Word Embedding Based Document Similarity for the Inferring of Penalty

Tieke He(✉), Hao Lian, Zemin Qin, Zhipeng Zou, and Bin Luo

State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210093, China
hetieke@gmail.com

Abstract. In this paper, we present a novel framework for the inferring of fine amount of judicial cases, which is based on word embedding when calculating the distances between documents. Our work is based on recent studies in word embeddings that learn semantically meaningful representations for words from local occurrences in sentences. This framework considers the context information of words by adopting the *word2vec* embedding, compared to traditional processing methods such as hierarchical clustering, kNN, k-means and traditional collaborative filtering that rely on vectors. In the area of judicial research, there exists the problem of deciding the amount of fine or penalty of legal cases, in this work we deal with it as a recommendation task, specifically, we divide all the legal cases into 7 classes by the amount of fine, and then for a target legal case, we try to infer which class this case belongs to. We conduct extensive experiments on a legal case dataset, and the results show that our proposed method outperforms all the comparative methods in metrics *Precision*, *Recall* and *F1-Score*.

Keywords: Collaborative Filtering · Word embedding
Penalty inferring

1 Introduction

Today, we live in the information age and are confronted with much information from all scopes of our lives. With the rapid development of the internet, we have entered an era of information explosion [16]. Due to this enormous amount of information, it is necessary to rely on techniques which are capable of filtering available data and allow the search for suitable data. In this sort of circumstance, recommender systems appear as a practical methodology [10]. Recommender systems adopts information filtering to recommend information of interest to a user and are defined as the system which suggests an appropriate product or service after learning the users' preferences and requirements [11,28].

This work is supported in part by the National Key Research and Development Program of China (2016YFC0800805), and the Fundamental Research Funds for the Central Universities (021714380013).

X. Meng et al. (Eds.): WISA 2018, LNCS 11242, pp. 240–251, 2018.
https://doi.org/10.1007/978-3-030-02934-0_22

As recommender systems are playing an increasingly critical role in many fields, the purpose of this paper is to seek opportunity of adopting this kind of technology in the judicial study, as well to introduce a novel framework in tackling the specific task as inferring the amount of fine or say penalty in legal cases.

Judicial study has witnessed progress in an amount of directions, such as speech recognition and computer vision in remote trial, as well as data compression, storage and transmission. However, there is little effort devoted to the knowledge discovery of law cases, which constitutes the most crucial part of the judicial big data. A legal case mainly consists of the description of the fact of the case, the rules or conditions that are applied, the basic information of the accuser and accused, as well as the lawyer and the court information. Getting to the final decision of a law case has always been a complex task for the court, it often involves with many rounds of negotiation and bargaining, along with referring to applicable codes and conditions, as well as historical similar cases. Each step of this process calls for intensive human effort and expertise. Thus, offering aid in some of these steps can greatly benefit the judicial cause, so as to improve the efficiency of the whole judicial system. In an attempt to mimic the process of court judgement, we seek to provide quantitative analysis of certain steps from the view of computer science. It is widely acknowledged that, for most parties the threat of being fined or punished provides incentives to take care not to harm others [1]. For instance, industrial firms may resist fouling the air, motorists may obey traffic regulations, and manufactures try to produce safe toys – all to avoid fines for violation of standards. Motivated by this, we start out by inferring the penalty or the amount of a fine, which functions as a practical utility in guiding judges to the final decision of a legal case.

Previous studies such as incorporating the court trial in their models [12,18], is not often explicitly analyzed. The outcome of the court is somehow predetermined and the decision moment in their models falls before the actual verdict. In their study, Polinsky and shavell [19] made an extensive economic analysis of the optimal level of punitive damages in a variety of circumstances. Following, Daughety and Reinganum [6] model both the settlement and the litigation process for decisions made in court, allowing for incomplete information about the damages incurred by the plaintiff on the part of both the defendant and the court. According to the analysis of Earnhar [7], driving factors behind penalty decisions can be divided into five categories: causes of accidents, measured damages, environmental factors, regional factors and political influence. And their study shows significant differences between the different regions and the between different political systems. Also, the size and the cause of the violation influence the level of the fine.

From a traditional point of view, we start off the task of inferring the penalty in the way of classic recommendation. That is, we hire methods such as hierarchical clustering, kNN, k-means and traditional collaborative filtering, which are based on the results of TF-IDF [20]. Specifically, TF-IDF is adopted to extract the words in the textual description of legal cases, and then a vector of words is generated for each legal case, which constitutes as the foundation of following processing in clustering algorithms and similarity calculation. The drawback of this approach is that, it overlooks the fact that words possess context information in a document, except the weight they carry according to the document or the whole corpus. And this leads to the sacrifice of accuracy of recommendation when generating the recommendation list.

In seeing this, we bring in the technique of word embedding, it deals with documents as a matrix of words, i.e., the embedding methods learns a vector for each word following the way of *word2vec* by using a shallow neural network language model, specifically, a neural network architecture (the skip-gram model) that consists of an input layer, a projection layer, and an output layer to predict nearby words is adopted, and then a matrix is constructed for each document through these vectors. There are some plausible embeddings that are available, such as [14,15], [4], [17] and [24].

We are dealing with the task of inferring the penalty in two ways, the first is by clustering the legal cases, and then the penalty of the targeted case is obtained by the majority voting strategy, the other is through collaborative filtering, that is, finding the neighbors of the targeted case, and then again adopting the voting strategy. Technique details will be introduced in the following section that presenting our framework.

In summary, our work makes the following contributions:

- We are among the beginners in the judicial research of introducing the recommendation techniques to tackle the problem of penalty inferring, and a recommendation framework is presented to deal with this task, which consists of two sorts of methodologies, one is clustering methods, and the other is collaborative filtering.
- Embedding methods are adopted to improve the accuracy of the recommendation, which considers the context information of words in the document, rather than traditional methods such as TF-IDF that treats word as independent to document.
- Extensive experiments are conducted on a judicial dataset, and the results show that the recommendation framework is effective, and the embedding method outperforms all the traditional methods.

The rest of this paper is organized as follows. Section 2 reviews the related work. We present our framework in Sect. 3, as well as the embedding method in aiding the similarity calculation. We report the experimental results in Sect. 4. We conclude our paper in Sect. 5.

2 Related Work

In our work, we mainly compare the effects of four different methods on the penalty recommendations. These four methods are hierarchical clustering, kNN, k-mans and collaborative filtering methods. Following this, we introduce the related work in this section.

Hierarchical Clustering is a simple while practical clustering algorithm. In detail, there are two main types of Hierarchical Clustering which are top-down approach and bottom-up approach. When performing Hierarchical Clustering on tags, using the co-occurrence as the distance, Hierarchical Clustering can get the relative sets of tags with chosen distance, which can then be used as the 'topics'. In Hierarchical Clustering, each word or tag can only be allocated in only one set of tags or 'topics'.

There are two relatively novel algorithms of Hierarchical clustering method. BIRCH [25] (Balanced Iterative Reducing and Clustering Using Hierarchies) is mainly used when the amount of data is large, and the data type is numerical. In this method, the tree structure is first used to divide the object set, and then using other clustering methods to optimize these clusters. ROCK [8] (A Hierarchical Clustering Algorithm for Categorical Attributes) are mainly used on categorical data type.

The advantages of hierarchical clustering are mainly in the following aspects: (1) In this algorithm, the similarity between the distance and the rules is easy to define and less limited. (2) The algorithm does not need to pre-set the number of clusters. (3) The hierarchical relationship of the classes can be found. However, hierarchical clustering has the disadvantages of high computational complexity and sensitivity to singular values.

The K-means [27] algorithm is a single iterative clustering algorithm that partitions a given dataset into a user-specified number of clusters, i.e., K. It is simple to implement and run, relatively fast, easy to adapt, and common in practice. Via a clustering algorithm, data are grouped by some notion of "closeness" or "similarity". In K-means, the default measure of closeness is Euclidean distance. The K-means algorithm has some drawbacks: (1) it is very sensitive to its initial value as different ones may lead to different solutions; and (2) it is based on an objective function simply and usually solves the extreme value problem by a gradient method.

K-means is simple, which is easy to understand and implement with low time complexity. However, there are also the following defects: (1) K-means needs manually set the number of classes and it is sensitive to the initial value setting, so K-means++, intelligent K-means, and genetic K-means are used to make up for this deficiency; (2) K-means is very sensitive to noise and outliers, so K-medoids and K-medians are available to compensate for this defect; (3) K-means is only used for numerical data, not for categorical data.

KNN [9] is an instance-based learning algorithm. Although it is most often used for classification, it also can be used in estimation and prediction. Given a set of training data, a new data may be classified simply by comparing it to the most similar data in the training dataset. The process of building KNN classifier involves identifying k value, the number of the most similar classes to be considered in the training dataset. The process involves also measuring the similarity based on defining the distance function. The most commonly used distance function is Euclidean distance.

The KNN algorithm has high accuracy and is insensitive to outliers. It can be used both for classification and regression, including nonlinear classification. While the training time complexity is only $O(n)$. However, this algorithm requires a large amount of storage memory, which is one of its disadvantages. There is also the problem of sample imbalance (i.e., there are a large number of samples in some categories and a small number of other samples).

Collaborative filtering (CF) is a technique mostly used by recommender systems. There are two categories of Collaborative Filtering, user-based collaborative and item-based collaborative. User-based method first find out several similar users to the active user, and then aggregates items preferred by these similar users to generate a recommendation list. Item-based method has been proposed to address the scalability problem of user-based method. Many adjustments to original collaborative filtering have been proposed. Resnick et al. [21] computed user similarity based on ratings on items which have been co-rated by both users. Chee et al. [2] used the average ratings of a small group of users as default ratings. Inverse user frequency was proposed in [3], and the concept of inverse user frequency is that popular items preferred by most users should not be recommended.

3 Framework

We introduce the proposed framework in this section. As mentioned above, our solution of tackling the problem of penalty inferring consists of two sorts of methodologies, one is by clustering, and the other is through collaborative filtering. The framework is illustrated in Fig. 1.

From Fig. 1 we can see the overall workflow of penalty inferring process. We first perform some data preprocessing on the original documents. We perform the data preprocessing as segmentation, filtering the stop words and property labelling. We left the option of stemming because previous research [5] shows that it has less impact.

Segmentation is first performed on the textual description. For the Chinese document, the results of segmentation greatly affect the effect of the whole task, we compared a series of Chinese lexical analyzer and finally choose the *NLPIR* [29] to do the segmentation. Once we get the result of segmentation, we need to remove the stop words. Stop words are the set of common words such as *the*, *is*, *an* etc., these words are not important for our task of bug management and should be removed. Mainly, we adopt a commonly used Chinese stop word list in this step. Afterwards, we adopt the *NLPIR* to label the words. And an original feature space is obtained for each legal case in this step.

The left part of Fig. 1 depicts the methods that are based on feature vector, while the right part describes the method based on feature matrix.

For the left part, to best illustrate our idea in this step but without loss of generality, we choose TF-IDF to construct the vector space, *Term Frequency-Inverse Document Frequency* [22] is a numerical statistic model that is intended to show how important a word is to a document in a collection or corpus. We first

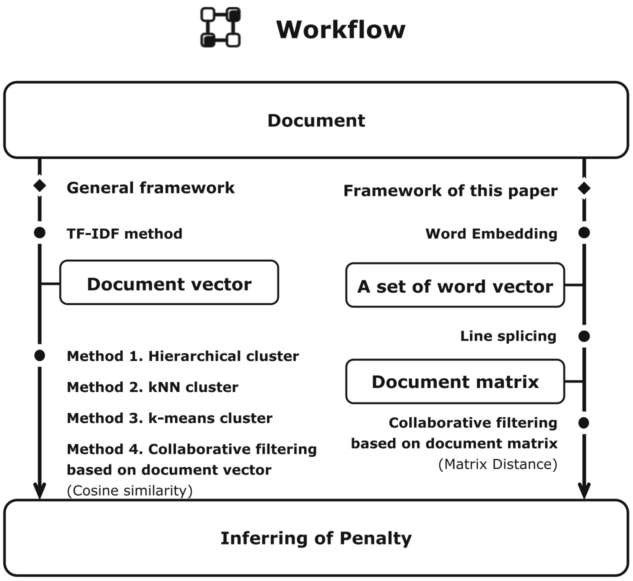

Fig. 1. The proposed framework for penalty inferring

compute the TF-IDF for all the words. That is, after choosing the feature words in the previous step, we should identify the weight of them, as different words vary in importance to legal cases, a numerical value is required to indicate the difference. Normally, words that are important to legal cases shall have a large value, and those less important will have a small value. In this paper, we adopt the TF-IDF to compute the numerical value for the feature words, the TF-IDF not only considers the frequency of words, it also takes into account the influence of the frequency of documents, the formula is defined as follows:

$$w_{ik} = tf_{ik} * \log \frac{N}{df_i}$$

in which, tf_{ik} is the frequency of feature word in legal case k, N is the number of legal cases, and df_i is the frequency of feature word in all the legal cases. In this way, we fulfill the process of constructing vector space for all the legal cases.

Based on the results of TF-IDF, clustering methods such as hierarchical clustering, kNN and k-means are conducted to cluster all the legal cases, and then the clusters are used to infer the penalty of target legal case, through a voting strategy.

The technique details are as follows. Hierarchical Clustering is a simple while useful clustering algorithm [23,26]. In detail, there are two main types of Hierarchical Clustering which are top-down approach and bottom-up approach. We adopt the bottom-up approach in this paper. Given a set of vectors $T = \{t_1, t_2, \ldots, t_n\}$, t_i denotes a certain vector. At first, each word is placed in a single cluster, so the initial set of clusters is

$$C = \{c_1 = \{t_1\}, c_2 = \{t_2\}, \ldots, c_n = \{t_n\}\}$$

in each iteration, two nearest clusters are picked out and aggregated together, using some distance measurements. In this paper, the distance between clusters is computed based on vectors.

For the kNN based method, the recommendation goes as follows. We build the KNN classifier by identifying k value, and the number of most similar classes to be considered in the training dataset, its process involves measuring the similarity based on defining the distance function, in our case, we adopt the Euclidean distance as our distance function. Once we get the resulting clusters, for the test case, we find its cluster, and then a voting strategy is applied to select an amount category, through which the recommendation is realized.

The K-means based recommendation is most similar to the kNN based method. Via a clustering algorithm, data are grouped by the notion of "closeness" or "similarity". In K-means, we measure the closeness using Euclidean distance. Similarly, we get the resulting clusters, then for the test case, we find the cluster it belongs to, and then a voting strategy is applied to select an amount category, which is the recommendation process.

While for the collaborative filtering recommendation based on document vector, the technique details are as follows. Collaborative filtering (CF) has been widely used in business situations. CF methods consist of User-Based CF, Item-Based CF and other variations. The main idea of CF is that similar items may share similar preferences. A higher similarity means they are much more similar. Given legal case list $U = \{u_1, u_2, \ldots, u_n\}$ and feature list $\{i_1, i_2, \ldots, i_m\}$, a case u can be represented by its feature vector $r_u = (r_{u,1}, r_{u,2}, \ldots, r_{u,m})$. The similarity between case u and v can be measured by the distance between r_u and r_v, using Cosine Similarity. After all the similarities are calculated, for the target test case, the most of k similar cases are used to predict the final amount of penalty of it, then just as other methods, a voting strategy is applied to generate the final category of penalty.

Word embedding is a method of using neural networks to calculate the degree of association of adjacent words. Compared with other methods, it not only considers word frequency, but also considers the context of the article. Therefore, when using this method, the relative distance of the article will be closer to reality.

For the right part of Fig. 1. The recommendation goes like this, the first stage is the same as the left part, i.e., segmentation, remove the stop words, and then the labelling. Afterwards, we adopt the iterative process of word embedding, through which a set of word vector is generated. Then by the line splicing, a matrix is composed for each document (legal case). In the next step, the collaborative filtering based on document matrix is used to generate the final recommendation list, the main difference is that it is based on the matrix distance, in this paper we use the word travel cost approach [13] to compute the distances between matrixes. And afterwards, for the target test case, the most of k similar cases are used to predict the final amount of penalty of it, then just as other methods, a voting strategy is applied to generate the final category of penalty.

4 Experiments

4.1 Experimental Settings

Datasets. We obtained the legal case dataset by crawler, which is encrypted. Each legal case is originally a textual description, which means it is not directly usable as input for our evaluation. Also, the explicit penalty amount is hidden from the law case, we can only get the category that amount falls into. Specifically, there are 8 categories of penalty amount, which are (0–1,000], (1,000–2,000], (2,000–3,000], (3,000–4,000], (4,000–5,000], (5,000–10,000], (10,000–500,000] and (500,000–*max*]. Then, through a filtering process, that is, we filter out those legal cases. A threshold of 30 feature words is adopted to remove law cases with no sufficient topic words. In our measurement, we only adopt the first 7 categories, and the number of each category is 500, which means we obtained $3,500$ legal cases in total. In this way, we get a legal case dataset that is applicable for our comparison, as well as other comparative methods, i.e., hierarchical clustering, kNN, k-means and traditional collaborative filtering that is based on the feature vector.

Comparative Approaches. In this part, we present a set of comparative approaches for the evaluation of our proposed framework, in detail, we want to exploit the ability of the word embedding based method in inferring the penalty or the amount of a fine. We mainly compare its effectiveness with the following methods:

- *Hierarchical Clustering.* The clustering method based on hierarchical clustering.
- *kNN.* The clustering method based on kNN.
- *K-means.* The clustering method based on K-means.
- *Vector based CF.* This is the traditional feature vector based collaborative filtering method.
- *Martrix based CF.* This is the word embedding based collaborative filtering method proposed in this paper.

Evaluation Methods. To make an overall evaluation of the performance of our proposed PTM model, we first design the following setting, i.e., penalty inferring with PTM.

In the primary setting, we adopt 5/6 of the ground-truth dataset as the training set, and the rest 1/6 as the testing set. Furthermore, we apply different dividing strategies as to see the influence of ratios. For the testing, we just mark-off the class of penalty amount of testing law cases, and then the inferred class of amount is calculated through all the comparative methods.

For the ease of description, we define some notations as follows. Al is the number of test cases that are correctly clustered into the specific original cluster, Bl is the number of test cases that are incorrectly assigned to this certain cluster, and Cl is the number of test cases that are not assigned to their original cluster of certain cluster, while k is the number of clusters. And in this way, we define the *Precision* and *Recall* in the following way.

$$Precision = \frac{\sum \frac{Al}{Al+Bl}}{k} \tag{1}$$

$$Recall = \frac{\sum \frac{Al}{Al+Cl}}{k} \tag{2}$$

We expect both *Precision* and *Recall* to be good. However, they usually conflict with each other, improving one is usually at the expense of the other. Thus, F_1 measure is introduced to combine *Precision* and *Recall*. F_1 measure is calculated as follows:

$$F_1 = \frac{2 \times Precision \times Recall}{Precison + Recall} \tag{3}$$

4.2 Experimental Results

Table 1 shows the result of matrix based collaborative filtering method, we mainly compared between different ratios, i.e., training set V.S testing set as the $2:1$, $3:1$, $4:1$, $5:1$, and choose K nearest legal cases for the voting process, which K we set as 5, 10, 15, 20, 25.

From the result, we can infer that, with the increasing of training set, the effect of recommendation increases. Also, with more neighbors voting for the final penalty, the result tends to be more accurate.

We also compared with other methods mentioned in the previous section, the result is illustrated in Fig. 2.

Table 1. Matrix based collaborative filtering results

		5	10	15	20	25
2:1	*Precision*	0.2857	0.3091	0.3564	0.2219	0.3061
	Recall	0.2087	0.2027	0.2415	0.2743	0.3039
	F1	0.2412	0.2449	0.2879	0.2453	**0.3049**
3:1	*Precision*	0.3615	0.3768	0.3688	0.3355	0.3365
	Recall	0.1977	0.2506	0.2928	0.2608	0.2720
	F1	0.2556	0.3010	0.3264	0.2934	**0.3008**
4:1	*Precision*	0.4158	0.2441	0.3648	0.3254	0.4680
	Recall	0.2057	0.3000	0.2940	0.3175	0.2800
	F1	**0.2752**	**0.2619**	**0.3256**	**0.3214**	**0.3504**
5:1	*Precision*	0.4089	0.2793	0.4150	0.2784	0.4654
	Recall	0.2057	0.3047	0.3000	0.3422	0.3928
	F1	0.2738	0.2915	0.3482	0.3070	**0.4260**

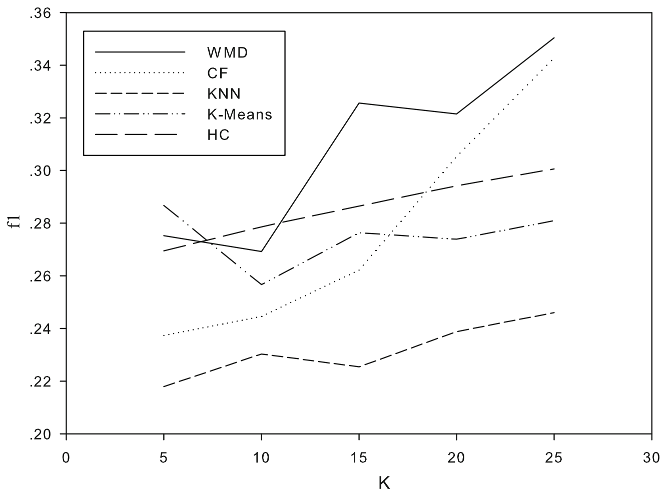

Fig. 2. Results of comparative methods in F1

From Fig. 2, we can see the proposed word embedding method outperforms all the comparative methods, in which it adopts the WMD distance measure, i.e., the Word Mover Distance. And, of all the methods, KNN works the worst.

5 Conclusions

In this paper, we introduced the penalty recommendation techniques for the judicial study, which provides efficient utility as to help judges on decision of the final penalty or amount of the fine. Specifically, we developed a word embedding based collaborative filtering method to generate recommendations for penalty. We conduct extensive experiments to evaluate the performance of the proposed framework on a real legal case dataset. The experimental results demonstrated the superiority of the proposed framework.

References

1. Boyer, M., Lewis, T.R., Liu, W.L.: Setting standards for credible compliance and law enforcement. Can. J. Econ./Rev. Can. D'économique **33**(2), 319–340 (2000)
2. Chee, S.H.S., Han, J., Wang, K.: RecTree: an efficient collaborative filtering method. In: Kambayashi, Y., Winiwarter, W., Arikawa, M. (eds.) DaWaK 2001. LNCS, vol. 2114, pp. 141–151. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-44801-2_15
3. Chowdhury, G.G.: Introduction to Modern Information Retrieval. Facet Publishing, London (2010)
4. Collobert, R., Weston, J.: A unified architecture for natural language processing: deep neural networks with multitask learning. In: Proceedings of the 25th International Conference on Machine Learning, pp. 160–167. ACM (2008)

5. Čubranić, D.: Automatic bug triage using text categorization. In: SEKE (2004)
6. Daughety, A.F., Reinganum, J.F.: Keeping society in the dark: on the admissibility of pretrial negotiations as evidence in court. RAND J. Econ. 203–221 (1995)
7. Earnhart, D.: Enforcement of environmental protection laws under communism and democracy. J. Law Econ. **40**(2), 377–402 (1997)
8. Guha, S., Rastogi, R., Shim, K.: ROCK: a robust clustering algorithm for categorical attributes. Inf. Syst. **25**(5), 345–366 (2000)
9. Guo, G., Wang, H., Bell, D., Bi, Y., Greer, K.: KNN model-based approach in classification. In: Meersman, R., Tari, Z., Schmidt, D.C. (eds.) OTM 2003. LNCS, vol. 2888, pp. 986–996. Springer, Heidelberg (2003). https://doi.org/10.1007/978-3-540-39964-3_62
10. He, T., Chen, Z., Liu, J., Zhou, X., Du, X., Wang, W.: An empirical study on user-topic rating based collaborative filtering methods. World Wide Web **20**(4), 815–829 (2017)
11. He, T., Yin, H., Chen, Z., Zhou, X., Sadiq, S., Luo, B.: A spatial-temporal topic model for the semantic annotation of POIs in LBSNs. ACM Trans. Intell. Syst. Technol. (TIST) **8**(1), 12 (2016)
12. Kilgour, D.M., Fang, L., Hipel, K.W.: Game-theoretic analyses of enforcement of environmental laws and regulations. JAWRA J. Am. Water Resour. Assoc. **28**(1), 141–153 (1992)
13. Kusner, M., Sun, Y., Kolkin, N., Weinberger, K.: From word embeddings to document distances. In: International Conference on Machine Learning, pp. 957–966 (2015)
14. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781 (2013)
15. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Advances in Neural Information Processing Systems, pp. 3111–3119 (2013)
16. Milani, B.A., Navimipour, N.J.: A systematic literature review of the data replication techniques in the cloud environments. Big Data Research (2017)
17. Mnih, A., Hinton, G.E.: A scalable hierarchical distributed language model. In: Advances in Neural Information Processing Systems, pp. 1081–1088 (2009)
18. P'ng, I.P.: Strategic behavior in suit, settlement, and trial. Bell J. Econ. 539–550 (1983)
19. Polinsky, A.M., Shavell, S.: Punitive damages: an economic analysis. Harv. Law Rev. **111**, 869–962 (1998)
20. Ramos, J., et al.: Using TF-IDF to determine word relevance in document queries. In: Proceedings of the First Instructional Conference on Machine Learning, vol. 242, pp. 133–142 (2003)
21. Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., Riedl, J.: GroupLens: an open architecture for collaborative filtering of netnews. In: Proceedings of the 1994 ACM Conference on Computer Supported Cooperative Work, pp. 175–186. ACM (1994)
22. Salton, G., Wong, A., Yang, C.S.: A vector space model for automatic indexing. Commun. ACM **18**(11), 613–620 (1975)
23. Shepitsen, A., Gemmell, J., Mobasher, B., Burke, R.: Personalized recommendation in social tagging systems using hierarchical clustering. In: Proceedings of the 2008 ACM Conference on Recommender Systems, pp. 259–266. ACM (2008)
24. Turian, J., Ratinov, L., Bengio, Y.: Word representations: a simple and general method for semi-supervised learning. In: Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, pp. 384–394. Association for Computational Linguistics (2010)

25. Virpioja, S.: BIRCH: balanced iterative reducing and clustering using hierarchies (2008)
26. Wang, W., Chen, Z., Liu, J., Qi, Q., Zhao, Z.: User-based collaborative filtering on cross domain by tag transfer learning. In: Proceedings of the 1st International Workshop on Cross Domain Knowledge Discovery in Web and Social Network Mining, pp. 10–17. ACM (2012)
27. Wilkin, G.A., Huang, X.: K-means clustering algorithms: implementation and comparison. In: 2007 Second International Multi-Symposiums on Computer and Computational Sciences. IMSCCS 2007, pp. 133–136. IEEE (2007)
28. Yin, H., Wang, W., Wang, H., Chen, L., Zhou, X.: Spatial-aware hierarchical collaborative deep learning for POI recommendation. IEEE Trans. Knowl. Data Eng. **29**(11), 2537–2551 (2017)
29. Zhou, L., Zhang, D.: NLPIR: a theoretical framework for applying natural language processing to information retrieval. J. Assoc. Inf. Sci. Technol. **54**(2), 115–123 (2003)