# Domain Representative Keywords Selection: A Probabilistic Approach

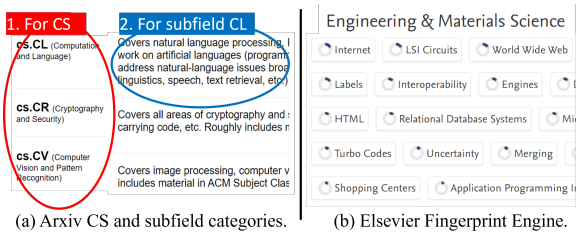
**Pritom Saha Akash**[1] **Jie Huang**[1] **Kevin Chen-Chuan Chang**[1]
**Yunyao Li**[2*] **Lucian Popa**[3] **ChengXiang Zhai**[1]
[1]University of Illinois at Urbana-Champaign, USA
[2]Apple, USA
[3]IBM Research, USA
{pakash2, jeffhj, kcchang, czhai}@illinois.edu
yunyaoli@apple.com, lpopa@us.ibm.com

## Abstract

We propose a probabilistic approach to select a subset of a *target domain representative keywords* from a candidate set, contrasting with a context domain. Such a task is crucial for many downstream tasks in natural language processing. To contrast the target domain and the context domain, we adapt the *two-component mixture model* concept to generate a distribution of candidate keywords. It provides more importance to the *distinctive* keywords of the target domain than common keywords contrasting with the context domain. To support the *representativeness* of the selected keywords towards the target domain, we introduce an *optimization algorithm* for selecting the subset from the generated candidate distribution. We have shown that the optimization algorithm can be efficiently implemented with a near-optimal approximation guarantee. Finally, extensive experiments on multiple domains demonstrate the superiority of our approach over other baselines for the tasks of keyword summary generation and trending keywords selection.[1]

## 1 Introduction

*Domain representative keywords* are the core knowledge of a *target domain* of interest. A target domain can be a broad area of science like *computer science (CS)* or its sub-field *artificial intelligence (AI)*. Acquiring domain representative keywords benefits various natural language processing (NLP) tasks such as information summarization, organization, and extraction. For instance, acquiring a set of domain representative keywords is an important first step in organizing domain knowledge with a taxonomy of keywords (Zhang et al., 2018). Moreover, tagging documents (Chen et al., 2017) with domain representative keywords helps




(a) Arxiv CS and subfield categories. (b) Elsevier Fingerprint Engine.

Figure 1: Screenshots of example applications.

to facilitate search or recommendation in a domain. For another example, summarizing a domain using its trending keywords for a specific time frame helps researchers get a snapshot of research trends or emerging areas of interest, e.g., new emerging security vulnerabilities.

In reality, while representing a domain, the desired keywords often depend on a given *context domain*. E.g., if we are interested in representing the CS domain with the context of general knowledge (all areas of knowledge), keywords like *model*, *data*, *information* make sense in distinguishing CS from general knowledge. However, if the context is general science, those keywords are not *distinctive* enough to distinguish CS from other areas like mathematics or physics. Instead, the keywords *machine learning*, *data mining*, *deep learning* make more sense in this case. Therefore, it is important to contrast with a known context domain while representing a target domain, but most of the existing work ignored this. An application of this is shown in Fig. 1 (a) from the *arxiv category taxonomy*[2]. We can see a shift of categories from general to more specific subcategories of research areas: CS → CL. Knowing CS categories (partially shown in (1)) as the context, the keywords specified in (2) (i.e., *speech*, *text retrieval*) are more appropriate to represent CL than keywords overlapped with other CS categories.

Moreover, the number of keywords that need to

[*] This work was done while the author was at IBM Research, USA.

[1]Code and data are available at https://github.com/pritomsaha/keyword-selection

[2]https://arxiv.org/category_taxonomy



*Findings of the Association for Computational Linguistics: ACL 2022*, pages 679 - 692
May 22-27, 2022 ©2022 Association for Computational Linguistics

be selected depends on the nature of the applications. E.g., users are interested in a quick, high-level overview with fewer keywords while summarizing a particular domain. On the other hand, while building a controlled vocabulary representative of a certain domain, the number of keywords is naturally large. An application of the controlled vocabulary is illustrated in Figure 1 (b). It shows the fingerprint visualization of a CS researcher generated by Elsevier Fingerprint Engine[3], a system for research profiling. The researcher's profile is summarized using keywords in *Engineering & Material Science* domain. However, we can see that some non-representative keywords like *labels* and *merging* are used in the summarization. Therefore, having a representative controlled vocabulary for each domain will facilitate this application for expressively representing a researcher profile.

We thus propose the **problem** of *domain representative keywords selection*. As input, we are given a set of candidate keywords, a target and a context domain represented by their corresponding corpora, and a size $k$. As output, we aim to select a subset consisting of $k$ keywords from the given candidate set such that the subset best represents the target domain contrasting with the context domain. Here, we assume that the candidate keywords are from the target domain and can be implicitly extracted from the given target domain corpus or externally given keywords for that domain.

From the above problem and discussion, we have identified that the solution for the problem needs to meet the following two **requirements**: (1) the selected keywords should be distinctive to the target domain contrastive with a context domain; (2) the selected keywords should represent the target domain as a whole within the specific size constraint. None of the existing work satisfies all of them. Previously, research has been conducted on automatic keyword extraction (Hätty et al., 2017; Meng et al., 2017; Alzaidy et al., 2019; Wang et al., 2020) and phrase mining (Liu et al., 2015; Shang et al., 2018). However, their main focus is to extract terms from single/multiple documents without considering whether the extracted terms are distinctive to a target domain contrastive with a context. There is also some previous research (Liu et al., 2015; Shang et al., 2018; Lu et al., 2019; Huang et al., 2021) that tries to find fine-grained domain-specific keywords from the text. However, these approaches mostly rank keywords based on their specificity to a corpus (or domain) rather than selecting a predefined number of keywords with a global objective of representing the target domain. Therefore, in this work, we propose a solution to satisfy all the specified requirements.

The first **challenge** on fulfilling the requirements is *contrasting the target and context domains*. Among candidate keywords, the distinctive keywords may have similar corpus statistics (i.e., frequency from target domain corpus) with many non-distinctive popular keywords. Therefore, simply filtering out highly frequent keywords may lose many distinctive keywords for a target domain. Instead, it is more intuitive to say that the keywords that frequently appear in both target and context corpora are often not distinctive keywords for the target domain. It inspires us to leverage the two-component mixture model (MM) (Zhai and Lafferty, 2001) concept to generate the candidate keywords distribution contrasting with the context domain. As far as we know, this is the first work to utilize a mixture model mechanism for keywords selection.

The second **challenge** is the *representation under a size constraint*. If we simply select the top distinctive keywords based on the MM-generated distribution, we may end up with redundant keywords that may fall short in representing the target domain as a whole. Hence, it is more intuitive to consider selecting keywords with a domain representation objective. Therefore, we cast this as an optimization problem of selecting $k$ keywords that *coarsen* the candidate distribution adapting the concept of *statistical machine translation* (Brown et al., 1993) with the objective of minimizing the divergence between the initial and coarsened distributions of candidate keywords.

In summary, as our **contributions** in this paper, **firstly**, we propose a new problem formulation named *domain representative keywords selection*. **Secondly**, we propose a framework for solving the problem consisting of two steps: (1) generating candidate keywords distribution using a two-component mixture model mechanism and (2) selecting a subset of keywords utilizing the generated distribution with an introduced optimization algorithm. **Thirdly**, we prove that our proposed optimization problem can be efficiently solved with a near-optimal approximation ratio. **Finally**, to validate the effectiveness of our approaches, we con-

[3]https://www.elsevier.com/solutions/elsevier-fingerprint-engine

duct extensive experiments on multiple domains for different tasks demonstrating the superiority of our framework against strongly designed baselines.

## 2 Related Work

The problem of *domain representative keywords selection* is related to the automatic keyword extraction (AKE) problem. AKE focuses on extracting or generating the most prominent keywords from single/multiple documents. Existing methods for AKE can be classified into two categories: supervised and unsupervised keyword extraction. Early supervised methods consider AKE as a binary classification problem (Witten et al., 1999; Turney, 2000) by learning a classifier from annotated documents to predict whether a candidate phrase is a keyword or not. Recently, *deep learning* has been used for the supervised AKE. E.g., (Meng et al., 2017) uses an encoder-decoder-based framework to generate keywords where (Alzaidy et al., 2019) addresses AKE as a sequence labeling problem. Unsupervised AKE methods mostly apply graph-based ranking mechanisms utilizing semantic relatedness measure between keywords (Mihalcea and Tarau, 2004). Besides, linguistic (Handler et al., 2016) and semantic (Bennani-Smires et al., 2018) approaches have also been used for unsupervised AKE.

However, the main focus of the above studies is to describe single/multiple documents rather than domain-specific keywords extraction. To solve this problem, several researches (Liu et al., 2015; Shang et al., 2018; Lu et al., 2019; Wang et al., 2020; Huang et al., 2021) have been conducted on domain-specific fine-grained keyword extraction. E.g., (Huang et al., 2021) propose an algorithm for measuring the relevance of a keyword in a particular domain. However, this approach requires a user to provide some seed domain-relevant terms for supervising the algorithm. Moreover, the above approaches only consider ranking keywords based on their domain specificity (or relevance). None of them deals with the problem of domain representative keyword selection with a specific size constraint.

The mixture model used for generating keywords distribution in our approach is related to the research on probabilistic topic models (Hofmann, 2001; Blei et al., 2003) and comparative text mining (Sarawagi et al., 2003; Zhai et al., 2004). However, the difference between our approach and these studies is that rather than finding multiple latent topics or themes from a collection or multiple collections of documents, we model a target domain corpus as a distribution of unigram language model contrastive with a context model.

## 3 Proposed Methodology

Our proposed framework consists of two steps: (1) generating distribution for the candidate keywords and (2) selecting a subset that best represents the target domain utilizing the generated distribution.

### 3.1 Keywords Distribution Generation

To select keywords, how do we represent a target domain in contrast with a context one? One naive solution can be the frequency distribution of keywords in the target corpus. However, this distribution is biased towards common but possibly non-distinctive keywords (e.g., *data*, *method* and *model* in CS), which may not differentiate the target (e.g., CS) from the context (e.g., Physics) domain. On the other hand, among candidate keywords, the distinctive keywords may have similar corpus statistics (i.e., frequency from target domain corpus) with many non-distinctive common keywords. Therefore, it is not easy to separate those desired target domain keywords from non-distinctive common keywords using simple statistics calculated from the target domain corpus. E.g., keyword *algorithm* is more distinctive than *method*, but both are popular keywords in CS domain. Therefore, simply filtering out highly frequent keywords may lose many distinctive keywords for a target domain.

To handle the above problem, we regard the target corpus as a mixture of two unigram language models. Specifically, the corpus is assumed to be generated from a mixture of two multinomial component models. One model is the known background model $\theta_B$ (computed from the context corpus), which models the non-distinctive common keywords in the target and context corpora. The other one is the target domain model ($\theta_D$) that needs to be estimated and concerned for prioritizing distinctive keywords in that domain.

Formally, let $\mathcal{C}$ be the target domain corpus from which we are interested to find the keyword distribution, then the log-likelihood value (LLV) of generating $\mathcal{C}$ from this mixture model is

$$\log p(\mathcal{C}|\theta_D) = \sum_{t_i \in V} c(t_i, \mathcal{C}) \log[(1-\lambda)p(t_i|\theta_D) + \lambda p(t_i|\theta_B)], \quad (1)$$

where $V$ is the candidate keywords set and $c(t_i, \mathcal{C})$ is the frequency of keyword $t_i$ in $\mathcal{C}$. $\lambda$ refers to the mixing weight of the $\theta_B$. In other words, $\lambda$ controls the amount of "background noise" in the corpus we want to be modeled by $\theta_B$. We assume $\theta_B$ and $\lambda$ to be known, and $\theta_D$ be estimated. Specifically, $\theta_B$ is the probability distribution calculated from the context domain corpus.

In principle, we can estimate $\theta_D$ using any optimization methods. E.g., the Expectation-Maximization (EM) algorithm (Dempster et al., 1977) is one of them and can be used to compute a maximum likelihood estimate with the following updating formulas:

$$p^{(n)}(z=0|t_i) = \frac{(1-\lambda)p^{(n)}(t_i|\theta_D)}{(1-\lambda)p^{(n)}(t_i|\theta_D) + \lambda p^{(n)}(t_i|\theta_B)},$$

$$p^{(n+1)}(t_i|\theta_D) = \frac{c(t_i, \mathcal{C})p^{(n)}(z=0|t_i)}{\sum_{t_j \in V} c(t_j, \mathcal{C})p^{(n)}(z=0|t_j)},$$

where $p(z=0|t_i)$ refers how likely $t_i$ is from $\theta_D$. The estimated $\{p(t_i|\theta_D) \cdots p(t_N|\theta_D)\}$ is used as candidate keywords distribution.

### 3.2 Keyword Subset Selection

After acquiring a distribution of candidate keywords, we find a subset with a size $k$ to represent the target domain. One possible solution is to select top $k$ keywords based on the candidate distribution ($\theta_D$) generated by the *mixture model* (MM). Hence, the keywords with high distinctiveness to the target domain contrasting with the context domain will be selected. However, one problem with this approach is that the selected keywords may fall short in representing the target domain by only selecting some redundant distinctive keywords.

To solve the above problem, we view the subset selection as a *distribution coarsening* problem. Specifically, we want to use a subset to estimate the candidate distribution (i.e., coarsened distribution). As defined in the previous section, a domain is a distribution of keywords (i.e., candidate distribution). Therefore, for a subset of keywords to represent the domain, the coarsened distribution by the subset should closely approximate the candidate distribution of that domain.

Formally, let $\mathcal{P} = \{p(t_i) \cdots p(t_N)\}$ be the candidate distribution, we compute a coarsened distribution $\tilde{\mathcal{P}} = \{\tilde{p}(t_i) \cdots \tilde{p}(t_N)\}$ by subset $S$ and $\tilde{p}(t_i)$ for each $t_i \in V$ is calculated as:

$$\tilde{p}(t_i) = \sum_{t_j \in S} p(t_i|t_j)p(t_j), \quad (2)$$

where $p(t_i|t_j)$ refers to the probability of *semantically translating* $t_j$ into $t_i$. This idea of estimating the probability of each keyword from candidates by a subset is adapted from the statistical machine translation from the same language used in information retrieval (Berger and Lafferty, 1999).

Now to find the subset, we introduce an optimization problem with objective of selecting a subset ($S$) with size $k$ from candidates ($V$) that minimizes the difference between the LLV of generating $\mathcal{C}$ by $\mathcal{P}$ and $\tilde{\mathcal{P}}$, respectively. We know that the LLV of generating $\mathcal{C}$ by $\mathcal{P}$ is $\log p(\mathcal{C}) = \sum_{i \in V} c(t_i, \mathcal{C}) \log p(t_i)$ where $c(t_i, \mathcal{C})$ is the frequency of $t_i$ in $\mathcal{C}$. Similarly, the LLV of generating $\mathcal{C}$ from $\tilde{\mathcal{P}}$ is $\log \tilde{p}(\mathcal{C})$. Hence, given $|S| = k$, our optimization objective is:

$$\begin{aligned} S &= \arg\min_{S \subseteq V} \| \log p(\mathcal{C}) - \log \tilde{p}(\mathcal{C}) \| \\ &= \arg\min_{S \subseteq V} \| \sum_{t_i \in V} c(t_i, \mathcal{C}) \log \frac{p(t_i)}{\tilde{p}(t_i)} \| \\ &= \arg\min_{S \subseteq V} \| \sum_{t_i \in V} p(t_i) \log \frac{p(t_i)}{\tilde{p}(t_i)} \| \\ &= \arg\min_{S \subseteq V} D_{KL}(\mathcal{P} \| \tilde{\mathcal{P}}) = \arg\min_{S \subseteq V} \phi(S), \end{aligned} \quad (3)$$

where $\phi(S)$ is our objective function, and $D_{KL}(\mathcal{P}\|\tilde{\mathcal{P}})$ is Kullback–Leibler (KL) divergence (Kullback and Leibler, 1951) between $\mathcal{P}$ and $\tilde{\mathcal{P}}$.

From (2), one obvious question is how to calculate $p(t_i|t_j)$. For this, we use *mutual information* (MI) to estimate $p(t_i|t_j)$ inspired from (Karimzadehgan and Zhai, 2010) where MI is used to estimate a similar model for information retrieval. MI is a good measure to judge relatedness between two terms. In our model, for any two terms $t_i$ and $t_j$, we first compute MI ($I(t_i; t_j)$) between them and normalize it into a probability as below:

$$I(t_i; t_j) = \sum_{b_i, b_j} p(b_i, b_j) \log \frac{p(b_i, b_j)}{p(b_i)p(b_j)}, \quad (4)$$

$$p(t_i|t_j) \approx p_{MI}(t_i|t_j) = \frac{I(t_i; t_j)}{\sum_{t'_j \in V} I(t_i; t'_j)}, \quad (5)$$

where $b_i$ is a binary variable indicating the presence/absence of $t_i$. E.g., $p(b_i = 1)$ indicates the ratio of documents containing $t_i$ and $p(b_i = 1, b_j = 1)$ indicates the ratio of documents where both $t_i$ and $t_j$ co-occur. Here, $p_{MI}(t_i|t_j)$ gives us the probability of how $t_j$ relates to $t_i$; intuitively, this probability would be higher when these two terms frequently co-occur in the same document in the target corpus.

**Optimization.** We are interested in finding a subset $S$ with size $k$ from $V$ such that $\phi(S)$ is minimized, i.e., $\arg\min_{S \in V} \phi(S)$ s.t. $|S| = k$. This is referred as the *cardinality-constrained optimization* and proven to be NP-hard (Feige, 1998). However, if the objective function $\phi(S)$ is monotone and submodular, a simple greedy algorithm is guaranteed to obtain an approximation of $1 - \frac{1}{e}$. We call a non-negative real valued function $F$ (to be maximized) *submodular* if it has the property of *diminishing returns* that is $F(X \cup \{v\}) - F(\{v\}) \geq F(Y \cup \{v\}) - F(\{v\})$ for all $v \in V$ and $X \subseteq Y \subseteq V$. Moreover, $F$ is said to be *monotone* if $F(X) \leq F(Y)$ for all $X \subseteq Y$.

**Theorem 1.** *For minimizing the objective function $\phi(\cdot)$, a simple greedy algorithm obtains an approximation guarantee of $1 - \frac{1}{e}$.*

*Proof.* The proof can be found in Appendix A. □

So, as per Theorem 1, we can obtain a near optimal solution using a simple greedy algorithm. Initially, we have $S = \emptyset$, then iteratively update $S = S \cup \arg\max_{t \in V \setminus S} \mathcal{G}(t|S)$ until $|S| = k$ where $\mathcal{G}(t|S) = \phi(S) - \phi(S \cup t)$ is the gain of adding a new term $t$ to $S$. Thanks to the submodularity property of $\phi(\cdot)$, this simple greedy algorithm can further be accelerated by lazy greedy algorithm (Minoux, 1978). More specifically, instead of recomputing $\mathcal{G}(t_i|S), \forall t_i \in V$ in every step, we use a priority queue of sorted gains $g(t_i), \forall t_i \in V$. Starting with $g(t_i) = -\phi(\{t_i\}), \forall t_i \in V$, the algorithm adds a term $t_i$ to $S$ if $g(t_i) \geq \mathcal{G}(t_i|S)$, otherwise we update $g(t_i)$ to $\mathcal{G}(t_i|S)$ and resort the priority queue. This largely improves the efficiency of the algorithm.

## 4 Experiments

This section evaluates our models from different perspectives: (1) the ability to select representative summary keywords for a target domain; (2) the performance for trending keywords selection task in a domain for different time frames.

### 4.1 Experiment Setup

**Datasets.** In our experiments, to test the generality of the proposed approaches, we use two document collections from two domains for constructing target and context corpora for each of the domains. One is abstracts collections from the *arxiv* repository (version 47)[4], and the other is a collection of newsgroup documents[5].

**Candidate Keywords.** In our experiments, we use different sets of candidate keywords. For the CS domain, we collected keywords from two external sources named Springer and Aminer (Tang et al., 2008). The Springer CS keyword list is collected through web scraping from Springer[6] and trimmed to 83K based on frequency $\geq 5$. The Aminer keyword list is the collection of keywords assigned by authors in CS research papers, and there are approximately 50K keywords in this list. Alongside keywords from external sources, we also created candidate sets extracted from concerning corpus using AutoPhrase (Shang et al., 2018) tool. All the candidate keywords are lemmatized, and several filtering rules are used. For instance, keywords containing only letters, numbers, hyphens are used; stop and single-letter words are removed.

**Baselines.** We compare our models with the following four baseline keyword selection algorithms.

- **Relative Frequency (RF):** Since a keyword is likely to be domain representative when it frequently appears in a domain corpus, we consider a simple approach that selects the top $k$ frequent keywords based on the relative frequency calculated from the target corpus.
- **Log-odds (LO):** We adapted a method (Monroe et al., 2008) for keyword selection which was introduced to compare words used by two political parties. Recently, (Hughes et al., 2020) used this method for detecting trending terms in *Cybersecurity* forum discussion. In this baseline, we adapt this method to model keywords as a function of a particular domain or time to compute the likelihood of keywords in that domain or time as log-likelihoods ("log-odds").
- **Page Rank (PR):** This baseline is a graph-based keyword selection method using PageRank (Mihalcea and Tarau, 2004). We build the graph of candidate keywords where each edge weight denotes how closely two keywords are related.

[4] https://www.kaggle.com/Cornell-University/arxiv
[5] http://qwone.com/ jason/20Newsgroups
[6] https://www.springer.com/gp

- **Facility Location (FL) Function:** Facility location function is a representation based subset selection measure (Mirchandani and Francis, 1990) used for finding a representative subset of items. Recently, this measure is used for training-data subset selection (Kaushal et al., 2019). In this paper, we adapt this measure as a baseline for selecting subset from candidate keywords set. Specifically, denoting $rel(t_i, t_j)$ as the relatedness of two keywords $t_i$ and $t_j$, the objective is to select a subset $S \in V$ that maximizes FL function $f(S) = \sum_{t_i \in V} \max_{t_j \in S} rel(t_i, t_j)$.

**Proposed Models.** We have the following three variants of our proposed framework.

- **KL divergence + RF ($\mathbf{KL_{rf}}$):** This model is a simple version of our proposed objective function $D_{KL}(\mathcal{P}\|\tilde{\mathcal{P}})$ defined in (3). In this model, $\mathcal{P}$ is the relative frequency distribution calculated from the target corpus and $\tilde{\mathcal{P}}$ is coarsened distribution defined in (2).
- **Mixture Model (MM):** In this proposed model, keywords are ranked based on the estimated distribution for the target domain contrasting with a context domain using the mixture model defined in Section 3.1. Based on the distribution, the top $k$ keywords are selected.
- **KL Divergence + MM ($\mathbf{KL_{mm}}$):** This proposed model is similar to $\mathbf{KL_{rf}}$. In $\mathbf{KL_{mm}}$, instead of using relative frequency, the mixture model estimated keyword distribution is used as $\mathcal{P}$ in $D_{KL}(\mathcal{P}\|\tilde{\mathcal{P}})$.

**Implementation Details.** There are some parameters both in baselines and the proposed models we have to set. E.g., the mixing weight $\lambda$ for the background model in the mixture model is set to two different values based on the specificity of the target domains. Particularly, when we set $\lambda$ to a small value, the model favors frequent non-informative terms (i.e., domain-specific stop words). Therefore, the larger values are set for $\lambda$. In our experiments, for a broad domain like CS, we set $\lambda$ to 0.9, and for more specific domains (i.e., AI and subtopics in newsgroup), we set $\lambda$ to 0.99. The reason for these two different values of $\lambda$ is that more specific domains demand larger $\lambda$ for selecting distinctive keywords. For optimizing MM, we use Expectation-Maximization (EM) algorithm (Dempster et al., 1977). Since EM does not guarantee the global maxima, in our experiment, we run the algorithm multiple times with random initialization, and the one with the best MLE is chosen to reduce the chance of getting local maxima. As we use mutual information (MI) based on document co-occurrence statistics in our model (defined in (5)), for the fair comparison, in the baseline FL, we also use MI between two keywords $t_i$ and $t_j$ to encode the relatedness between them (i.e., $rel(t_i, t_j)$). Similarly, MI is used for computing edge weight in the PR method.

| $k$ | RF | LO | PR | FL | $\mathbf{KL_{rf}}$ | MM | $\mathbf{KL_{mm}}$ |
|---|---|---|---|---|---|---|---|
| 10 | 1.0651 | 1.1001 | 1.1035 | 1.0722 | 1.0651 | 2.0981 | **2.1212** |
| 20 | 1.1440 | 3.2476 | 2.2682 | 1.1345 | 1.1451 | 4.2626 | **4.2972** |
| 30 | 2.2134 | 4.3965 | 3.3607 | 3.2682 | 3.2875 | **4.4321** | 4.4169 |
| 40 | 3.3273 | 4.4929 | 4.4902 | 3.3515 | 3.3660 | 4.6000 | **4.6018** |
| 50 | 3.4530 | 4.6896 | 4.5734 | 3.4505 | 3.4496 | **5.6826** | **5.6826** |
| 100 | 4.7812 | 8.2382 | 7.0399 | 4.7626 | 4.8761 | 8.2708 | **8.2824** |
| 200 | 9.7166 | 11.1047 | 8.9403 | 9.5908 | 8.7045 | 11.1082 | **12.0233** |
| 500 | 18.902 | 19.1719 | 18.0464 | 16.6171 | 17.7441 | **19.3221** | 19.2353 |

Table 1: Category correspondence results

## 4.2 Experiment Results

### 4.2.1 Summary Keywords Selection

We conduct both quantitative and qualitative studies to evaluate the ability of proposed models to select domain representative summary keywords. For this purpose, we use the abstracts from the arxiv under CS categories as the target corpus. The context corpus is composed of all abstracts in the arxiv repository.

**Quantitative Evaluation.** We create keyword summaries for the CS domain with varying sizes ($k$) for quantitative evaluation. We collected 52 known category keywords from arxiv categories as CS representative ground keywords to evaluate the ability of selected $k$ summary keywords to represent the target domain when $k$ varies. The correspondence between $k$ selected keywords $S = \{t_1 \cdots t_k\}$ and $m$ category keywords $C = \{c_1 \cdots c_m\}$, $CC(S, C)$ is calculated as the summation of the pairwise normalized mutual information (NMI) (Bouma, 2009) between $S$ and $C$ i.e., $CC(S, C) = \sum_{i,j} \frac{I(t_i;c_j)}{H(t_i;c_j)}$ where $I(t_i; c_j)$ is calculated following formula from (4) and $H(t_i; c_j) = -\sum_{b_i,b_j} p(b_i, b_j) \log p(b_i, b_j)$ is the joint entropy of $t_i$ and $c_j$.

From the results on Table 1, using AutoPhrase extracted candidate keywords, we can see that even though no supervision is used, our methods $\mathbf{KL_{mm}}$ and MM select keywords that best correspond with the known categories outperforming all the baselines (similar results from two more candidate sets are shown in Appendix B). We ob-

| Models | Selected 20 keywords in CS |
|---|---|
| RF | paper, model, **algorithm**, datum, result, information, **graph**, state, high, art, single, order, human, research, general, design, **deep learning**, **semantic**, knowledge, **neural network** |
| LO | **algorithm**, art, information, **semantic**, **graph**, human, **deep learning**, paper, datum, **neural network**, **machine learning**, real world, research, video, **robot**, communication, language, **security**, **architecture**, knowledge |
| PR | polynomial, research, channel, paper, energy, **graph**, datum, model, experimental, information, **machine learning**, **software**, binary, english, propose method, function, acoustic, upper, solution, algebraic |
| FL | art, paper, datum, **algorithm**, model, result, high, information, **graph**, channel, research, order, single, human, general, **deep learning**, design, experimental, solution, knowledge |
| $\mathbf{KL_{rf}}$ | model, **algorithm**, paper, datum, state, **graph**, result, information, high, art, human, research, design, single, **semantic**, order, **deep learning**, energy, general, **neural network** |
| MM | **algorithm**, art, **semantic**, **deep learning**, human, **neural network**, **convolutional neural network**, **machine learning**, real world, video, information, **robot**, research, language, communication, **security**, **architecture**, **privacy**, **deep neural network**, **label** |
| $\mathbf{KL_{mm}}$ | **algorithm**, art, **semantic**, **deep learning**, human, **security**, **neural network**, real world, **convolutional neural network**, communication, **machine learning**, **robot**, language, video, research, **privacy**, **label**, information, **software**, **architecture** |

Keywords distinctive to the CS domain are **highlighted** (annotated by authors).

Table 2: Summary keywords in CS Domain

serve that there is a good improvement of result from MM to $\mathbf{KL_{mm}}$. However, this is not true for $\mathbf{KL_{rf}}$ and the RF baseline. The reason is that the relative frequencies from the target corpus favor the non-distinctive common keywords (e.g., *model* and *method*). As described in Section 3.2, $\mathbf{KL_{rf}}$ tries to select the subset of keywords that best estimate the original candidate distribution. Hence, it also favors those common keywords to attain the nearest estimation of the original distribution.

On the other hand, the MM-generated distribution assigns larger probabilities to distinctive keywords of the target domain, contrasting with the context domain. Therefore, selecting a keyword subset by $\mathbf{KL_{mm}}$ with close estimation of the MM generated distribution also favors distinctive keywords with the domain representative objective. Furthermore, one interesting observation is that when $k$ is smaller, the selected keywords by $\mathbf{KL_{mm}}$ tend to summarize the domain better than that of MM. The primary reason for this is that $\mathbf{KL_{mm}}$ prefers to select more non-redundant keywords than MM while $k$ is smaller, which we later discuss from Table 2.

| Models | 2000-2009 | 2010-2019 | 2020-2021 |
|---|---|---|---|
| RF | 0.6289 | 0.6640 | 0.6493 |
| LO | 0.6813 | 0.7199 | 0.7238 |
| PR | 0.6626 | 0.6970 | 0.6826 |
| FL | 0.6172 | 0.6848 | 0.6528 |
| $\mathbf{KL_{rf}}$ | 0.6282 | 0.6792 | 0.6516 |
| MM | **0.6908** | 0.7331 | 0.7898 |
| $\mathbf{KL_{mm}}$ | 0.6898 | **0.7763** | **0.7944** |

Table 3: Results using trending ground truth keywords

| Models | 2000-2009 | 2010-2019 | 2020-2021 |
|---|---|---|---|
| RF | 0.3593 | 0.3195 | 0.323 |
| LO | 0.3956 | 0.3326 | 0.4043 |
| PR | 0.3705 | 0.3211 | 0.3641 |
| FL | 0.3591 | 0.3217 | 0.3336 |
| $\mathbf{KL_{rf}}$ | 0.3583 | 0.3189 | 0.3239 |
| MM | 0.4104 | 0.3468 | 0.5145 |
| $\mathbf{KL_{mm}}$ | **0.4165** | **0.3523** | **0.5215** |

Table 4: Results generated using Google Trends

**Qualitative Evaluation.** For the qualitative evaluation, we show the summary keywords selected by different algorithms in the CS domain from AutoPhrase extracted candidate keywords in Table 2 (simmilar additional results are shown in Appendix C). This study aims to observe the difference between the proposed models and baselines in selecting summary keywords. We can see that our models (MM and $\mathbf{KL_{mm}}$) outperform all the baselines by selecting the most number of CS representative keywords. We also observe that the LO baseline method also selects a comparable amount of distinctive keywords. The reason is its use of a contrastive method like MM for selecting keywords for a particular corpus compared to a context corpus.

However, our models MM and $\mathbf{KL_{mm}}$ tend to select more representative keywords than the LO method. E.g., we can see that our methods select keywords like *privacy*, *software* and *convolutional neural network* instead of keywords that LO selects like *graph* and *paper*, *data*. Another observation is that the keywords selected by PR are mostly those keywords (i.e., *experimental*, *data* and *function*) that have a broad association with other words. However, these keywords as an unit do not convey much information about the domain.

Now to see the difference between our models MM and $\mathbf{KL_{mm}}$, we see the difference between their selected keywords. As stated before, we can

| | 2000-2009 | 2010-2019 | 2020-2021 |
|---|---|---|---|
| RF | paper, problem, **algorithm**, model, method, approach, **system**, information, result, datum, set, application, number, user, word, performance, language, order, time, case | model, method, paper, approach, image, problem, datum, task, algorithm, dataset, performance, network, result, feature, system, training, application, work, number, object | model, method, task, datum, approach, dataset, image, paper, performance, problem, training, algorithm, network, feature, result, system, work, **application**, **deep learning**, experiment |
| LO | **logic program**, rule, manipulator, **genetic algorithm**, workspace, **parallel manipulator**, **logic programming**, document, grammar, stable model, **artificial immune system**, logic, word, web site, **answer set**, global constraint, **machining**, fitness, belief, evolvability | image, dataset, method, feature, task, **convolutional neural network**, object, training, **deep learning**, classification, classifier, **deep neural network**, **neural network**, robot, model, video, **recurrent neural network**, word, **segmentation**, representation | model, dataset, task, training, **transformer**, image, **deep learning**, **neural network**, prediction, label, **federated learning**, learning, method, **machine learning**, **language model**, explanation, experiment, **covid 19**, **reinforcement learning**, feature |
| PR | problem, **algorithm**, paper, user, datum, model, word, method, image, information, approach, **system**, constraint, set, solution, performance, application, document, result, rule | image, algorithm, user, robot, object, network, word, model, dataset, agent, environment, datum, task, video, method, training, **language**, system, policy, **segmentation** | image, robot, model, object, algorithm, dataset, task, agent, environment, user, datum, policy, language, graph, **reinforcement learning**, network, method, video, training, **deep learning** |
| FL | paper, problem, **algorithm**, model, method, approach, **system**, result, information, set, application, datum, number, user, word, order, performance, case, image, time | image, model, method, paper, problem, approach, datum, algorithm, task, network, performance, dataset, result, user, application, work, feature, system, training, number | image, model, method, paper, task, datum, approach, dataset, performance, problem, training, algorithm, work, result, network, experiment, **application**, system, feature, **deep learning** |
| $\mathbf{KL_{mm}}$ | problem, paper, **algorithm**, method, model, **system**, approach, information, datum, word, set, result, user, application, **agent**, number, network, performance, **language**, order | image, model, method, algorithm, datum, paper, task, network, problem, approach, dataset, system, user, feature, performance, training, object, application, result, information | model, method, image, task, datum, dataset, problem, network, approach, paper, training, algorithm, system, performance, feature, object, **application**, user, **deep learning**, result |
| MM | **logic program**, manipulator, **genetic algorithm**, workspace, **parallel manipulator**, **logic programming**, grammar, **stable model**, **artificial immune system**, web site, **answer set**, **global constraint**, **machining**, fitness, **evolvability**, **radial distortion**, **soft constraint**, **nonmonotonic reasoning**, **stable model semantic**, **belief revision** | image, **convolutional neural network**, **recurrent neural network**, classifier, **deep convolutional neural network**, **deep network**, **cnn**, **computer vision**, **lstm**, **deep neural network**, **bayesian network**, **rnn**, **word embedding**, **svm**, **segmentation**, **convolutional network**, **descriptor**, **neural machine translation**, recognition, sentence | **transformer**, training, **federated learning**, **language model**, **covid 19**, **graph neural network**, dataset, explanation, **deep learning**, **pre training**, **adversarial attack**, **fine tuning**, **meta learning**, **deep learning** model, lidar, **self attention**, point cloud, **reinforcement learning**, **bert**, label |
| $\mathbf{KL_{mm}}$ | **logic program**, workspace, **genetic algorithm**, grammar, manipulator, **logic programming**, web site, **global constraint**, **artificial immune system**, **evolvability**, **parallel manipulator**, synonym, **stable model**, **som**, **belief revision**, **unification**, **soft constraint**, **language resource**, fitness, **wordnet** | image, **convolutional neural network**, **recurrent neural network**, classifier, **deep network**, **deep convolutional neural network**, **bayesian network**, **word embedding**, **computer vision**, **descriptor**, **svm**, **crf**, **lstm**, **neural machine translation**, dictionary, **convolutional network**, **deep neural network**, **recognition**, **cnn**, **segmentation** | **transformer**, training, explanation, **language model**, **covid 19**, **federated learning**, **graph neural network**, dataset, **pre training**, lidar, **deep learning**, **adversarial attack**, label, **meta learning**, **knowledge distillation**, **fine tuning**, **deep learning model**, **latent space**, **datum augmentation**, **target domain** |

Keywords representative of its corresponding time frame are **highlighted** (annotated by authors).

Table 5: Keyword summaries (top 20 keywords) of three different time frames in AI domain

see $\mathbf{KL_{mm}}$ prefers non-redundant keywords than MM. E.g, $\mathbf{KL_{mm}}$, instead of selecting *deep neural network* as it already selects keywords like *neural network* and *deep learning*, it selects a different keyword *software* where MM prefers redundant keyword *deep neural network*. Therefore, while the only requirement is to rank keywords based on their distinctiveness for a target domain contrastive with a context domain, MM is more practical to use. On the other hand, if the objective is also selecting diverse representative keywords, $\mathbf{KL_{mm}}$ is preferable. See Appendix D for more qualitative study using newsgroup dataset.

### 4.2.2 Trending Keywords Selection

As an important application of our problem, we evaluate the performance of proposed approaches for *trending keywords selection* in the AI domain. This study conducts quantitative and qualitative evaluations considering three different time frames: 2000-2009, 2010-2019, and 2020-2021. For this purpose, we compose a corpus representative of each of the specified time frames by collecting abstracts from the Arxiv repository under AI-related categories: cs.AI, cs.CL, cs.CV, cs.IR, cs.LG, cs.NE and cs.RO. The entire dataset under all CS categories is used for the context corpus.

**Quantitative Evaluation.** Since there is no ground truth trending keywords available for the AI domain, it is not easy to quantitatively evaluate the selected ones for a specific time. Instead, we have created three ground truth sets by collecting related keywords from topic areas used in the call for papers (CFP) of an AI conference called AAAI[7] over the three specified time frames. However, the topics that appear in the CFP are not necessarily trending topics, and many topics appear throughout all the time frames. For this, we collect only the changing topics from a time frame to another. Further, to expand the ground truth sets, we also add keywords related to the collected topics. E.g., *word embedding* was a popular keyword in NLP during the 2010s, and one related of this is *word2vec*.

Evaluation using Ground Truths. For evaluation, we compute the selected keywords' ability to cover the ground truth keywords using a *representativeness* measure. Formally, similar to (Kaushal et al., 2019), say $s_{ij}$ denotes the similarity between two keywords $t_i$ and $t_j$, $R(S) = \frac{1}{|\mathcal{G}|}\sum_{t_i \in \mathcal{G}} \max_{t_j \in} s_{ij}$ is used as the *representativeness* score of selected keyword set $S$ to represent the ground truth set $\mathcal{G}$. For $s_{ij}$, we compute the cosine similarity between vector representation of $t_i$ and $t_j$. The vector for

[7] https://www.aaai.org/

each keyword is the concatenation of two word-vectors; one is word2vec (300d) (Mikolov et al., 2013) learned from corresponding corpus, and the other is the compositional GloVe embedding (Pennington et al., 2014) (element-wise addition of the pre-trained 300d word embeddings). The reason for using pre-trained word vectors is that many keywords in ground truth sets do not appear in the corresponding corpus, and thus vectors cannot be learned from that corpus. Table 3 shows the detailed results over three time frames. We can see that our proposed model $\mathbf{KL_{mm}}$ outperforms the other methods with large margins followed by MM.

Evaluation using Google Trends. Alongside using ground truths, we also design a quantitative evaluation measure (shown in Table 4) using Google Trends (GT) API[8]. GT[9] awards a score for a term called *interest over time* that expresses the term's popularity over a specified time range. Since GT does not have data before 2004, we have to use data from 2004 till 2009 for the 2000-2009 time frame. As our three specified time frames are not equal, we first take the average of provided interest scores for each keyword in each time frame to make the score comparable across different time frames. Then, we calculate the probability of each term's interest over three specified time frames. Finally, the average of computed probability scores of 50 selected terms is calculated for each method. This score represents the average probability of selected terms to be trending in each time frame. From Table 4, we can see that our method $\mathbf{KL_{mm}}$ achieves the best score over others, followed by comparable results from MM. It indicates that our solutions are more appropriate in finding trending keywords for a specified time frame.

**Qualitative Evaluation.** We qualitatively evaluate the performance of different algorithms by directly comparing their selected keywords in each time frame from Table 5. We can see PR selects keywords that are either CS stop words or the keywords that are not distinctive for a perspective time frame compared to others (similar results by RF, FL, $\mathbf{KL_{rf}}$). Because PR primarily depends on the popularity of a keyword and some keywords always appear frequently in any time frames (e.g., *task*, *dataset*, *model*, *etc* ). Here, the LO again provides comparable results. E.g., similar to our methods MM and $\mathbf{KL_{mm}}$, LO also can select very relevant trending keywords during the 2020s like *covid 19*. However, while selecting trending keywords, the LO also tends to select many domain-specific stop words overlapped over different time frames (e.g., *method, task, model*). As discussed before, the reason is that LO does not have the objective of representing the target domain. Therefore, it is not that effective in identifying trending keywords representative for a target domain compared to our models.

[8]https://github.com/GeneralMills/pytrends

[9]https://trends.google.com

## 5 Conclusion

This paper proposes an approach for solving an important but understudied problem of a domain representative keywords selection from candidates contrasting with a context domain. Our approach utilizes a two-component mixture model mechanism followed by a novel subset selection optimization algorithm to tackle the problem. We believe this work will encourage the automated text structuring problem and help a wide range of downstream applications in NLP. For future research direction, we want to focus on adapting the proposed approach in a more challenging task like single document summarization where the scope of information is limited. Besides, our proposed techniques are general and thus can be used in many applications such as information extraction, topic modeling, and concept indexing. Exploration of those applications is an interesting future direction.

## Acknowledgements

We thank the anonymous reviewers for their valuable comments and suggestions. This material is based upon work supported by the National Science Foundation IIS 16-19302 and IIS 16-33755, Zhejiang University ZJU Research 083650, Futurewei Technologies HF2017060011 and 094013, IBM-Illinois Center for Cognitive Computing Systems Research (C3SR)- a research collaboration as part of the IBM Cognitive Horizon Network, grants from eBay and Microsoft Azure, UIUC OVCR CCIL Planning Grant 434S34, UIUC CSBS Small Grant 434C8U, and UIUC New Frontiers Initiative. Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the author(s) and do not necessarily reflect the views of the funding agencies.

## References


Rabah Alzaidy, Cornelia Caragea, and C Lee Giles. 2019. Bi-lstm-crf sequence labeling for keyphrase extraction from scholarly documents. In *The world wide web conference*, pages 2551–2557.

Kamil Bennani-Smires, Claudiu Musat, Andreea Hossmann, Michael Baeriswyl, and Martin Jaggi. 2018. Simple unsupervised keyphrase extraction using sentence embeddings. *arXiv preprint arXiv:1801.04470*.

Adam L. Berger and John D. Lafferty. 1999. Information retrieval as statistical translation. In *SIGIR '99: Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, August 15-19, 1999, Berkeley, CA, USA*, pages 222–229. ACM.

David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *the Journal of machine Learning research*, 3:993–1022.

Gerlof Bouma. 2009. Normalized (pointwise) mutual information in collocation extraction. *Proceedings of GSCL*, 30:31–40.

Peter F Brown, Stephen A Della Pietra, Vincent J Della Pietra, and Robert L Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational linguistics*, 19(2):263–311.

Sheng Chen, Akshay Soni, Aasish Pappu, and Yashar Mehdad. 2017. Doctag2vec: An embedding based multi-label learning approach for document tagging. *arXiv preprint arXiv:1707.04596*.

Arthur P Dempster, Nan M Laird, and Donald B Rubin. 1977. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society: Series B (Methodological)*, 39(1):1–22.

Uriel Feige. 1998. A threshold of ln n for approximating set cover. *Journal of the ACM (JACM)*, 45(4):634–652.

Abram Handler, Matthew Denny, Hanna Wallach, and Brendan O'Connor. 2016. Bag of what? simple noun phrase extraction for text analysis. In *Proceedings of the First Workshop on NLP and Computational Social Science*, pages 114–124.

Anna Hätty, Michael Dorna, and Sabine Schulte im Walde. 2017. Evaluating the reliability and interaction of recursively used feature classes for terminology extraction. In *Proceedings of the student research workshop at the 15th conference of the European chapter of the association for computational linguistics*, pages 113–121.

Thomas Hofmann. 2001. Unsupervised learning by probabilistic latent semantic analysis. *Machine learning*, 42(1):177–196.

Jie Huang, Kevin Chen-Chuan Chang, Jinjun Xiong, and Wen-mei Hwu. 2021. Measuring fine-grained domain relevance of terms: A hierarchical core-fringe approach. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*.

Jack Hughes, Seth Aycock, Andrew Caines, Paula Buttery, and Alice Hutchings. 2020. Detecting trending terms in cybersecurity forum discussions. In *Proceedings of the Sixth Workshop on Noisy User-generated Text (W-NUT 2020)*, pages 107–115.

Maryam Karimzadehgan and ChengXiang Zhai. 2010. Estimation of statistical translation models based on mutual information for ad hoc information retrieval. In *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval*, pages 323–330.

Vishal Kaushal, Rishabh Iyer, Suraj Kothawade, Rohan Mahadev, Khoshrav Doctor, and Ganesh Ramakrishnan. 2019. Learning from less data: A unified data subset selection and active learning framework for computer vision. In *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1289–1299. IEEE.

Solomon Kullback and Richard A Leibler. 1951. On information and sufficiency. *The annals of mathematical statistics*, 22(1):79–86.

Jialu Liu, Jingbo Shang, Chi Wang, Xiang Ren, and Jiawei Han. 2015. Mining quality phrases from massive text corpora. In *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*, pages 1729–1744.

Weiming Lu, Yangfan Zhou, Jiale Yu, and Chenhao Jia. 2019. Concept extraction and prerequisite relation learning from educational data. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 9678–9685.

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

Rada Mihalcea and Paul Tarau. 2004. Textrank: Bringing order into text. In *Proceedings of the 2004 conference on empirical methods in natural language processing*, pages 404–411.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Michel Minoux. 1978. Accelerated greedy algorithms for maximizing submodular set functions. In *Optimization techniques*, pages 234–243. Springer.

Pitu B Mirchandani and Richard L Francis. 1990. *Discrete location theory*.

Burt L Monroe, Michael P Colaresi, and Kevin M Quinn. 2008. Fightin'words: Lexical feature selection and evaluation for identifying the content of political conflict. *Political Analysis*, 16(4):372–403.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Sunita Sarawagi, Soumen Chakrabarti, and Shantanu Godbole. 2003. Cross-training: Learning probabilistic mappings between topics. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 177–186.

Jingbo Shang, Jialu Liu, Meng Jiang, Xiang Ren, Clare R Voss, and Jiawei Han. 2018. Automated phrase mining from massive text corpora. *IEEE Transactions on Knowledge and Data Engineering*, 30(10):1825–1837.

Jie Tang, Jing Zhang, Limin Yao, Juanzi Li, Li Zhang, and Zhong Su. 2008. Arnetminer: extraction and mining of academic social networks. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 990–998.

Peter D Turney. 2000. Learning algorithms for keyphrase extraction. *Information retrieval*, 2(4):303–336.

Li Wang, Wei Zhu, Sihang Jiang, Sheng Zhang, Keqiang Wang, Yuan Ni, Guotong Xie, and Yanghua Xiao. 2020. Mining infrequent high-quality phrases from domain-specific corpora. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, pages 1535–1544.

Ian H. Witten, Gordon W. Paynter, Eibe Frank, Carl Gutwin, and Craig G. Nevill-Manning. 1999. KEA: practical automatic keyphrase extraction. In *Proceedings of the Fourth ACM conference on Digital Libraries, August 11-14, 1999, Berkeley, CA, USA*, pages 254–255. ACM.

Chengxiang Zhai and John Lafferty. 2001. Model-based feedback in the language modeling approach to information retrieval. In *Proceedings of the tenth international conference on Information and knowledge management*, pages 403–410.

ChengXiang Zhai, Atulya Velivelli, and Bei Yu. 2004. A cross-collection mixture model for comparative text mining. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 743–748.

Chao Zhang, Fangbo Tao, Xiusi Chen, Jiaming Shen, Meng Jiang, Brian Sadler, Michelle Vanni, and Jiawei Han. 2018. Taxogen: Constructing topical concept taxonomy by adaptive term embedding and clustering. *Proc. KDDI*.

| $k$ | RF | LO | PR | FL | $\mathbf{KL_{rf}}$ | MM | $\mathbf{KL_{mm}}$ |
|---|---|---|---|---|---|---|---|
| Candidate Keywords from Springer | | | | | | | |
| 10 | 2.0691 | 2.1157 | 1.1273 | 1.0852 | 1.0998 | 2.1401 | **2.1432** |
| 20 | 2.1745 | 2.2391 | 2.2557 | 2.1745 | 2.1805 | 2.2915 | **3.3260** |
| 30 | 2.2302 | 3.3938 | 3.4220 | 2.2261 | 2.2556 | 3.4259 | **3.4262** |
| 40 | 2.2873 | 3.5040 | 3.5053 | 2.2847 | 2.3223 | 5.5379 | **5.5404** |
| 50 | 2.3591 | 5.6136 | 3.6136 | 2.3591 | 2.3689 | 5.6346 | **5.6576** |
| 100 | 3.7701 | 7.0986 | 6.0387 | 3.7601 | 3.7846 | **8.1300** | 7.1193 |
| 200 | 6.4205 | 12.0418 | 8.8759 | 5.3433 | 6.3992 | **12.1043** | 12.0591 |
| 500 | 13.3390 | 19.4237 | 13.9664 | 13.2692 | 12.2365 | **19.4987** | 19.4857 |
| Candidate Keywords from Aminer | | | | | | | |
| 10 | 1.1020 | 2.1205 | 1.1101 | 1.1073 | 1.1109 | 2.1205 | **2.1645** |
| 20 | 2.1782 | 2.3050 | 1.2272 | 2.1699 | 2.1956 | 3.3048 | **3.3079** |
| 30 | 2.2433 | 3.4320 | 2.3747 | 2.2508 | 2.2795 | 3.4394 | **4.4522** |
| 40 | 2.3159 | 3.5312 | 3.5345 | 2.3012 | 2.3944 | 5.5693 | **5.5700** |
| 50 | 2.4617 | 5.6519 | 4.6396 | 3.4657 | 2.4675 | 5.6463 | **5.6829** |
| 100 | 3.8432 | 8.1523 | 6.1264 | 3.8038 | 3.8429 | **9.1788** | 8.1679 |
| 200 | 6.5778 | 12.1533 | 8.9075 | 6.5320 | 6.5431 | **12.2007** | 12.1013 |
| 500 | 13.668 | 19.5232 | 15.1445 | 13.4622 | 12.4942 | 19.5580 | **20.5255** |

Table 6: Results of selected summary keywords' correspondence with arxiv category keywords

| | Selected 20 keywords in CS |
|---|---|
| RF | model, method, paper, problem, approach, **algorithm**, datum, **network**, **system**, **performance**, task, result, **image**, number, user, application, time, **dataset**, **graph**, work |
| LO | task, **algorithm**, user, **network**, **performance**, **dataset**, **image**, problem, **training**, approach, **deep learning**, method, **node**, **agent**, language, **neural network**, paper, video, challenge, **architecture** |
| PR | **image**, **graph**, **dataset**, user, method, model, **network**, task, **algorithm**, datum, problem, **system**, training, channel, node, performance, object, agent, deep learning, language |
| FL | **image**, model, paper, problem, method, network, datum, approach, **algorithm**, **system**, user, performance, result, **graph**, task, application, number, work, time, dataset |
| $\mathbf{KL_{rf}}$ | model, method, problem, **system**, **network**, datum, **algorithm**, paper, **image**, task, user, approach, performance, graph, application, dataset, time, result, number, information |
| MM | task, **algorithm**, user, **network**, **performance**, **dataset**, image, **training**, **deep learning**, **node**, **agent**, language, **neural network**, video, **architecture**, challenge, **robot**, real world, **attack**, **learning** |
| $\mathbf{KL_{mm}}$ | task, user, **algorithm**, **dataset**, **network**, **performance**, **image**, **training**, **agent**, language, **deep learning**, **attack**, **robot**, video, challenge, **node**, **neural network**, **query**, **code**, **machine learning** |

Keywords distinctive to the CS domain are **highlighted** (annotated by authors).

Table 7: Summary keywords selected from Springer candidate keywords in CS domain

| | Selected 20 keywords in CS |
|---|---|
| RF | model, method, **network**, **algorithm**, **system**, datum, problem, user, **image**, time, **graph**, application, **performance**, state, feature, **dataset**, number, art, work, information |
| LO | **network**, user, **algorithm**, **dataset**, art, **performance**, training, **image**, task, **deep learning**, **node**, **learning**, **agent**, **attack**, **neural network**, language, video, problem, **robot**, **graph** |
| PR | **image**, art, **graph**, **dataset**, state, user, model, **network**, method, **algorithm**, datum, vertex, training, channel, **system**, feature, **node**, **deep learning**, experiment, object |
| FL | art, model, method, **network**, **algorithm**, **image**, datum, **system**, problem, user, **graph**, application, **performance**, time, work, number, **dataset**, feature, order, experiment |
| $\mathbf{KL_{rf}}$ | model, method, **algorithm**, **network**, **system**, datum, **image**, user, problem, **graph**, **dataset**, application, **performance**, time, **feature**, **agent**, art, information, number, **training** |
| MM | **network**, user, **algorithm**, **dataset**, art, **performance**, **training**, **image**, task, **deep learning**, **node**, **learning**, **agent**, **neural network**, **attack**, language, video, **robot**, **architecture**, **machine learning** |
| $\mathbf{KL_{mm}}$ | **dataset**, user, **network**, **algorithm**, **training**, **image**, **agent**, task, **performance**, **attack**, art, language, **deep learning**, **robot**, **learning**, video, **node**, **machine learning**, **code**, **neural network** |

Keywords distinctive to the CS domain are **highlighted** (annotated by authors).

Table 8: Summary keywords selected from Aminer candidate keywords in CS domain

| Religion | Recreation | Science | Politics |
|---|---|---|---|
| talk.religion.misc | rec.autos | sci.crypt | talk.politics.misc |
| alt.atheism | rec.motorcycles | sci.electronics | talk.politics.guns |
| soc.religion.christian | rec.sport.baseball | sci.med | talk.politics.mideast |
| | rec.sport.hockey | sci.space | |

Table 9: Subtopics in each of the four topics in the newsgroup dataset

## A Proof of Theorem 1

To prove this, we need to first show that $\phi(\cdot)$ is *submodular* and *monotone*. As, we are concerned on minimizing $\phi(\cdot)$, it is equivalent to maximizing $F(\cdot) = -\phi(\cdot)$. Hence, it is sufficient to prove that $F(.)$ is submodular and monotone. Let, $X \subseteq Y \subseteq V$ and $v \in V$, then we get

$$
\begin{aligned}
&F(X \cup \{v\}) - F(\{X\}) = \\
&\sum_{t_i \in V} p(t_i) \log \frac{\sum_{t_j \in X \cup \{v\}} p(t_i|t_j)p(t_j)}{p(t_i)} \\
&- \sum_{t_i \in V} p(t_i) \log \frac{\sum_{t_j \in X} p(t_i|t_j)p(t_j)}{p(t_i)} \\
&= \sum_{t_i \in V} p(t_i) \log \frac{\sum_{t_j \in X \cup \{v\}} p(t_i|t_j)p(t_j)}{\sum_{t_j \in X} p(t_i|t_j)p(t_j)} \\
&= \sum_{t_i \in V} p(t_i) \log \frac{\sum_{t_j \in X} p(t_i|t_j)p(t_j) + p(t_i|v)p(v)}{\sum_{t_j \in X} p(t_i|t_j)p(t_j)} \\
&= \sum_{t_i \in V} p(t_i) \log(1 + \frac{p(t_i|v)p(v)}{\sum_{t_j \in X} p(t_i|t_j)p(t_j)}).
\end{aligned}
$$

Similarly, $F(Y \cup \{v\}) - F(\{Y\}) =$

$$
\sum_{t_i \in V} p(t_i) \log(1 + \frac{p(t_i|v)p(v)}{\sum_{t_j \in Y} p(t_i|t_j)p(t_j)}).
$$

As $X \subseteq Y$, then,

$$
\sum_{t_j \in X} p(t_i|t_j)p(t_j) \leq \sum_{t_j \in Y} p(t_i|t_j)p(t_j).
$$

Therefore, $F(X \cup \{v\}) - F(\{X\}) \geq F(Y \cup \{v\}) - F(\{Y\})$ which proves that $F(\cdot)$ is *submodular*. Moreover, we can show that $F(Y) - F(X) = \sum_{t_i \in V} p(t_i) \log \frac{\sum_{t_j \in Y} p(t_i|t_j)p(t_j)}{\sum_{t_j \in X} p(t_i|t_j)p(t_j)} \geq 1$. Hence, $F(Y) \geq F(X)$ for $X \subseteq Y \subseteq V$ which proves that $F(\cdot)$ is monotone. Therefore, it proves that minimizing $\phi(\cdot)$ using simple greedy algorithm guarantees an approximation of $1 - \frac{1}{e}$.

| | Religion | Recreation | Science | Politics |
|---|---|---|---|---|
| RF | thing, **church**, life, word, man, **religion**, **bible**, **faith**, question, **belief**, book, point, law, evidence, **sin**, reason, world, truth, child, **god** | **game**, **car**, **team**, **player**, **bike**, **season**, **point**, **hockey**, problem, lot, **goal**, **baseball**, guy, **engine**, power, number, year, line, question, **run** | key, information, thing, government, **space**, **encryption**, datum, **clipper**, **chip**, case, number, phone, **bit**, **privacy**, drug, earth, power, **security**, **program**, disease | **government**, **gun**, child, state, law, country, man, **president**, case, **war**, group, fact, **firearm**, number, **crime**, question, **weapon**, world, history, population |
| LO | **church**, **bible**, **faith**, **religion**, **belief**, **sin**, **god**, **scripture**, life, word, **atheist**, truth, **atheism**, **homosexuality**, love, man, evidence, son, **morality**, book | **game**, **team**, **car**, **player**, **bike**, **season**, **hockey**, **baseball**, **playoff**, **engine**, **goal**, **pitcher**, **tire**, **run**, pen, **league**, **puck**, **motorcycle**, dog, **clutch** | key, **encryption**, **space**, **clipper**, **privacy**, **satellite**, **mission**, **disease**, **shuttle**, phone, **orbit**, **escrow**, **moon**, **cancer**, **algorithm**, **spacecraft**, **security**, launch, **vitamin**, health | **gun**, **government**, **firearm**, **president**, country, **weapon**, **crime**, village, **soldier**, **genocide**, **war**, population, state, child, police, **turk**, **massacre**, **handgun**, compound, new york |
| PR | man, word, thing, life, world, history, **church**, book, question, **bible**, **faith**, point, truth, reason, matter, law, year, **religion**, earth, mind | **game**, **team**, **player**, **car**, **season**, **goal**, **point**, **hockey**, shot, year, power, number, **engine**, **bike**, **win**, **league**, speed, line, **run**, end | information, datum, year, study, number, united states, **space**, nature, **security**, mail, government, thing, **encryption**, case, archive, key, life, book, law, **science** | **government**, man, group, **war**, world, village, child, fact, year, history, life, state, house, end, woman, home, **power**, arm, law, population |
| FL | man, thing, **church**, life, word, **religion**, book, question, history, point, **bible**, law, evidence, reason, **sin**, world, **belief**, child, **faith**, case | **game**, **car**, **team**, **player**, **bike**, **season**, pit, problem, **hockey**, lot, power, **point**, **baseball**, **engine**, **goal**, **run**, question, guy, standing, **speed** | information, **space**, key, study, thing, government, datum, year, case, number, patient, **software**, power, book, archive, food, **clipper**, **mission**, hicnet medical newsletter page, **encryption** | **government**, man, village, **gun**, **president**, **sumgait**, history, state, case, child, **law**, world, country, population, fact, **war**, los angeles, number, group, year |
| $\mathbf{KL_{rf}}$ | thing, life, **church**, word, **belief**, man, question, **religion**, **bible**, **faith**, book, law, evidence, point, **sin**, world, reason, child, truth, **god** | **game**, **car**, **team**, **bike**, **player**, **point**, problem, **season**, lot, **hockey**, **engine**, guy, power, **baseball**, question, **goal**, number, list, road, year | key, information, **space**, thing, government, **encryption**, datum, case, **clipper**, number, **chip**, **disease**, power, phone, **earth**, drug, **program**, **bit**, book, **privacy** | **government**, **gun**, child, state, country, man, law, president, **war**, case, group, fact, question, **crime**, population, number, world, **firearm**, history, woman |
| MM | **bible**, **church**, **faith**, **sin**, **scripture**, **atheism**, **god**, **gospel**, **christianity**, **prophecy**, **mcconkie**, **jesus christ**, **prophet**, **new testament**, **atheist**, **disciple**, **holy spirit**, **theist**, **christian**, **homosexuality** | **team**, pen, **player**, **bike**, **hockey**, **season**, **second period**, **puck**, **playoff**, **first period**, **ranger**, **schedule**, **pitcher**, **baseball**, **nhl**, **cub**, **tire**, **injury**, **league**, respect | **encryption**, **clipper**, **privacy**, **satellite**, **shuttle**, **orbit**, **vitamin**, **infection**, **escrow**, **moon**, **pgp**, **mission**, **spacecraft**, **cryptography**, **cancer**, **circuit**, **astronaut**, **asteroid**, **cipher**, **telescope** | **gun**, **firearm**, **soldier**, village, **genocide**, **bayonet**, **turk**, **handgun**, **massacre**, new york, **tartar**, **homicide**, **civilian**, **weapon**, **human right**, **gun control**, **bullet**, **troop**, **ottoman**, **sumgait** |
| $\mathbf{KL_{mm}}$ | **bible**, **church**, **faith**, **sin**, **scripture**, **atheism**, **god**, **gospel**, **jesus christ**, **prophecy**, **christianity**, **new testament**, **mcconkie**, **prophet**, **holy spirit**, **morality**, **theist**, **disciple**, **homosexuality**, **atheist** | **team**, pen, **bike**, **player**, **season**, **hockey**, **second period**, **playoff**, **pitcher**, **puck**, **schedule**, **ranger**, **cub**, **baseball**, **tire**, **clutch**, **first period**, favor, respect, **nhl** | **encryption**, **satellite**, **clipper**, **vitamin**, **shuttle**, **infection**, **orbit**, **moon**, **cancer**, **privacy**, **spacecraft**, **circuit**, **mission**, **pgp**, **escrow**, **allergy**, **yeast**, **cryptography**, **diet**, **solar sail** | **gun**, village, **firearm**, **soldier**, **genocide**, **turk**, new york, **massacre**, **human right**, **handgun**, **bayonet**, **civilian**, **croat**, **tartar**, **weapon**, **gun control**, **troop**, **homicide**, **well regulated**, **sumgait** |

Keywords distinctive to the subtopics in a respected topic are **highlighted** (annotated by authors).

Table 10: Summary keywords selected by different algorithms on four topics from newsgroups dataset

## B Additional Quantitative Results on CS Domain

Results are shown in Table 6.

## C Additional Qualitative Results on CS Domain

Results are shown in Tables 7 and 8.

## D Evaluation Using Newsgroup Dataset

We use newsgroup dataset covering four known topics named *Religion*, *Recreation*, *Science* and *Politics*. In this study, we split the whole dataset into these four topic groups represented by their corpus and use the whole newsgroup dataset as our background corpus. Table 9 shows the subtopics for each of the four topics. For each topic, we show the selected top 20 keywords using different algorithms in Table 10. This study aims to evaluate the capability of the proposed models to select distinctive keywords for each topic compared to the baselines. We can see that almost all the keywords selected by our methods MM and $\mathbf{KL_{mm}}$ are distinctive for each topic relating closely with respected subtopics shown in Table 9 and do not overlap with other topics. Similarly, as previously, the results from LO come close to ours with some anomalies. For instance, our methods select informative keywords like *jesus christ*, *holy spirit* and *new testament* for *religion* topic rather than non-distinctive keywords like *word*, *man* and *son*.