding (Mikolov et al., 2013) is a popular and powerful technique in natural language processing (NLP). It builds a mapping from raw words or phrases to low-dimension vectors in order to overcome data sparsity, which is caused by unique and discrete representations of words in the vocabulary. Word-Embedding has this nice property that after vectorization, semantically similar words are more likely to be mapped to nearby points, i.e. embedded nearby each other in a continuous vector space. Hence, word-embedding can provide us a handy tool to observe word clustering phenomena and study similarity among words. The methodology of word-embedding is discussed in detail in section 4.

2.3. Interaction of Node Features and Network Structure

When detecting communities, there are two possible sources of information we can use: the network structure, and the features and attributes of nodes(Yang et al., 2013). Since these two approaches both yield reasonable performance in detecting dense communities/clusters, it is reasonable to put forward the hypothesis that there might be some intrinsic nature of the network that couples network structure and node attributes.

We expect that the aggregation of node attributes within a community serves as good depiction of the characteristic of this community. By visualizing (1) the feature space of aggregated node attributes and (2) the relationship between communities, we can better understand the correlation between node attributes and network structure. In our problem, node attributes are defined as the abstract section of a paper, which can well summarize the topic of the paper. Aggregated node attributes of a community is obtained by doing some processing and aggregation on the feature of nodes within a community.

(Yang et al., 2013) has shown that combining network structure and node attributes can boost community detection performance. Our work serves as a justification, using visualization to better understand the reason behind. Meanwhile, our work introduces using word-embedding, which might be a higher-quality distributed vector representation for word features.

3. Problem Formulation and Dataset

3.1. Problem Overview

We model each paper as a node. An undirected edge appears when there is citation relationship between two nodes. We apply BigCLAM (Yang & Leskovec, 2012b) to detect communities in this network, after which we visualize community detection result via graph coloring. Next we try to understand community topics. For each community, we model the topic of the community by extracting tf-idf features for all titles contained in this community. Then we keep top 100 words with largest tf-idf feature values, which are conceptually the keywords that best describes the community topic. After this, we train word-embedding representation with all paper titles. Applying the word-embedding representation and a novel iterative algorithm for community keyword selection that we proposed, we find a much more concise and precise keyword profile for each community. We visualize each community's keyword embedding with t-SNE and find clear clustering behavior of community keywords, i.e. strong coupling between node attribute cluster and network topology cluster. Finally, as a by-product, we model topic drift over every 5 years and do

visualization with t-SNE, and find topic drift behavior not only on semantic level, but also on visualized graph.

3.2. The ACM Dataset

Our dataset is a citation network extracted from ACM digital library [2]. The network contains 2,381,688 papers and 10,476,564 citation relationships. For each paper, there are records of paper title, authors, year, references and abstract.

4. Methodology

4.1. Community Detection using BigCLAM

BigCLAM is a highly scalable community detection algorithm (Yang & Leskovec, 2013). A community membership graph is represented as $B(V, C, M, \{p_c\})$, where V is node set, C is community set, M is membership assignment, and $\{p_c\}$ is the probability that nodes in community c are connected. A node u may be a member of multiple communities. BigCLAM makes the assumption that memberships have strengths. We denote membership strength of node u to community A as F_{uA} ($F_{uA} \geq 0$). If $F_{uA} = 0$, then u has no membership in community A. BigCLAM also assumes that each community links nodes independently, i.e. for any two nodes u, v, the probability that u, v is linked by an edge because they both belong to community A is

$$P_A(u, v) = 1 - \exp(-F_{uA} \cdot F(vA)) \tag{1}$$

If we define F as a matrix, where each row corresponds to a node and each column corresponds to a community, then F_{uA} means the membership strength of node v to community A. Thus the probability that u, v is linked by an edge (because of any one of the communities) is

$$\begin{aligned} P(u, v) &= 1 - \Pi_C \left(1 - P_C(u, v)\right) \\ &= 1 - \exp(-\sum_C F_{uC} \cdot F_{vC}) \\ &= 1 - \exp(-F_u \cdot F_v^T) \end{aligned} \tag{2}$$

Given a network, we would like to estimate F. We would like to find F that maximizes the probability of network data, i.e. the likelihood (or log likelihood $l(F)$) of parameters F.

$$\arg\max_F \sum_{(u,v)\in E} \log P(u, v) + \sum_{(u,v)\notin E} \log(1 - P(u, v)) \tag{3}$$

where $P(u, v)$ is defined in (2). Computing the gradient $\nabla_F l(F)$, we can use stochastic gradient descent to obtain F. Next we iteratively update the model B by a series of random transitions, as described in (Yang & Leskovec, 2012a).

To limit the number of communities detected, first we fit a candidate community $B_0(V, C_0, M_0)$ with a very large number of communities, then an L_1 regularization term is added to penalize the number of communities.

$$\arg\max_{\{p_c\}} P(G|B_0, \{p_c\}) - \lambda \sum_c |p_c| \tag{4}$$

The regularization term will force some $|p_c|$ to zero. Thus these communities can be ignored and removed from B_0. The log likelihood $B(\lambda)$ exhibits a step-like behavior (Yang & Leskovec, 2012a). We take the value of λ at which $B(\lambda)$ experiences step-like transition, denoted as $\lambda*$. The number of communities at $\lambda*$ is taken as the estimation for the number of communities.

Also, using some runtime optimizations in implementation, BigCLAM is very fast and can easily deal with large scale datasets.

4.2. Word-Embedding

The reason that we utilize word-embedding is to find a suitable representation for visualizing relationships among words via possible clustering. Intuitively, words are very likely to gather around if they belong to the same topic or are commonly used in a certain research area. Analyzing the word-embedding from paper abstracts may give us access to the understanding of what topic a new paper might belong to given its title and abstract, and what characteristic other papers in the same community may have in common.

However, one major drawback of producing word-embedding might be its huge computational cost when training on large text data. As a computationally-efficient algorithm, therefore, word2vec has been commonly used for learning word-embedding. The two model architectures word2vec utilizes to produce distributed representations of words are continuous bag-of-words (CBOW) and continuous skip-gram. Specifically CBOW model predicts the current word from a window of surrounding context words; while skip-gram model does the opposite, i.e. using the current word to predict the surrounding window of context words. According to (Mikolov et al., 2013), CBOW is faster while skip-gram is slower but does a better job for infrequent words. Therefore, We will focus on the skip-gram model in our project.

4.2.1. The Skip-gram Model

Assume we are given a sequence of words $w_1, w_2, \cdot, w_T$, the objective we're trying to maximize is the log likelihood

$$\sum_{t=1}^{T} \sum_{-c<j<c, j\neq 0} p(w_{t+j}|w_t) \tag{5}$$

where the conditional probability is defined using softmax function

$$p(w_o|w_t) = \frac{\exp\left(z_o^T z_t\right)}{\sum_{i=1}^{W} \exp\left(z_i^T z_t\right)} \tag{6}$$

Here z_j represents the word vector of word w_j and W is the total number of words in the vocabulary. Normally, W is large ($10^5 - 10^7$) which makes this formulation impractical.

1. *Hierarchical Softmax* To avoid heavy computation, an approximation of the above formulation is using hierarchical sotfmax. The basic idea is to use a binary tree representation of the output layer with the W words as its leaves and, for each node, explicitly represents the relative probabilities of its child nodes. The major simplification or approximation of this approach is to use same word vector representation z_j for every inner node of the the same depth j from root. The details about the effectiveness and correctness of hierarchical sotfmax can be found in (Mikolov et al., 2013), which is beyond the scope of this proposal.

2. *Negative Sampling and Noise Contrastive Estimation* As an alternative approach to the hierarchical sotfmax introduced above, Negative Sampling and Noise Contrastive Estimation (NCE) choose to optimize the probability of softmax from a different angle, which is minimizing the log likelihood of sampled negative instances. The task becomes distinguishing the target word w_o from draws draws from the noise distribution $Pn(w)$ using logistic regression.

   Theoretically Negative Sampling is more efficient than NCE because Negative Sampling only needs samples while NCE needs both sample and noise distribution. However, we might choose to use NCE instead since tensorflow has already implemented it as a loss function **tf.nn.nce_loss()**.

3. *Subsampling* One of the most important parameterizations of word2vec algorithm is subsampling, which aims for reducing the influence of frequent but meaningless words like *a, the, it, that* etc. In fact, with high probability, this kind of words may compose a huge fraction in real data. As a data preprocessing, we use subsampling to clean the raw text data.

   For each word w_i in the training set, the probability of discarding it is formulated as

$$p(w_i) = 1 - \sqrt{\frac{t}{f(w_i)}} \tag{7}$$

   Here $f(w_i)$ is the frequency of word w_i and t is a hyperparameter. Intuitively, when $f(w_i)$ is high, then with high probability, it will not be contained in the vocabulary.

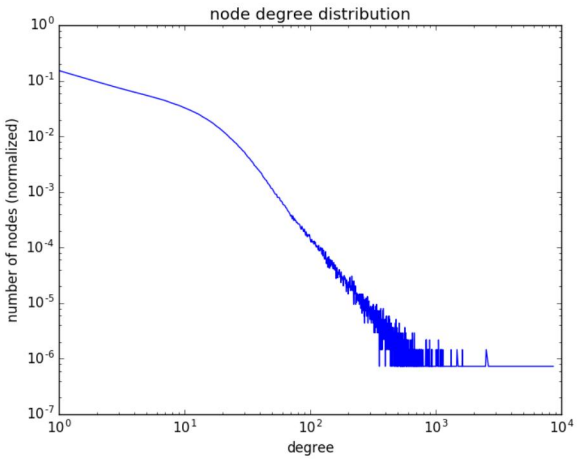

Figure 1. Node distribution

5. Experiment

5.1. Graph Structure

First we pre-processed the citation data. For each paper, we extracted its id, abstract and references. We model the citation network as an undirected graph: each node represents a paper, and if one paper references another, there will be an edge between them. The citation graph we got contains 1,369,055 nodes and 8,641,929 edges. Figure 1 shows the node distribution(log-log scale) of this graph, it follows a power law distribution. Average node degree is 6.31, so in average every paper has around 6 references.

5.2. Community Detection

We use BigCLAM (Yang & Leskovec, 2012b) for community detection. The number of communities is detected automatically via searching. The optimal number of communities is 100, and the size of each community is illustrated in Figure 2. There are around 30 large communities with more than 1000 nodes. These larger communities are those we will emphasize on. We visualize community structure as shown in Figure 4 using Gephi (Bastian et al., 2009). Each community is distinguished by a unique color. There are 30 large communities, however we only visualize the first 10 of them for visual-friendliness. Nodes that belong to multiple communities are colored using a color-average of all communities that it belongs to, in order to make the graph visually smooth and more accurate. The structure of this graph is further adjusted using `ForceAtlas2` layout.

It is shown via visualization that there is some overlap between different academic communities, but the overlap is generally very small. The number of connections within community is much denser than those between communities.

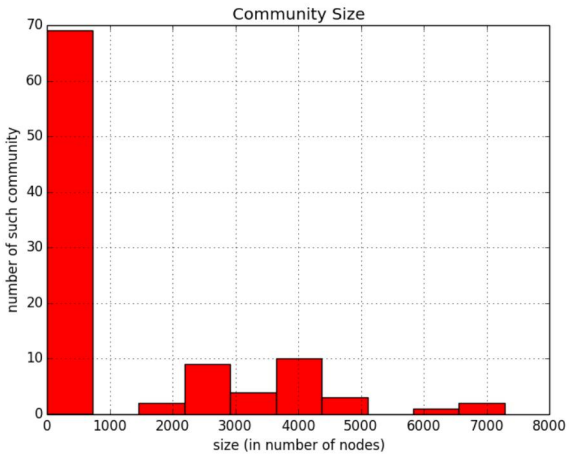

Figure 2. Community size distribution

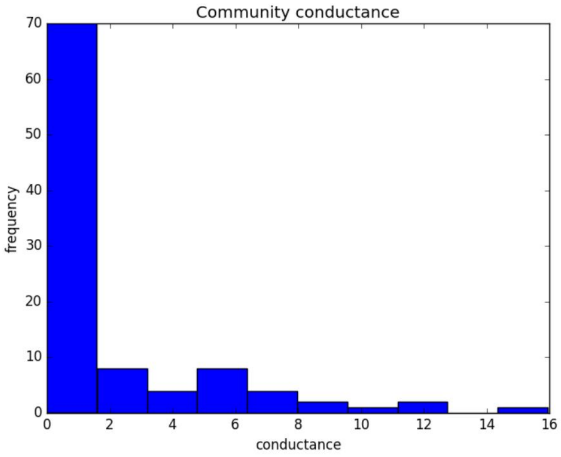

Figure 3. Community conductance

Figure 3 shows the conductance distribution of communities detected. Low conductance can illustrate good quality of clustering. As we can see from the figure, around 70% of communities has 0 conductance, which means the number of edges pointing outside of these communities is 0. These communities mostly are small communities. For large communities, most of them have conductance less than 8.

5.3. Community Topic Modeling

5.3.1. TF-IDF FEATURE

In order to find a set of tokens that best describes a community, we extract tf-idf features (Hofmann, 1999) for all titles that belong to a community. Concretely, to calculate inverse document frequency, we treat all titles within a community as a single file. Stop words are removed and stemming is performed. Intuitively, tokens with the highest tf-idf feature values best describe a community. Therefore for each community, we take 100 tokens with highest tf-idf values as keywords.

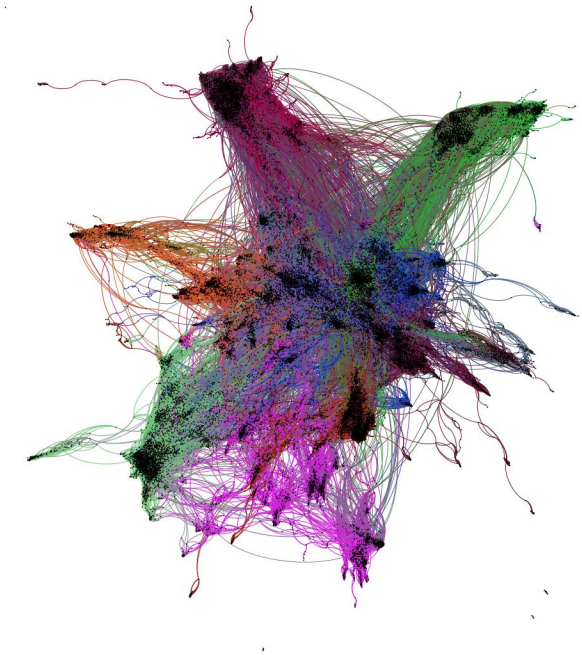

Figure 4. Visualization of community structure

We then counted the most frequent conference/journal names that papers within the community belong to. We compare community keywords with the most common academic conferences/journals that papers within the community belong to. We sample the comparison result for several communities in Table 1. We show the truncated top 100 keywords (because otherwise it is too long to display), and the most frequent conference names.

The intuition is the following: (1) If within a community, words with highest tf-idf weights are semantically similar, and that the academic fields of conferences are similar, then we can validate that BigCLAM gives a good clustering result: the communities it detected are actually meaningful and indicative of different academic fields. (2) If within a community, the keyword profile is semantically relevant to the conference names, then using tf-idf feature is a reasonable approach of constructing keywords profiles for community.

Next we analyze the results in this table. It is surprisingly obvious that most communities have only one or two topics. Meanwhile, it is obvious that the keywords are highly relevant to the academic fields implied by conference names. For example, it is obvious that commu-

Community	keyword profile	most frequent conference	# nodes
ID=0 Guess: **Graphics**, **Theory**	original,**world**,**architecture**,defined, **theory**,**discrete**,**local**,**parallel**,applied, complex,**surface**,rate,test,mobile,proof, al,**consider**,**group**,shown,**bound**, approaches,developed,type,**field**,**channel**, dynamic,memory	Journal of Computational Physics,Theoretical Computer Science, Journal of Computational and Applied Mathematics,IEEE Transactions on Visualization and Computer Graphics,IEEE Transactions on Information Theory,Information Processing Letters,Computer Communications,IEEE Transactions on Computers	7239
ID=6 Guess: **E-Commerce**	range,dynamic,products,book,students, **current**,**researchers**,**online**,change, **security**,**memory**,**consumer**,**strategies**, **industry**,**experience**,efficient,form, including,**consumers**,sharing, community	MIS Quarterly,Proceedings of the SIGCHI Conference on Human Factors in Computing Systems,Journal of Cognitive Neuroscience, Management Science,Journal of Management Information Systems, Marketing Science,International Journal of Electronic Commerce, Expert Systems with Applications: An International Journal,Computers in Human Behavior,	4328
ID=9, Guess: **Linguistics**	types,**chinese**,atis,component,measure, **german**,**optimal**,complex,uses,**patterns**, key,related,**pairs**,provides,**probabilistic**, **graph**,discuss,**constraints**,**retrieval**, **significant**,users,web,**spoken**	IEEE Transactions on Signal Processing,COLING '04 Proceedings of the 20th international conference on Computational Linguistics, HLT '10 Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics,IEEE Transactions on Audio, Speech, and Language Processing	4081

Table 1. Community top 100 keywords (truncated) and most frequent conferences of paper submission in each community. We also included community ID, our guess of its academic field, and community size.

nity ID=0 is "Graphic, Theory" papers, because they are in computational physics, graphics and theory conferences, and have keywords including "world, theory, discrete, local, parallel, surface, bound, field, channel". Same with other academic fields, see details in Table 1. Thus both of our intuitions (1) and (2) have been verified.

5.3.2. Iterative Selection of Community Topic based on Word-Embedding

Tokens extracted via tf-idf features have been proven to be quite powerful for describing the topic of a community. However, there are many spurious words in the 100 keywords selected by tf-idf features that are not descriptive of community topic. To obtain a cleaner and more precise description for each community, we bring in word embedding to distill those keyword candidates into a better keyword profile.

To represent each keyword as a vector in the word embedding, all title words in the dataset are extracted, preprocessed and trained in Google word2vec framework (Mikolov et al., 2013). The pre-processing phase includes stemming, lower-casing and removing stop words. Using default parameter setup, the training takes less than a minute.

We choose the top 30 largest communities and 500 keyword candidates for each community. The keyword candidates are extracted using tf-idf features, as described in Sec. 5.3.1. After training, we obtain, for each word, a 200-dimensional vector as its distributive representation, i.e. the word-embedding representation for each word. Based on this word-embedding representation, we go ahead and refine community keyword profile to be more concise and precise. In order to refine community keyword profile, we propose a novel algorithm, the *Iterative Selection of Community Topic* algorithm, illustrated as follows.

- Initialize **community centroids** $\mu_1, \cdots, \mu_k \in \mathbb{R}^n$ by

$$\mu_i = \frac{1}{m_i} \sum_{j=1}^{m_i} z_j^{(i)}$$

- Repeat until convergence: {
  - For each word $w_j^{(i)}$, compute

$$c_j^{(i)} = \arg\min_c \text{cos\_dist}(\mu_c, z_j^{(i)})$$

    if $c_j^{(i)} \neq i$, remove $w_j^{(i)}$ from community i, decrease m_i by 1
  - For each community i, compute

$$\mu_i = \frac{1}{m_i} \sum_{j=1}^{m_i} z_j^{(i)}$$

}

where k is the number of communities, m_i is the number of keyword candidates in the i-th community extracted using tf-idf feature, n is the dimension of word2vec vectors, and $z_j^{(i)}$ represents the word vector of the j-th word $w_j^{(i)}$ in the i-th community.

In the first step, each word is assigned to the closest community centroid. $c_j^{(i)}$ represents the community assignment for word $w_j^{(i)}$. If $c_j^{(i)}$ is different from the word's original assignment i, then this word does not serve as a qualified

keyword for community i. In the second step, community centroids are re-computed.

The intuition of this algorithm is to first compute the centroid of each community, and discard those words which are actually closer to other communities. After this distillation, the words left in every community will serve as identical marks which can distinguish its own community from others. We denote those leftovers as keyword profiles refined by wording embedding. Figure 5 visualizes the embedding of keywords from 6 communities. Dimension reduction is done via t-SNE. Table 2 shows the community keyword profiles extracted via *Iterative Selection of Community Topic* algorithm.

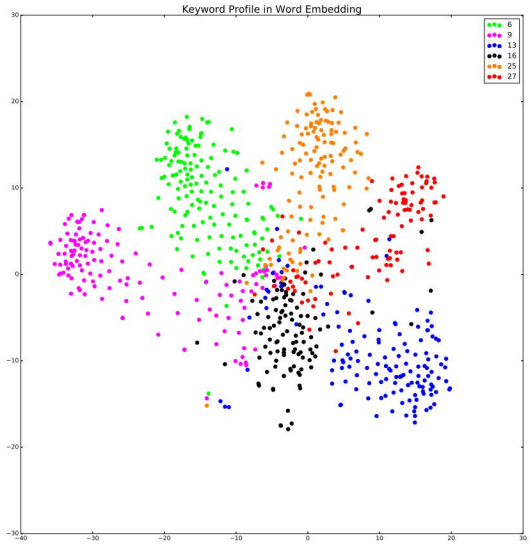

Figure 5. Word embedding: community keywords from community 6, 9, 13, 16, 25, 27

Community	keyword profile
ID=6 **E-Commerce**	market, commercial, demand, supply, trust, retail, enterprise, portfolio, inter-organization
ID=9, **Linguistics**	translation, language, semantic, text, speech, phrase, parser, spoken, retrieval, dialogue
ID=13, **Circuit**	circuit, fault, simulation, power, synthesis, memory, delay, chip, processor
ID=16, **Vision**	video, image, motion, wavelet, mpeg, distort, enhance, frame, decode, fingerprint, camera
ID=25, **Math**	nonlinear, approximation, finite, differential, singularity, spectral, chebyshev, asymptotic
ID=27, **Communication**	network, channel, wireless, cdma, access, packet, antenna, ofdm, router, bandwidth

Table 2. Community keywords (from word embedding) for community 6, 9, 13, 16, 25, 27 and our guess of their academic fields.

This part of work has the following significance.

(1) We can see significant improvement in the precision of community keywords by comparing the keyword profiles in Table 2 (extracted by word-embedding) and Table 1 (extracted by tf-idf features). Keywords extracted by tf-idf features has lots of spurious detection, where non-representative words (e.g. "original","types","form","including") are chosen to be community keywords. On the other hand, this iterative selection algorithm can construct a distinct profile with few spurious words that distinguishes communities from each other very well.

(2) Nodes within a community has linguistic attributes that also densely cluster in the space of word-embedding representation, as shown in Figure 5. We know that network nodes that are densely connected in network topology form communities. It is surprising when these network clusters also correspond to keyword clusters in word-embedding vector space, because word-embedding is trained with complete absence of network knowledge.

5.3.3. Intra- and Inter-Community Topic Variance

To model topic variance within each community i (i.e. intra-community topic variance), we take the set of keywords K_i that describes this community, and calculate the mean pair-wise cosine distance for each word pair in K_i. More concretely, to calculate pair-wise cosine distance, we take the word-embedding representation of each word and compute their cosine distance. Similarly, to model topic variance across two communities i, j (i.e. inter-community topic variance), we consider the cosine distance between every word pair w_1, w_2, where $w_1 \in K_i, w_2 \in K_j$, and calculate mean cosine distance of all word pairs. Formally, using the same notation as Sec. 5.3.2, we have the topic variance between community i and j is (when $i = j$, it is simply intra-class topic variance)

$$D_{ij} = \sum_{u,v} \frac{z_u^{(i)}}{\|z_u^{(i)}\|_2} \cdot \frac{z_v^{(j)}}{\|z_v^{(j)}\|_2}$$

where $z_j^{(i)}$ represents the word-embedding representation of the j-th word $w_j^{(i)}$ in the i-th community.

To obtain keyword set K_i associated with community i, we used two methods: (1) tf-idf, and (2) iterative selection algorithm with word-embedding to construct community keyword profiles. These two methods are exactly as described in Sec. 5.3.1 and Sec. 5.3.2. The comparison of intra- and inter-community topic variance is illustrated in Figure 6. The variance of topic within a community is smaller than across communities, which shows that each community is inclined toward talking about the same topic. Using the refined keyword profile extracted by iterative se-

lection algorithm gives us a much larger gap, which indicates that community keyword profile is better depicted by this algorithm. We also visualize in Figure 7 the community similarity matrix $S = \{1 - D_{ij}\}$, where D_{ij} is the pair-wise cosine distance of words as described above. Cells on the diagnal show intra-community word similarity, while other cells show inter-community word similarity. This visualization just uses another way to show that intra-community topic variance is much smaller than inter-community.

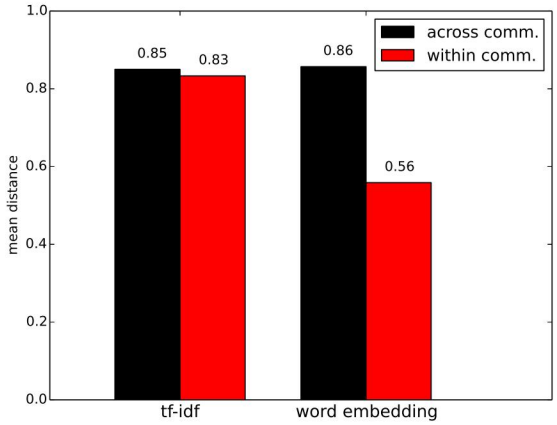

Figure 6. Intra- and Inter-Community topic variance, using both tf-idf and iterative selection method with word-embedding to obtain community keyword profile.

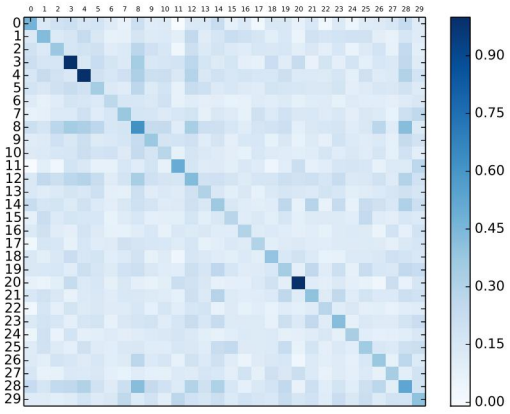

Figure 7. Similarity matrix of 30 largest communities. Each cell shows mean pair-wise cosine distance among word pairs in community keyword profiles extracted by iterative selection algorithm.

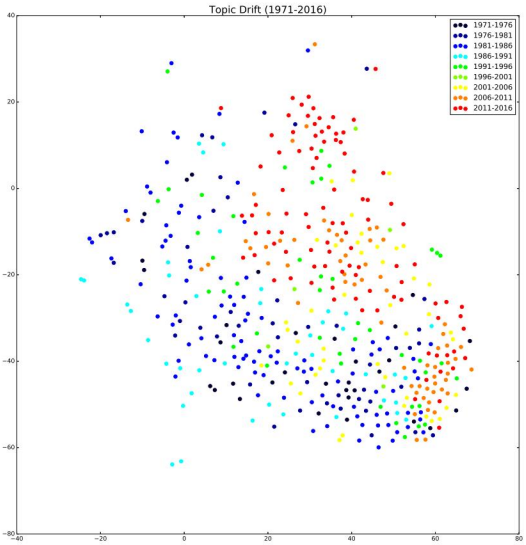

Figure 8. Topic drift of *communication* community from 1971 to 2016

5.4. Topic Drift

Being able to extract a clean community profile and reasonable spatial embedding gives us much more extra stuff to play with, among which an interesting question is: does each academic community's topic drift over the years, i.e. is popular research topics within an academic subfield changing over the years, as new research areas of computer science emerge and old areas die? To answer this question, we divide papers into 5-year bins from 1971 to 2015. We perform the pipeline of (1) extract tf-idf keyword profiles, (2) refine keyword profiles with iterative selection algorithm based on word-embedding, and (3) visualize word-embedding with t-SNE. Steps (1) and (2) are exactly as described in Sec. 5.3.1 and Sec. 5.3.2. Step (3) is slightly different, where we only visualize the set difference of keyword profiles. More concretely, for each 5-year bin from 1971 to 2015, we extract keyword profiles (step (1),(2)), which are sets of words denoted as $K_{1971\sim1975}, K_{1976\sim1980}, ..., K_{2010\sim2015}$. Topic drift is modeled as newly-emerged keywords in each community's keyword profile, because those keywords represent the cutting-edge and most popular topic of this research subfield. More formally, to model newly-emerged keywords, we use

$$K_{t\sim(t+5)} \backslash K_{(t-5)\sim(t)}, t \in [1971, 1975, ..., 2010]$$

We conduct the above steps on a well-known community *communication* (ID=27) and visualize its newly-emerged

keywords per 5 years in the word-embedding, as shown in Figure 8. Blue dots represent emerging words around 1980s while red dots are much more recent words emerging after 2010. We can see that the keywords in *communication* community has changed gradually over the past 30 years, moving from bottom-left corner to top-right corner. The blue dots are keywords e.g. **digitize, pattern, model, kalman, gaussian**, which is exactly the major research focus of *communication* back in 1980s. On the other hand, the red dots represent more modern topics in *communication*, e.g. **broadband, multicast, CDMA2000, constellation, wireless, multiplex, OFDM, MIMO**, which represent new research areas that have emerged in the last decade.

6. Code

Our code is publicly available at our github repo.

7. Conclusion

(1) There is strong coupling between network structure and node attributes in ACM academic network. Nodes within a community has linguistic attributes that also densely cluster in word-embedding vector space, i.e. topological clusters in network corresponds to semantic clusters. It is very surprising result given that word-embedding representation is trained with complete absence of network knowledge.

(2) Using tf-idf feature, we can obtain a reasonably good keyword profile for each community, which is indicative of community's academic subfield (as verified by most frequent conferences of papers in the community). However there are lots of spurious words. To solve this problem, we trained word-embedding on all paper titles, and proposed the *Iterative Selection of Community Topic* algorithm, which proved to be an effective approach that can be used to construct concise and precise community keyword profiles.

(3) Topic drift within research communities can be modeled, and clearly visualized using t-SNE of word-embedding representation. Vanished and emerged keywords over the years appear clustered by 5-year bins, and the topic drift within communities exactly corresponds with our common sense on computer science history.

8. Contribution

Chenyue Meng: run BigCLAM and tune parameters, visualize community structure using GePhi, code up, run and analyze word-embedding and t-SNE visualization, come up with and code up iterative selection algorithm, data preparation for intra- and inter-class topic variance, code up and analyze topic drift visualization. Write up abstract, 2.2, 4.2, 5.2, 5.3.2, 5.3.3, 5.4, 6).

Chenjie Yang: write-up of part of report with teammates, including introduction, related work, description problem overview and dataset. Using BigCLAM to detect communities together with teammates (not including visualization using GePhi). Code up the graph structure part and analyze the community structure.(1, 2.1, 3.1, 3.2, 4.1, 5.1, 5.2).

Yixin Wang: code up data parsing, run BigCLAM and tune parameters, visualize community structure using GePhi, code up, run and analyze tf-idf features, come up with and code up intra- and inter-class topic variance, code up topic drift data preparation. Write up 2.3, 3.1, 4.1, 5.2, 5.3.1, 5.3.3, 5.4, 6, 7.

References

Bastian, Mathieu, Heymann, Sebastien, Jacomy, Mathieu, et al. Gephi: an open source software for exploring and manipulating networks. *ICWSM*, 8:361–362, 2009.

Beyer, Kevin, Goldstein, Jonathan, Ramakrishnan, Raghu, and Shaft, Uri. When is nearest neighbor meaningful? In *International conference on database theory*, pp. 217–235. Springer, 1999.

Egghe, Leo and Rousseau, Ronald. Introduction to informetrics: Quantitative methods in library, documentation and information science. 1990.

Girvan, Michelle and Newman, Mark EJ. Community structure in social and biological networks. *Proceedings of the national academy of sciences*, 99(12):7821–7826, 2002.

Hofmann, Thomas. Probabilistic latent semantic indexing. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 50–57. ACM, 1999.

Mikolov, Tomas, Sutskever, Ilya, Chen, Kai, Corrado, Greg S, and Dean, Jeff. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pp. 3111–3119, 2013.

Newman, Mark EJ. Modularity and community structure in networks. *Proceedings of the national academy of sciences*, 103(23):8577–8582, 2006.

Palla, Gergely, Derényi, Imre, Farkas, Illés, and Vicsek, Tamás. Uncovering the overlapping community structure of complex networks in nature and society. *Nature*, 435(7043):814–818, 2005.

Schaeffer, Satu Elisa. Graph clustering. *Computer science review*, 1(1):27–64, 2007.

Yang, Jaewon and Leskovec, Jure. Community-affiliation graph model for overlapping network community detection. In *2012 IEEE 12th International Conference on Data Mining*, pp. 1170–1175. IEEE, 2012a.

Yang, Jaewon and Leskovec, Jure. Structure and overlaps of communities in networks. *arXiv preprint arXiv:1205.6228*, 2012b.

Yang, Jaewon and Leskovec, Jure. Overlapping community detection at scale: a nonnegative matrix factorization approach. In *Proceedings of the sixth ACM international conference on Web search and data mining*, pp. 587–596. ACM, 2013.

Yang, Jaewon, McAuley, Julian, and Leskovec, Jure. Community detection in networks with node attributes. In *2013 IEEE 13th International Conference on Data Mining*, pp. 1151–1156. IEEE, 2013.