Unsupervised Keyphrase Extraction

Subjects: Computer Science, Information Systems

Contributor: Chengyu Sun , Liang Hu , Shuai Li , Tuohang Li , Hongtu Li , Ling Chi

A review of unsupervised keyphrase extraction methods.

keyphrase extraction unsupervised method feature selection

1. Introduction

Keyphrase extraction is divided into supervised methods and unsupervised methods based on the training set. The difference between them is whether there is a labeled training set in the learning process. Among them, the supervised method[1] transforms the keyphrase extraction task into a classification problem[2][3] or regression problem[4]. It trains the model on the labeled training set and uses the trained model to determine whether a candidate word in a text is a keyphrase. For example, KEA (Automatic keyphrase extraction)[2] determines whether a candidate word is a keyphrase by calculating the TF-IDF (Term Frequency–Inverse Document Frequency)[5] value of each candidate word and the location where it first appears in the text and inputs these two values into the Naive Bayes classifier. Generally, the supervised method is superior to the unsupervised method[6]. However, compared with the past, the explosive growth of all kinds of information makes the types and quantity of information increase significantly, and the supervised method requires many labeled training sets, thus it require large amounts of manual labor[7]. Moreover, there are no labeled datasets that can serve as references in many fields, especially in some languages that are not well known by human beings, such as the translation tasks of hieroglyphs and cuneiform characters, which makes unsupervised methods without human intervention essential.

Based on the features of the unsupervised keyphrase extraction methods selected by researchers, unsupervised methods can be divided into the statistics-based method, graph-based method, topic-based method, language model-based method, and these methods can be classified into two schools: the linguistic school and the statistical school. The first school mainly extracts keyphrases by analyzing texts using linguistic methods, among which the most common method is to analyze the topic distribution of articles, such as KeyCluster[8] and CommunityCluster[9]. The statistical school mainly analyzes an article's probability features such as KP-Miner[10]and YAKE[11] based on TF-IDF, TextRank[12], or SingleRank[13]. The linguistic school and statistical school have been influencing and promoting each other. As time has passed, researchers have proposed new methods to cross-utilize the two schools' knowledge, such as TopicRank based on clustering (linguistic school) and graphs (statistical school).

2. Classification of Unsupervised Keyphrase Extraction Methods

The mainstream unsupervised keyphrase extraction methods are divided into four categories: statistics-based method, graph-based method, topic-based method, and language model-based method. The methods covered in this article are summarized in Figure 1.

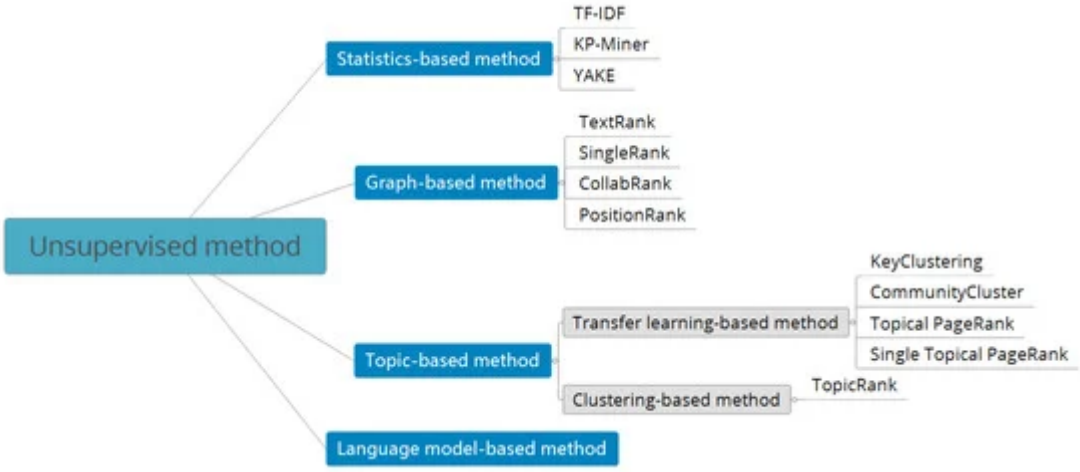

Figure 1. Summary of unsupervised methods.

The mainstream unsupervised methods usually preprocess documents when performing keyphrase extraction task. Because the keyphrases are the lexical words (nouns, adjectives, verbs, and adverbs), deleting other words except lexical words in the document is necessary. Since some words have different forms but have similar meanings (such as play and playing), they are restored to their root forms. Finally, the remaining words are treated as candidate keyphrases, where the real keyphrases are extracted. The unsupervised method described below does not introduce this step.

3. Unsupervised Keyphrase Extraction Methods

3.1. Statistics-Based Methods

- **TF-IDF:**

The TF-IDF is a common baseline method in the keyphrase extraction field, in which the Term Frequency (TF) represents the frequency of a word in a document. To prevent the frequency of words in a long document from being too high, the TF usually uses the document length to normalize the value, that is, $TF = \frac{TN}{DL}$ in which TN represents how many times the word T appears in a specific document D, and DL represents the length of document D. The Inverse Document Frequency (IDF) represents how many documents the word T has appeared in. The main idea of the TF-IDF is that, when the frequency of T in a document is very high (that is, TF is very large) while other documents containing T are very few (that is, IDF is huge), it indicates that T has a good ability to

distinguish keyphrases. Among them, $IDF = log(\frac{DN}{DC+1})$where DN represents the total number of documents and DC represents the number of documents containing the word T.

- **KP-miner:**

The TF-IDF is generally only used as a statistical method applied by other unsupervised keyphrase extraction methods to calculate keyphrases' importance. For example, El-Beltagy and Rafea proposed the KP-miner[10] in 2009. This method is a typical unsupervised keyphrase extraction method using the TF-IDF, divided into three steps. The first step is to select the candidate words from documents, the second step is to calculate the candidate words' score, and the third step is to select the candidate word with the highest score as the final keyphrase. KP-miner introduced two new statistical features in the candidate word selection stage. (i) The least allowable seen frequency (lasf) factor means that only words that appear more than n times in a document can be regarded as candidate words. (ii) CutOff is based on the fact that, if a word appears after a given threshold position in a long document, it will not be a keyphrase, which means the word appearing after CutOff will be filtered out. Finally, the final keyphrases are selected by combining the candidate words' positions and the TF-IDF score. Experiments show that the efficiency of the algorithm is higher than Extractor[14] and KEA.

- **YAKE:**

Campos et al. proposed YAKE[11] in 2018, as a typical unsupervised keyphrase extraction method using the TF-IDF. The difference between YAKE and KP-miner is that it uses candidate word locations or TF-IDF information and introduces a new feature set, which contains five features. The Word Casing (*WC*) reflects the cases of the candidate words. The Word Position (*WP*) reflects the position of a word, which means the more often the word is in the front of the document, the greater its value. The Word Frequency (*WF*) reflects that the higher is the frequency of a word in a document, the greater is its value. The Word Relatedness to Context (*WRC*) indicates the number of different words appearing on both sides of a candidate word. The Word DifSentence (*WD*) indicates the frequency of a candidate word in different sentences. The five values are combined to calculate $S(\omega)$, as shown in the following formula.

$$S(w) = \frac{WR*WP}{WC+\frac{WF}{WRC}+\frac{WD}{WR}} \quad (1)$$

Finally, the final $S(k\omega)$ of each candidate word is calculated by using the 3-gram model.

$$S\left(kw\right)=\frac{\prod_{w\in kw}S(w)}{TF(kw)*\left(1+\sum_{w\in kw}S(w)\right)} \quad (2)$$

Where kw represents the candidate word and *TF* represents the frequency of the keyphrase. The smaller is $S(k\omega)$, the more likely kw is to be a keyphrase.

3.2. Graph-Based Methods

The keyphrase extraction task is transformed into a graph sorting problem using a graph-based algorithm based on the basic assumption that more connections mean more important candidate words. The idea originated from PageRank[15] of Google with a basic idea of voting or recommendations, which means the graph's edges are considered votes. The more votes a node gets, the higher its score, and the more critical it is. Specifically, PageRank generates a directed graph containing all pages with a single page as a node. If there is a link pointing to B in web page A, node A in the graph has an edge pointing to B, regarded as A "voting" for B. The more votes a node receives the higher its score is and the higher its web page ranking. Moreover, the voting of high score nodes will contribute higher scores to the voted nodes[16]. Combining PageRank and word embedding[17], the performance on Chinese and English datasets exceeds TF-IDF and PositionRank.

- **TextRank:**

Based on the idea of PageRank, Mihalcea and Tarau proposed TextRank[12] in 2004, which is the first algorithm to use PageRank for keyphrase extraction. The first thing TextRank does for a document is to delete the function words in the document. Only certain words with fixed parts of speech (such as adjectives and names) can be candidate words. The algorithm then links the selected candidate words according to the co-occurrence relationships between words to generate a directed and powerless graph. The initial score of each node is 1. If two words are within the window of w (w takes a random value from 2 to 20), the two words are connected by lines in the graph. Next, PageRank is run to calculate each node's final score, where the score of node VK is determined by the Formula (1). Finally, the document's consecutive candidate words will be connected into multi-word keyphrases, where the score is the sum of the scores of each candidate word, and the top-ranked candidate words are taken as the keyphrases.

$$S\left(Vk\right)=\left(1-\alpha\right)+\alpha\sum_{m\in NB(Vk)}\frac{1}{|NB(Vm)|}S\left(Vm\right) \quad (3)$$

To prevent TextRank from encountering a dead cycle in the recursive computations, it introduces a damping factor (α). *NB* (vi) represents the neighboring node set of node vi.

- **SingleRank:**

In view of the fact that the graphs constructed by TextRank are unweighted graphs and the weights of the edges can reflect the strength of the semantic relationship between the two nodes, using the weighted graph may be better in the keyphrase extraction task. Based on this assumption, Wan and Xiao proposed SingleRank[13] in 2008, which added weights on the basis of the TextRank between nodes appearing in the window of w at the same time, and the weight value was determined by the number of times the two words appeared in the window of w at the same time. The final score of nodes is determined by Formula (2), where *C(Vj,Vm)* represents the number of times that node *Vj* and node *Vm* appear together in a document.

$$S(Vk) = (1-\alpha) + \alpha \sum_{m \in NB(Vk)} \frac{C(Vj, Vm)}{\sum_{Vk \in NB(Vm)} C(Vm, Vj)} S(Vm) \quad (4)$$

- **ExpandRank:**

In 2008, based on SingleRank, Wan and Xiao proposed ExpandRank[13], which takes the neighboring documents in the same dataset into account to provide the background knowledge when extracting keyphrases from a specific document. Specifically, ExpandRank first uses vectors to represent the documents in the dataset. Next, it calculates the k neighboring documents similar to the extracted document d0 to form a k + 1 document set D. Then, it builds a global graph to assist in extracting the keyphrases by using D, where the edge weight WE (Vk, Vm) between nodes Vk and Vm in the global graph is determined by Formula (3). Sim (d0, di) represents the similarity of documents d0 and di and Fdi (Vk, Vm) represents the number of times that nodes Vk and Vm appear in document di at the same time. The efficiency of ExpandRank is not significantly better than that of SingleRank.

$$WE(Vk,\ Vm) = \sum_{di \in D} sim(d0,\ di) \cdot Fdi(Vk,\ Vm) \quad (5)$$

- **PositionRank:**

Florescu et al. proposed PositionRank[18] in 2017, which introduces location information based on SingleRank according to the idea that the earlier the candidate words appear in a document, the more important they are. As shown in Formula (4), where each item in vector *P* represents the normalized location information of a candidate word, the final score of each candidate word can be calculated by bringing the location information of each node into Formulas (5) and (6), where *pk* is the *k*th element in *P*, that is, the ratio of the position of the *k*th candidate word to the sum of positions of all candidate words; *w* is the weight of the edge; and *adj(v)* is the adjacent node of *v*.

$$P = \left[\frac{p1}{p1 + p2 + \ldots + pn}, \frac{p2}{p1 + p2 + \ldots + pn}, \cdots, \frac{pn}{p1 + p2 + \ldots + pn} \right] \quad (6)$$

$$S(Vk) = (1 - \alpha)\, pk + a \cdot \sum_{vm \in adj(Vk)} \frac{Wmk}{O(Vm)} S(Vm) \quad (7)$$

$$O(Vm) = \sum_{vi \in adj(Vm)} Wmi \quad (8)$$

Graph-based algorithms have some disadvantages. As far as multi-topics documents (such as news) are concerned, human language habits determine that a new topic will have corresponding new keyphrases. However, in graph-based methods, all candidate words (node) are uniformly sorted, and the node with the highest score is taken as the keyphrase. This does not completely guarantee that the keyphrases output by the algorithm can cover all topics, and it may cause the phenomenon that all the keyphrases describe the same topic [7][8][7], which is improved by topic-based methods.

3.3. Topic-Based Methods

Topic-based methods can be further divided into transfer learning-based methods and clustering-based methods.

3.3.1. Transfer Learning-Based Methods

Applying the knowledge acquired from one problem to another different but related problem is the primary motivation of transfer learning[19]. Common knowledge in keyphrase extraction includes Wikipedia[20] and citation networks[21]. Because some background knowledge is needed to classify candidate words in topic-based methods, transfer learning is widely used. The following introduces several mainstream transfer learning-based methods.

- **KeyCluster:**

Applying the knowledge acquired from one problem to another different but related problem is the primary motivation of transfer learning[19]. Common knowledge in keyphrase extraction includes Wikipedia[20] and citation networks[21]. In 2009, Liu et al. proposed KeyCluster[8], divided into four steps. As with other methods, the first step is to preprocess the document, delete the function words, and use the remaining words as candidate words. The second step is to use the Wikipedia-based method to calculate the semantic relationships of candidate words. The Wikipedia-based method regards each word as a vector with each item being the TF-IDF value in Wikipedia. The correlation between the two words can be measured by comparing the vector representations of the two words. The third step is to group the candidate words based on these semantic relationships and find each group's exemplar. The fourth step is to extract the final keyphrases from the exemplar. The experimental results show that the performance of KeyCluster is better than TextRank, and the extracted keyphrases cover the whole document.

- **CommunityCluster:**

In 2009, Grineva et al. proposed CommunityCluster[9] based on the assumption that the words related to the same topic are generally aggregated into a subgraph (or community), and the most connected subgraph generally corresponds to a theme of a document. CommunityCluster uses Girvan–Newman network analysis to detect communities and uses all words in the most closely connected communities as keyphrases. According to the experimental results, CommunityCluster is superior to the baseline system, such as TF-IDF, Yahoo!, and Wikify![22], in precision and recall.

- **Topical PageRank (TPR):**

In 2010, Liu et al. proposed TPR[7], which uses Wikipedia articles as resources to train the potential Dirichlet Distribution (LDA) and uses the trained LDA model to calculate the topic distribution of documents. Then, it uses PageRank for each topic, as shown in Formula (15), to calculate the topic-specific importance scores. Finally, it combines these scores to calculate the candidate words' total score and selects the top-ranked word as keyphrases. TPR, similar to KeyCluster, ensures that the extracted keyphrases cover the entire document. According to the experimental results, TPR is better than the baseline methods, such as TF-IDF and PageRank, in precision, recall, F-score, Bpref, MRR, and MR.

$$St\left(Vk\right)=\alpha \sum_{m:Vm\to Vk} \frac{Wmk}{O(Vm)} St\left(Vm\right)+\left(1-\alpha\right) pt\left(Vk\right) \quad (9)$$

Here, *t* represents a topic and *pt* represents the LDA distribution of *t*.

It is worth mentioning that the introduction of LDA makes each topic have different weights when using TPR, and topics with low weights may not output related keyphrases, which is more in line with human language habits. For example, when we write an article on natural language processing, we may use 20% of the content to describe human language habits, 70% of the content to write about how a computer deals with human language, and 10%

to write other things, and this 10% may not be needed to extract keyphrases, which is a feature that KeyCluster does not have.

- **Single Topical PageRank:**

Because the TPR needs to run PageRank once for each topic, its running efficiency is reduced. Based on this weakness of TPR, Sterckx et al. improved TPR in 2015 and proposed Single Topical PageRank (Single TPR)[23]. Single TPR only needs to run PageRank once for a document, which significantly improves the running efficiency on the premise of accuracy, especially when dealing with large datasets.

The topic-based method includes the use of transfer learning and the use of hierarchical aggregative clustering to complete the keyphrase extraction task.

3.3.2. Clustering-Based Methods

- **TopicRank:**

In 2013, Bougouin et al. proposed topicRank[5] that is similar to TextRank using candidate words as graph nodes, as TopicRank uses topics as graph nodes. Specifically, TopicRank first uses hierarchical agglomerative clustering[20] to divide the document into multiple topics, uses PageRank to score each topic, then selects the first candidate word from each top-ranked topic, and finally uses all the selected candidate words as the keyphrases. According to the experimental results, the method makes the extracted keyphrases cover all topics, and the performance is better than TF-IDF, SingleRank, and TextRank in precision, recall, and F-score.

3.4. Language Model-Based Methods

Based on Kullback–Leibler (KL) divergence that can measure the loss of two language models, Tomokiya et al. used two kinds of datasets with different functions, foreground corpus and background corpus, to assist in keyphrase extraction[24]. The foreground corpus is the dataset for keyphrase extraction, while the background corpus provides background knowledge. Similar to TF-IDF, this method reflects each keyphrase's unique extent by using background knowledge and introduces two new features, namely phraseness and informativeness. Phraseness represents the extent to which a word sequence can be used as a phrase, while informativeness represents the extent to which the phrase can express a document's central idea. This method uses the n-gram model to learn these two features in the foreground corpus and the background corpus. The phraseness and informativeness determine the final scores of the candidate words.

In the above three types of methods (statistics-based methods, graph-based methods, and topic-based methods), each method often contains more than one idea. For example, TPR uses two ideas of topic and graph. This connection is described in detail in Figure 2.

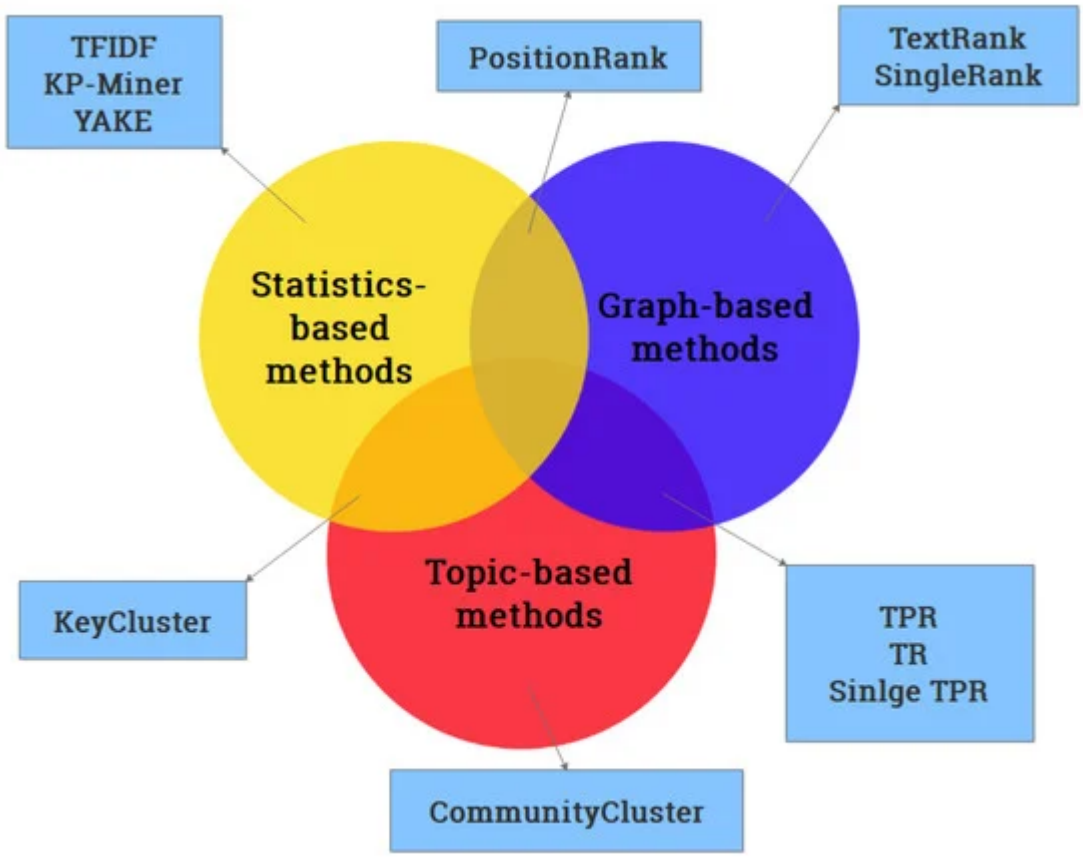

Figure 2. All methods are classified according to the technology applied. The overlapping part represents that the method uses multiple technologies.

References

1. keyphrase extraction;unsupervised method;feature selection
2. format change
3. Wang, R.; Wang, G. Web Text Categorization Based on Statistical Merging Algorithm in Big Data. Environ. Int. J. Ambient. Comput. Intell. 2019, 10, 17–32.
4. Gutierrez, C.E.; Alsharif, M.R.; Khosravy, M.; Yamashita, K.; Miyagi, H.; Villa, R. Main Large Data Set Features Detection by a Linear Predictor Model. Available online: https://aip.scitation.org/doi/abs/10.1063/1.4897836.
5. Salton, G.; Buckley, C. Term-weighting approaches in automatic text retrieval. Inf. Process. Manag. 1988, 24, 513–523.
6. Kim, S.N.; Kan, M.Y. Re-examining automatic keyphrase extraction approaches in scientific articles. In Proceedings of the Workshop on Multiword Expressions Identification, Interpretation,

Disambiguation and Applications—MWE'09, Singapore, 25–27 November 2009; p. 9.

7. Liu, Z.; Huang, W.; Heng, Y.; Sun, M. Automatic Keyphrase Extraction via Topic Decomposition. Conference on Empirical Methods in Natural Language Processing; Association for Computational Linguistics: Singapore, 2010.
8. Liu, Z.; Li, P.; Zheng, Y.; Sun, M. Clustering to find exemplar terms for keyphrase extraction. In Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing—EMNLP'09, Singapore, 6–7 August 2009; Association for Computational Linguistics: Singapore, 2009; Volume 1, p. 257.
9. Hasan, K.S.; Ng, V. Automatic Keyphrase Extraction: A Survey of the State of the Art. In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, Baltimore, MD, USA, 23–25 June 2014; Association for Computational Linguistics: Baltimore, MD, USA, 2014; pp. 1262–1273.
10. El-Beltagy, S.R.; Rafea, A. KP-Miner: A keyphrase extraction system for English and Arabic documents. Inf. Syst. 2009, 34, 132–144.
11. Campos, R.; Mangaravite, V.; Pasquali, A.; Jorge, A.M.; Nunes, C.; Jatowt, A. YAKE! Collection-Independent Automatic Keyword Extractor. In Advances in Information Retrieval; Pasi, G., Piwowarski, B., Azzopardi, L., Hanbury, A., Eds.; Lecture Notes in Computer Science; Springer International Publishing: Cham, Switzerland, 2018; Volume 10772, pp. 806–810. ISBN 978-3-319-76940-0.
12. Mihalcea, R.; Tarau, P. TextRank: Bringing Order into Texts. In Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2004), Barcelona, Spain, 25–26 July 2004.
13. Wan, X.; Xiao, J. CollabRank: Towards a collaborative approach to single-document keyphrase extraction. In Proceedings of the 22nd International Conference on Computational Linguistics—COLING'08, Manchester, UK, 18–22 August 2008; Association for Computational Linguistics: Manchester, UK, 2008; Volume 1, pp. 969–976.
14. Peter D. Turney; Learning Algorithms for Keyphrase Extraction. *Information Retrieval Journal* **2000**, *2*, 303-336, 10.1023/a:1009976227802.
15. Page, L.; Brin, S.; Motwani, R.; Winograd, T. The PageRank Citation Ranking: Bringing Order to the Web. Available online: http://ilpubs.stanford.edu:8090/422/
16. HongBin Wang; Jingzhen Ye; Zhengtao Yu; Jian Wang; Cunli Mao; Unsupervised Keyword Extraction Methods Based on a Word Graph Network. *International Journal of Ambient Computing and Intelligence* **2020**, *11*, 68-79, 10.4018/ijaci.2020040104.
17. Mikolov, T.; Chen, K.; Corrado, G.; Dean, J. Efficient Estimation of Word Representations in Vector Space. Available online: https://ui.adsabs.harvard.edu/abs/2013arXiv1301.3781M/abstract

18. Florescu, C.; Caragea, C. PositionRank: An Unsupervised Approach to Keyphrase Extraction from Scholarly Documents. In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, Vancouver, BC, Canada, 30 July–4 August 2017; pp. 1105–1115.
19. Sinno Jialin Pan; Qiang Yang; A Survey on Transfer Learning. *Design of LDV: a multilevel secure relational database management system* **2009**, *22*, 1345-1359, 10.1109/tkde.2009.191.
20. Medelyan, O.; Frank, E.; Witten, I.H. Human-competitive tagging using automatic keyphrase extraction. In Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing—EMNLP'09, Singapore, 6–7 August 2009; Association for Computational Linguistics: Singapore, 2009; Volume 3, p. 1318.
21. Gollapalli; Sujatha, D.; Cornelia, C. Extracting Keyphrases from Research Papers Using Citation Networks. In Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence, Québec City, QC, Canada, 27–31 July 2014; pp. 1629–1635.
22. Mihalcea, R.; Csomai, A. Wikify!: Linking documents to encyclopedic knowledge. In Proceedings of the Sixteenth ACM Conference on Conference on information and Knowledge Management—CIKM'07, Lisbon, Portugal, 6–10 November 2007; p. 233.
23. Sterckx, L.; Demeester, T.; Deleu, J.; Develder, C. Topical word importance for fast keyphrase extraction. Presented at the 24th International Conference on World Wide Web Companion, WWW 2015, Florence, Italy, 18–22 May 2015; pp. 121–122.
24. Tomokiyo, T.; Hurst, M. A language model approach to keyphrase extraction. In Proceedings of the ACL 2003 Workshop on Multiword Expressions Analysis, Acquisition and Treatment, Sapporo, Japan, 12 July 2003; Volume 18, pp. 33–40.