Analyzing the Research Trend of NIPS 2015 Papers with Hierarchical Clustering and Visualization

Haoming Jiang
School of Gifted Young
University of Science and Technology of China
Hefei, Anhui
Email: jinghm@mail.ustc.edu.cn

***Abstract*—NIPS (Conference on Neural Information Processing Systems) is one of the most well known conferences on AI (Artificial Intelligence) and ML (Machine Learning) area. In this work, I use the data set of NIPS 2015 Papers from Kaggle, an online data science platform, to analyze the research interest trend of AI and ML area. First I used TF-IDF weighting to extract the feature from papers and use cosine of two feature vectors measure their similarity. Based on the defined similarity, I cluster the papers in different groups and find the topics which can represent these groups. Finally, some visualization techniques are applied in revealing these papers' relation.**

All data and codes are available in http://home.ustc.edu.cn/~jinghm/files/Research%20Trend%20Data.rar

I. Background

A. Topics of NIPS

Neural Information Processing Systems (NIPS) is one of the top machine learning conferences in the world [11]. The conference continues to span a wide range of topics and continues to grow, with over 2,500 registered participants in 2014. Besides machine learning and neuroscience, other fields represented at NIPS include cognitive science, psychology, computer vision, statistical linguistics, and information theory. Although the 'Neural' in the NIPS acronym was something of a historical relic, the recent resurgence of deep learning in neural networks, which has a wide range of practical applications in speech recognition, object recognition, natural language processing and brain imaging, has revived the initial impetus for the conference.

Since it contains various topics, whether common pattern of topics exits among these papers become a question. This is a **text mining task**. The first challenge is how to convert texts into a minable structure. I choose TF-IDF method to map the text into the vector space. In analyzing part, I applied the hierarchical clustering method to disclose the pattern first. The clustering results are discussed. In addition, some visualization techniques are utilized, including Word Cloud and T-SNE.

B. Data Set Description

The data set is obtained from Kaggle [12]. Ben Hamner built the data set from the raw pdf version of papers in NIPS 2015. Information has been categorized into three tables stored in *.csv* documents. A single SQLite database, which consist of all three tables, is also available. The *Paper* table contains one row for each of the 403 NIPS papers from this year's conference. The *Authors* table contains id's and names for each of the authors on this year's NIPS papers. The *PaperAuthors* table links papers to their corresponding authors.

I mainly focus on the content of papers. Some feature has been extracted in the *Paper* table. Specifically, it provide the following attributes:

- **Id** - unique identifier for the paper (equivalent to the one in NIPS's system)
- **Title** - title of the paper
- **EventType** - whether it's a poster, oral, or spotlight presentation
- **PdfName** - filename for the PDF document
- **Abstract** - text for the abstract (scraped from the NIPS website)
- **PaperText** - raw text from the PDF document (created using the tool pdftotext)

Among them, only *Title, Abstract and PaperText* are useful in my text mining task. Although *PdfName* could carry some information, it is not accurate owing to that people usually use abbreviation and some may even just a serial number. Actually, the author information can also be study by analyzing the author affiliation. Since researcher has their own research area, the relation between authors may also reflect the relation between papers. This is an interesting research direction, but it is not included in this work.

II. Measuring the Similarity of Papers

In this section, I describe the process of transforming the raw texts into a minable structure. First, some preprocessing procedures are discussed. Then, the vector space and TF-IDF representation, which is the foundation of the following discussion, is introduced.

A. Text Preprocessing

Before actually process the text data, some preprocessing procedure have to be done, in order to reach better performance. The main purpose of this part is to correct the potential mistakes, eliminate trivial information and turn the text into minable form. They are listed below.

- remove '$\backslash n$' '$\backslash x0c$' and other mistakes including some non-English words (e.g. Foreign Names)
- transform all words into lower case
- extract Tokens from the text

- remove stop words and extremely infrequent words.

The text is converted from pdf automatically by the tool *pdftotext*. Some escape character such as '$\backslash n$' and '$\backslash x0c$' will remain. It is necessary to replace them with a space.

When extracting tokens, for paper text and abstract, the sentence tokenizer is applied first. And then, the word tokenizer is applied. For titles, which is usually one sentence, the word tokenizer is applied directly.

We do not have to consider the stems of words, which is usually a problem in text processing (e.g. a verb may have different forms: 'd', 'does', 'did', 'done' etc.). Because the critical words in this area are usually nouns (e.g. 'Deep Learning', 'CNN', etc.) which do not have that much transformations, we can just process the raw text without turning words into stems. Also, in this way, the features can own high interpretability.

In computing, **Stop Words**, such as *we, and, or, than* , are words which are filtered out before or after processing of natural language data (text) [10]. Though stop words usually refer to the most common words in a language, there is no single universal list of stop words used by all natural language processing tools, and indeed not all tools even use such a list. Some tools specifically avoid removing these stop words to support phrase search. However, in my code I did not explicitly remove these stop words. They are removed automatically by the following process of TF-IDF Representation. In the word cloud section, stop words and some trivial words are removed explicitly.

Infrequent words do not add anything to the similarity computation. In some cases, such words may be misspellings or typographical errors in documents. Since the text is converted from pdf by the the tool *pdftotext*, some converting mistakes are inevitable.

A sample text before transformation:

> Predtron: A Family of Online Algorithms for General\nPrediction Problems\nPrateek Jain\nMicrosoft Research, INDIA\nprajain@microsoft.com\n\nNagarajan Natarajan\nUniversity of Texas at Austin, USA\nnaga86@cs.utexas.edu\n\nAmbuj Tewari\nUniversity of Michigan, Ann Arbor, USA\ntewaria@umich.edu\n\nAbstract\nModern prediction problems arising in multilabel learning and learning to rank\npose unique challenges to the classical theory of supervised learning. These problems have large prediction and label spaces of a combinatorial nature and involve\nsophisticated loss functions.

After the transformation, the text turns into:

> predtron a family of online algorithms for general prediction problems prateek jain microsoft research india prajain microsoft com nagarajan natarajan university of texas at austin usa naga cs utexas edu ambuj tewari university of michigan ann arbor usa tewaria umich edu abstract modern prediction problems arising in multilabel learning and learning to rank pose unique challenges to the classical theory of supervised learning these problems have large prediction and label spaces of a combinatorial nature and involve sophisticated loss functions

After transformation, the text is much more clearer.

B. Vector Space and TF-IDF

In the **vector space model** [3], we are given a set of documents D. Each document is a set of words. The goal is to convert these textual documents to (feature) vectors. We can represent document i with vector d_i,

$$d_i = (w_{1,i}, w_{2,i}, ..., w_{N,i})$$

, where $w_{i,j}$ represent the weight for word j that occurs in document i and N is the number of words used for vectorization.

A **binary representation** is to set $w_{i,j}$ to 1 when the word j exist in document i and 0 when it does not. A more generalized approach is to use the **term frequency-inverse document frequency(TF-IDF)** weighting scheme. In TF-IDF, $w_{i,j}$ is calculated as:

$$w_{i,j} = tf_{i,j} \times idf_j$$

, where $tf_{i,j}$ is the frequency of word j in document i. idf_j is the inverse TF-IDF frequency of word j across all documents,

$$idf_j = \log_2 \frac{|D|}{|\{document \in D | j \in document\}|}$$

which is the logarithm of the total number of documents divided by the number of documents that contain word j. TF-IDF assigns higher weights to words that are less frequent across documents and, at the same time, have higher frequencies within the document they are used. This property enable the TF-IDF to filter stop words, such as 'the' 'a',which are common in all documents. In order to illustrate that TF-IDF can really discover the key topic words, I extract the words with the highest $w_{i,j}$ in the sample document.

index	Based on PaperText	Based on Abstract
0	submodular	submodular
1	drsubmodular	naive
2	cs	cover
3	lattice	running
4	xi1921	bicriteria
5	sensor	logfactor
6	ks	logr
7	diminishing	onlog
8	1992\017	diminishing
9	greedy	integer

From the above result, applying TF-IDF to abstract can obtain a better key words extraction. The abstract of a paper

have contain the most essential information of the paper, which is sufficient for analyzing. Although the content of paper provides more than that, it include a large amount of noise at the same time, such as auxiliary numbers, mathematical symbols and latent typesettings.

III. Clustering Approach of Discovering the Pattern

After obtaining the vector representation of text, the clustering technique **AHC(Agglomerative Hierarchical Clustering)** is applied.

A. Cosine Similarity

I use the **cosine similarity** to measure the distance between two vector representations. As the book [1] mentioned, the *cosine similarity* between two documents U and V is defined as follow:

$$cosine(U,V) = \frac{\sum_{i=1}^{k} f(u_i)f(v_i)}{\sqrt{\sum_{i=1}^{k} f(u_i)^2}\sqrt{\sum_{i=1}^{k} f(v_i)^2}}$$

u_i and v_i are the components of the vector representation of U and V, while $f(\cdot)$ represent the damping function. Usually, damping function for $f(\cdot)$ could represent either square-root or the logarithm [1]. Here, because the vector representation is really sparse, which means many components are 0s, the logarithm is not applicable. Compare to that the square-root seems like a more reasonable choice, owing to that it can magnify the terms which are not 0s. Each term of the vector representation must be smaller than 1, which is an obvious fact indicated by the definition of the TF-IDF Representation, as a result the magnifying ability of damping function is established.

The dissimilarity between two documents U and V is also easy to obtained:

$$dissim(U,V) = 1 - cosine(U,V)$$

B. Agglomerative Hierarchical Clustering

In order to make a clear instinct about the similarity I double the similarity except those measured between it and itself and equal 1s. Since the maximum similarity is about 0.49, doubling will not make these similarity exceed 1. And it will also not affect the clustering results with a better clustering plot.

Here I choose the Ward [5] and average linkage method to decide which groups to merge. After applying AHC, the tree hierarchical plot is obtained in Figure 1 and Figure 2. From Figure 1, we can see that some papers (on the left part of the figure) do not belongs to any cluster group. The clustering result is not satisfying. However, the Ward method gives a better cluster result as the Figure 2 shows.

After gaining the hierarchical tree, the subgroups are achieved by cutting the tree. However, choosing the number of cluster is also a problem. So, I use the strategy of keeping separating the largest group. Specifically, the number of clusters keep being enlarge until when at some step, the divided group is not the largest one. For the AHC tree in Figure 2, the process is presented bellow:

$$5\ clusters:\ 250\ 108\ 15\ 24\ 6$$

$$6\ clusters:\ 229\ 108\ 15\ 24\ 21\ 6$$

$$7\ clusters:\ 229\ 99\ 9\ 15\ 24\ 21\ 6$$

Finally, I choose 6 for the number of clusters. Next, we exam whether these clusters are really related. The title of these are sampled. Since the first cluster is so large that the meaning it represent will be really complex or general. More valuable will be discovered in the other clusters.

For cluster2, which contains 108 papers and is the second large group, the sampled titles are as follows:

- "Rectified Factor Networks"
- "Neural Adaptive Sequential Monte Carlo"
- "Recovering Communities in the General Stochastic Block Model Without Knowing the Parameters"
- "Analysis of Robust PCA via Local Incoherence"
- "Sparse Local Embeddings for Extreme Multi-label Classification"

From the sampled titles, we can tell that these papers are related to *Feature Selection, Dimension Reduction (paper 1,3,4,5)*, and *Stochastic Models (paper 1,2,3,4)*

For cluster5, which contains 21 papers and is the middle size group, the sampled titles are as follows:

- "Probabilistic Line Searches for Stochastic Optimization"
- "On the Global Linear Convergence of Frank-Wolfe Optimization Variants"
- "A Normative Theory of Adaptive Dimensionality Reduction in Neural Networks"
- "Beyond Sub-Gaussian Measurements: High-Dimensional Structured Estimation with Sub-Exponential Designs"
- "Learning Bayesian Networks with Thousands of Variables"

From the sampled titles, we can tell that these papers are related to *High-Dimension (paper 3,4,5)*, *Estimation and Optimization (paper 1,2,4)* and *Network Structure (paper 3,5)*

From the above result, although it can really discover some similarity between papers. But each cluster represent multi-concepts. And also these clusters have some intersection, for example, cluster2 and cluster5 all contain the papers dealing with High-Dimension. All of the above happen, because the information contained limited information. The algorithm can not find out deeper information of these words without further knowledge, for example it can not know the relation between 'High Dimension' and 'Dimension Reduction'.

For the above consideration, a more rational data can be used – *Abstract* of papers. Although the paper content contains more words and more information, it meanwhile contains more noise, such as mathematical signs, names and other text related symbols. Similar process is done with the abstract papers. The AHC result is presented in Figure 3. The best cluster number

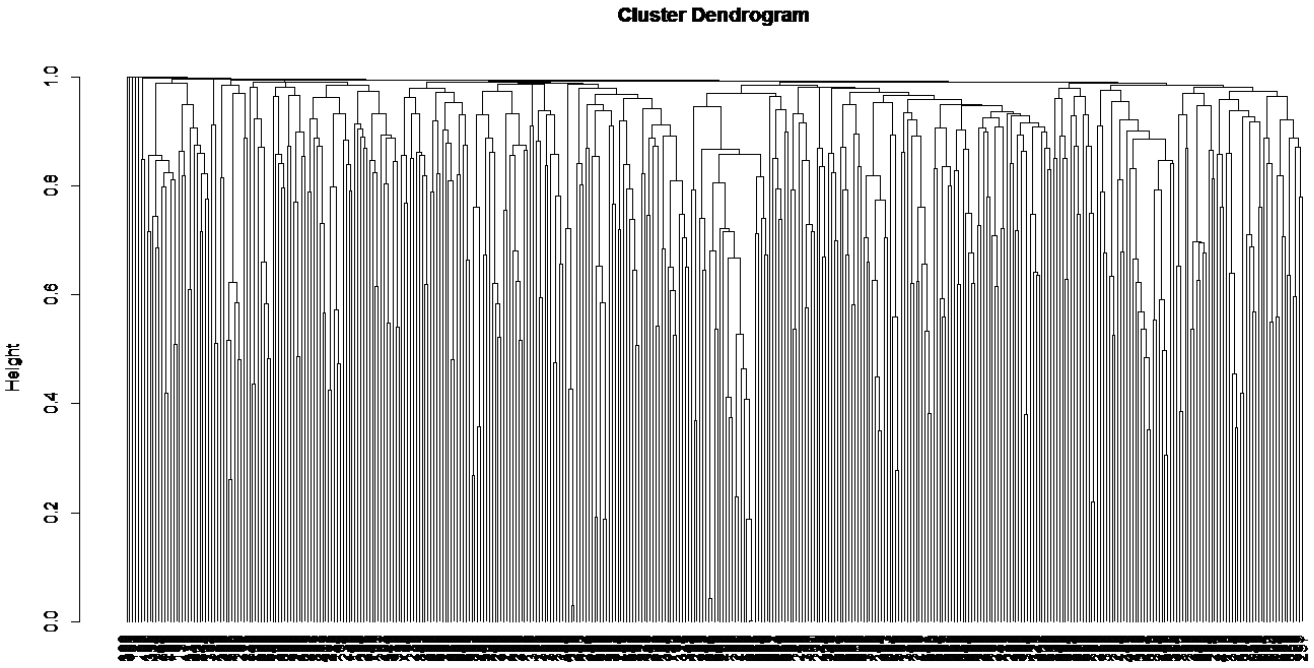

Fig. 1. The Clustering Result of AHC with Average Linkage and Title Data

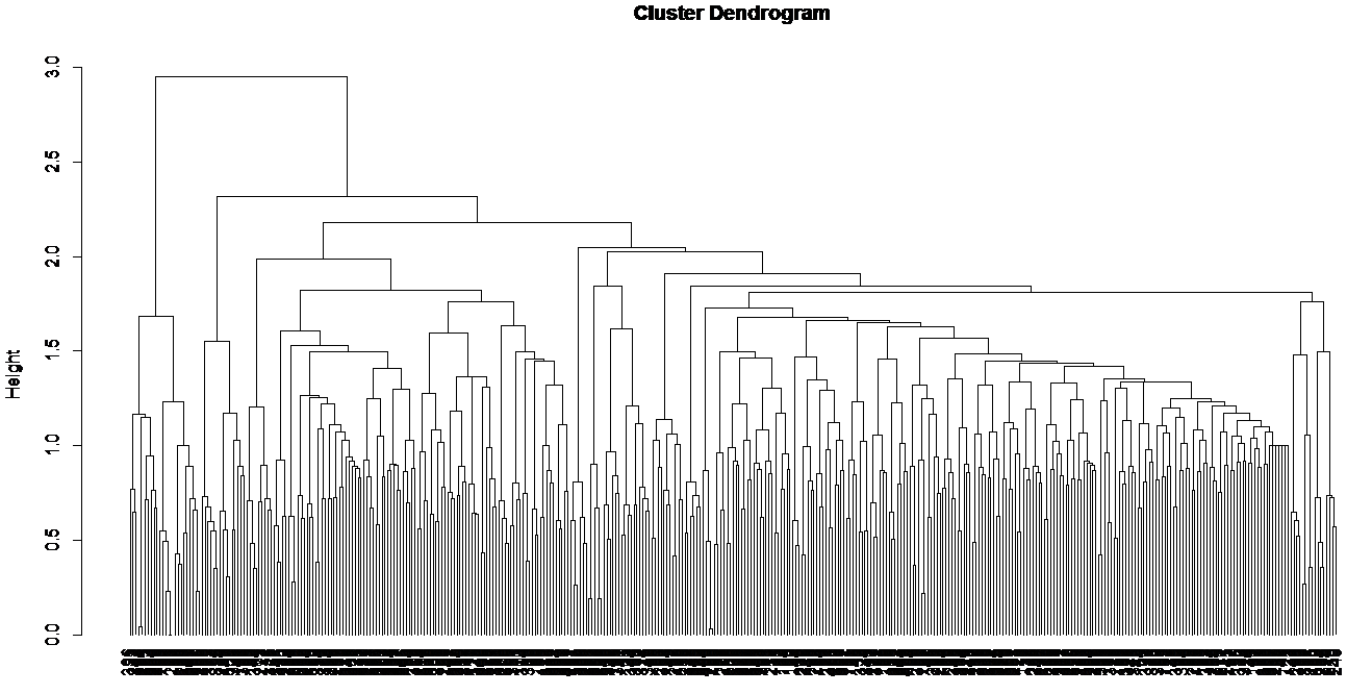

Fig. 2. The Clustering Result of AHC with Ward Method and Title Data

is 20. It can be discovered by both the tree plot (choosing the threshold around 1.5) and search strategy of continuously checking the size of clusters. The amount of papers which consist of each cluster is presented as follows.

$$20\ clusters:\ 73\ 12\ 18\ 15\ 9\ 12\ 23\ 31$$

$$12\ 15\ 22\ 22\ 43\ 20\ 16\ 24\ 9\ 17\ 4\ 6$$

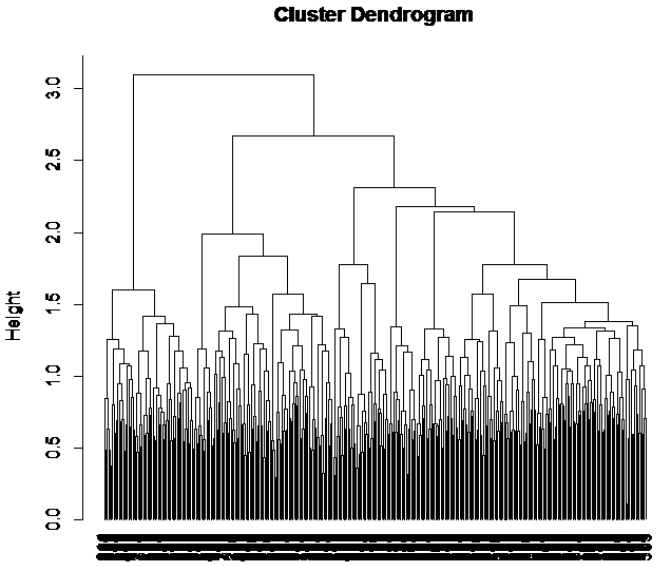

Fig. 3. The Clustering Result of AHC with Ward Method and Abstract Data

Instead of investigating the titles of the paper, in order to find out what meaning the cluster represent, we use the visualization technique of **Word Cloud** to investigate the topic in different clusters. Here I select the cluster1 which is the largest cluster, cluster7 which is a representative of middle size cluster and cluster19 which is the smallest cluster to analyze. Other results are presented in the next section.

Figure 4 shows the word cloud of the largest cluster. Since it is a large cluster containing 73 papers, it contains sophisticated contents and topics. Most words of it are large. However, the topic of *INFORMATION* seems like a stem of these papers.

Fig. 4. The Word Cloud of the Largest Cluster (No.1)

Figure 5 shows the word cloud of the middle size cluster. The word *GREEDY* is prominent. while some words including *GRADIENT, DESCENT* appears. It can be indicated that the contents of these paper must be related to optimizing techniques like *GRADIENT DESCENT* and other *Greedy Methods* (In some cases Gradient Descent is also a greedy methods).

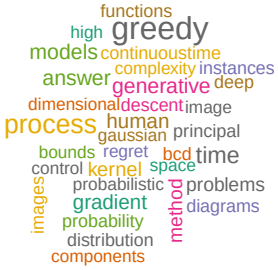

Fig. 5. The Word Cloud of the Largest Cluster (No.7)

Figure 6 shows the word cloud of the smallest size cluster. These papers' research are related to the structure of the *SPACE*, as *COMPLETION, SUBSPACE* in word cloud illustrated. Since it only contains 4 papers, I can examine them one by one.

- "b-bit Marginal Regression": It is a way of processing *SPARSE* signals.
- "SubmodBoxes: Near-Optimal Search for a Set of Diverse Object Proposals": It formulates the search for a set of bounding boxes from the *LARGE SPACE* of all possible bounding boxes in an image.
- "Differentially private subspace clustering": It just as the title mentioned *SUBSPACE*.
- "Unified View of Matrix Completion under General Structural Constraints": It present a unified analysis of matrix completion under general *LOW-DIMENSIONAL* structural constraints.

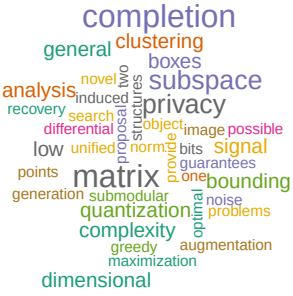

Fig. 6. The Word Cloud of the Largest Cluster (No.19)

IV. Visualization

A. Word Cloud

Word Cloud All word clouds in this paper are constructed on TF-IDF. The word cloud can be constructed by TF(Term

Frequency) and other measurement.

It also can be used a tool of examining the result of text clustering as the previous section mentioned. All cluster result is presented in Figure 7. Most of them have a prominent topics, which means the cluster result can really discover some pattern of the research interest in NIPS papers.

The results of word cloud and clustering are summarized (some clusters, which contain multiple topics, are denoted as 'Complex'):

	Topic	Paper Amount
01	INFORMATION	73
02	COMPLEX	12
03	REGULARIZATION	18
04	COMPLEX	15
05	CAUSAL STUDY	09
06	STOCHASTIC	12
07	GREEDY METHOD	23
08	INFERENCE AND ESTIMATORS	31
09	RELATION	12
10	COMPLEX	15
11	RISK	22
12	COMPLEX	22
13	NETWORKS	43
14	PATTERNS	20
15	MATRIX	16
16	PHYSICAL	24
17	BOUNDS	09
18	SCREENING	17
19	SPARSE	04
20	COMPLEX	06

Comparison between word cloud data base, Figure 8 is presented. From the figure, the one which is based on the titles perform better than the abstract one in magnifying the key topic words. It can be concluded that a larger group of papers can be analyze by the titles, while the smaller ones prefer the abstracts. This conclusion is really natural and is also indicated by the discussion in the clustering section.

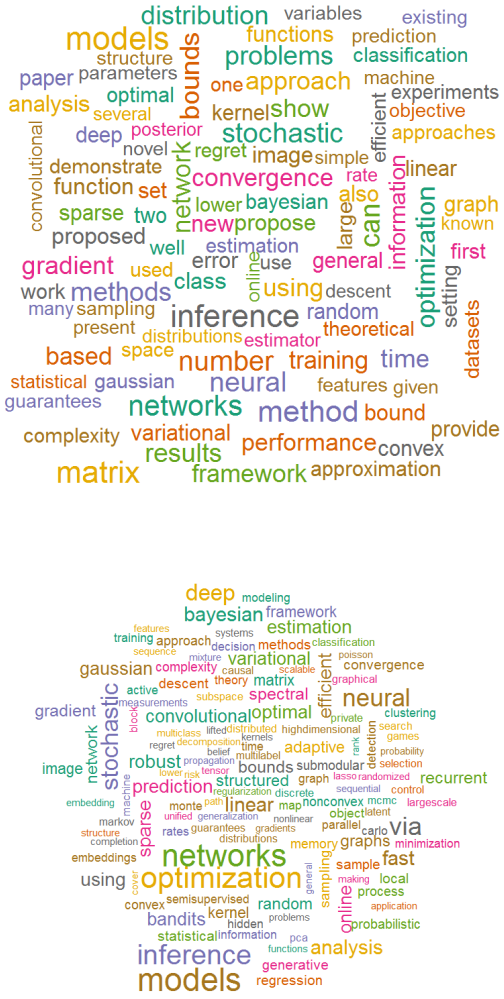

Fig. 8. The Word Cloud Based on Article Abstracts and Titles. The upper one is constructed from abstracts and the other one is from titles.

B. Large Data Visualization

Currently, many research about visualizing large-scale and high-dimension data (e.g. text mining) have been done [6]. Some typical ways are *SNE* (proposed by Hinton), *t-SNE* (proposed by Maaten) and LargeVis (proposed by Jian). All of them turn the High-Dimensional data into 2-Dimension which is easy to draw. I embed the high-dimensional (thousands of features) into 2-Dimension via different approach to find out the which one is good for visualizing the textual data. In addition, I use the results from hierarchical clustering to label the data. With the help of these labels, we can validate the visualization. Conversely, visualization can also validate clustering result.

PCA is a well-known Dimension Reduction techniques. Applying it to TF-IDF representation of abstracts, Figure 9 is obtained. I set it as a fundamental baseline of visualization.

In Figure 9, some clusters are tangled together. Although, it can disclose some patterns of the data, it does not clear enough.

Isomap [8] is a nonlinear dimensionality reduction method. Isomap is used for computing a quasi-isometric, low-dimensional embedding of a set of high-dimensional data points. The algorithm provides a simple method for estimating the intrinsic geometry of a data manifold based on a rough estimate of each data points neighbors on the manifold. Applying it to TF-IDF representation of abstracts, Figure 10 is obtained.

Although, compared to PCA, Isomap is able to distinguish some clusters from tangling, such as the red and green ones, some other clusters are not well separated, for example the purple ones are clustering together but are tangled with the red ones.

Locally Linear Embedding(LLE) [9] is an unsupervised learning algorithm that computes low-dimensional,

Fig. 7. The Word Cloud of Every Cluster Based on Article Abstract. The index of the word cloud start from the first row and is counted by columns first. The cluster4,6,7,8,9,11,14,15,16,17,19 have magnificent key topic words.

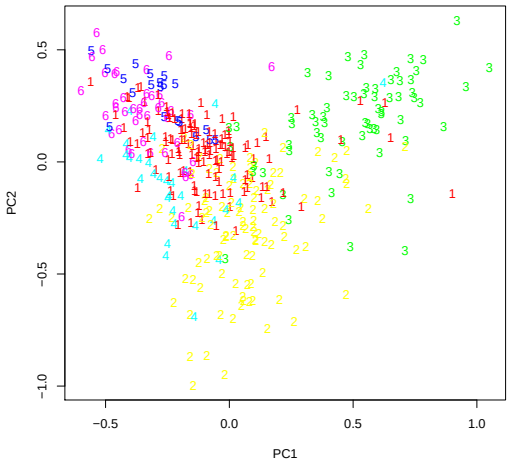

Fig. 9. Visualization of TF-IDF Representation of Abstracts via PCA Approach

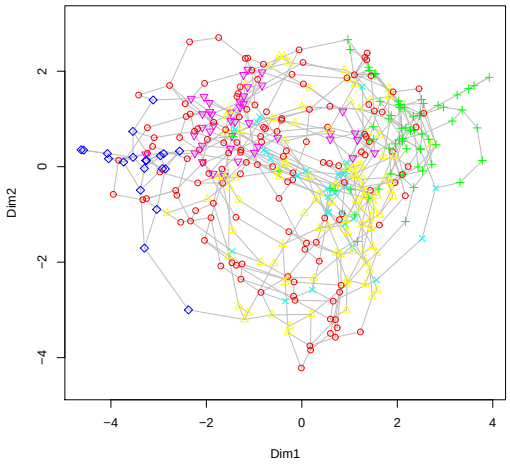

Fig. 10. Visualization of TF-IDF Representation of Abstracts via Isomap Approach

neighborhood-preserving embeddings of high-dimensional inputs. Unlike clustering methods for local dimensionality reduction, LLE maps its inputs into a single global coordinate system of lower dimensionality, and its optimizations do not involve local minima. By exploiting the local symmetries of linear reconstructions, LLE is able to learn the global structure of nonlinear manifolds, such as those generated by images of faces or documents of text. Applying it to TF-IDF representation of abstracts, Figure 11 is obtained.

The pattern LLE found is similar to the one PCA found. Compared to the PCA one, this one is just like the rotated version with higher density. Owing to that the data is highly concentrated, some originally distinctive clusters are tangled together. The performance is the worst.

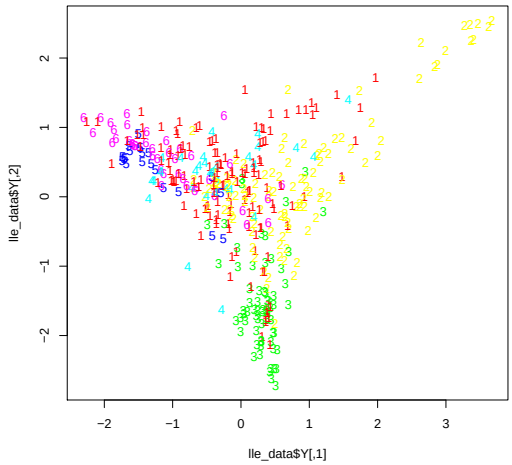

Fig. 11. Visualization of TF-IDF Representation of Abstracts via LLE Approach

t-Distributed Stochastic Neighbor Embedding (t-SNE) [7] is a technique for dimensionality reduction that is particularly well suited for the visualization of high-dimensional datasets. The technique can be implemented via Barnes-Hut approximations, allowing it to be applied on large real-world datasets. T-SNE represents each object by a point in a two-dimensional scatter plot, and arranges the points in such a way that similar objects are modeled by nearby points and dissimilar objects are modeled by distant points. Applying it to TF-IDF representation of abstracts, Figure 12 is obtained.

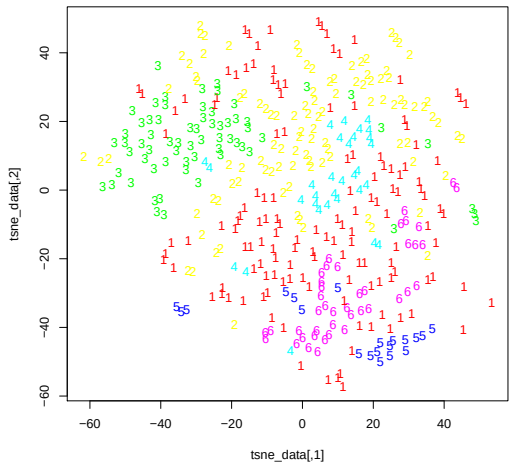

Fig. 12. Visualization of TF-IDF Representation of Abstracts via t-SNE Approach

Compared to PCA, t-SNE performance better and match the result of clustering, owing to two characteristics: (1) t-SNE mainly focuses on appropriately modeling small pairwise distances, i.e. local structure, in the map and (2) because

t-SNE has a way to correct for the enormous difference in volume of a high-dimensional feature space and a two-dimensional map. As a result of these two characteristics, t-SNE generally produces maps that provide much clearer insight into the underlying (cluster) structure of the data than alternative techniques.

	Speed	Performance
PCA	fast	Soso
Isomap	very fast	Good
LLE	very slow	Worst
t-SNE	slow	Brilliant

Among all techniques, t-SNE has the best performance. From that visualization result, the clustering result is validated. It also gives us an instinctive insight about the documents.

V. Conclusion

Finding the research trend from papers is a text mining task. The text must be preprocessed into minable structure. Some cleaning and representing techniques are applied (e.g. TF-IDF, which the basis of the following research). After obtaining the minable vector representation, Different Agglomerative Hierarchical Clustering are applied to find subgroups of the paper. In the procedure, both the title data and abstract data are used, in order to find better result by comparison. Eventually, it comes to the conclusion that abstracts are more appropriate for this text mining task. The clustering result is validated and presented by both hierarchical tree plot and word cloud. Three representative (the largest, the smallest and a middle size one) clusters are analyzed in details. Other clusters are presented in the word cloud where we can find the research trend. Besides word cloud, other alternative visualization method are applied in investigating the textual pattern (e.g. PCA, Isomap, t-SNE). Although the visualization result is directly interpretable, it shows us the cluster pattern and an instinctive insight of the documents.

References

[1] Aggarwal C C, Zhai C X. Mining text data[M]. Springer Science & Business Media, 2012.

[2] Salton G, McGill M J. Introduction to modern information retrieval[J]. 1986.

[3] Zafarani R, Abbasi M A, Liu H. Social media mining: an introduction[M]. Cambridge University Press, 2014.

[4] Salton G, Buckley C. Term-weighting approaches in automatic text retrieval[J]. Information processing & management, 1988, 24(5): 513-523.

[5] Ward J H. Hierarchical Grouping to Optimize an Objective Function[J]. Journal of the American Statistical Association, 1963, 58(301):236-244.

[6] Tang, Jian, et al. "Visualizing Large-scale and High-dimensional Data." Proceedings of the 25th International Conference on World Wide Web. International World Wide Web Conferences Steering Committee, 2016.

[7] L.J.P. van der Maaten and G.E. Hinton. Visualizing High-Dimensional Data Using t-SNE. Journal of Machine Learning Research 9(Nov):2579-2605, 2008

[8] J. B. Tenenbaum, V. de Silva, J. C. Langford, A Global Geometric Framework for Nonlinear Dimensionality Reduction, Science 290, (2000), 2319C2323.

[9] Sam T. Roweis and Lawrence K. Saul. Nonlinear Dimensionality Reduction by Locally Linear Embedding, Science, Dec 22 2000:2323-2326

[10] Wikipedia, Stop Words, https://en.wikipedia.org/wiki/Stop_words, 2016/6/16

[11] Wikipedia, Conference on Neural Information Processing Systems, https://en.wikipedia.org/wiki/Conference_on_Neural_Information_Processing_Systems, 2016/6/16

[12] Ben Hamner, https://www.kaggle.com/benhamner/nips-2015-papers, 2016/6/16