UNIVERSITY OF DELHI

Undergraduate Thesis

Reconstructing Dependency Trees for Unsupervised Keyphrase Extraction

Meenal Jhajharia

Supervisor:	Prof. Shobha Bagai
Department:	Cluster Innovation Centre
Program:	Bachelor of Technology
Specialization:	IT and Mathematical Innovation
Course:	Semester Long Project

April 2021

Acknowledgement

The time I spent working on this project was extremely fruitful and a worthwhile learning experience. In my desire to write this report, I have in no way any claim to come out with a perfect piece of work. These few details lead me to realize that, like all human endeavours, this report is not perfect and may contain errors and shortcomings. Thus, I remain open to all criticisms and suggestions which could present me with new sources of inspiration as I develop in my ability to research and learn. This report would not have been possible without the contribution and collaboration of others. My sincere gratitude:

- To Prof. Shobha Bagai for guiding me and lending her valuable advice.
- To Harshit Joshi, my collabarator who made notable contributions.

To all of you, I extend my deepest gratitude.

Declaration of Authorship

I, Meenal Jhajharia, declare that this thesis titled, 'Reconstructing Dependency Trees for Unsupervised Keyphrase Extraction' and the work presented in it are my own. I confirm that:

- This work was done while in candidature for a degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, has been clearly stated.
- I have clearly attributed references of published work of others.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- If present, I have clearly mentioned contributions made by others.

Signed:
Meenal Jhajharia

Abstract

Meenal Jhajharia

Extracting Keyphrases from a multitude of documents can be an exhausting and challenging task. While recent advances in Natural Language Processing have made strides in Automatic Keyphrase Extraction, a critical limitation of existing algorithms is that they ignore the inherent hierarchical syntactic relations across words. In view of the tree-like structure of syntactic graphs, we reconstruct dependency trees in Hyperbolic space to locate key information in a document. To this end, we propose ReTree, an unsupervised, graph-based algorithm for keyphrase extraction that exploits syntactic relations using dependency parsing, augmented with local text attributes. To our knowledge, this is the first work that use hyperbolic geometry for keyphrase extraction. We outperform state-of-the-art unsupervised methods on three widely used datasets and expect these findings would help researchers to further study graphs in hyperbolic spaces for the task of keyphrase extraction.

Keywords: Graph-Based algorithm, Keyword Extraction, Natural Language Procesing, Unsupervised, Hyperbolic, Metric Tree, Dependency Parsing

Contents

List of Figures

1 Introduction

Compared to supervised methods, unsupervised key extraction has several advantages. In most cases, supervised key extraction necessitates the existence of a (large) annotated corpus of both documents and their manually selected key terms on which they are trained. Beyond the specific domain represented by the training corpus, supervised methods perform poorly - a significant issue given that the domain of new documents might not be identified at all. Unsupervised key extraction approaches these context-constrained cases by focusing on in-corpus statistical information (e.g., the reciprocal document frequency of the words) and only extracting information about the text in hand. Key extraction is conceived as a ranking task in unsupervised practices, with graph-based ranking techniques considered state-of-the-art [1]. Such graph-based methods use each target document to create a word graph of nodes mapped to words and edges corresponding to relations between words. The top-ranked phrases are returned as keywords using graph centrality measures such as PageRank [2]. Many graph-based extensions have been proposed since their inception to model various concepts.

Given a set of documents, we want to retrieve ones with content most accurately matching the user's specific and unique interest, not only including terms he/she often uses than other people in the domain. For this purpose, we have to extract terms representing the author's point in the current document. Using a dependency tree leverages the contextual information inside syntactic properties in an ordinary text(frequency, position, word similarity). In our model, the construction of the initial graph through dependency parsing takes into account the entire document and its semantic context (instead of certain Part-of-speech tags or certain words based on position, frequency, stopword elimination). Dependency parsing is a naturally occurring hierarchical structure present in the text. Additionally, it incorporates the semantic effects of stopwords as well (i.e., words related with of, are not the same as words related with of). Hence, the initial text graph construction is more valuable and contains more semantic information than all the unsupervised State-of-the-art approaches. We use the dependency tree as the segmentation process for words, which identifies word meaning beyond the crude measure of co-occurrence. While parsing, the model tries to interpret the meaning of various word relationships based on the sentence flow. We define the dependency tree as the fundamental concept of natural language from the Stanford NLP parser. Dependency trees are created for each sentence, which are later merged. A few things are kept in mind during the combination of Dependency Trees - a word might have different POS-tag classification resulting in multiple nodes labeled with the same word. We merge each tree into a

graph by prioritizing the co-occurrence pair of nodes and keep the same words as a part of identification. In this way, we can capture the significance of both terms and phrases. This connects various word contexts, examines the interconnectedness of sentences depending on a word context. As a limitation, it might distort the dependency structure to a certain degree.

Our method is based on the notable progress of Hyperbolic embeddings in Natural Language Processing. Multiple standard datasets such as Wordnet or real-time data such as social networks depict a hierarchal structure in data [3]. It is present in both syntactical or abstract forms. For such hierarchal data, hyperbolic spaces have recently been advocated as alternatives to regular Euclidean spaces when learning representations of certain datasets to represent the hierarchical structure[4]. Hyperbolic spaces are Non-Euclidean geometric spaces that inherently depict hierarchal relations; these can denote continuous versions of trees [5]. Hyperbolic embedding of datasets like Wordnet showed improved reconstruction error and link prediction[4]. Finally, for hyperbolic metric tree construction, we use treerep. The intuition behind hyperbolic spaces is that it works for hierarchies. Based on its value in hypernymy detection and hierarchal analogies[6], which show that dependency parsing is a generalized idea of Hearst patterns[7].

This study proposes ReTree, an unsupervised approach for extracting keyphrases from a single text requires only the current document itself instead of an entire associated corpus for training purposes. ReTree is a graph-based method that leverages syntactic relations between words and local text statistical features. Our model outperforms all *statistical* and *graph-based* baselines.

2 Methods

Given a text document $\mathcal{D} = \{w_1, w_2, \cdots, w_m\}$, with m words, we extract a set of keyphrases $\hat{\mathcal{Y}} = \{\hat{y_1}, \hat{y_2}, \cdots, \hat{y_n}\}$. Each keyhrase is a sequence of one or more words, such that $\hat{y_i} = \{w_{s_1}, w_{s_2}, \cdots, w_{s_x}\}$, where $1 \leq x < m$, and each word $w_s \in \mathcal{D}$. In the following, we present **ReTree**:**Re**constructing dependency **Tree** for Keyphrase extraction, an unsupervised graph-based method that leverages syntactic relations between words, and local text statistical features.

2.1 Dependency Graph

Dependency tree representations of sentences allow us to exploit lexico-syntactic relations among words [8] and can be utilized to locate critical information in a doc-

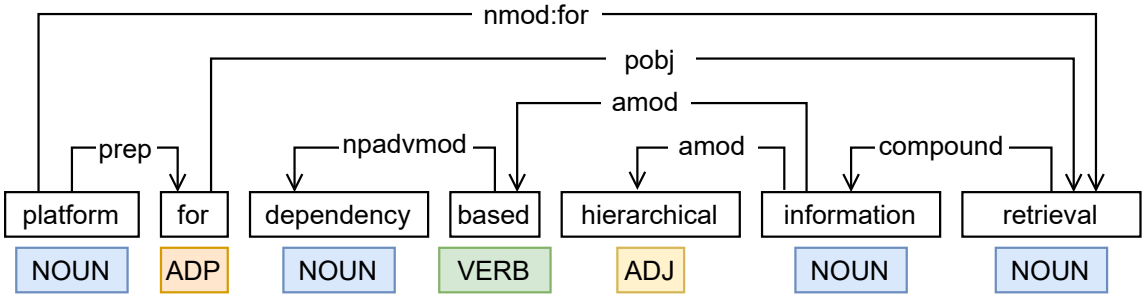

Figure 1: Syntactic relations based on dependency parsing

ument [9, 10]. For instance, in Figure 1, the dependency tree aids in finding relations between two distant words, "*platform*" and "*retrieval*". Furthermore, it emphasizes the relevance of "*retrieval*", as illustrated by the maximum number of connections. Building on the propensity of intra-word syntactical relations, we extract dependency tree representations for each sentence to procure meaning beyond the co-occurrence with other words.

Graph Construction: For a document $\mathcal{D}$, we split the text into sentences and extract dependency relations using Stanford Dependency Parser [11]. In every sentence, we treat words as nodes and each dependency relation as an edge. Thus, forming a sentence level dependency tree $T_i \in T = \{T_1, T_2, \cdots, T_l\}$ for l sentences in a document. Finally, we construct a Dependency Graph $\mathcal{G} = (V, E)$ by combining disjoint trees $\mathcal{T}$ through words occurring in their corresponding sentences, prioritizing the frequently appearing words in a document.[1] Thus, we capture the significance of both terms and phrases. Formally, $\mathcal{G}$ is a directed graph, where each edge $e \in E$ denotes one of the two types of edges between two words $w_x, w_y \in V$: i) Word w_x is syntactically dependent on w_y, ii) Words w_x and w_y are same words occurring in two different sentences. Note that we do not differentiate between theses two types of edges in the graph, as shown in Figure 2. To preclude the possibility of deforming dependency tree structure in the combined graph, we use Gromov's hyperbolicity(δ) to measure resemblance of G to a tree($\delta = 0$ for a tree). The average value of Gromov's hyperbolicity (δ) of the final dependency graph $\mathcal{G}$ across all the datasets used is 0.4. The low value of δ indicates a hierarchical tree-like structure of graph $\mathcal{G}$; for trees $\delta = 0$.

[1]We do not make an edge between stopwords and words smaller than 4 characters.

2.2 Tree Reconstruction

Dependency graphs are inherently hierarchical, tree-like structures [12]. Recent methods utilizing dependency parsing in NLP work on Euclidean Spaces [13, 14, 15]. However, when tree-like structures such as the dependency graph $\mathcal{G}$, are embedded in Low Dimensional Euclidean Spaces they exhibit a highly distorted representation. In contrast, Hyperbolic Space is more suited for hierarchical relationships as distance increases exponentially as we move from parent to child node. Moreover, several hyperbolic embedding algorithms learn an embedding followed by approximating a suitable metric for distance [16]. Building on a recent Tree Reconstruction algorithm, TreeRep [17], we adopt a metric-first approach to obtain a low distortion representation of the Dependency Graph [2]. We chose TreeRep owing to its ability to reliably embed any weighted tree while preserving distance. We map the Dependency graph $G = (V, E, W)$ where $W \in \{0,\ 1\}$, to a metric space (X, d). X is the set of nodes of the Dependency graph and d is the shortest distance metric between these nodes measured along the edges joining them. A δ-hyperbolic metric space adheres to Hyperbolic Geometry for a parameter $\delta > 0$, as explained in §2.1. (X, d) is Gromov(or δ) Hyperbolic if it satisfies the following condition for any $x, y, z, w \in X$:

$$|w - x| + |y - z| \leq max\{|w - y| + |x - z|, |w - z| + |x - y|\} + 2\delta \tag{1}$$

This very idea can be defined in terms of Gromov Product as well. When a metric space (X, d) satisfies the Gromov four point condition [18], for any $x, y, z, w \in X$ it is called δ-hyperbolic. The Gromov product of two points $x, y \in X$ with respect to a third one $w \in X$ is computed as follows:

$$(x, y)_w = \frac{1}{2}(d(x, w) + d(y, w) - d(x, y) \tag{2}$$

Formally, Gromov's four point condition is defined as:

$$(x, y)_w \geq min\{(x, z)_w, (y, z)_w\} - \delta \tag{3}$$

Upon expanding the Gromov Products $(x, y)_w$, $(x, z)_w$, $(y, z)_w$ we observe that for $\delta = 0$, this inequality can be rewritten as:

$$|w - x| + |y - z| \leq max\{|w - y| + |x - z|, |w - z| + |x - y|\} \tag{4}$$

We observe that when $\delta = 0$, Equation 4 satisfies the four point condition, guaranteeing the existence of a unique Metric Tree T that contains the members of X and induces d [19].

Identifying Zones: At $\delta = 0$, the four point condition implies the lack of a unique maximum among the three terms. We create two zones using the possibilities inferred

[2] TreeRep introduces an additive distortion of at most δ

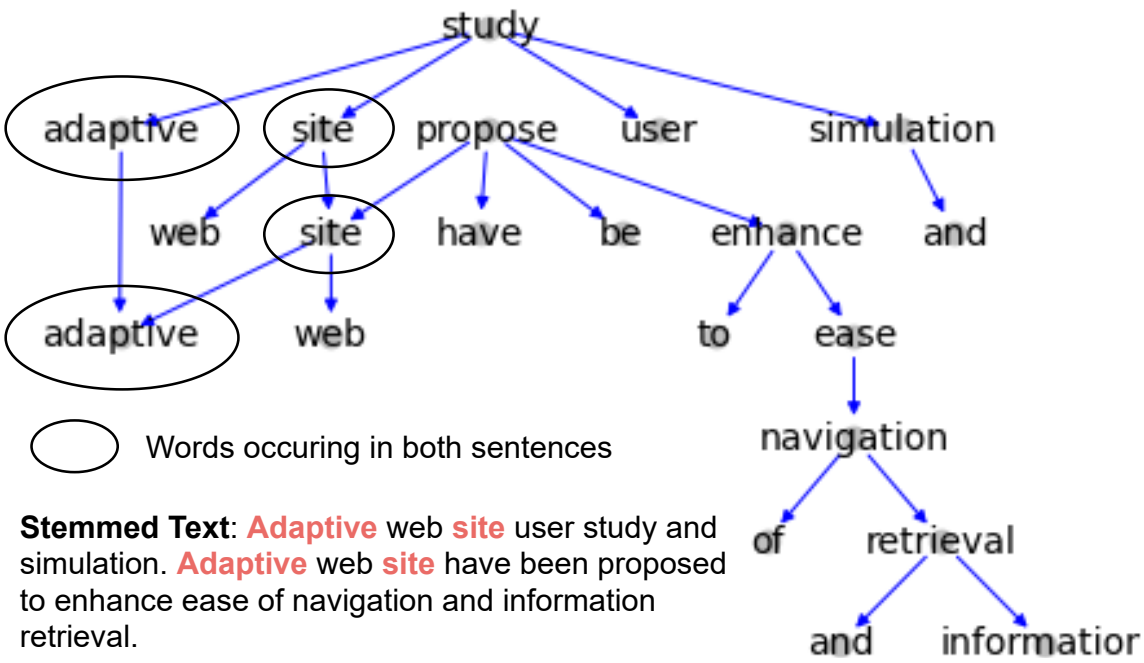

Figure 2: Dependency Graph constructed using a snippet of text. Red denotes words occurring in multiple sentences.

Table 1: Performance Comparison

Models	Type	Single Document	KDD (5.07 keyphrases per document) P	R	F@O	MAP	WWW (5.80 keyphrases per document) P	R	F@O	MAP
Tf-Idf	Statistical	×	13.55*	13.21*	13.33*	12.31*	12.89	12.33	12.49	11.13*
YAKE		✓	9.67	9.33	9.45	8.62	9.9	9.34	9.5	7.85
Text Rank	Graph based	✓	5.6	5.26	5.37	5.71	6.43	5.88	6.03	5.58
Single Rank		✓	9.4	9.06	9.17	7.86	9.66	9.11	9.27	7.36
Position Rank		✓	11.05	10.71	10.82	9.18	11.48	10.93	11.09	8.47
Topic Rank	Topic-Graph based	✓	11.33	10.95	11.08	9.24	12.04	11.47	11.64	8.7
Multipartite Rank		✓	12.71†	12.37†	12.48†	11.27†	12.94*†	12.39*†	12.55*†	10.45†
ReTree	Graph + Statistics	✓	**15.33**	**14.98**	**15.09**	**13.13**	**13.86**	**13.31**	**13.44**	**11.23**

for Equation 1 corresponding to i) *largest two terms are equal*, ii) *all three terms are equal*. **Construction of Universal Tree and Recursive Sorting:** Finally, for any metric d on three points in X and a Steiner node,[3] ReTree constructs a universal tree $\mathcal{T}$ with a metric d_T, such that $d_T = d$. We recursively sort all other points in X into one of the two zones. Thus, computing a Metric Tree (T, d_T) from (X, d), where $\delta = 0$, $d_T = d$ and T has the least possible nodes. The resulting metric tree (T, d_T) encodes crucial notions of hierarchy, such as syntactical dependencies and word-context frequency, such that relevant word in the document $\mathcal{D}$ form $\mathcal{T}$ We call these relevant words obtained from the reconstructed tree $\mathcal{T}$ *cue words*. This framework relies on the hypothesis that keywords are nodes closer to the root node or appearing in multiple contexts an will have a higher degree in the metric tree, thus reducing those nodes' shortest path from other points.

2.3 Candidate Generation and Ranking

Candidate Generation: Cue words extracted from reconstructing a tree from the dependency graph are subject to low coverage since cue words are one-word keyphrases candidates. Hence to increase the coverage, we extract multi-word key phrases following previous works [20, 21, 22, 23] and selecting adjacent nouns with one or more preceding adjectives (`Adjective*Noun+`), up to a length of 5 words. We only consider those phrases in which at least one cue word is present. Additionally, we also use all the cue words as candidates.

Ranking: ReTree utilizes phrase frequency, term frequency, and position in the document to rank the keyphrase candidates. For each keyphrase, ReTree comuptes the following:

i) Phase Score ($\mathbf{P_F}$): Studies show that phrases having a higher count are more relevant to a document [24]. Hence, we calculate the frequency of the entire keyphrase in the document while preserving the order of words in the keyphrase.

(ii) Token Frequency Score ($\mathbf{T_F}$): We sort the frequencies (t_f) for each stemmed word[4] in the keyphrase candidate and calculate the token frequency score (T_F) for a

[3]Steiner nodes are extra nodes added to a tree for Geometrical Optimization

[4]We use the Porter Stemming Algorithm [25](a process for removing the commoner morphological and inflexional endings from words) as implemented in `nltk.org` [26]

keyphrase of n words as:

$$T_F = \begin{cases} \text{frequency}(\lceil \frac{n}{2} \rceil^{\text{th}} \text{ word }), & \text{if } n > 1 \text{ and } n\%2 = 1 \\ \text{frequency}(\frac{n}{2}^{\text{th}} \text{ word }), & \text{if } n > 1 \text{ and } n\%2 = 0 \\ 1, & \text{otherwise} \end{cases} \quad (5)$$

iii) Position Score ($\mathbf{P_S}$): The position of words occurring in a document can indicate their importance in a document [22]. Generally, words appearing in the early parts of text tend to be more relevant keywords [27, 20]. Building on this, we use the first occurrence of each key phrase as their position score.

Finally, each keyphrase is assigned a score (S_p) as the product of phase score (P_F) and token frequency score (T_F), where a higher score signifies higher relevance of the keyphrase:

$$S_p = P_F * T_F \text{ (Higher is more important)} \quad (6)$$

When two or more phrases have the same score (S_p), we assign a higher rank to candidates having lower position score (P_S).

Although simple, this ranking method gives reasonably good results. Note that other ranking algorithms can also be applied.

3 Experiments

3.1 Data

We evaluate our algorithm on three public english datasets:

KDD: [28] Collection of 755 abstracts of papers from the ACM Conference on Knowledge Discovery and Data Mining (KDD), published during 2004-2014. The keywords are assigned by editors.

WWW: [28] Consists editor assigned keyphrases for 1330 abstracts of scientific papers collected between 2004 and 2014, from the World Wide Web Conference (WWW).

500N-KPCrowd: [29] Contains 500 English broadcast news stories distributed over 10 categories (e.g. Politics, Sports, Art). The keyphrases consists of words selected at least by 90% readers.

3.2 Evaluation

For a given text document $\mathcal{D}$ and its corresponding set of ground-truth keyphrases $\mathcal{Y}$, top k extracted keyphrases $\hat{\mathcal{Y}}$ based on their ranks are evaluated. **Metrics:** We follow Yuan et. al. [30] and calculate precision (P), recall (R), Mean Average Precision (MAP) and F_1 score for top O extracted keyphrases $\hat{\mathcal{Y}} = \{\hat{y_1}, \hat{y_2}, \cdots, \hat{y_n}\}$. Where O denotes the number of ground truth keyphrases.

Baselines: We compare ReTree to several competitive baselines, implemented using PKE [31].

Statistical Methods

- TF-IDF [32]: Candidates containing words with the highest TF-IDF weights are assigned as keyphrases. Utilizes other documents in the dataset to calculate IDF.
- YAKE [27]: Statistical method that relies on text features such as word casing, frequency, position, relatedness to context.

Graph-based Methods

- TextRank [2]: Generates a directed, unweighted graph with words as nodes based on their occurrence relationship. Finally, uses PageRank algorithm [33] to score words.
- SingleRank [23]: Like TextRank, weighs the graph by counting the appearance of the two words in a window.
- PositionRank [22]: Proposes a position biased PageRank algorithm, leveraging a word's position and its frequency.
- TopicRank [21]: Utilizes Hierarchical Agglomerative Clustering to generate topics that form nodes of a complete graph.
- MultipartiteRank [20]: Building on TopicRank, constructs a multipartite graph with topics as graph nodes, followed by a weight adjustment mechanism and ranking.

Title: Extracting **temporal signatures** for comprehending **systems biology** models.
Abstract: **Systems biology** has made massive strides in recent years, with capabilities to model complex systems including cell division, stress response, energy metabolism, and signaling pathways. Concomitant with their improved modeling capabilities, however, such biochemical network models have also become notoriously complex for humans to comprehend. We propose network comprehension as a key problem for the KDD community, where the goal is to create explainable representations of complex **biological networks**. We formulate this problem as one of extracting **temporal signatures** from multi-variate time series data, where the signatures are composed of ordinal comparisons between time series components. We show how such signatures can be inferred by formulating the data mining problem as one of **feature selection** in **rank-order space**. We propose five new **feature selection** strategies for **rank-order space** and assess their selective superiorities. Experimental results on budding yeast cell cycle models demonstrate compelling results comparable to human interpretations of the cell cycle
Ground Truth: **biological networks, feature selection, rank-order space, systems biology, temporal signatures**

ReTREE: **systems biology**, **temporal signatures**, **rank-order space**, **feature selection**, time series

Multipartite Rank: complex systems, cell division, capabilities, **temporal signatures**, key problem

TF-IDF: signatures, systems, models, extracting, extracting temporal

Figure 3: ReTree vs other algorithms on given Sample Text

4 Results

4.1 Performance Comparison

Our model outperforms all *statistical* and *graph-based* baselines as presented in Table 1. We note that graph-based models that use Local text features (LF) such as word frequency and positioning, have the worst performance across both datasets. Text Rank and Single Rank suffer especially on small texts as they rank nodes without considering the syntactic context. Syntactic feature(SF) Graph-based models improve performance with additional syntactic features such as Part of Speech tagging. Position Rank uses a biased Text Rank algorithm, which leverages position and frequency. While it outperforms simple node ranking algorithms, it fails to accommodate topical diversity. Topic Rank and its successor MultiPartite Rank are also built on Text Rank, using topic modeling instead of words, thus improving earlier Graph-Based methods. MultiPartite Rank uses a complete multipartite graph, an insightful approach that struggles because of clustering error among similar topics. Although TF-IDF performs well, it is not document-independent as it needs a set of documents to generate keywords. YAKE does not analyze the syntactic context; it is an entirely statistical model based on position, frequency, and co-occurrence. However, KDD and WWW are short documents, making semantic relationships unignorable, leading to YAKE's poor performance. ReTree performs better than purely statistical models and LF Graph-based models, owing to the additional context in the form of lexico-syntactic properties and information about the text's inherent structure in Dependency Parsing. SF Graph-based models suffer through a significant error in Topic Generation and weigh all topics equally. However, keyphrase candidates are of-

ten unevenly distributed among topics. ReTree reconstructs the Dependency Graph with low distortion and preserves syntactical information. In Hyperbolic Geometry, the combined Graph conserves topical (or sentence-based) bias in keyword extraction. Thus ReTree outperforms all the baselines analytically and experimentally. We address crucial drawbacks in these models by leveraging syntactic relations and using metric tree representation to prevent distortion of hierarchal lexico-syntactic dependencies.

4.2 Efficacy of Tree Reconstruction

Additionally, we perform another experiment on the KP500 dataset to evaluate our unigram keywords. This dataset mostly has unigram keywords, so we do not generate multi-word keyphrases. We observe that without a sophisticated ranking algorithm, a simple tree reconstruction outperforms all baseline models significantly. These results encourage further exploration of hyperbolic geometry in keyphrase extraction and NLP in general.

Table 2: Unigram

Models	**n-gram**	**P**	**R**	**F@O**	**MAP**
Single Rank	> 1	29.91	27.22	28.33	11.54
Position Rank	1	32.22*	32.19*	32.2*	23.8*
Topic Rank	1	31.19	31.16	31.17	22.85
Multipartite Rank	> 1	31.94	29.25	30.36	17.1
YAKE	> 1	31.36	29.29	30.4	14.89
ReTree	1	**35.18**	**34.66**	**34.9**	**26.2**

5 Conclusion

This paper introduced a new unsupervised graph-based keyphrase extraction model named ReTree: Reconstructing dependency TREE for Keyphrase Extraction, which leverages syntactic relations between words and local text statistical features. It reconstructs sentence-level dependency trees based on a hyperbolic metric for extracting candidate keyphrases. After all the candidate keyphrases are extracted from the reconstructed metric tree, they are ranked, taking statistical factors into account. At last, top-N keyphrases are selected from the sorted list and returned. Our proposed technique is compared with other prominent unsupervised keyphrase extraction techniques on a uniform experimental setup. The results are acquired for three standard

publicly available datasets. According to the acquired results, ReTree outperforms the rest of the compared techniques in terms of F_1 scores for all considered parameters. Our code and data are available at `https://github.com/almostmeenal/poincare2key`https://github.com/almostmeenal/poincare2key.

References

[1] Kazi Saidul Hasan and Vincent Ng. Automatic keyphrase extraction: A survey of the state of the art. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1262–1273, 2014.

[2] Rada Mihalcea and Paul Tarau. TextRank: Bringing order into text. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 404–411, Barcelona, Spain, July 2004. Association for Computational Linguistics. URL `https://www.aclweb.org/anthology/W04-3252`.

[3] Aaron B Adcock, Blair D Sullivan, and Michael W Mahoney. Tree-like structure in large social and information networks. In *2013 IEEE 13th International Conference on Data Mining*, pages 1–10. IEEE, 2013.

[4] Maximilian Nickel and Douwe Kiela. Poincar\'e embeddings for learning hierarchical representations. *arXiv preprint arXiv:1705.08039*, 2017.

[5] Dmitri Krioukov, Fragkiskos Papadopoulos, Maksim Kitsak, Amin Vahdat, and Marián Boguná. Hyperbolic geometry of complex networks. *Physical Review E*, 82(3):036106, 2010.

[6] Matt Le, Stephen Roller, Laetitia Papaxanthos, Douwe Kiela, and Maximilian Nickel. Inferring concept hierarchies from text corpora via hyperbolic embeddings. *arXiv preprint arXiv:1902.00913*, 2019.

[7] Nigel Franciscus, Xuguang Ren, and Bela Stantic. Dependency graph for short text extraction and summarization. *Journal of Information and Telecommunication*, 3(4):413–429, 2019. doi: 10.1080/24751839.2019.1598771. URL `https://doi.org/10.1080/24751839.2019.1598771`.

[8] Vamshi Ambati. Dependency structure trees in syntax based machine translation. In *Adv. MT Seminar Course Report*, volume 137, 2008.

[9] Mark Stevenson and Mark A Greenwood. Dependency pattern models for information extraction. *Research on Language and Computation*, 7(1):13, 2009.

[10] Haoyu Zhang, Dingkun Long, Guangwei Xu, Pengjun Xie, Fei Huang, and Ji Wang. Keyphrase extraction with dynamic graph convolutional networks and diversified inference. *arXiv preprint arXiv:2010.12828*, 2020.

[11] Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D. Manning. Stanza: A python natural language processing toolkit for many human languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 101–108, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-demos.14. URL `https://www.aclweb.org/anthology/2020.acl-demos.14`.

[12] Marie-Catherine De Marneffe and Christopher D Manning. Stanford typed dependencies manual. Technical report, Technical report, Stanford University, 2008.

[13] Tao Ji, Yuanbin Wu, and Man Lan. Graph-based dependency parsing with graph neural networks. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2475–2485, 2019.

[14] Wenzhe Pei, Tao Ge, and Baobao Chang. An effective neural network model for graph-based dependency parsing. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 313–322, 2015.

[15] Yue Zhang and Stephen Clark. A tale of two parsers: Investigating and combining graph-based and transition-based dependency parsing. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 562–571, 2008.

[16] Frederic Sala, Chris De Sa, Albert Gu, and Christopher Ré. Representation tradeoffs for hyperbolic embeddings. In *International conference on machine learning*, pages 4460–4469. PMLR, 2018.

[17] Rishi Sonthalia and Anna C Gilbert. Tree! i am no tree! i am a low dimensional hyperbolic embedding. *arXiv preprint arXiv:2005.03847*, 2020.

[18] JMS Simoes Pereira. A note on the tree realizability of a distance matrix. *Journal of combinatorial theory*, 6(3):303–310, 1969.

[19] Gusfield Dan. Algorithms on strings, trees and sequences: computer science and computational biology. *Cambridge, UK: Press Syndacate of the Univestity of Cambridge*, 1997.

[20] Florian Boudin. Unsupervised keyphrase extraction with multipartite graphs. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 667–672, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-2105. URL `https://www.aclweb.org/anthology/N18-2105`.

[21] Adrien Bougouin, Florian Boudin, and Béatrice Daille. TopicRank: Graph-based topic ranking for keyphrase extraction. In *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, pages 543–551, Nagoya, Japan, October 2013. Asian Federation of Natural Language Processing. URL `https://www.aclweb.org/anthology/I13-1062`.

[22] Corina Florescu and Cornelia Caragea. PositionRank: An unsupervised approach to keyphrase extraction from scholarly documents. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1105–1115, Vancouver, Canada, July 2017. Association for Computational Linguistics. doi: 10.18653/v1/P17-1102. URL `https://www.aclweb.org/anthology/P17-1102`.

[23] Xiaojun Wan and Jianguo Xiao. Single document keyphrase extraction using neighborhood knowledge. In *AAAI*, volume 8, pages 855–860, 2008.

[24] Ricardo Campos, Vítor Mangaravite, Arian Pasquali, Alípio Mário Jorge, Célia Nunes, and Adam Jatowt. A text feature based automatic keyword extraction method for single documents. In *European conference on information retrieval*, pages 684–691. Springer, 2018.

[25] M. Porter. An algorithm for suffix stripping. *Program*, 40:211–218, 1980.

[26] Steven Bird, E. Klein, and E. Loper. Natural language processing with python. In *Natural Language Processing with Python*, 2009.

[27] Ricardo Campos, Vítor Mangaravite, Arian Pasquali, A. Jorge, C. Nunes, and Adam Jatowt. Yake! keyword extraction from single documents using multiple local features. *Inf. Sci.*, 509:257–289, 2020.

[28] Cornelia Caragea, Florin Adrian Bulgarov, Andreea Godea, and Sujatha Das Gollapalli. Citation-enhanced keyphrase extraction from research papers: A supervised approach. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1435–1446,

Doha, Qatar, October 2014. Association for Computational Linguistics. doi: 10.3115/v1/D14-1150. URL https://www.aclweb.org/anthology/D14-1150.

[29] Luis Marujo, Anatole Gershman, Jaime Carbonell, and Robert Frederking. Jo ao p. neto. 2012. supervised topical key phrase extraction of news stories using crowdsourcing, light filtering and co-reference normalization. In *Proceedings of LREC*, 2012.

[30] Xingdi Yuan, Tong Wang, Rui Meng, Khushboo Thaker, Peter Brusilovsky, Daqing He, and Adam Trischler. One size does not fit all: Generating and evaluating variable number of keyphrases. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7961–7975, 2020.

[31] Florian Boudin. pke: an open source python-based keyphrase extraction toolkit. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: System Demonstrations*, pages 69–73, Osaka, Japan, December 2016. URL http://aclweb.org/anthology/C16-2015.

[32] Karen Sparck Jones. A statistical interpretation of term specificity and its application in retrieval. *Journal of documentation*, 1972.

[33] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford InfoLab, 1999.