MotifClass: Weakly Supervised Text Classification with Higher-order Metadata Information

Yu Zhang[1*], Shweta Garg[1*], Yu Meng[1], Xiusi Chen[2], Jiawei Han[1]

[1]Department of Computer Science, University of Illinois at Urbana-Champaign, IL, USA
[2]Department of Computer Science, University of California, Los Angeles, CA, USA
[1]{yuz9, shwetag2, yumeng5, hanj}@illinois.edu,   [2]xchen@cs.ucla.edu

ABSTRACT

We study the problem of weakly supervised text classification, which aims to classify text documents into a set of pre-defined categories with category surface names only and without any annotated training document provided. Most existing classifiers leverage textual information in each document. However, in many domains, documents are accompanied by various types of metadata (e.g., authors, venue, and year of a research paper). These metadata and their combinations may serve as strong category indicators in addition to textual contents. In this paper, we explore the potential of using metadata to help weakly supervised text classification. To be specific, we model the relationships between documents and metadata via a heterogeneous information network. To effectively capture higher-order structures in the network, we use motifs to describe metadata combinations. We propose a novel framework, named MotifClass, which (1) selects category-indicative motif instances, (2) retrieves and generates pseudo-labeled training samples based on category names and indicative motif instances, and (3) trains a text classifier using the pseudo training data. Extensive experiments on real-world datasets demonstrate the superior performance of MotifClass to existing weakly supervised text classification approaches. Further analysis shows the benefit of considering higher-order metadata information in our framework.

CCS CONCEPTS

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

KEYWORDS

text classification; weak supervision; metadata

---

*Equal Contribution.

(a) Two documents connecting with their metadata and text information
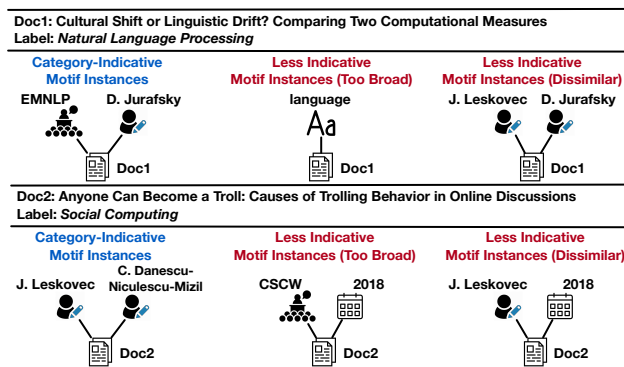
(b) Examples of category-indicative and non-indicative motif instances associated with the two documents

Figure 1: A network view of documents with metadata. Some metadata or metadata combinations (i.e., motifs) are category-indicative, while others are not.

1 INTRODUCTION

Text classification is a fundamental task in text mining with a wide spectrum of applications such as text geolocalization [4], sentiment analysis [25], and email intent detection [33]. Following the routine of supervised learning, one can build a text classifier from human-annotated training documents. Many deep learning-based models (e.g., [9, 12, 37]) have achieved great performance in text classification when trained on a large-scale annotated corpus. Despite such a success, a frequent bottleneck of applying these models to a new domain is the acquisition of abundant annotated documents.

Weakly supervised text classification, which relies on only category names or a few descriptive keywords to train a classifier, has recently gained increasing attention as it eliminates the need for human annotations. Under the weakly supervised setting, most existing approaches leverage only the text data in each document [2, 17, 21, 22, 35]. However, in various domains, documents are beyond plain text sequences and are accompanied by different types of metadata (e.g., authors, venue, and year of a scientific paper; user and product of an e-commerce review). These metadata, together

with text, provide better clues of the inter-relationship between multiple documents and thus are useful for inferring their categories. Figure 1(a) provides a network view of an academic paper corpus with metadata. We can see that some metadata neighbors are helpful for predicting the category of a document. For example, the venue node *EMNLP* suggests *Doc1*'s relevance to Natural Language Processing.

Recent studies [18, 40, 41] have confirmed that metadata signals are beneficial to weakly supervised text classification. However, in their work, the authors are less concerned with two important factors: higher-order metadata information and metadata specificity.

Higher-order Metadata Information[1]**.** Different types of metadata should be considered collectively in classification. For example, the combination of *AAAI* and *1990* is a strong indicator that the paper belongs to the traditional AI domain because the scope of AAAI was more focused in the early years. In comparison, either the venue or the year alone becomes a weaker signal. As another example, in Figure 1, *Doc2* has two authors *J. Leskovec* and *C. Danescu-Niculescu-Mizil*. Neither of them alone is enough to predict the category Social Computing, but their co-authorship becomes category-indicative. Such higher-order information (called *motifs* in network science [1, 24, 27]) is not explored in [40, 41].

Metadata Specificity. To suggest the category of a document, a motif instance should be not only semantically close to that category but also specific enough to indicate only one category. For example, in Figure 1, the venue *CSCW* may be linked with many papers related to Social Computing, but purely relying on *CSCW* (or even the combination of *CSCW* and *2018*) will introduce noises because it is broader than the category. Similarly, the term *language* in *Doc1* is too broad to predict Natural Language Processing. Such metadata and text specificity is not considered in [18, 40, 41].

Contributions. In this paper, we study the problem of weakly supervised metadata-aware text classification. Being aware of higher-order metadata information and metadata specificity, we propose to discover discriminative and specific motifs for each category to help text classification. Specifically, we propose MotifClass, a framework that is built in three steps. (1) *Indicative motif instance selection*: We leverage motif patterns (e.g., Venue & Author) to obtain candidate motif instances (e.g., *KDD & J. Leskovec*) in the dataset. Then, for each category, we select category-indicative motif instances based on their similarity with the label surface name as well as their specificity. To facilitate this, we propose a joint representation learning method to learn motif instance embedding and specificity simultaneously. (2) *Pseudo-labeled training data collection*: By matching unlabeled documents with selected motif instances, we can retrieve documents that likely belong to a certain category. Besides retrieval, we propose to generate artificial training documents based on motif-aware text embeddings. The retrieval and the generation strategies are proved to be complementary to each other in creating pseudo training data. (3) *Text classifier training*: We finally train a text classifier using collected pseudo training data. Note that our framework is compatible with any text classifier.

To summarize, this work makes the following contributions:
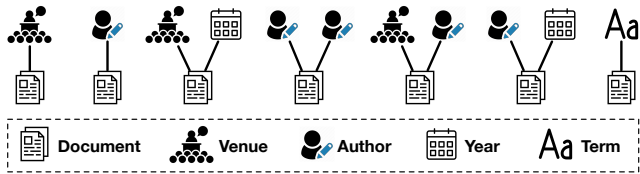
Figure 2: Motif patterns used in an academic paper corpus.

- We propose a weakly supervised text classification model MotifClass. It does not need any human-annotated document for training. Instead, it relies on category names and utilizes higher-order document metadata as additional supervision.
- We design an instance-level motif selection method to discover category-indicative metadata signals. The method is featured by a joint representation learning process that simultaneously learns the embedding and specificity of each motif instance.
- We conduct experiments on two real-world datasets to show the superiority of MotifClass to existing weakly supervised metadata-aware text classification methods.

2 PRELIMINARIES

2.1 Text, Metadata, and Motif

Text. We assume the text information of each document is a sequence of terms, denoted as $w_1 w_2 ... w_N$. Each term w_i here can be either a word or a phrase[2]. To simplify our discussion, if a document has multiple text fields (e.g., title and abstract of a paper), we concatenate them into one sequence.

Metadata and Metadata Instance. Documents are often associated with metadata [18, 41]. For example, research papers can have Author, Venue, and Year fields. Each metadata type has its instances appearing in the dataset (e.g., Venue[*EMNLP*], Author[*D. Jurafsky*], Year[*2016*]).

As illustrated in Figure 1, we can construct a heterogeneous information network (HIN) [29] to describe documents with metadata. The formal definition of HIN is as follows.

Definition 2.1. (Heterogeneous Information Network [29]) An HIN is a graph $G = (\mathcal{V}, \mathcal{E})$ with a node type mapping $\phi : \mathcal{V} \to \mathcal{T}_{\mathcal{V}}$ and an edge type mapping $\psi : \mathcal{E} \to \mathcal{T}_{\mathcal{E}}$. Either the number of node types $|\mathcal{T}_{\mathcal{V}}|$ or the number of edge types $|\mathcal{T}_{\mathcal{E}}|$ is larger than 1.

In our constructed HIN, $\mathcal{V}$ consists of document nodes, term nodes, and all metadata instances; $\mathcal{E}$ includes edges connecting each document with its metadata information and words/phrases.

Motif Pattern and Motif Instance. In an HIN, a motif pattern refers to a subgraph at the type level.

Definition 2.2. (Motif Pattern [27]) A motif pattern in an HIN is a connected graph p. Each node in p is a node type $\in \mathcal{T}_{\mathcal{V}}$, and each edge in p is an edge type $\in \mathcal{T}_{\mathcal{E}}$.

In the document classification task, we focus on motif patterns with one Document node. In this way, motif patterns essentially describe the semantics of metadata and their combinations. For example, Figure 2 shows the motif patterns that can be used in an

---

[1]The term "higher-order" in this paper refers to higher-order network structures [1] represented by certain subgraph patterns [27], such as one document linked with two authors. It does not refer to multi-hop relationships or higher-order logic here.

[2]We include phrases into our discussion because, in many scenarios, category names are not single words (e.g., Data Mining, Video Games). In practice, given a sequence of words, one can use existing phrase chunking tools [15, 26] to detect phrases in it.
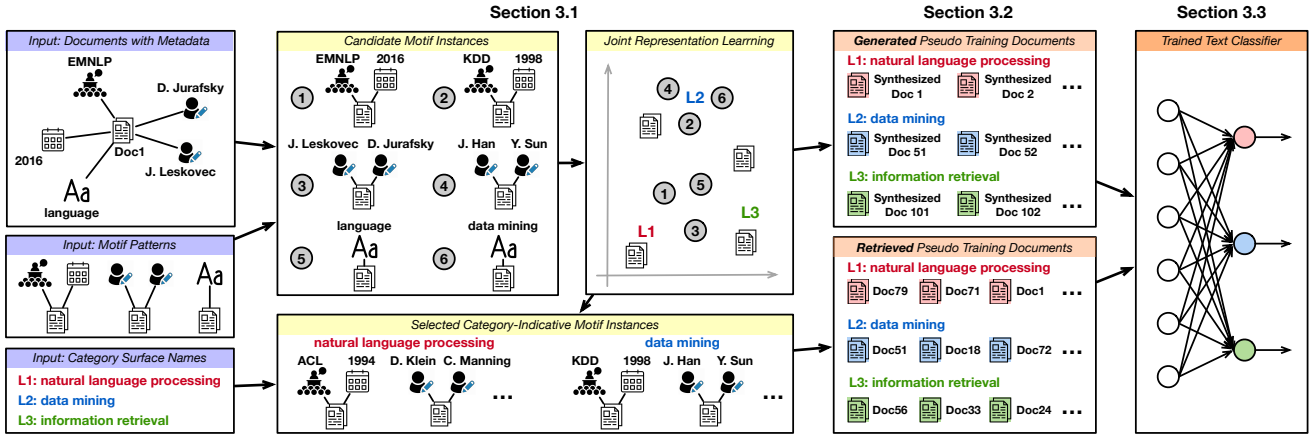
Figure 3: The overview of our MotifClass framework. We first discover category-indicative motif instances from documents with metadata. Pseudo-labeled training documents are then collected according to the selected motif signals. A text classifier is finally trained on the pseudo training data.

academic paper corpus. They are able to model the relationship between documents and (higher-order) metadata. For ease of notation, in this paper, when representing a motif pattern, we omit the Document node and write the metadata node(s) only. For example, in Figure 2, the third pattern from the left can be written as Venue-Year, and the fourth one can be written as Author-Author. We view the connection between a document and a term also as a motif pattern (i.e., Term), so that we can describe text information.

Similar to a single metadata type, a motif pattern has its instances (e.g., Venue[*EMNLP*]-Year[*2016*], Term[*data mining*]). Note that each word/phrase w_i appearing in the corpus will be viewed as a motif instance Term[w_i].

2.2 Problem Definition

Given a collection of documents $\mathcal{D} = \{d_i\}_{i=1}^{|\mathcal{D}|}$ and a set of categories $\mathcal{L} = \{l_j\}_{j=1}^{|\mathcal{L}|}$, the text classification task is to assign a category label l_j to each document d_i. In this paper, we study the *weakly supervised* setting, where no human-annotated training data is needed, and the only descriptive signal of each category is the surface text of its category name. Each category l has only one surface name (a term, denoted as n_l). We allow the category name to be either a single word (e.g., database) or a phrase (e.g., data mining). This assumption is more relaxed than that of previous studies using BERT-based models [17, 22, 35], where the category name must be a single word in the vocabulary of BERT.

Following the setting in [18, 27], we ask users to specify a set of possibly useful motif patterns $\mathcal{P} = \{p_i\}_{i=1}^{|\mathcal{P}|}$ as input to our model, like what we show in Figure 2. (This setting is aligned with many HIN embedding studies [6, 34, 36] that prefer users to input a set of meta-paths.) If users do not have such prior knowledge, we can also enumerate all possible metadata combinations (including those with ≥ 3 metadata nodes) as candidate motif patterns. Note that our framework automatically refines motif signals through instance-level selection, thus is robust to the existence of unreliable input motif patterns.

To summarize, our problem definition is as follows.

Definition 2.3. (Problem Definition) Given a set of unlabeled documents $\mathcal{D}$ with metadata, a label space $\mathcal{L}$, the surface name of each category $\{n_l : l \in \mathcal{L}\}$, and a set of candidate motif patterns $\mathcal{P}$, the task is to assign a label $l \in \mathcal{L}$ to each $d \in \mathcal{D}$.

3 FRAMEWORK

Figure 3 illustrates the overall MotifClass framework. The core idea is to use category names and higher-order metadata information to create pseudo-labeled training data. To implement this idea, we first discover category-indicative motif instances for each category through a joint representation learning process (**Section 3.1**). Then, we retrieve and generate pseudo-labeled training documents based on selected motif instances and learned motif-aware embeddings, respectively (**Section 3.2**). Finally, using pseudo-labeled documents, we train a text classifier (**Section 3.3**).

3.1 Selecting Indicative Motif Instances

Given the candidate motif patterns, we first find all instances of these motifs in the corpus $\mathcal{D}$ (instances with frequency below a certain threshold will be discarded). We denote the set of candidate motif instances as $\mathcal{M} = \{m_i\}_{i=1}^{|\mathcal{M}|}$. For each category $l \in \mathcal{L}$, our first step is to select a group of category-indicative motif instances $\mathcal{M}_l \subseteq \mathcal{M}$. We assume the category name Term[n_l] must be an indicative motif instance of l. Then the goal is to find other indicative instances according to Term[n_l]. We propose the following two criteria.

Similarity. The selected motif instance should be semantically similar to the corresponding category name. In other words, if we embed all motif instances into the same latent space, we expect each selected motif instance (e.g., Venue[*EMNLP*]-Author[*D. Jurafsky*]) to have high embedding cosine similarity with the category name (e.g., Term[*natural language processing*]).

Specificity. The selected motif instances should not indicate multiple categories at the same time (e.g., Venue[*AAAI*]), so that we can use these instances to infer high-quality pseudo training data for each category. To facilitate this, we require the selected motif instances to be semantically more specific than the category name.

3.1.1 Joint Embedding and Specificity Learning. Based on these two requirements, for each motif instance $m \in \mathcal{M}$, we propose to learn two parameters $\boldsymbol{e}_m$ and κ_m from the corpus. Here, $\boldsymbol{e}_m$ is the embedding vector of m, and κ_m is the specificity of the instance (a scalar). The larger κ_m is, the more focused semantics the instance should indicate. For example, we should expect $\kappa_{\text{VENUE}[AAAI]} < \kappa_{\text{VENUE}[EMNLP]} < \kappa_{\text{AUTHOR}[D. Jurafsky]}$.

To simultaneously estimate $\boldsymbol{e}_m$ and κ_m, we propose a joint representation learning process that embeds motif instances, categories, and documents into the same latent space. It considers the following two types of proximity in the learning objective.

Motif Instance–Document Proximity. Previous studies on word embedding [19, 31] encourage the proximity between each word and its belonging document. This idea can be directly generalized from words to motif instances. Given an instance m and a document d, where m appears in d [3], we aim to maximize the following probability:

$$p(d|m) = \frac{\exp(\kappa_m \boldsymbol{e}_m^T \boldsymbol{e}_d)}{\sum_{d' \in \mathcal{D}} \exp(\kappa_m \boldsymbol{e}_m^T \boldsymbol{e}_{d'})}. \tag{1}$$

If we ignore κ_m, Eq. (1) is essentially a softmax function widely used in embedding learning. Meng et al. [19] first introduce "κ" into the softmax function to model word specificity. We extend their technique to the motif case. To explain why κ_m can represent the specificity of m, we follow [19] and introduce the von Mises-Fisher (vMF) distribution [7].

Definition 3.1. (The Von Mises-Fisher Distribution [7]) The vMF distribution is defined on a unit sphere $\mathbb{S}^{\delta-1} = \{\boldsymbol{x} \in \mathbb{R}^\delta : ||\boldsymbol{x}||_2 = 1\}$. It is parameterized by the mean direction vector $\boldsymbol{\mu}$ and the concentration parameter κ. The probability density function is

$$\text{vMF}(\boldsymbol{x}|\boldsymbol{\mu}, \kappa) = c_\delta(\kappa) \exp(\kappa \boldsymbol{\mu}^T \boldsymbol{x}), \tag{2}$$

where $\boldsymbol{x} \in \mathbb{S}^{\delta-1}$, $\boldsymbol{\mu} \in \mathbb{S}^{\delta-1}$, and $\kappa \geq 0$. Here, $c_\delta(\kappa)$ is a constant related to κ and δ only.

Intuitively, the vMF distribution can be viewed as an analogue of the Gaussian distribution on a sphere. The distribution concentrates around $\boldsymbol{\mu}$, and is more concentrated if κ is larger.

Motivated by the fact that directional similarity is more effective in capturing semantics [19, 20], we require all embedding vectors $\boldsymbol{e}_m$ and $\boldsymbol{e}_d$ in Eq. (1) to reside on a unit sphere, then we have

$$\lim_{|\mathcal{D}| \to \infty} p(d|m) = \frac{\exp(\kappa_m \boldsymbol{e}_m^T \boldsymbol{e}_d)}{\int_{\mathbb{S}^{\delta-1}} \exp(\kappa_m \boldsymbol{e}_m^T \boldsymbol{e}_{d'}) d\boldsymbol{e}_{d'}} = \frac{\exp(\kappa_m \boldsymbol{e}_m^T \boldsymbol{e}_d)}{1/c_\delta(\kappa_m)}$$
$$= c_\delta(\kappa_m) \exp(\kappa_m \boldsymbol{e}_m^T \boldsymbol{e}_d) = \text{vMF}(\boldsymbol{e}_d|\boldsymbol{e}_m, \kappa_m). \tag{3}$$

The second step holds because $\int_{\mathbb{S}^{\delta-1}} c_\delta(\kappa_m) \exp(\kappa_m \boldsymbol{e}_m^T \boldsymbol{e}_{d'}) d\boldsymbol{e}_{d'} = \int_{\mathbb{S}^{\delta-1}} \text{vMF}(\boldsymbol{e}_{d'}|\boldsymbol{e}_m, \kappa_m) d\boldsymbol{e}_{d'} \equiv 1$. Eq. (3) essentially assumes that given the motif instance m, the embeddings of documents containing m are generated from $\text{vMF}(\cdot|\boldsymbol{e}_m, \kappa_m)$. For a motif instance with more general meaning (e.g., $\text{VENUE}[AAAI]$), it will appear in more diverse documents. Therefore, its learned vMF distribution will have a lower concentration parameter κ_m than that of a more specific instance (e.g., $\text{VENUE}[EMNLP]$). This explains why κ_m can represent the specificity of m.

---

[3] We say a motif instance m appears in a document d if and only if d contains all metadata instances of m. For example, in Figure 1, the instance $\text{VENUE}[EMNLP]$-$\text{AUTHOR}[D. Jurafsky]$ appears in *Doc1*.

Given the probability $p(d|m)$ in Eq. (1), we aim to maximize the log-likelihood

$$\mathcal{J}_{\text{Doc}} = \sum_{m \in \mathcal{M}} \sum_{d:\, m \text{ appears in } d} \log p(d|m). \tag{4}$$

(TERM) Motif Instance–Context Proximity. Words/phrases have local context information. To be specific, given a text sequence $w_1 w_2 ... w_N$, Mikolov et al. [23] define the local context of w_i as $C(w_i, h) = \{w_j : i - h \leq j \leq i + h, j \neq i\}$, where h is the context window size. We view each word/phrase as a TERM instance, so the local context of a TERM motif instance can be written as $C(\text{TERM}[w_i], h) = \{\text{TERM}[w_j] : i - h \leq j \leq i + h, j \neq i\}$. According to the Skip-Gram model [23], we consider the following proximity.

$$p(C(m, h)|m) = \prod_{m_+ \in C(m,h)} \frac{\exp(\kappa_m \boldsymbol{e}_m^T \boldsymbol{e}_{m_+})}{\sum_{m_-} \exp(\kappa_m \boldsymbol{e}_m^T \boldsymbol{e}_{m_-})}. \tag{5}$$

Similar to Eq. (1), the specificity κ_m is added into the softmax function. The log-likelihood is given by

$$\mathcal{J}_{\text{Ctxt}} = \sum_{d \in \mathcal{D}} \sum_{m:\, \text{TERM instance, appears in } d} \log p(C(m, h)|m). \tag{6}$$

Based on the two types of proximity, our joint representation learning process can be cast as an optimization problem:

$$\max_{\boldsymbol{e}_m, \boldsymbol{e}_d, \kappa_m} \mathcal{J} = \mathcal{J}_{\text{Doc}} + \mathcal{J}_{\text{Ctxt}}, \quad \text{s.t. } \boldsymbol{e}_m, \boldsymbol{e}_d \in \mathbb{S}^{\delta-1}, \kappa_m \geq 0. \tag{7}$$

The optimization process of this objective can be found in Appendix A.1.

3.1.2 Motif Instance Selection. After obtaining the embedding vector and specificity of each motif instance, we are able to select a set of indicative motif instances $\mathcal{M}_l$ for each category. First, we assume the category name n_l must be indicative, so we have the TERM instance $m_l = \text{TERM}[n_l]$ in $\mathcal{M}_l$. Then, we find top-ranked instances and add them into $\mathcal{M}_l$. The ranking criterion is

$$\max_{m \in \mathcal{M}} \cos(\boldsymbol{e}_m, \boldsymbol{e}_{m_l}), \quad \text{where } \kappa_m \geq \eta \cdot \kappa_{m_l}. \tag{8}$$

Here, $\eta > 1$ is a hyperparameter. Intuitively, from all motif instances that are more specific than the category name (i.e., the *specificity* criterion), we select a number of instances closest to the category name in the embedding space (i.e., the *similarity* criterion).

3.2 Retrieving and Generating Pseudo-Labeled Training Data

Based on the selected motif instances and motif-aware embeddings, we aim to collect pseudo-labeled training data $\mathcal{D}_l$ for each category l. In this paper, we propose two ways, *retrieval* and *generation*. The idea of retrieval is to use category-indicative motif instances $\mathcal{M}_l$ to find existing unlabeled documents which likely belong to l. In contrast, the idea of generation is to generate artificial documents (i.e., sequences of text and metadata) that have close meaning to l.

Retrieval. Given a document $d \in \mathcal{D}$ and a category $l \in \mathcal{L}$, we calculate the score that d belongs to l by counting the number of l's indicative motif instances appearing in d. Formally,

$$\text{score}(d, l) = \sum_{m \in \mathcal{M}_l} \mathbf{1}(m \text{ appears in } d). \tag{9}$$

Here, $\mathbf{1}(\cdot)$ is the indicator function.

For each category l, we retrieve a set of documents $\mathcal{D}_l^{\mathrm{R}} \subseteq \mathcal{D}$ as the pseudo training data with label l. The retrieved documents should have a high score with l and a score of 0 with any other category. In other words, the ranking criterion is

$$\max_{d \in \mathcal{D}} \mathrm{score}(d, l), \quad \text{where } \mathrm{score}(d, l') = 0 \quad (\forall l' \neq l). \quad (10)$$

By Eq. (10), top-ranked documents are selected and added to $\mathcal{D}_l^{\mathrm{R}}$.

Generation. Given $l \in \mathcal{L}$, we generate a set of synthesized documents $\mathcal{D}_l^{\mathrm{G}}$ that belong to the category l. To generate text related to a certain topic, we follow the idea in [41] and leverage our joint representation learning space. Specifically, there are two major steps: *Step 1*: given a category, generate a document embedding e_d that is semantically close to the category. *Step 2*: given the document embedding e_d, generate a sequence of metadata instances and words/phrases that are coherent with the document semantics.

Step 1: We have obtained the category name embedding e_{m_l} in the joint representation learning step. When generating the document embedding, we expect e_d to be close to e_{m_l} in the embedding space, thus we adopt the vMF distribution.

$$e_d \sim \mathrm{vMF}(\cdot | e_{m_l}, \kappa). \quad (11)$$

Note that we cannot use a softmax function here because we are "creating" a new document instead of sampling one from the existing pool. Therefore, we use the vMF distribution which, according to Eq. (3), is a good approximation of a softmax function.

Step 2: Now, to form a complete document, we aim to generate a sequence of metadata instances and words/phrases. Note that most words/phrases and metadata instances appearing in the dataset can be represented as a motif instance (e.g., VENUE[*EMNLP*], AUTHOR[*D. Jurafsky*], TERM[*language*]). We have learned the embeddings of all motif instances above a certain frequency threshold. Based on these embeddings, the probability of generating a word/phrase or metadata instance m in a document d is given by

$$p(m|e_d) = \frac{\exp(e_d^T e_m)}{\sum_{m' \in \mathcal{M}_{\mathrm{Gen}}} \exp(e_d^T e_{m'})} \quad (\forall m \in \mathcal{M}_{\mathrm{Gen}}). \quad (12)$$

Here, $\mathcal{M}_{\mathrm{Gen}}$ is the set of words/phrases and metadata instances used to generate d. In practice, we set $\mathcal{M}_{\mathrm{Gen}}$ to be the top-50 nearest neighbors of e_d in the embedding space. We do not use all words/phrases and metadata instances in the embedding space because the computational cost of $\sum_{m' \in \mathcal{M}_{\mathrm{Gen}}} \exp(e_d^T e_{m'})$ will be very high in that case. Using Eq. (12) repeatedly, we can obtain a sequence of metadata instances and words/phrases $m_1 m_2 ... m_N$.

The final set of pseudo-labeled training documents $\mathcal{D}_l$ is the union of the retrieved ones $\mathcal{D}_l^{\mathrm{R}}$ and the generated ones $\mathcal{D}_l^{\mathrm{G}}$. We use the combination of retrieval and generation strategies because they have different merits. Retrieved documents are real, thus have higher linguistic quality. However, the input corpus $\mathcal{D}$ may not have lots of documents whose pseudo-label predictions are confident enough. In contrast, generated documents are artificial, but the number of generated documents is not limited by the size of $\mathcal{D}$.

3.3 Training a Text Classifier

Our framework is compatible with any text classification model as a classifier (e.g., CNN [12], HAN [37], Transformer [32]). The goal of this paper is not to develop a novel classifier. Therefore,

Table 1: Dataset statistics.

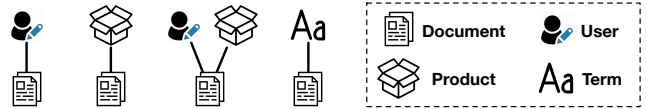|	MAG-CS [42]	Amazon [16]
#Documents	203,157	100,000
Avg Doc Length	125	120
#Categories	20	10
Text Fields	title, abstract	headline, review
Metadata Fields	AUTHOR, VENUE, YEAR	USER, PRODUCT

Figure 4: Motif patterns used in the Amazon dataset.

following previous studies [21, 40, 41], we adopt Kim-CNN [12] as our classifier, with all parameter settings the same as those in [21].

Given a pseudo-labeled training document $d \in \mathcal{D}_l$, we feed both its text and metadata information into the classifier. Specifically, if d is retrieved, we concatenate its metadata and text information into one sequence. For example, given the paper *Doc1* in Figure 1, the input sequence is

"AUTHOR[*W. Hamilton*] AUTHOR[*J. Leskovec*] AUTHOR[*D. Jurafsky*] VENUE[*EMNLP*] YEAR[*2016*] *cultural shift or linguistic drift* ..."

If d is generated, it already has a mixed sequence of metadata instances and words/phrases. We train Kim-CNN on each d with its pseudo-label. The training loss is the negative log-likelihood. The initialized word/phrase and metadata embeddings are those learned in joint representation learning (i.e., e_m) since most words/phrases or metadata instances can be viewed as a motif instance.

We would like to report that we have also tried BERT [5] as our classifier, but its performance is not so good as that of Kim-CNN, possibly because the fixed vocabulary of BERT restricts its capacity to deal with metadata instances (e.g., author names, product IDs).

4 EXPERIMENTS

4.1 Setup

Datasets. We use two real-world datasets from different domains for evaluation[4]. Dataset statistics are listed in Table 1.

- **MAG-CS [42]** is constructed from the Microsoft Academic Graph (MAG). It consists of papers published in 105 top CS conferences from 1990 to 2020. Each paper has labels at different levels of the MAG taxonomy. We use labels in the highest level for classification, and we remove papers that belong to two or more categories because our problem setting is single-label classification. The candidate motif patterns are listed in Figure 2.
- **Amazon [16]** is a crawl of Amazon product reviews. Each review is associated with its user (i.e., reviewer) and product IDs. 10 large categories are selected and 10,000 reviews are sampled from each category. The used motif patterns are listed in Figure 4.

In MAG-CS, phrase terms are already recognized. In Amazon, we use AutoPhrase [26] to detect phrases in the text. The whole datasets are used for evaluation because our weakly supervised setting does not require any training document with ground-truth labels.

---

[4]The code and datasets are available at
https://github.com/yuzhimanhua/MotifClass.

Compared Methods. We compare MotifClass with the following methods, including both weakly supervised text classification approaches and HIN embedding methods.

- **WeSTClass [21]** is a weakly supervised text classification approach. It can take category surface names as supervision and applies a pre-training and self-training scheme.
- **WeSTClass+Metadata** is an easy extension of WeSTClass. Since WeSTClass considers text information only, we concatenate all metadata instances of a document with its text as the input sequence, so that WeSTClass can utilize metadata signals.
- **MetaCat [41]** is a weakly supervised metadata-aware text classification approach. It takes a small set of labeled documents, instead of category names, as supervision. To align the experiment setting, we use the pseudo-labeled documents retrieved by MotifClass (i.e., $\mathcal{D}_l^R$) as the supervision of MetaCat.
- **META [18]** is a weakly supervised metadata-aware text classification approach. It can take category names as supervision and iteratively performs classification and motif instance expansion.
- **LOTClass [22]** is a weakly supervised text classification approach based on BERT. It takes category names or keywords as supervision, but each category name/keyword must be a single word in the vocabulary of BERT. To apply it here, we separate each phrase category name into single words and remove those common words that appear in multiple category names.
- **Metapath2Vec [6]** is an HIN embedding method. We use it to embed terms, documents, and metadata instances into the same space. The category of each document is given by its nearest category name in the embedding space.
- **HIN2Vec [8]** is an HIN embedding method that considers metapath embeddings in addition to node embeddings. We perform nearest neighbor search after learning the embeddings to classify each document.
- **HGT [10]** is a recent heterogeneous graph neural network model. We adopt the unsupervised unattributed version of HGT and perform nearest neighbor search after learning node embeddings.
- **MotifClass-NoHigherOrder** is an ablation version of MotifClass that does not leverage higher-order metadata information. Specifically, it only considers single metadata types (e.g., Venue, Author, and Term in Figure 2) as input motifs.
- **MotifClass-NoSpecificity** is another ablation version of MotifClass that does not consider specificity of motif instances. In other words, for each m, κ_m is fixed to be 1. There is no specificity requirement when selecting motif instances.

We also present the performance of BERT [5] under the fully supervised setting (shown as **Supervised BERT** in Table 2), where we perform a random 80%-10%-10% train-dev-test split of the datasets.

Implementation and Hyperparameters. We discard infrequent motif instances that appear in less than 5 documents. The embedding dimension $\delta = 100$. The context window size $h = 5$. During embedding learning, for each positive sample, we collect 5 negative samples. The size of selected motif instances $|\mathcal{M}_l| = 50$. The specificity criterion is $\kappa_m \geq 2\kappa_{m_l}$ (i.e., $\eta = 2$). The size of retrieved and generated training set $|\mathcal{D}_l^R| = |\mathcal{D}_l^G| = 50$ for MAG-CS, and $|\mathcal{D}_l^R| = |\mathcal{D}_l^G| = 100$ for Amazon. For the CNN classifier [12], following [21], we use filters with widths 2, 3, 4, and 5. For each width, we generate 20 feature maps. The maximum input sequence length is set to be 200 for both datasets. The CNN classifier is trained

Table 2: Performance of compared methods on MAG-CS and Amazon. Bold: the highest score of weakly supervised methods. *: significantly worse than MotifClass (p-value < 0.05). **: significantly worse than MotifClass (p-value < 0.01).

Algorithm	MAG-CS		Amazon	
	Micro-F1	Macro-F1	Micro-F1	Macro-F1
WeSTClass [21]	0.464**	0.326**	0.519**	0.547**
WeSTClass+Metadata	0.525**	0.369**	0.610**	0.603**
MetaCat [41]	0.488**	0.403**	0.664**	0.657**
META [18]	0.398**	0.373**	0.664**	0.662
LOTClass [22]	0.124**	0.107**	0.658*	0.589**
Metapath2Vec [6]	0.436**	0.414**	0.619**	0.611**
HIN2Vec [8]	0.408**	0.350**	0.628**	0.566**
HGT [10]	0.151**	0.136**	0.272**	0.211**
MotifClass-NoHigherOrder	0.549*	0.476**	0.682	**0.670**
MotifClass-NoSpecificity	0.553*	0.499	0.675*	0.664
MotifClass	**0.571**	**0.501**	**0.689**	**0.670**
Supervised BERT [5]	0.798	0.717	0.952	0.952

using SGD with the training batch size of 256. The hyperparameter configuration of all baselines can be found in the Appendix.

4.2 Performance Comparison

Table 2 shows the Micro/Macro-F1 scores of compared algorithms. We repeat each experiment 5 times with the mean reported. To measure statistical significance, we conduct a two-tailed unpaired t-test to compare MotifClass with each baseline approach. The significance level is also marked in Table 2.

From Table 2, we observe that: (1) MotifClass consistently achieves the best performance. In most cases, the gap between MotifClass and baselines is statistically significant. (2) The full MotifClass model outperforms MotifClass-NoHigherOrder and MotifClass-NoSpecificity on both datasets (although not significant in some cases), which validates our claim that higher-order metadata information and metadata specificity are helpful to text classification. (3) Some weakly supervised text classification methods, such as WeSTClass+Metadata, MetaCat, and MotifClass-NoHigherOrder, consider single metadata types only. The advantage of MotifClass over these methods is larger on MAG-CS than it is on Amazon. This is possibly because higher-order motif structures can be better exploited in the MAG-CS network. Specifically, 4 out of 7 candidate motif patterns used in MAG-CS are higher-order, while only 1 out of 4 is higher-order in Amazon. (4) LOTClass is a strong baseline on Amazon but performs quite poorly on MAG-CS. This is because most category names in MAG-CS are phrases, and separating them into single words actually distorts the meaning of those categories. (5) MotifClass outperforms HIN embedding approaches (i.e., Metapath2Vec, HIN2Vec, and HGT) by a clear margin. We believe this is because the constructed HIN losses local context information of each term in the text. In contrast, MotifClass models context proximity (i.e., $\mathcal{J}_{\text{Ctxt}}$ in Eq. (6)) in addition to the HIN.

4.3 Analysis of Higher-order Metadata

The comparison between the full MotifClass model and two ablation versions already show the positive contribution of both higher-order metadata and metadata specificity. In Sections 4.3 and 4.4, we would like to give more detailed analyses of these two factors. We start from higher-order metadata in this section.

Table 3: Proportion of each motif pattern in selected motif instances on MAG-CS. We show 10 (out of 20) categories. V: Venue, A: Author, Y: Year, T: Term.

| Category	V	A	T	V-Y	V-A	A-A	A-Y
computer security	0	0.24	0.24	0.34	0.14	0.04	0
computer vision	0	0.20	0.14	0.36	0.20	0.06	0.04
data mining	0	0.24	0.18	0.36	0.22	0	0
database	0	0.06	0.38	0.44	0.10	0.02	0
embedded system	0	0.24	0.54	0	0.14	0.08	0
information retrieval	0	0.22	0.02	0.44	0.32	0	0
machine learning	0	0.30	0.38	0	0.16	0.06	0.10
multimedia	0	0.10	0.06	0.48	0.34	0.02	0
real time computing	0.04	0.16	0.54	0.08	0.12	0.06	0
theoretical comp. sci.	0	0.42	0	0.54	0.04	0	0
overall	0.003	0.211	0.301	0.248	0.186	0.032	0.019

(a) MAG-CS

(b) Amazon
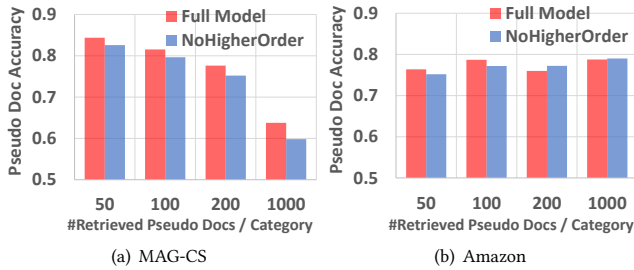
Figure 5: Accuracy of retrieved pseudo-labeled training documents with and without using higher-order metadata.

Observation 1: Higher-order instances cover a large proportion of selected instances. Table 3 presents the proportion of each motif pattern in selected motif instances on MAG-CS. (For example, if we select $|\mathcal{M}_l| = 50$ motif instances for a category l, and 10 of them are instances of Venue-Year, then the proportion of Venue-Year is $10/50 = 0.20$.) Due to space limit, we show 10 (out of 20) categories. We also show the overall proportion of each pattern across all 20 categories. It can be observed that: (1) The overall proportion of Venue-Year, Venue-Author, Author-Author, and Author-Year is 48.5% in total. In other words, nearly half of the motif instances selected by MotifClass are higher-order. (2) The same motif pattern can play very different roles in different categories. For example, for "theoretical computer science", the proportion of Venue-Year is more than 50%. However, for "embedded system", MotifClass does not pick any Venue-Year instance, possibly because conferences related to "embedded system" often have papers belonging to "real-time computing" as well. Besides, only "computer vision" and "machine learning" categories have Author-Year instances selected. This is possibly because most Author-Year instances are infrequent, but machine learning and computer vision researchers have relatively more papers per year on average, so that the corresponding Author-Year instances can pass the minimum frequency threshold.

Observation 2: Higher-order instances improve the quality of retrieved pseudo training data. To explore the benefit of leveraging higher-order metadata signals in pseudo training data collection, we calculate the accuracies of pseudo-labeled training documents retrieved by the full MotifClass model and MotifClass-NoHigherOrder. (For example, if 1000 documents are retrieved in total, and for 800 of them, the pseudo label is the same as the true label, then the accuracy is $800/1000 = 0.80$.) Figure 5 demonstrates the pseudo training data accuracy when we retrieve 50, 100, 200, and 1000 documents per category.

On MAG-CS, we can see the advantage of MotifClass over MotifClass-NoHigherOrder in retrieving pseudo training data. In fact, MotifClass consistently outperforms MotifClass-NoHigherOrder in terms of the retrieved training document accuracy. Intuitively, the accuracy of pseudo-labeled training data will affect the quality of the trained text classifier. By considering higher-order motif patterns, the full MotifClass model is able to find more category-indicative instances and collect more accurate training samples, which can explain why it finally outperforms MotifClass-NoHigherOrder in Table 2. On Amazon, the gap between MotifClass and MotifClass-NoHigherOrder is not significant in terms of the retrieved training document accuracy, which is also reflected in their final classification performance in Table 2.

4.4 Analysis of Specificity

Now we proceed to metadata specificity. To explain why considering specificity is important in motif instance selection, we list motif instances that are close to each category at different specificity levels in Table 4. We choose three categories in MAG-CS – "database", "data mining", and "information retrieval". Note that in the hyperparameter settings of MotifClass, we require $\kappa_m \geq 2\kappa_{m_l}$. Therefore, instances in the first two rows (in red) will not be selected by MotifClass.

We have two findings from Table 4. (1) When κ_m is smaller, the instances have broader semantic coverage, meanwhile become less category-indicative. For example, Venue[$CIKM$] is broader than the three categories because CIKM accepts papers from all these three areas. Although close to "database" and "information retrieval" in the embedding space, it should be filtered out since it is not discriminative enough to indicate either category. (2) When κ_m becomes larger, more higher-order motif instances emerge. For example, when $2\kappa_{m_l} \leq \kappa_m < 3\kappa_{m_l}$, we start to see Venue-Year instances; when $3\kappa_{m_l} \leq \kappa_m < 4\kappa_{m_l}$, Venue-Author instances emerge; when $4\kappa_{m_l} \leq \kappa_m$, we can find Author-Author instances. Such metadata combinations express more accurate semantics, meanwhile cover fewer documents than single metadata instances. Overall, we believe setting $\eta = 2$ can strike a good balance here.

4.5 Efficiency

Table 5 shows the running time of all weakly supervised text classification methods on Intel Xeon E5-2680 v2 @ 2.80GHz and one NVIDIA GeForce GTX 1080. Since MotifClass considers higher-order network structures, its running time is longer than WeSTClass and MetaCat. However, compared with META which also leverages metadata combinations, MotifClass is 26.6 times faster on MAG-CS and 5.2 times faster on Amazon.

5 RELATED WORK

Weakly Supervised Text Classification. Weakly supervised text classification aims to classify documents solely based on label surface names or category-indicative keywords. A pioneering approach is dataless classification [2, 28, 38] which relies on Wikipedia to

Table 4: Motif instances close to each category at different specificity levels (from coarse to fine) on MAG-CS. Too general instances (in red) will not be selected by MotifClass. κ_{m_l}: specificity of the category, κ_m: specificity of the motif instance.

Choice of κ_m	database ($\kappa_{m_l} = 0.498$)	data mining ($\kappa_{m_l} = 0.588$)	information retrieval ($\kappa_{m_l} = 0.576$)
$0 \leq \kappa_m < \kappa_{m_l}$ **Not Selected**	TERM[*records*] TERM[*index*] VENUE[*CIKM*]	TERM[*mining*] VENUE[*KDD*] TERM[*big data*]	TERM[*retrieval*] TERM[*documents*] VENUE[*CIKM*]
$\kappa_{m_l} \leq \kappa_m < 2\kappa_{m_l}$ **Not Selected**	TERM[*sql*] TERM[*relational database management system*] VENUE[*SIGMOD*]	TERM[*knowledge extraction*] TERM[*association rule learning*] TERM[*data mining algorithm*]	VENUE[*SIGIR*] TERM[*document retrieval*] TERM[*text retrieval*]
$2\kappa_{m_l} \leq \kappa_m < 3\kappa_{m_l}$	TERM[*dbmss*] TERM[*database research*] VENUE[*SIGMOD*]-YEAR[*2018*]	TERM[*apriori algorithm*] VENUE[*KDD*]-YEAR[*2008*] AUTHOR[*Jiawei Han*]	VENUE[*SIGIR*]-YEAR[*2019*] TERM[*ir evaluation*] VENUE[*CLEF*]
$3\kappa_{m_l} \leq \kappa_m < 4\kappa_{m_l}$	TERM[*sql database*] VENUE[*VLDB*]-YEAR[*2008*] VENUE[*ICDE*]-AUTHOR[*David B. Lomet*]	VENUE[*KDD*]-YEAR[*2007*] VENUE[*KDD*]-AUTHOR[*Usama M. Fayyad*] VENUE[*KDD*]-AUTHOR[*Mohammed Zaki*]	VENUE[*SIGIR*]-YEAR[*1994*] AUTHOR[*Donna Harman*] TERM[*faceted search*]
$4\kappa_{m_l} \leq \kappa_m$	VENUE[*VLDB*]-YEAR[*1998*] AUTHOR[*H.-P. Kriegel*]-AUTHOR[*Daniel A. Keim*] VENUE[*VLDB*]-AUTHOR[*Michael J. Carey*]	VENUE[*KDD*]-YEAR[*1996*] VENUE[*KDD*]-AUTHOR[*Heikki Mannila*] VENUE[*KDD*]-AUTHOR[*Charu C. Aggarwal*]	VENUE[*SIGIR*]-YEAR[*2004*] VENUE[*SIGIR*]-AUTHOR[*Noriko Kando*] VENUE[*SIGIR*]-AUTHOR[*Nicholas J. Belkin*]

Table 5: Running time (in hours) of weakly supervised text classification methods on the two datasets.

	WeSTClass	MetaCat	LOTClass	META	MotifClass
MAG-CS	2.9	0.4	6.1	74.7	2.8
Amazon	0.2	0.3	1.1	11.0	2.1

map labels and documents into the same semantic space and derive their relevance. Along another line, seed-guided topic models [3, 13] infer topics from descriptive keywords and predict labels from posterior category-topic assignments. Recently, neural models have been applied to weakly supervised text classification. Meng et al. [21] propose to generate documents to train a neural classifier and refine the classifier via self-training. Their approach is further improved by introducing pre-trained language models. For example, ConWea [17] utilizes contextualized word representations to detect category-indicative words for pseudo-label generation. LOTClass [22] uses BERT to predict masked category names to find category-indicative keywords. X-Class [35] leverages BERT representation of each word to cluster and align documents to categories.

However, all these approaches consider only the text information and do not make use of metadata signals. Moreover, BERT-based approaches require the category names or keywords to be a single word in the vocabulary of BERT, while our MotifClass framework can take phrases as category names.

Metadata-Aware Text Classification. There are many efforts to incorporate metadata into text classification in a specific domain. For example, Tang et al. [30] consider user and product information in document-level sentiment analysis; Zubiaga et al. [45] and Zhang et al. [43] leverage user profile information for tweet geolocalization. To deal with the general metadata-aware text classification task, Kim et al. [11] add categorical metadata representation into a neural classifier; Zhang et al. [42] present a Transformer architecture to encode metadata. While achieving inspiring performance, these approaches are fully supervised and require massive annotated training data. In contrast, our method only requires label surface names as supervision.

Recently, Zhang et al. [39, 40, 41, 44] use a small set of labeled documents or keywords as supervision to categorize text with metadata. However, their methods consider each metadata instance separately and fail to model higher-order interactions between different types of metadata. Mekala et al. [18] adopt motif patterns to iteratively discover topic-related motif instances and retrieve pseudo-labeled training data. Compared with their method, MotifClass is able to model the specificity of each motif instance, which is crucial when selecting category-indicative motif instances.

6 CONCLUSIONS AND FUTURE WORK

In this paper, we propose to study weakly supervised metadata-aware text classification from the HIN perspective, which avails us with additional higher-order network structures besides corpus. We identify the importance of modeling higher-order metadata information and metadata specificity. We then propose the MotifClass framework that discovers indicative motif instances for each category to create pseudo-labeled training documents. Experimental results demonstrate the effectiveness of MotifClass as well as the utility of considering higher-order metadata and specificity.

In the future, it is of interest to extend our framework to hierarchical or multi-label text classification, where each document can belong to more than one category. In this setting, categories are no longer mutually exclusive, and one needs to reconsider how to select representative motif instances and assign pseudo labels. Besides, we would like to explore the potential of automatically detecting candidate motif patterns under weak supervision.

REFERENCES

[1] Austin R Benson, David F Gleich, and Jure Leskovec. 2016. Higher-order organization of complex networks. *Science* 353, 6295 (2016), 163–166.

[2] Ming-Wei Chang, Lev-Arie Ratinov, Dan Roth, and Vivek Srikumar. 2008. Importance of Semantic Representation: Dataless Classification. In *AAAI'08*. 830–835.

[3] Xingyuan Chen, Yunqing Xia, Peng Jin, and John Carroll. 2015. Dataless text classification with descriptive LDA. In *AAAI'15*.

[4] Zhiyuan Cheng, James Caverlee, and Kyumin Lee. 2010. You are where you tweet: a content-based approach to geo-locating twitter users. In *CIKM'10*. 759–768.

[5] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *NAACL-HLT'19*. 4171–4186.

[6] Yuxiao Dong, Nitesh V Chawla, and Ananthram Swami. 2017. metapath2vec: Scalable representation learning for heterogeneous networks. In *KDD'17*. 135–144.

[7] Ronald Aylmer Fisher. 1953. Dispersion on a sphere. *Proceedings of the Royal Society of London. Series A. Mathematical and Physical Sciences* 217, 1130 (1953), 295–305.

[8] Tao-yang Fu, Wang-Chien Lee, and Zhen Lei. 2017. Hin2vec: Explore meta-paths in heterogeneous information networks for representation learning. In *CIKM'17*. 1797–1806.

[9] Qipeng Guo, Xipeng Qiu, Pengfei Liu, Yunfan Shao, Xiangyang Xue, and Zheng Zhang. 2019. Star-Transformer. In *ACL'19*. 1315–1325.

[10] Ziniu Hu, Yuxiao Dong, Kuansan Wang, and Yizhou Sun. 2020. Heterogeneous graph transformer. In *WWW'20*. 2704–2710.

[11] Jihyeok Kim, Reinald Kim Amplayo, Kyungjae Lee, Sua Sung, Minji Seo, and Seung-won Hwang. 2019. Categorical Metadata Representation for Customized Text Classification. *TACL* 7 (2019), 201–215.

[12] Yoon Kim. 2014. Convolutional Neural Networks for Sentence Classification. In *EMNLP'14*. 1746–1751.

[13] Chenliang Li, Jian Xing, Aixin Sun, and Zongyang Ma. 2016. Effective document labeling with very few seed words: A topic model approach. In *CIKM'16*. 85–94.

[14] Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of Machine Learning Research* 9, Nov (2008), 2579–2605.

[15] Christopher D Manning, Mihai Surdeanu, John Bauer, Jenny Rose Finkel, Steven Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *ACL'14, System Demonstrations*. 55–60.

[16] Julian McAuley and Jure Leskovec. 2013. Hidden factors and hidden topics: understanding rating dimensions with review text. In *RecSys'13*. 165–172.

[17] Dheeraj Mekala and Jingbo Shang. 2020. Contextualized Weak Supervision for Text Classification. In *ACL'20*. 323–333.

[18] Dheeraj Mekala, Xinyang Zhang, and Jingbo Shang. 2020. META: Metadata-Empowered Weak Supervision for Text Classification. In *EMNLP'20*. 8351–8361.

[19] Yu Meng, Jiaxin Huang, Guangyuan Wang, Zihan Wang, Chao Zhang, Yu Zhang, and Jiawei Han. 2020. Discriminative Topic Mining via Category-Name Guided Text Embedding. In *WWW'20*. 2121–2132.

[20] Yu Meng, Jiaxin Huang, Guangyuan Wang, Chao Zhang, Honglei Zhuang, Lance Kaplan, and Jiawei Han. 2019. Spherical text embedding. In *NeurIPS'19*. 8208–8217.

[21] Yu Meng, Jiaming Shen, Chao Zhang, and Jiawei Han. 2018. Weakly-supervised neural text classification. In *CIKM'18*. 983–992.

[22] Yu Meng, Yunyi Zhang, Jiaxin Huang, Chenyan Xiong, Heng Ji, Chao Zhang, and Jiawei Han. 2020. Text Classification Using Label Names Only: A Language Model Self-Training Approach. In *EMNLP'20*. 9006–9017.

[23] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *NIPS'13*. 3111–3119.

[24] Ron Milo, Shai Shen-Orr, Shalev Itzkovitz, Nadav Kashtan, Dmitri Chklovskii, and Uri Alon. 2002. Network motifs: simple building blocks of complex networks.

[25] Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. 2002. Thumbs up? Sentiment Classification using Machine Learning Techniques. In *EMNLP'02*. 79–86.

[26] Jingbo Shang, Jialu Liu, Meng Jiang, Xiang Ren, Clare R Voss, and Jiawei Han. 2018. Automated phrase mining from massive text corpora. *IEEE TKDE* 30, 10 (2018), 1825–1837.

[27] Jingbo Shang, Xinyang Zhang, Liyuan Liu, Sha Li, and Jiawei Han. 2020. Nettaxo: Automated topic taxonomy construction from text-rich network. In *WWW'20*. 1908–1919.

[28] Yangqiu Song and Dan Roth. 2014. On dataless hierarchical text classification. In *AAAI'14*. 1579–1585.

[29] Yizhou Sun and Jiawei Han. 2012. Mining heterogeneous information networks: principles and methodologies. *Synthesis Lectures on Data Mining and Knowledge Discovery* 3, 2 (2012), 1–159.

[30] Duyu Tang, Bing Qin, and Ting Liu. 2015. Learning semantic representations of users and products for document level sentiment classification. In *ACL'15*. 1014–1023.

[31] Jian Tang, Meng Qu, and Qiaozhu Mei. 2015. Pte: Predictive text embedding through large-scale heterogeneous text networks. In *KDD'15*. 1165–1174.

[32] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NIPS'17*. 5998–6008.

[33] Wei Wang, Saghar Hosseini, Ahmed Hassan Awadallah, Paul N Bennett, and Chris Quirk. 2019. Context-aware intent identification in email conversations. In *SIGIR'19*. 585–594.

[34] Xiao Wang, Houye Ji, Chuan Shi, Bai Wang, Yanfang Ye, Peng Cui, and Philip S Yu. 2019. Heterogeneous graph attention network. In *WWW'19*. 2022–2032.

[35] Zihan Wang, Dheeraj Mekala, and Jingbo Shang. 2021. X-Class: Text Classification with Extremely Weak Supervision. In *NAACL-HLT'21*. 3043–3053.

[36] Carl Yang, Yuxin Xiao, Yu Zhang, Yizhou Sun, and Jiawei Han. 2020. Heterogeneous network representation learning: A unified framework with survey and benchmark. *IEEE TKDE* (2020).

[37] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *NAACL'16*. 1480–1489.

[38] Wenpeng Yin, Jamaal Hay, and Dan Roth. 2019. Benchmarking Zero-shot Text Classification: Datasets, Evaluation and Entailment Approach. In *EMNLP'19*. 3905–3914.

[39] Xinyang Zhang, Chenwei Zhang, Xin Luna Dong, Jingbo Shang, and Jiawei Han. 2021. Minimally-Supervised Structure-Rich Text Categorization via Learning on Text-Rich Networks. In *WWW'21*. 3258–3268.

[40] Yu Zhang, Xiusi Chen, Yu Meng, and Jiawei Han. 2021. Hierarchical Metadata-Aware Document Categorization under Weak Supervision. In *WSDM'21*. 770–778.

[41] Yu Zhang, Yu Meng, Jiaxin Huang, Frank F. Xu, Xuan Wang, and Jiawei Han. 2020. Minimally Supervised Categorization of Text with Metadata. In *SIGIR'20*. 1231–1240.

[42] Yu Zhang, Zhihong Shen, Yuxiao Dong, Kuansan Wang, and Jiawei Han. 2021. MATCH: Metadata-Aware Text Classification in A Large Hierarchy. In *WWW'21*. 3246–3257.

[43] Yu Zhang, Wei Wei, Binxuan Huang, Kathleen M Carley, and Yan Zhang. 2017. RATE: Overcoming Noise and Sparsity of Textual Features in Real-Time Location Estimation. In *CIKM'17*. 2423–2426.

[44] Yu Zhang, Frank F. Xu, Sha Li, Yu Meng, Xuan Wang, Qi Li, and Jiawei Han. 2019. HiGitClass: Keyword-Driven Hierarchical Classification of GitHub Repositories. In *ICDM'19*. 876–885.

[45] Arkaitz Zubiaga, Alex Voss, Rob Procter, Maria Liakata, Bo Wang, and Adam Tsakalidis. 2017. Towards real-time, country-level location classification of worldwide tweets. *IEEE TKDE* 29, 9 (2017), 2053–2066.

Science 298, 5594 (2002), 824–827.

A APPENDIX

In the Appendix, we first show how we optimize the objective of our joint embedding and specificity learning step (i.e., Eq. (7)). Then, we give more detailed information of the two datasets and all compared methods. Finally, we present additional experimental results including the effect of retrieval and generation strategies as well as the visualization of our joint embedding space.

A.1 Optimization of the Objective

To optimize Eq. (7), we adopt the negative sampling [23] technique. Following [31], each time, we alternately select one term from the objective. Taking $\mathcal{J}_{\text{Doc}}$ as an example. We first randomly sample a motif instance m. Given m, we randomly sample a positive document d (i.e., m appears in d) and several negative documents d' from $\mathcal{D}$ [5]. Then, we need to optimize the following objective.

$$\mathcal{J}_{\text{Doc}} = -\log \sigma(\kappa_m \boldsymbol{e}_m^T \boldsymbol{e}_d) - \sum_{d'} \sigma(-\kappa_m \boldsymbol{e}_m^T \boldsymbol{e}_{d'}) + \text{const.} \quad (13)$$

Here, $\sigma(\cdot)$ is the sigmoid function. Given a parameter θ (θ can be $\boldsymbol{e}_u, \boldsymbol{e}_d, \boldsymbol{e}_{d'}$ or κ_m), we have

$$\frac{\partial \mathcal{J}_{\text{Doc}}}{\partial \theta} = \left(\sigma(\kappa_m \boldsymbol{e}_m^T \boldsymbol{e}_d) - 1 \right) \frac{\partial \kappa_m \boldsymbol{e}_m^T \boldsymbol{e}_d}{\partial \theta} - \sum_{d'} \sigma(\kappa_m \boldsymbol{e}_m^T \boldsymbol{e}_{d'}) \frac{\partial \kappa_m \boldsymbol{e}_m^T \boldsymbol{e}_{d'}}{\partial \theta},$$

where

$$\frac{\partial \kappa_m \boldsymbol{e}_m^T \boldsymbol{e}_d}{\partial \boldsymbol{e}_m} = \kappa_m \boldsymbol{e}_d, \quad \frac{\partial \kappa_m \boldsymbol{e}_m^T \boldsymbol{e}_d}{\partial \boldsymbol{e}_d} = \kappa_m \boldsymbol{e}_m, \quad \frac{\partial \kappa_m \boldsymbol{e}_m^T \boldsymbol{e}_d}{\partial \kappa_m} = \boldsymbol{e}_m^T \boldsymbol{e}_d. \quad (14)$$

Knowing the gradient, we optimize each embedding vector and specificity using gradient descent. To satisfy the constraints, after each update, one can do $\boldsymbol{e} \leftarrow \boldsymbol{e}/\|\boldsymbol{e}\|_2$ if the embedding is not on the unit sphere, and $\kappa \leftarrow 0$ if $\kappa < 0$.

A.2 Datasets

Table 6 shows the surface names of 20 categories in MAG-CS and 10 categories in Amazon.

A.3 Compared Methods

The code source and hyperparameter configuration of each compared method are explained below.

A.3.1 WeSTClass and WeSTClass+Metadata [21]. We run the code from the first author's GitHub[6]. The maximum number of self-training iterations is changed from 5000 to 1000 as we find the performance starts to drop after 1000 iterations. When concatenating metadata and text together as the input to WeSTClass+Metadata, the order is AUTHOR, VENUE, YEAR, text for MAG-CS and USER, PRODUCT, text for Amazon. The maximum input sequence length is 200. Other hyperparameters are set by default.

A.3.2 MetaCat [41]. We use the code from the first author's GitHub[7]. MetaCat takes a small set of labeled documents, instead of category names, as supervision. To align the experiment setting, we use the pseudo-labeled documents retrieved by MOTIFCLASS as the supervision of MetaCat. The maximum input sequence length

---

[5]Inspired by [23], the negative sample distribution $\propto \#\text{motif}(d)^{3/4}$, where $\#\text{motif}(d)$ is the number of motif instances appearing in d.
[6]https://github.com/yumeng5/WeSTClass
[7]https://github.com/yuzhimanhua/MetaCat

Table 6: List of categories in MAG-CS and Amazon.

	MAG-CS [42]	Amazon [16]
Categories	information retrieval computer hardware programming language theoretical computer science speech recognition real time computing database embedded system multimedia machine learning natural language processing software engineering computer network world wide web computer security computer graphics parallel computing data mining human computer interaction computer vision	android books cd clothing electronics health kitchen movies sports video games

Table 7: Meta-paths used by baselines on the two datasets.

	MAG-CS [42]	Amazon [16]
Meta-paths	VENUE→PAPER AUTHOR→PAPER VENUE→PAPER→YEAR AUTHOR→PAPER→AUTHOR VENUE→PAPER→AUTHOR AUTHOR→PAPER→YEAR TERM→PAPER	USER→REVIEW PRODUCT→REVIEW USER→REVIEW→PRODUCT TERM→REVIEW

is 200. The number of training epochs is 40. Other hyperparameters are by default.

A.3.3 META [17]. The code is from the first author's GitHub[8]. The input motif patterns of META are the same as those of our MOTIFCLASS model. Instead of using their phrase mining code, we directly input our phrase chunking results into their model. All hyperparameters are by default.

A.3.4 LOTClass [22]. The code is from the first author's GitHub[9]. LOTClass takes category names or keywords as supervision, but each category name/keyword must be a single word in the vocabulary of BERT. (Other BERT-based weakly supervised text classifiers [17, 35] also suffer from this problem.) To apply it to our datasets, we separate each phrase category name into single words and remove those common words that appear in multiple category names. The maximum input sequence length is 120. The keyword matching threshold is 10 for MAG-CS and 20 for Amazon. Other hyperparameters are by default.

A.3.5 Metapath2Vec [6], HIN2Vec [8], and HGT [10]. Yang et al. [36] integrate 13 popular heterogeneous network representation learning models into one GitHub repository[10], which contains the code of Metapath2Vec, HIN2Vec, and HGT. We use their code. Our constructed HIN has (DOCUMENT, METADATA) and (DOCUMENT, TERM) edges. Metapath2Vec requires users to specify meta-paths.

---

[8]https://github.com/dheeraj7596/META
[9]https://github.com/yumeng5/LOTClass
[10]https://github.com/yangji9181/HNE

Table 8: Effect of retrieval and generation strategies in creating pseudo-labeled training data. Bold: the highest score. *: significantly worse than MotifClass (p-value < 0.05). **: significantly worse than MotifClass (p-value < 0.01).

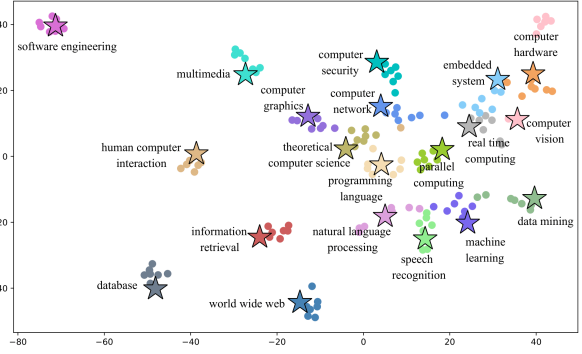| Algorithm	MAG-CS		Amazon	
	Micro-F1	Macro-F1	Micro-F1	Macro-F1
MotifClass (X retrieved + X generated)	**0.571**	**0.501**	**0.689**	**0.670**
X retrieved + 0 generated	0.555	0.489*	0.657**	0.642*
$2X$ retrieved + 0 generated	0.527**	0.469**	0.667**	0.662
0 retrieved + X generated	0.491**	0.449**	0.614**	0.598**
0 retrieved + $2X$ generated	0.486**	0.452**	0.623**	0.610**

Thus, we view the motif patterns used by MotifClass as meta-paths. Table 7 shows the meta-paths used on the two datasets. For all three baselines, we change the embedding dimension from 50 to 100 for consistency with MotifClass. For HGT, we change the number of training epochs from 100 to 500 as we observe higher performance. Other hyperparameters are by default.

A.3.6 Supervised BERT [5]. We use the BertForSequenceClassification class[11] from HuggingFace Transformers. The batch size is 16. The number of training epochs is 10. The model is optimized using AdamW with lr = 5e−5. Other hyperparameters are by default.
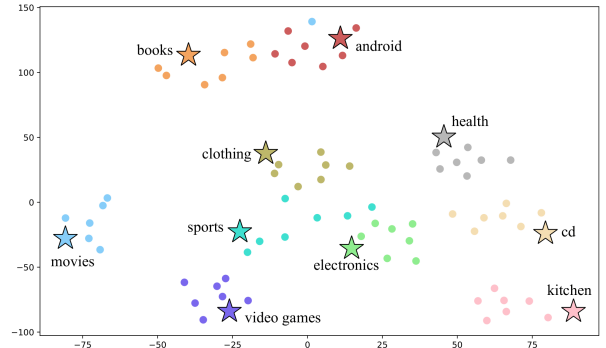
A.4 Additional Experiments: Effect of Retrieval and Generation

We adopt two strategies to collect pseudo-labeled training data – retrieval and generation. At the end of Section 3.2, we have already explained their respective merits. Now, we empirically show the advantage of combining these two strategies. In MotifClass, we set $|\mathcal{D}_l^R| = |\mathcal{D}_l^G| = X$, where $X = 50$ for MAG-CS and $X = 100$ for Amazon. In other words, we collect X retrieved pseudo training documents and X generated ones for each category. We compare this strategy with four variants. Two of the variants do not generate any training data but collect X and $2X$ retrieved documents, respectively, for each category. In contrast, the other two variants do not retrieve any training data but generate X and $2X$ pseudo documents, respectively, for each category. Table 8 compares the classification performance of MotifClass and the four variants.

As we can see from Table 8: (1) MotifClass consistently outperforms the four variants on both datasets. In most cases, the gap is statistically significant. This validates our claim that retrieval and generation strategies are complementary to each other. (2) If we compare "X retrieved + 0 generated" with "$2X$ retrieved + 0 generated", the former performs better on MAG-CS while the latter is better on Amazon. This observation can be explained by Figure 5. On MAG-CS, the accuracy of retrieved pseudo training documents drops significantly when X becomes larger. On Amazon, the accuracy just slightly fluctuates as X increases, thus the model can perform better when using more retrieved training data. (3) If we compare "0 retrieved + X generated" with "0 retrieved + $2X$ generated", the latter is slightly better. This is because the quality of generated training data is not affected by X, as each document is sampled independently. Therefore, it is always better to have

---
[11] https://huggingface.co/transformers/model_doc/bert.html#bertforsequenceclassification

(a) MAG-CS

(b) Amazon

Figure 6: Embedding space visualization. Category name embeddings are denoted as stars, and the embeddings of selected motif instances are denoted as points with the same color as their corresponding categories.

more generated training data. (4) In general, retrieval-only variants perform better than generation-only variants.

A.5 Additional Experiments: Embedding Visualization

To reveal how categories and selected motif instances are distributed in our joint embedding space, we apply t-SNE [14] to visualize their embeddings in Figure 6. Category name embeddings (i.e., $\{e_{m_l} : l \in \mathcal{L}\}$) are denoted as stars; embeddings of top-7 selected motif instances (i.e., $\{e_m : m \in \mathcal{M}_l\}$) are denoted as points with the same color as their corresponding categories. We observe that: (1) Selected motif instances surround their category names in most cases. (2) Semantically similar categories (e.g., "data mining" and "machine learning" in MAG-CS, "clothing" and "sports" in Amazon) are embedded closer.