AutoName: A Corpus-Based Set Naming Framework

Zhiqi Huang[1], Razieh Rahimi[1], Puxuan Yu[1], Jingbo Shang[2], James Allan[1]
[1]University of Massachusetts Amherst [2]University of California San Diego
{zhiqihuang,rahimi,pxyu,allan}@cs.umass.edu,jshang@ucsd.edu

ABSTRACT

We propose AutoName, an unsupervised framework that extracts a name for a set of query entities from a large-scale text corpus. Entity-set naming is useful in many tasks related to natural language processing and information retrieval such as session-based and conversational information seeking. Previous studies mainly extract set names from knowledge bases which provide highly reliable entity relations, but suffer from limited coverage of entities and set names that represent broad semantic classes. To address these problems, AutoName generates hypernym-anchored candidate phrases via probing a pre-trained language model and the entities' context in documents. Phrases are then clustered to identify ones that describe common concepts among query entities. Finally, AutoName ranks refined phrases based on the co-occurrences of their words with query entities and the conceptual integrity of their respective clusters. We built a new benchmark dataset for this task, consisting of 130 entity sets with name labels. Experimental results show that AutoName generates coherent and meaningful set names and significantly outperforms all baselines.

CCS CONCEPTS

• **Information systems** → **Data mining**; *Information retrieval.*

KEYWORDS

Entity Set Naming; Language Model Probing; Conceptual Clustering

ACM Reference Format:
Zhiqi Huang, Razieh Rahimi, Puxuan Yu, Jingbo Shang and James Allan. 2021. AutoName: A Corpus-Based Set Naming Framework. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '21), July 11–15, 2021, Virtual Event, Canada.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3404835.3463100

1 INTRODUCTION

Entity set naming refers to the task of inferring a name, i.e., a short, multi-word phrase, for a set of semantically-coherent entities. We propose AutoName, an unsupervised framework that extracts a name for a set of query entities from a large-scale text corpus.

Entities play an important role in understanding queries and documents to provide users with precise and relevant information. This is particularly important in session-based [5] and conversational information seeking [27]. Consider an ongoing information-seeking session between a user and search system, where a set of entities have been used in previous queries and responses. Being able to name that set of entities allows the search system to generate clarifying questions or to lead the session based on the underlying concept of entities. In addition, entity set naming benefits other tasks in natural language processing and information retrieval such as automatic captioning of tables [12, 42], topic model labeling [16, 20, 23, 34], or entity set expansion [31, 32, 40].

Previous studies [18, 33] mostly focus on identifying set names from knowledge bases such as Probase [37] and Freebase [3]. The natural incompleteness of knowledge bases makes it impossible to name all sets of entities. In addition, entity sets to be named in practical settings are usually small, which makes names from knowledge bases not ideal as they more often describe broad semantic classes of entities which do not provide the most specific relation between query entities. To address these problems, we explore generating entity-set names from massive text corpora.

The abundance of text corpora provides high coverage of entities, but it comes with a large number of occurrences for many entities. Selecting which contexts of query entities are useful for the set naming task, and which contexts among those are about common properties of all query entities are challenging. This is even more challenging for queries with multi-sense entities.

An ideal set name should describe the most specific concept common to all query entities. We hypothesize that a set name consists of two parts: (1) a hypernym of the entities describing their broad semantic class and (2) one or more modifiers that narrow down the semantic class to the smallest that include the query entities. Accordingly, we first generate general hypernyms via probing a pre-trained neural language model [9] using Hearst patterns [13] and query entities. These hypernyms then become anchors to extract specified candidate names from entities' contexts in large text corpora. The obtained candidates are then clustered to filter out noise and to find out ones that express the same concept across all query entities. Finally, the refined candidate names will be scored based on the integrity score of their respective clusters and their mutual co-occurrence with all query entities. In addition to extracting a name for an entity set, AutoName also justifies the set name with its most similar contextual sentences that have query entities. This allows users to readily judge the accuracy of provided names.

As AutoName is the first model for naming small sets of query entities using a corpus, to the best of our knowledge, we build an evaluation dataset of 130 entity sets with their set names. Entity sets are automatically obtained and sampled to build user queries of lengths 3 to 5. Sampled queries with the name of their respective sets are then judged by human annotators to make sure that the reference name for the larger set of entities is still appropriate

SIGIR '21, July 11–15, 2021, Virtual Event, Canada

ACM ISBN 978-1-4503-8037-9/21/07…$15.00
https://doi.org/10.1145/3404835.3463100

for the queries in terms of specificity and accuracy. The evaluation dataset contains 1,767 queries of entities with their reference names.

We conduct extensive experiments, comparing AutoName against both the state-of-the-art knowledge-base approach [33] and the powerful text generation model T5 [28] that needs training data to be fine-tuned for entity-set naming. The results demonstrate that AutoName outperforms the baselines by 45% in ROUGE [19]. Since automatic evaluation models may not always correlate with human judgments, we asked people to judge the quality of extracted set names by AutoName and baselines. AutoName achieves both the highest average accuracy and Kappa agreements among annotators. Finally, evaluation results show that AutoName, as a successful and completely unsupervised model, can be used to generate a large amount of training data for training of language generation models, such as T5, to generate set names. This is because we demonstrated that 1,125 labeled samples are not sufficient to fully fine-tune T5. The evaluation dataset and implementation of AutoName are available at https://github.com/zhiqihuang/autoname.

2 RELATED WORK

We review related research topics and their differences with entity set naming.

Table captioning is supported by web tables that provide abundant training data. Leveraging table headers and page structures, Hancock et al. [12] train a sequence-to-sequence model to generate titles. When the table contains only a single column of entities, the table captioning task degenerates to entity set naming task. We thus prepare our benchmark dataset by choosing an appropriate subset of the web tables in Wikipedia. Our problem is more challenging because of the unstructured text data with unsupervised setting.

Topic modeling [2] is to discover latent topics in a collection of documents. Treeratpituk and Callan [34] assign labels to topic via hierarchical clustering. Neural model with different structures and loss functions are also designed for this task [11, 15]. In general, topic labeling is focusing on document-level inputs to generate keywords to represent the latent topic. On the other hand, our task focuses on entity-level inputs and extract specific set name from large textual corpus.

Extracting keyphrases that are salient to a document's meaning is an essential step to semantic document understanding [38]. The input for a keyphrase extraction task is usually documents and the goal is to extract or generate phrases which cover the topic of the input documents. Recent approaches based on neural networks have shown promising results [6, 24, 39, 44]. Rather than finding a description of a document's content, the set naming task is to find descriptive concept for a group of semantically coherent entities.

The **entity set expansion** problem identifies entities that belong to the same semantic class given a few seed entities [7, 8, 31, 32, 35, 36, 40, 41]. In set expansion, the user would like to see more semantically similar entities; in set naming, the user has a group of entities and seeks to understand why they can be grouped together.

Conceptual labeling, proposed by Sun et al. [33], a task we believe is the most similar to entity set naming. The method extracts labels from a knowledge graph to summarize a concept towards a bag of words. We use it as a strong baseline to evaluate the performance of our framework.

3 THE AUTONAME FRAMEWORK

Given a set of semantically similar entities as query $Q = \{e_1, e_2, \ldots e_q\}$ where $|Q|$ is small, the goal of set naming is to identify a name for Q from a text corpus. AutoName addresses the problem by extracting and ranking phrases from the contexts of the query entities.

3.1 Candidate Name Generation

A set name typically contains two components: a hypernym for the broad semantic class and one or more modifiers that narrow the semantic class of the hypernym. Therefore, candidate set names for an entity set are generated as a two step process: hypernym generation and then phrase enrichment.

Hypernym generation identifies general hypernyms for the given query entities. We probe a pre-trained LM with query entities to obtain the hypernym component of the set name. The probing queries are constructed based on six Hearst patterns [14] filling the hyponym slots by query entities and leave the hypernym slot as a masked token to predict. For example, the query {*Toronto*, *Ottawa*, *Waterloo*} and pattern "*NP such as NP, NP and NP*", the probing query is "[MASK] *such as Toronto, Ottawa and Waterloo*". The LM predicts a probability distribution over vocabulary for the masked token. To achieve high recall, we collect the top 5 predicted tokens from each Hearst pattern as hypernym candidates. The ranking step (Section 3.3) controls the accuracy of candidates as set names. Given query entities and a set of likely hypernyms, we retrieve all sentences in the corpus that have at least one occurrence of a query entity and a hypernym, providing contexts of query entities. Restricting the query contexts to sentences that contain a hypernym greatly reduces the search space of candidate phrases.

Phrase enrichment. Each hypernym is used to extract a set of candidate noun phrases from the contexts of query entities in the text corpus. First, tokens around the hypernym in obtained contexts are labeled with part-of-speech (POS) tags. We then extract text spans that match the phrase grammar pattern defined on a coarse tag set of adjectives (A), nouns (N), prepositions (P), and determiners (T) as $(A|N)*N(PT*(A|N)*N)*$. We apply a finite-state transducer algorithm [30] on the POS sequence to efficiently extract overlapping and nested spans which can cover different levels of semantic classes for each hypernym. Considering both generality and specificity, we extract phrases of length 2 to 8. For each obtained phrase, we keep its corresponding sentence. Thus, the output is a set of phrase-sentence pairs, denoted $\mathbf{PS} = \{(p, s)_i\}_{i>0}$.

3.2 Density-based Phrase Clustering

Candidate phrase-sentence pairs were generated from contexts of query entities. However, the phrases can refer to different concepts related to query entities where some concepts are not common properties of all query entities. To identify similar concepts of entities, we cluster the candidate phrases using Hierarchical Density-based Spatial Clustering of Applications with Noise (HDBSCAN) [4, 21], chosen primarily because HDBSCAN does not require the number of clusters as a parameter. This is useful as clusters should represent different concepts of query entities, where the number of concepts differs between queries. To represent noun phrases for clustering, we use Sentence-BERT (SBERT) [29]. Mapping candidate phrases to a vector space, we measure phrase similarity using cosine distance.

We cluster the candidate phrases in **PS** using HDBSCAN to find clusters that represent common concepts across all query entities. For this purpose, we say a cluster is valid only if its constituent phrases cover all query entities. We use the parameter k of HDBSCAN that defines how conservative clustering should be [22] to get valid clusters. Starting from $k = 1$, the algorithm generates the largest number of clusters with minimum cluster size. Then, we check the validity of each obtained cluster. If there is no valid cluster, HDBSCAN is re-run with an increased value of k which leads to clusters of larger sizes. We stop the algorithm if either of two conditions is satisfied: (1) there exists at least one valid cluster or (2) k exceeds a pre-defined large value. When the second condition is satisfied, we assume there is no common concept in the corpus for the given query entities and do not continue. After clustering, only phrases in valid clusters are kept for further evaluation.

3.3 Set Name Ranking

We rank refined candidate phrases for set names. Ranking models mostly rely on exact or semantic term matching, but in this task, query entities do not occur in candidate names. Thus, we design a scoring function considering two factors: *cluster* and *phrase* scores.

Cluster score. To measure how well candidates within a cluster represent the same concept, we compute a score for each cluster based on the semantic similarity of sentences from its constituent (p, s) pairs. As we only consider valid clusters, these sentences as a group contain all query entities, and are encoded using SBERT. We denote by s_i^c a sentence in cluster c that contains query entity e_i. For each pair of query entities e_i and e_j, we then compute the cosine similarity matrix Δ_{ij} between all sentence pairs (s_i^c, s_j^c) in the cluster. The score of cluster c is then defined as

$$\text{score}(c) = \frac{1}{\binom{q}{2}} \sum_{1 \le i < j \le q} \max\left(\Delta_{ij}\right).$$

This score reflects the semantic coherence of the candidate phrases in a cluster as the name of the entity set.

Phrase score. To rank the candidate phrases within the cluster, we score each phrase as well. For each term ω in a candidate phrase p, we compute an importance weight based on its co-occurrences with query entities. We only count the co-occurrence of ω and a query entity in a sentence if their shortest dependency path (SDP) is smaller than a pre-defined value. The importance of ω for query entity e_i is calculated by the Mutual Information (MI) as

$$\text{MI}(\omega, e_i) = \sum_{X_i=0,1} \sum_{X_\omega=0,1} P(X_i, X_\omega) \log \frac{P(X_i, X_\omega)}{P(X_i)P(X_\omega)},$$

where X_i and X_ω are binary variables representing whether e_i or ω is present in a sentence. The importance of ω across all query entities are computed as the average of MI, $\mathcal{I}_\omega = \frac{1}{q} \sum_{i=1}^{q} \text{MI}(\omega, e_i)$. A phrase p is then scored based on its constituent terms as

$$\text{score}(p) = \frac{\sum_{\omega \in p} \mathcal{I}_\omega}{(1 + \log_2 L^*)} \tag{1}$$

where L^* is the number of non-stop tokens in p to penalize long phrases but provide stopword tolerance.

Score combination. The final ranking score of phrase p is the sum of its phrase score(p) and its respective cluster score(c).

4 EXPERIMENTS

4.1 Dataset and Experimental Setup

We prepare the **benchmark dataset for entity-set naming** based on the Wikipedia *Lists of lists of lists* page [10]. Following past works [1, 17], we restrict content extraction to *wikitable* class and fetch set entities from the subject column of tables and use the page title as their set names. The obtained collection consists of 130 semantic sets among various domains. From each set, we sample 15 queries, 5 each of length 3, 4, and 5. Sampled queries with the name of their respective sets are then judged by three human annotators to make sure that the reference name for the larger set of entities is still appropriate for the queries in terms of specificity and accuracy. Only queries where all annotators agree about the suitability of the set names are kept. The resulting dataset contains 1,767 queries.

We experiment on two text corpora: the news reports published by Associated Press in 1989 (AP89) and the English Wikipedia data dump from June 2019 (Wiki). For AP89, we use the full dataset with 242,819 documents. For Wiki, we exclude all list-type pages and focus on article pages with structured templates removed. Finally, we have a large text corpus with 981,923 documents.

Experimental setup. For LM probing, we use pre-trained BERT-base-uncased. We set k_{max} to be twice the query length as the stopping criteria for clustering. To find co-occurrences, we consider terms whose distance in the dependency tree is not greater than 3.

Evaluation Metrics. We use BLEU [26] and ROUGE [19] metrics for text generation evaluation. To accommodate paraphrasing, we use ROUGE with word embeddings (ROUGE-WE) [25] and BERTScore [43] to compare semantic similarity between reference and generated set names. We also report human evaluations of generated set names.

We compare AutoName against three **baseline** models. **CLBoW** labels a bag of words using *IsA* and *IsPropertyOf* relationships in Probase [33]. Their algorithm finds label with the minimum description length that cover the query. The **LMProbing** baseline is inspired by hypernym detection using LM probing [45]. The generated hypernym is fed to an auto-regressive LM to predict the next three tokens. The last baseline is **T5Gen** which fine-tunes the generative model T5 [28] for entity-set naming. All sentences including query entities are first scored using Eq.(1). The top 3 sentences are then selected as the T5 input. The generation target is the corresponding set name. Using 75 sets with their 1,125 queries as training data and 55 sets as test data, T5 is fine-tuned for 5 epochs.

We also study two ablations of AutoName. First, **AutoName-T** excludes the conceptual clustering and ranks candidate phrases only using the phrase score. The second ablation **AutoName-C** consider all possible phrases in the query-related documents as candidate names, instead of probing a LM to generate candidates.

4.2 Experimental Results

Overall performance. Table 1 reports the comparison results. AutoName and its ablated variants perform significantly better than baseline methods in terms of BLEU-1, BLEU-2, ROUGE-1, ROUGE-L and ROUGE-WE-1. AutoName improves performance over baseline models by a larger percentage in the Wiki than in the AP89. This behavior is expected because a larger text corpus provides more contextual information for the model to infer semantic classes.

Table 1: Comparison results on AP89 and Wiki dataset. BL-k and RG-k and refer to BLEU-k and ROUGE-k respectively. RGWE-k stands for ROUGE-k with word embeddings. This table is based on queries of length 3. (▲) indicates statistical significance comparing to baseline methods and bold in AutoName indicates statistical significance comparing to its ablations (p = 0.05).

Model	AP89								Wiki							
	BL-1	BL-2	RG-1	RG-2	RG-L	RGWE-1	RGWE-2	BERTScore	BL-1	BL-2	RG-1	RG-2	RG-L	RGWE-1	RGWE-2	BERTScore
LMProbing	0.2102	0.1465	0.3107	0.0372	0.2951	0.7406	0.6385	0.2043	0.2018	0.1542	0.3000	0.0636	0.2966	0.7621	0.6777	0.2127
CLBoW	0.1833	0.1452	0.2980	0.0335	0.2805	0.7396	0.6357	0.3451	0.1761	0.1436	0.2859	0.0532	0.2846	0.7420	0.6602	0.3958
AutoName	0.2818▲	**0.2162**▲	0.3735▲	0.0696▲	0.3606▲	0.7687▲	0.6314	0.2689	**0.4754**▲	**0.3693**▲	**0.5528**▲	**0.2418**▲	**0.5249**▲	0.8285▲	0.7327▲	**0.4474**▲
AutoName-T	0.2740▲	0.2068▲	0.3655▲	0.0633▲	0.3538▲	0.7776▲	0.6324	0.2664	0.4382▲	0.3311▲	0.5313▲	0.2229▲	0.4976▲	0.8339▲	0.7409▲	0.4129▲
AutoName-C	0.2709▲	0.2044▲	0.3613▲	0.0709▲	0.3498▲	0.7742▲	0.6331	0.2631	0.4194▲	0.3147▲	0.5103▲	0.2122▲	0.4789▲	0.8260▲	0.7380▲	0.3879

Table 2: Performance of AutoName and T5 models. (▲) indicates statistical significance difference (p = 0.05).

Model	AP89								Wiki							
	BL-1	BL-2	RG-1	RG-2	RG-L	RGWE-1	RGWE-2	BERTScore	BL-1	BL-2	RG-1	RG-2	RG-L	RGWE-1	RGWE-2	BERTScore
T5Gen	0.1639	0.1348	0.1876	0.0503	0.1876	0.6678	0.5913	0.1764	0.3413	0.2718	0.4280	0.1502	0.4226	0.8039	0.7293	0.3424
AutoName	0.2177▲	0.1602▲	0.3003▲	0.0516	0.2845▲	0.7239▲	0.6141▲	0.1974▲	0.4083▲	0.3078▲	0.4886▲	0.1971▲	0.4713▲	0.8172	0.7301	0.3804▲

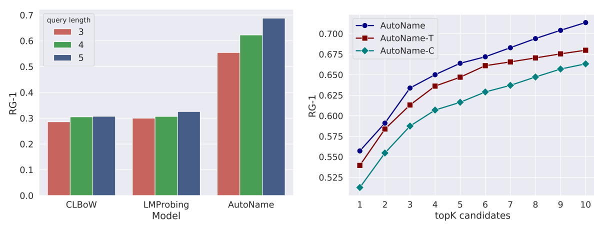
Table 3: Human evaluation of generated set names.

Model	Fleiss' kappa	Average accuracy
CLBoW	0.3604	23%
AutoName	**0.5866**	**53%**
T5Gen	0.4883	48%

LMProbing has nearly the same performance as CLBoW based on a knowledge base. The pre-trained LM tries to recover the masked token with the highest probable token based on its training data, and the Hearst patterns used for probing are lexico-syntactic patterns capturing the hypernym relation between query entities. Therefore, this model successfully generates the hypernym part of most set names, and provides a strong baseline. Comparing with CLBoW, the AutoName framework shows superior performance, especially achieving significant improvements over all metrics on the Wiki dataset. This is because AutoName not only generates hypernyms for query entities, but also narrows the concept by searching for a more fine-grained phrase from the text corpus. In AP89, two variants of our model reach the same performance as the combined approach. Yet, in Wiki, AutoName outperforms its ablated methods. Because of more occurrences and context of query entities, both probing and clustering are necessary to achieve optimal results in a larger and context-rich corpus.

In Table 2, we compare the results of our model to the supervised generative model on the same test set. With a limited amount of data available for fine-tuning, our model performs better than the T5 model. In Table 3, we compare AutoName and baseline methods based on human evaluation of 50 randomly sampled queries over the Wiki corpus. Pairs of target set names and outputs of a set-naming model are shuffled and anonymized for the annotation task where annotators judge which set name is more suitable for the query entities. Judged by 3 annotators, AutoName achieves the best average accuracy and Kappa agreement between annotators.

Performance analysis. We first study the impact of query length on the performance of different models. Figure 1a shows the performance of different models with respect to queries of length 3, 4, and 5 based on the ROUGE-1 metric. AutoName outperforms the baseline models for all query lengths, and shows a larger performance gain with length increment. This is because more input entities provide richer contextual information for the model to infer the semantic classes, thus, leading to better performance.



(a) **Effect of query length** (b) **Evaluation of top-K candidates**

Figure 1: Analysis of query length and top-K candidates.

Since AutoName and its ablations generate a ranked list of candidate phrases, we also consider the performance of top-K candidates. Figure 1b shows the best ROUGE-1 of top-K candidates where K varies from 1 to 10. AutoName shows consistently better performance than its ablations.

5 CONCLUSION

To address the entity set naming task, we propose AutoName, a corpus-based unsupervised framework that considers both the semantic concept of query entities and the structural features of candidate names to consolidate and rank hypernym-based phrases as the desirable set names. We collect a benchmark dataset with 130 entity sets and their set names from Wikipedia list pages. Our experiments show that the names generated by AutoName are of higher quality than all baseline models. In the future, we plan to combine the ranking model with the neural language generative model and improve it to extract multiple names learned from contexts.

ACKNOWLEDGMENTS

This work was supported in part by the Center for Intelligent Information Retrieval and in part by NSF grant number 1813662. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect those of the sponsors.

REFERENCES

[1] Chandra Sekhar Bhagavatula, Thanapon Noraset, and Doug Downey. 2013. Methods for exploring and mining tables on wikipedia. In *Proceedings of the ACM SIGKDD Workshop on Interactive Data Exploration and Analytics*. 18–26.

[2] David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *Journal of machine Learning research* 3, Jan (2003), 993–1022.

[3] Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*. 1247–1250.

[4] Ricardo JGB Campello, Davoud Moulavi, and Jörg Sander. 2013. Density-based clustering based on hierarchical density estimates. In *Pacific-Asia conference on knowledge discovery and data mining*. Springer, 160–172.

[5] Ben Carterette, Paul Clough, Mark Hall, Evangelos Kanoulas, and Mark Sanderson. 2016. Evaluating Retrieval over Sessions: The TREC Session Track 2011-2014. In *Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '16)*. 685–688.

[6] Jun Chen, Xiaoming Zhang, Yu Wu, Zhao Yan, and Zhoujun Li. 2018. Keyphrase generation with correlation constraints. *arXiv preprint arXiv:1808.07185* (2018).

[7] Zhe Chen, Michael Cafarella, and HV Jagadish. 2016. Long-tail vocabulary dictionary extraction from the web. In *Proceedings of the Ninth ACM international conference on Web search and data mining*. ACM, 625–634.

[8] Bhavana Bharat Dalvi, William W Cohen, and Jamie Callan. 2012. Websets: Extracting sets of entities from the web using unsupervised information extraction. In *Proceedings of the fifth ACM international conference on Web search and data mining*. ACM, 243–252.

[9] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).

[10] Wikimedia Foundation. 2020. Wikipedia List page. http://en.wikipedia.org/wiki/List_of_lists_of_lists.

[11] Lin Gui, Jia Leng, Gabriele Pergola, Ruifeng Xu, Yulan He, et al. 2019. Neural Topic Model with Reinforcement Learning. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. 3469–3474.

[12] Braden Hancock, Hongrae Lee, and Cong Yu. 2019. Generating Titles for Web Tables. In *The World Wide Web Conference (WWW '19)*. ACM, New York, NY, USA, 638–647. https://doi.org/10.1145/3308558.3313399

[13] Marti A Hearst. 1992. Automatic acquisition of hyponyms from large text corpora. In *Proceedings of the 14th conference on Computational linguistics-Volume 2*. Association for Computational Linguistics, 539–545.

[14] Marti A. Hearst. 1992. Automatic Acquisition of Hyponyms from Large Text Corpora. In *COLING 1992 Volume 2: The 15th International Conference on Computational Linguistics*. https://www.aclweb.org/anthology/C92-2082

[15] Minghui Huang, Yanghui Rao, Yuwei Liu, Haoran Xie, and Fu Lee Wang. 2018. Siamese Network-Based Supervised Topic Modeling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Brussels, Belgium, 4652–4662. https://doi.org/10.18653/v1/D18-1494

[16] Jey Han Lau, Karl Grieser, David Newman, and Timothy Baldwin. 2011. Automatic labelling of topic models. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*. Association for Computational Linguistics, 1536–1545.

[17] Oliver Lehmberg, Dominique Ritze, Robert Meusel, and Christian Bizer. 2016. A large public corpus of web tables containing time and context metadata. In *Proceedings of the 25th International Conference Companion on World Wide Web*. International World Wide Web Conferences Steering Committee, 75–76.

[18] Jiaqing Liang, Yanghua Xiao, Haixun Wang, Yi Zhang, and Wei Wang. 2017. Probase+: Inferring missing links in conceptual taxonomies. *IEEE Transactions on Knowledge and Data Engineering* 29, 6 (2017), 1281–1295.

[19] Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*. 74–81.

[20] Davide Magatti, Silvia Calegari, Davide Ciucci, and Fabio Stella. 2009. Automatic labeling of topics. In *2009 Ninth International Conference on Intelligent Systems Design and Applications*. IEEE, 1227–1232.

[21] Leland McInnes and John Healy. 2017. Accelerated hierarchical density clustering. *arXiv preprint arXiv:1705.07321* (2017).

[22] Leland McInnes, John Healy, and Steve Astels. 2017. hdbscan: Hierarchical density based clustering. *Journal of Open Source Software* 2, 11 (2017), 205.

[23] Qiaozhu Mei, Xuehua Shen, and ChengXiang Zhai. 2007. Automatic labeling of multinomial topic models. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*. 490–499.

[24] Rui Meng, Sanqiang Zhao, Shuguang Han, Daqing He, Peter Brusilovsky, and Yu Chi. 2017. Deep keyphrase generation. *arXiv preprint arXiv:1704.06879* (2017).

[25] Jun-Ping Ng and Viktoria Abrecht. 2015. Better Summarization Evaluation with Word Embeddings for ROUGE. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Lisbon, Portugal, 1925–1930. https://doi.org/10.18653/v1/D15-1222

[26] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*. 311–318.

[27] Filip Radlinski and Nick Craswell. 2017. A Theoretical Framework for Conversational Search *(CHIIR '17)*. 117–126.

[28] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683* (2019).

[29] Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084* (2019).

[30] Emmanuel Roche and Yves Schabes. 1997. *Finite-state language processing*. MIT press.

[31] Xin Rong, Zhe Chen, Qiaozhu Mei, and Eytan Adar. 2016. Egoset: Exploiting word ego-networks and user-generated ontology for multifaceted set expansion. In *Proceedings of the Ninth ACM international conference on Web search and data mining*. ACM, 645–654.

[32] Jiaming Shen, Zeqiu Wu, Dongming Lei, Jingbo Shang, Xiang Ren, and Jiawei Han. 2017. Setexpan: Corpus-based set expansion via context feature selection and rank ensemble. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 288–304.

[33] Xiangyan Sun, Yanghua Xiao, Haixun Wang, and Wei Wang. 2015. On conceptual labeling of a bag of words. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*.

[34] Pucktada Treeratpituk and Jamie Callan. 2006. Automatically labeling hierarchical clusters. In *Proceedings of the 2006 international conference on Digital government research*. 167–176.

[35] Chi Wang, Kaushik Chakrabarti, Yeye He, Kris Ganjam, Zhimin Chen, and Philip A Bernstein. 2015. Concept expansion using web tables. In *Proceedings of the 24th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 1198–1208.

[36] Richard C Wang, Nico Schlaefer, William W Cohen, and Eric Nyberg. 2008. Automatic set expansion for list question answering. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 947–954.

[37] Wentao Wu, Hongsong Li, Haixun Wang, and Kenny Q Zhu. 2012. Probase: A probabilistic taxonomy for text understanding. In *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data*. 481–492.

[38] Lee Xiong, Chuan Hu, Chenyan Xiong, Daniel Campos, and Arnold Overwijk. 2019. Open Domain Web Keyphrase Extraction Beyond Language Modeling. *arXiv preprint arXiv:1911.02671* (2019).

[39] Hai Ye and Lu Wang. 2018. Semi-Supervised Learning for Neural Keyphrase Generation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Brussels, Belgium, 4142–4153. https://doi.org/10.18653/v1/D18-1447

[40] Puxuan Yu, Zhiqi Huang, Razieh Rahimi, and James Allan. 2019. Corpus-based Set Expansion with Lexical Features and Distributed Representations. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 1153–1156.

[41] Puxuan Yu, Razieh Rahimi, Zhiqi Huang, and James Allan. 2020. Learning to Rank Entities for Set Expansion from Unstructured Data. In *Proceedings of the 2020 ACM SIGIR on International Conference on Theory of Information Retrieval*. 21–28.

[42] Shuo Zhang and Krisztian Balog. 2018. Ad hoc table retrieval using semantic similarity. In *Proceedings of the 2018 World Wide Web Conference*. International World Wide Web Conferences Steering Committee, 1553–1562.

[43] Tianyi Zhang*, Varsha Kishore*, Felix Wu*, Kilian Q. Weinberger, and Yoav Artzi. 2020. BERTScore: Evaluating Text Generation with BERT. In *International Conference on Learning Representations*. https://openreview.net/forum?id=SkeHuCVFDr

[44] Yong Zhang, Yang Fang, and Xiao Weidong. 2017. Deep keyphrase generation with a convolutional sequence to sequence model. In *2017 4th International Conference on Systems and Informatics (ICSAI)*. IEEE, 1477–1485.

[45] Yunyi Zhang, Jiaming Shen, Jingbo Shang, and Jiawei Han. 2020. Empower Entity Set Expansion via Language Model Probing. *arXiv preprint arXiv:2004.13897* (2020).