Open-world Multi-label Text Classification with Extremely Weak Supervision

Xintong Li[1], **Jinya Jiang**[1], **Ria Dharmani**[1]
Jayanth Srinivasa[2], **Gaowen Liu**[2], **Jingbo Shang**[1]
University of California, San Diego[1] Cisco[2]
{xil240, j9jiang, rdharmani, jshang}@ucsd.edu
{jasriniv, gaoliu}@cisco.com

Abstract

We study open-world multi-label text classification under extremely weak supervision (XWS), where the user only provides a brief description for classification objectives without any labels or ground-truth label space. Similar single-label XWS settings have been explored recently, however, these methods cannot be easily adapted for multi-label. We observe that (1) most documents have a dominant class covering the majority of content and (2) long-tail labels would appear in some documents as a dominant class. Therefore, we first utilize the user description to prompt a large language model (LLM) for dominant keyphrases of a subset of raw documents, and then construct a (initial) label space via clustering. We further apply a zero-shot multi-label classifier to locate the documents with small top predicted scores, so we can revisit their dominant keyphrases for more long-tail labels. We iterate this process to discover a comprehensive label space and construct a multi-label classifier as a novel method, X-MLClass. X-MLClass exhibits a remarkable increase in ground-truth label space coverage on various datasets, for example, a 40% improvement on the AAPD dataset over topic modeling and keyword extraction methods. Moreover, X-MLClass achieves the best end-to-end multi-label classification accuracy.

1 Introduction

Multi-label text classification (MLTC) aims to assign one or more labels to each input document in the corpus. Traditional methods (Liu et al., 2022; Xiong et al., 2021; Gera et al., 2022) require a complete list of class names, which is challenging to provide beforehand given the massive number of documents and diverse topics. This work focuses on a new problem, open-world[1] MLTC under extremely weak supervision (XWS), where the user only provides a brief description for classification objectives without any labels or ground-truth label space. Despite the considerable technical challenges posed by this task, its practical significance in real-world applications cannot be underestimated. For instance, the need for product tagging and categorization is ubiquitous in online shopping platforms. It requires identifying multiple labels for each product without access to a predefined label space, a challenge our model adeptly addresses.

The most related XWS problems are text clustering (Zhang et al., 2023; Wang et al., 2023b) and topic modeling (Grootendorst, 2022; Pham et al., 2023), where those methods are typically only capable of assigning a single label to each document. These single-label methods cannot be easily adapted for multi-label.

We observe that (1) most documents have a dominant class covering the majority of content and (2) long-tail labels would appear as the dominant class in some documents. Experiments reveal that over 90% of documents contain a dominant class, and 100% of labels appear as the dominant class in at least one document, as analyzed further in Section 4. Based on these observations, we propose a novel method, X-MLClass, to discover a pragmatic label space by iteratively adding (long-tailed) labels and construct a multi-label text classification classifier with the assistance of LLM (i.e., `llama-2-13b-chat` in our experiments), as shown in Figure 1. Our approach requires only a brief user description about the classification objective as prompt for LLM, which significantly reduces the cost of model training.

The first step in X-MLClass is to construct a high-quality label space. We start with a reasonably large subset of the raw documents. For each document, we partition it into chunks to better align with the context length of LLM and also ensure that each chunk contains a single topic, and then prompt

[1]Our "open-world" definition denotes the absence of any class information during the training and testing phase, which is more challenging than the traditional settings.

Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, pages 15084–15096

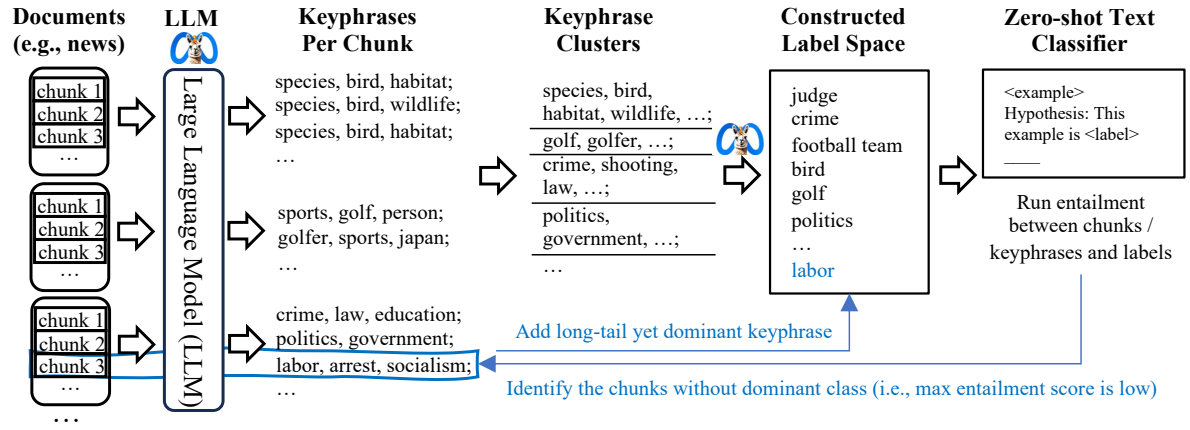

Figure 1: An overview of our X-MLClass framework. The only required supervision from the user is a brief description of the classification objective. During the first LLM prompting stage for keyphrases, X-MLClass leverages this description as a part of the prompt, so it will be helpful if the description includes some demonstrations.

the LLM to generate the most dominant keyphrases for each chunk. As previous LLM-based text clustering work has suggested (Wang et al., 2023b,a), there are very likely some semantically redundant yet lexically different keyphrases among the generated ones. We cluster these keyphrases, and within every cluster, we pull together the corresponding chunks of the keyphrases closest to the cluster center to prompt the LLM once again, generating one single label for each cluster. After eliminating labels with high similarity scores, we constitute an initial label space.

We then apply the state-of-the-art textual entailment-based classification methods (Pàmies et al., 2023) to construct a classifier that revisits the documents and identifies long-tail labels. Specifically, we query every text chunk against all the labels for the entailment score, identifying chunks with low top predicted scores that lack a dominant class. We revisit the keyphrases generated by these chunks to unveil more long-tail labels, selectively choosing keyphrases with a modest presence in the entire keyphrase set but absent in the original label space. These new keyphrases are included in the label set, and documents are reassessed with this updated set. By repeating these steps for a fixed number of iterations, the final label space contains a substantial number of long-tail labels.

Extensive experiments on 5 benchmark datasets reveal the superiority of X-MLClass outperforming all compared methods. Remarkably, compared with baselines, X-MLClass achieves a significant enhancement of 40% and 30% in ground-truth label space coverage on the AAPD and RCV1-V2 datasets. Furthermore, it achieves the highest accuracy in zero-shot MLTC, surpassing the top-ranking models on HuggingFace across all datasets.

Our contributions are summarized as:

- We attack a challenging problem, open-world MLTC with XWS, where the user only provides a brief description for classification objectives without any labels or ground-truth label space.
- We propose a novel framework X-MLClass which iteratively discovers the label space and builds an MLTC classifier.
- Compared with all traditional label generation methods, X-MLClass achieves a significantly higher coverage score along with superior end-to-end classification accuracy.

Our source code and dataset can be obtained here [2].

2 Related Work

Topic Modeling: Topic modeling has been widely adopted for discovering latent thematic structures within collections of text documents. Traditional models, such as Latent Dirichlet Allocation (LDA) (Blei et al., 2003) and Non-Negative Matrix Factorization (NMF) (Févotte and Idier, 2011) represent documents as mixtures of latent topics using bag-of-words representations. New techniques like Top2Vec (Angelov, 2020) and BERTopic (Grootendorst, 2022) build primarily on clustering embeddings, demonstrating the potential of embedding-based topic modeling approaches. Another recent method, TopicGPT (Pham et al., 2023), takes a different approach by prompting large language models for topic generation, aligning more closely with ground truth labels. However,

[2] https://github.com/Kaylee0501/X-MLClass

these existing methods typically provide a single topic for each document, which poses challenges when extending them to multi-label scenarios.

Multi-label Text Classification: Numerous approaches have been proposed to tackle the complexities of Multi-Label Text Classification (MLTC) problems. Bhatia and Jain (Bhatia et al., 2015) employ embedding-based methods, leveraging the power of embeddings to train individual classifiers for each label. Later, XML-CNN (Liu et al., 2017) uses a Convolutional Neural Network (CNN) to learn text representations, demonstrating improvements in MLTC accuracy. Recent works have started to tackle MLTC problems using a small amount of labeled data or even with no labels at all. For example, Shen et al. (2021) achieves impressive results by using only class names and taxonomies. Rios and Kavuluru (2018) train a neural architecture with both true labels and their natural language descriptor. However, these methods still require access to the ground-truth label space.

Open-world Single-label Text Classification: There has been a surge in open-world models utilizing LLM prompts to derive labels without relying on ground-truth label spaces. Notably, GOALEX (Wang et al., 2023b) generates labels for text samples based on users' specific goals, demonstrating a goal-driven approach. Another noteworthy model, CLUSTERLLM (Zhang et al., 2023), leverages API-based LLMs to guide text clustering, resulting in improved performance. The approach of intent discovery (Zhang et al., 2022), aiming to infer latent intents from a document set, has proven effective in generating label spaces. A newly introduced method, IDAS (De Raedt et al., 2023), prompts LLMs to succinctly summarize utterances, enhancing intent prediction.

3 Problem Formulation

Given an unlabeled corpus $\mathcal{D} = \{D_1, D_2, \ldots, D_n\}$, where $D_i \in \mathcal{D}$ represents a document in the collection. Our task is to (1) identify class names $\mathcal{C} = \{C_j\}_{j=1}^{K}$, where K is the *unknown* number of classes, and (2) build a text classifier $f(\cdot)$ to map any raw document D_i to its target labels $Y_i = \{y_i^j\}_{j=1}^{p}$, where y_i^j is the single label name and p is the number of target labels for D_i. The definition of "open world" denotes the absence of labeled information during training, which is more stringent than the original definition where new labels only appear in the test phase. Several existing works (Brunet et al., 2023; Zhu et al., 2023) adopt a similar "open world" definition to ours.

To the best of our knowledge, this is the first work that explores open-world multi-label text classification without the presence of a ground-truth label space. Given the challenging nature of the problem, we assume that human experts are willing to devote minimal effort, i.e., *extremely weak supervision*, typically in the form of brief classification objective descriptions.

4 Our Observations

Our two observations mentioned in Section 1 are confirmed by experiments based on 5 benchmark datasets: AAPD (Yang et al., 2018), Reuters-21578 (Debole and Sebastiani, 2005), RCV1-V2 (Lewis et al., 2004), DBPedia-298 (Lehmann et al., 2015), and Amazon-531 (McAuley and Leskovec, 2013). Specifically, we prompt a large language model (LLM) to check if any of the ground truth labels of a given document is dominant, i.e., covering more than 50% of the content; and if it exists, which one is the dominant label. [3]

We randomly sample 1000 examples from each dataset and calculate the percentage of documents with a dominant class. The dominance percentages across datasets are AAPD: 92%, RCV1-V2: 91.7%, DBPedia-298: 95.2%, Reuters-21578: 96.4%, and Amazon-531: 87.3%. Notably, Reuters dataset exhibits a higher proportion of dominant labels due to its mainly consisting of single-labeled examples. Conversely, Amazon presents a lower dominance percentage, attributed to the mix of fine-grained and coarse-grained labels, posing challenges in determining dominance. Overall, our analysis indicates that across all datasets, more than 90% of documents contain a dominant class. For our second observation, we identify labels present in less than 1% of the dataset as long-tail labels. Notably, we find that instances of these long-tail labels emerge as dominant classes in at least one document, demonstrating that 100% of the labels serve as dominant classes in some instances. These observations highlight the potential for generating labels, particularly long-tail labels, from raw documents and guide the design of our framework.

[3] The specific prompt can be found in Appendix A

5 Our X-MLClass Framework

X-MLClass consists of three key steps. First, every document is split into chunks and transformed into keyphrases by prompting an LLM to construct an initial label space. We further assign labels to each raw document D_i using a custom keyphrase-chunk zero-shot textual entailment classifier. Finally, we iteratively enhance the label space by incorporating additional long-tail labels. The framework overview is depicted in Figure 1, and the below sections provide a detailed discussion.

5.1 Initial Label Space Construction

The first step in X-MLClass is to construct a high-quality label space. To balance label coverage and the computational cost of LLM, X-MLClass is applied to a reasonably large subset of the corpus $\mathcal{D}$, denoted as $\mathcal{D}_{sub} \subset \mathcal{D}$.

Dominant Keyphrase Generation: For each document, we partition it into chunks to better align with the context length of LLM, and then prompt for the most dominant keyphrases per chunk. Specifically, each document $D_i \in \mathcal{D}_{sub}$ is segmented into chunks $\{S_i^1, S_i^2, \dots\}$, with a predefined chunk size of 50 tokens. This choice is also made to ensure each chunk primarily contains one label. To generate keyphrases for each chunk S_i^j, we employ an LLM and provide it with an instruction based on a brief user description of the classification objective. [4] The LLM then refines keyphrases p_i^j from the chunk S_i^j, serving as potential class candidates for subsequent stages of our X-MLClass model. Keyphrases generated from each chunk collectively form a set $\mathcal{P}$.

Keyphrase Clustering: As previous LLM-based text clustering work has suggested (Wang et al., 2023b,a), there are very likely some semantically redundant yet lexically different keyphrases among the generated ones, so we employ the instruction-tuned text embedding model, `instructor-large` (Su et al., 2022), to generate vector representations for all the keyphrases in $\mathcal{P}$. Traditional clustering methods face challenges in high-dimensional spaces (Aggarwal et al., 2001; Wang et al., 2020b). To address this, we apply UMAP (McInnes et al., 2018) for dimensionality reduction, effectively balancing local and global structures. Finally, we obtain the clusters using the Gaussian Mixture Model (GMM) in the projected low-dimensional space, renowned for its enhanced flexibility in capturing intricate data distributions.

Number of Clusters: The number of clusters is determined by considering both the insights of human experts regarding the magnitude of the label space and non-parametric clustering methods such as BERTopic (Grootendorst, 2022), a highly effective topic modeling method. For example, one can train BERTopic on the keyphrase set $\mathcal{P}$ to obtain the topic number K^0, serving as the hyperparameter to GMM. This approach also ensures a fair comparison with baseline methods by maintaining consistency in the number of clusters.

Redundant Keyphrase Removal: Within every cluster, we focus on the three keyphrases closest to the cluster center to synthesize one single label. Instead of directly employing the keyphrases for label space creation, we trace back to the original chunks that generated these keyphrases, as they likely contain similar content and represent the same label. Concatenating these three chunks for each cluster results in a new document. For each document, we prompt LLM with an instruction "*find one label for this document*", yielding the initial K^0 classes $\{C_j\}_{j=1}^{K^0}$. This strategy allows us to generate a single label that best represents the cluster.

This initial label space may contain redundant labels. Sentence-Transformer models (Wang et al., 2020a) are used to identify distinct pairs of classes with relatively high cosine similarity scores. The first class in each identified pair is then removed to eliminate redundancy. For those borderline similar label pairs, we prompt `GPT-4` (Achiam et al., 2023) with an instruction "*Do label pairs have similar meanings? If Yes, please output the label that we should delete*." to help us detect distinct pairs where cosine similarity scores alone are insufficient. This method proves effective in creating a robust label space $\{C_j\}_{j=1}^{K^1}$, and while human involvement can enhance the refinement process, it is not mandatory. Further details on human involvement in the label space refinement are provided in appendix C.

5.2 Textual Entailment-based Classifier

Given a label space, we build a zero-shot textual entailment-based classifier (Yin et al., 2019). Since each chunk is designed to have only one label, state-of-the-art zero-shot single-label text classification methods (Pàmies et al., 2023; Gera et al., 2022; He

[4] The specific prompt can be found in Appendix B

Dataset	# Train	# Text	# Class
AAPD	53,840	2,000	54
Reuters	7,769	3,019	90
RCV1-V2	643,531	160,883	103
DBPedia	196,665	49,167	298
Amazon	29,487	19,658	531

Table 1: Dataset statistics.

et al., 2021) are all applicable here. Specifically, we compare every text chunk against all the labels using a textual entailment model. For each chunk $s \in \mathcal{S}$ and each class name $c \in \mathcal{C}$, we derive $E_{s,c}$ representing the confidence for the chunk s entailing the hypothesis "*This example is constructed for* c", and similarly obtain $E_{p,c}$ for each keyphrase $p \in \mathcal{P}$. Subsequently, for each example in $\mathcal{S}$, we identify the label c^* with the top entailment score, denoted by $E_{s,c^*} > E_{s,c}, \forall c \neq c^*$.

Finally, we find all s and p belong to the same document D_i and group them into a new set Q. For each instance in Q, we rank the label candidates according to their entailment scores. We identify the labels that occur most frequently with the same ranking as the predicted labels for document D_i, progressing from the top-ranking to the lowest-ranking order.

5.3 Label Space Improvement

We further identify the chunks with lower top predicted scores — these chunks lack a dominant class in the initial label space. By ranking E_{s,c^*} in ascending order, we select a subset $\mathcal{S}_{sub} \subset \mathcal{S}$ with relatively lower entailment scores, suggesting potential association with tail classes not included in our label space. Since keyphrases generated by the LLM may include outliers that are too specific to their corresponding documents, we retain only keyphrases occurring more than 15 times. For each $s \in \mathcal{S}_{sub}$, we examine all keyphrases in the corresponding p. If a keyphrase is absent in the label space but occurs more than 15 times in $\mathcal{P}$, we incorporate it into the label space $\mathcal{C}$.

Additionally, we compute the frequency of each label c with the top entailment score. Labels with lower frequency are removed from the label space $\mathcal{C}$. The high-frequency labels, secured as a part of the label space, are temporarily excluded from the later label space improvement process. By iteratively training the classifier based on the updated label space, the label set gets finalized by adding more long-tail labels. In the concluding stages, all high-frequency labels are reintroduced, culminating in the formation of ultimate label space.

6 Experiments

We assess the performance of X-MLClass through two primary criteria: label space quality and zero-shot MLTC accuracy. Our evaluation involves a comparison of our model's label coverage with that of four topic modeling and three keyword extraction methods. In terms of end-to-end classification accuracy, we test our method with several top-ranking models available on HuggingFace. The subsequent section provides comprehensive details on the datasets, baseline methods, evaluation metrics, implementation specifics, and performance analysis.

6.1 Datasets

We perform experiments on five benchmark datasets for multi-label text classification across various domains: **AAPD**, **Reuters**-21578, **RCV1-V2**, **DBPedia**-298, and **Amazon**-531. Detailed information about each dataset is provided in Appendix D. Table 1 shows that the number of labels in these datasets varies from tens to hundreds. All the methods will be applied on the documents from the training set, and then evaluated on the test set.

6.2 Compared Methods

We compare our X-MLClass framework with two types of methods.

Label (Space) Generator: We select five representative *topic modeling* methods with distinct paradigms. **LDA** (Blei et al., 2003) and **NMF** (Févotte and Idier, 2011) extract topics based on word frequency within documents. **Topic2Vec** (Angelov, 2020) extends the Word2Vec model to embed topics, facilitating the exploration of semantic relationships between documents. **BERTopic** (Grootendorst, 2022) leverages BERT embeddings and the HDBSCAN clustering algorithm to identify topics. Meanwhile, **TopicGPT** (Pham et al., 2023) prompts LLMs to generate and assign topics to documents while providing quotations for verification.

The three *keyword extraction* methods include **PKE**(Boudin, 2016), **TextRank**(Mihalcea and Tarau, 2004), and **TF-IDF** (Ramos et al., 2003). PKE selects keyphrase candidates based on their confidence scores, TextRank is a graph-based ranking model that ranks keywords using a voting mechanism, and TF-IDF scores words based on their frequency in the document and inversely proportional to their frequency across documents.

Despite the above methods generating a single label per document, based on our observations, they often align with the dominant label for each document. Given their potential to cover all labels, it is reasonable to compare the label space coverage between our task and these label generators.

Zero-shot Text Classification: State-of-the-art zero-shot text classifiers typically follow textual entailment (Yin et al., 2019; Pàmies et al., 2023). Therefore, we choose three entailment models: (1) **bart-large-mnli** exclusively trained on the MNLI dataset, (2) **deberta-v3-large-all** trained on 33 datasets reformatted into the universal NLI format, and (3) **xlm-roberta-large-xnli** fine-tuned on the XNLI dataset. We apply these models using the HuggingFace Transformer pipeline, with a hypothesis template "*This example is {label}*".

6.3 Evaluation Metrics

Label Space Quality: We employ an automatic evaluation metric, **coverage**, to measure the alignment between the ground-truth (GT) label space and the predicted label space. A ground-truth label is considered "covered" if it exhibits a similarity score exceeding a predefined threshold when compared to a predicted label, or if it receives a positive evaluation from GPT-4. We compute the similarity scores using the `all-MiniLM-L6-v2` model from HuggingFace Sentence-Transformers. If the similarity score exceeds 0.75, the ground-truth label is considered "covered." For scores between 0.5 and 0.75, we prompt GPT-4 with specific instructions (see Appendix E). Labels with scores below 0.5 are not compared due to obvious dissimilarity.

The coverage score is computed as follows:

$$\text{Coverage} = \frac{1}{N}\mathbb{G}\left(\mathbb{I}\left(C^{\text{pred}}, C^{\text{GT}}\right)\right)$$

Here N is the total number of topics in the GT label set, $C^{\text{GT}}(C^{\text{pred}})$ denotes the set of ground-truth (predicted) labels. $\mathbb{I}$ is an indicator that returns 1 if the GT label is considered "covered". $\mathbb{G}$ represents the bipartite graph maximum match algorithm.

Classification Accuracy: Because of the large label space, multi-label text classification typically employs the rank-based evaluation metric precision at k, i.e., P@k. It captures the percentage of true labels among top-k score labels and is used for performance comparison. P@k can be defined as:

$$\text{P@k} = \frac{1}{N}\sum_{i=1}^{N}\frac{C_i^{r_k} \cap L_i}{\min(k, |L_i|)}$$

where L_i and C_i denote the true labels and predicted labels for document D_i, $|L_i|$ is the number of true labels for D_i, and r_k is the k-th highest predicted label. The ground-truth label is defined "covered" using the same **coverage** metric.

Model	AAPD	Reuters	RCV1-V2	DBPedia	Amazon
PKE	14.29	14.44	25.24	25.84	15.44
TextRank	18.37	11.11	11.65	14.77	12.43
TF-IDF	14.29	15.56	8.74	26.51	12.24
LDA	30.61	17.77	12.74	11.40	13.18
NMF	28.57	17.77	12.74	28.18	15.06
Top2Vec	32.65	20.00	28.43	30.87	14.50
BERTopic	20.41	20.00	10.68	33.22	16.76
TopicGPT	41.66	25.55	23.30	41.61	32.20
X-MLClass	**77.56**	**37.78**	**61.17**	**67.45**	**38.04**

Table 2: Label Space Coverage Comparison. Top2Vec and BERTopic generate topics with multiple keywords. The predicted label is determined by selecting the top-ranking keyword based on each model's setting.

6.4 Implementation Details

We use `llama-2-13b-chat` LLM for X-MLClass implementation. The chunk size is uniformly set to 50 across all datasets, ensuring a consistent approach. For TopicGPT, we prompt the LLM to generate a number of topics comparable to our method. For Top2Vec and BERTopic, which employ HDBSCAN as a clustering method, specifying an exact number of topics is not feasible. However, to maintain consistency, we ensure that these methods generate clusters neither exceeding nor falling below 10 in comparison to our label number. For the remaining methods, we modify their configurations to align the number of topics with our approach. By implementing these techniques, we ensure that all baseline methods produce similar labels to our proposed method, enabling a fair comparison.

We employ a dynamic addition strategy to determine the size of $\mathcal{D}_{sub}$ for each dataset. Starting with the empty set, we iteratively add 1000 examples at a time, generating the initial label space until no additional labels are generated. Consequently, $\mathcal{D}_{sub}$ contains 3000 documents for AAPD, Reuters-21578, and RCV1-V2; 8000 documents for DBPedia-298; and 14000 documents for Amazon-531. In the label space improvement phase, by ranking the top entailment scores in ascending order, we select a subset of chunks S with comparatively lower entailment scores. To ensure

Classifier	Method	AAPD		Reuter		RCV1-V2		DBPedia		Amazon	
		P@1	P@3	P@1	P@3	P@1	P@3	P@1	P@3	P@1	P@3
`bart-large-mnli`	Vanilla	0.2030	0.2585	0.0940	0.2547	0.4220	0.3760	0.6420	0.3657	0.5080	0.3930
	X-MLClass	**0.3323**	**0.3735**	**0.1630**	**0.3617**	**0.4530**	**0.3808**	**0.6890**	**0.4000**	**0.6620**	**0.4608**
`deberta-v3-large-all`	Vanilla	**0.3860**	0.3700	0.1290	0.3937	0.4660	0.4152	**0.6370**	0.3816	0.5970	0.4068
	X-MLClass	0.3573	**0.4009**	**0.1320**	**0.4150**	**0.4820**	**0.4307**	0.6350	**0.3953**	**0.6200**	**0.4650**
`xlm-roberta-large-xnli`	Vanilla	0.1860	0.2330	0.1450	0.3477	0.3270	0.3053	0.6500	0.3673	0.5060	0.3860
	X-MLClass	**0.2713**	**0.3467**	**0.2040**	**0.3968**	**0.4040**	**0.3383**	**0.6610**	**0.3910**	**0.5840**	**0.4547**

Table 3: Zero-Shot Multi-Label Text Classification Accuracy Comparison: Vanilla method represents baseline classifier trained on raw documents. We compare the Vanilla performance with X-MLClass performance.

consistency with $\mathcal{D}_{sub}$, we control the size of S proportionally. For AAPD, Reuters-21578, and RCV1-V2 datasets, we select the top 500 examples. For DBPedia-298, the subset size is set at 1,000, and for Amazon-531, we choose 1,500 examples.

We consider labels existing in less than 1% of the dataset as long-tail labels. Therefore, we select all keyphrases that occur in less than 1% of $\mathcal{P}$. By calculating the semantic similarity score between these selected keyphrases and all labels in C, we identify the highest score for each keyphrase and use the median of these scores as our threshold γ. Only labels with a semantic similarity score lower than γ compared to existing labels are added to the new label space. This methodology ensures a refined and relevant augmentation of the label space. Details on the time and computational resources required for model training and prediction are provided in Appendix F.

6.5 Label Space Coverage Results

We present the coverage of the predicted label space in comparison to topic modeling and keyword extraction baselines, as detailed in Table 2. Our method consistently outperforms all baseline approaches. Specifically, for the AAPD, RCV1-V2, and DBPedia-298 datasets, we achieve over 60% coverage of the ground-truth label space, showcasing a noteworthy increase of up to 40% compared to the baseline methods. These results demonstrate that our method excels at predicting labels that align closely with the ground-truth label space. For example, our model can generate long-tail labels, a task that is particularly challenging for all baselines. However, our model exhibits comparatively lower performance on the Reuters-21578 and Amazon-531 datasets. Regarding Reuters-21578, the lower performance is due to a higher proportion of long-tail labels and the use of abbreviations in ground-truth labels. For Amazon dataset, the initially generated label space by X-MLClass is only one-third of the ground-truth size. Even after adding more labels in the improvement stage, the predicted label space is still less than half of the ground-truth space, leading to lower coverage scores.

Dataset	Initial	Improvement	Δ
AAPD	44.90	77.56	+32.66%
Reuters-21578	24.44	37.78	+13.34%
RCV1-V2	49.51	61.17	+11.66%
DBPedia-298	55.70	67.45	+11.75%
Amazon-531	23.35	38.04	+14.69%

Table 4: Label Coverage Score Improvement Results.

We also use `Meta-Llama-3-8B-Instruct` for model implementation and test on AAPD and RCV1-V2 datasets. The coverage scores were 75.51% and 60.19%, respectively, similar to the results obtained using `llama-2-13b-chat`.

6.6 Zero-shot Text Classification Accuracy

We present the comprehensive zero-shot performance across all methods in Table 3. The results unequivocally demonstrate that our framework consistently outperforms nearly all baseline models. Notably, the P@3 scores of X-MLClass surpass those of the baseline methods across all datasets. This observation implies that training the zero-shot classifier for both the keyphrases set and the chunk set, followed by merging the results, enhances the multi-label performance. Specifically, our chunk-splitting procedure increases the likelihood of finding the less dominant labels for each document, as these labels may become dominant in smaller chunks. Similarly, our approach improves the accurate prediction of tail labels by the classifier, contributing to the overall MLTC performance.

6.7 Label Space Coverage Improvement

Table 4 shows that iteratively updating the label space leads to an enhancement in label coverage across all datasets. Figure 2 visually represents

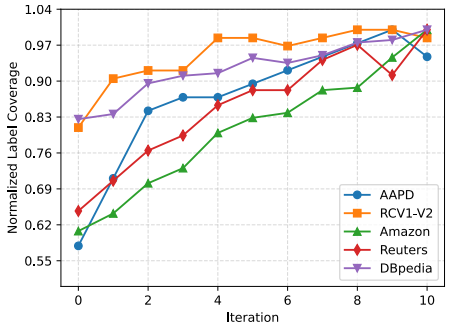

Figure 2: Coverage Improvement across Iterations.

the normalized incremental coverage during each iteration across datasets. Notably, the improvement is more pronounced for datasets with smaller initial label space sizes or lower initial coverage scores. This finding aligns with expectations, as AAPD exhibits significantly smaller label space sizes compared to all other datasets, rendering label space improvement easier. For the Reuters-21578 and Amazon-531 datasets, the initial coverage score is low, meaning that many matching labels are not initially included in the label space. This results in a higher potential for improvement by adding these labels. Additionally, the criteria for adding new labels must align with all existing ones in the generated label space, presenting a greater challenge in expanding larger label spaces like DBPedia-298.

6.8 Label Coverage with Human Involvement/Evaluation

We can seek expert assistance for refinement when generating the initial label space, especially for borderline similar label pairs. Human judgment helps determine whether to keep both labels, typically requiring inspection of only about 30 pairs on average. Table 5 shows that the coverage score of initial label space across all datasets is lower without human involvement, as expected.

Our model also encounters challenges in generating labels exactly matching the ground-truth label space. For instance, within Reuters-298, certain ground-truth labels are abbreviations, while our model generates the full-word version, leading to a lower semantic similarity score than the actual score. As shown in Table 6, the ground-truth label "acq" corresponds to our predicted label "acquisitions," possessing identical meanings, yet their semantic similarity score falls below 40%. To address this issue, we use GPT-4 to expand all abbreviations into their full-form version. This adjustment increased the label coverage to 35.56%. In the Amazon-531 dataset, many ground-truth labels

Dataset	w/o Human	w/ Human	Δ
AAPD	40.81	44.90	+4.09%
Reuters-21578	18.89	24.44	+5.55%
RCV1-V2	45.63	49.51	+3.88%
DBPedia-298	46.31	55.70	+9.39%
Amazon-531	21.28	23.35	+2.07%

Table 5: Initial Coverage w/wo Human Involvement.

Ground-truth	Predicted Label
acq	acquisitions
money-fx	monetary policy
earn	earnings
plug_play_video_games	gaming_electronics
electrical_safety	electronics_troubleshooting
teether	baby_dental_care

Table 6: Matching pairs between the ground-truth labels and the predicted labels through human evaluation.

consist of phrases, which complicates the coverage evaluation. Achieving high scores requires precise matches, but predicting similar-meaning phrases with different words is common, resulting in lower scores. As evident in Table 6, "electrical_safety" and "electronics_troubleshooting" are identical labels, but their semantic similarity scores are lower, treated as distinct labels in our setting. Expert evaluation can assist in such cases.

Considering these factors, the actual coverage score of our predicted label space compared to the ground-truth label space is likely higher than the presented result in Table 2.

6.9 Label coverage performance with varying chunk sizes

We conduct a literature review to explore the relationship between the number of words in a document and the number of ground-truth labels it contains across several multi-label datasets. By dividing the word count by the number of labels, we determine that a chunk size of 50 tokens would generally contain one label. To further evaluate our model's performance, we test a chunk size of 250 tokens. Aside from a few very long documents, this larger chunk size typically ensures that most documents remain within a single chunk. This setup allows us to compare the results of generating keyphrases from entire documents versus smaller chunks, each containing primarily one label. For the AAPD and RCV1 datasets, label coverage with a chunk size of 250 is 70.83% and 58.25%, respectively, which is slightly lower than the 50-token chunk size. These findings suggest that ensuring each chunk primarily contains one label improves

the generated label space quality.

6.10 Ablation Study for Amazon-531 Dataset

The label space for the Amazon-531 dataset is significantly larger than that of the other datasets. To address this discrepancy and enhance label coverage, we increase the number of iterations to add more long-tailed labels. Using the same number of iterations as for the other datasets would result in a final label space only half the size of the ground-truth label space. As depicted in Figure 3, increasing the number of iterations facilitates the addition of more labels to the predicted label space, leading to an improvement in the coverage score.

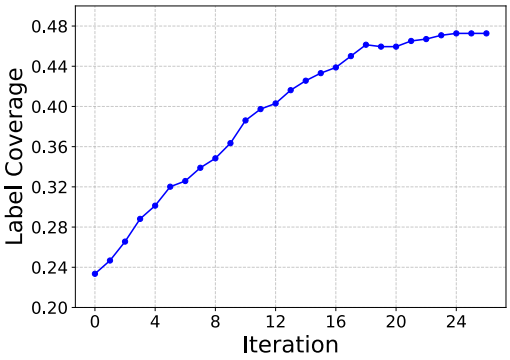

Figure 3: Improvement of Label Coverage for Amazon-531 by increasing the number of iterations.

7 Conclusion and Future Work

We attack a novel and challenging problem, open-world MLTC with extremely weak supervision. In this scenario, user only provides a brief description for classification objectives without any labels or ground-truth label space. Our LLM-based framework, X-MLClass, is designed to overcome this challenge by discovering a practical label space and constructing an MLTC classifier for label prediction. Notably, it excels in identifying long-tail labels, arguably the most challenging aspect in MLTC problems. Our experiment results show that X-MLClass surpasses baselines in terms of ground-truth label coverage and exhibits higher zero-shot text classification performance compared to top-ranking models.

Despite our model's success in generating some tail labels, a considerable number of tail labels remain undiscovered. Future work should focus on refining our approach to capture more long-tail labels. Subsequent studies could explore methodologies tailored for datasets featuring significantly larger label spaces, contributing to the broader applicability of our model.

Limitations

Our work aims to discover the label space from extensive input text documents and then construct a multi-label text classifier. The most formidable challenge in this problem setting revolves around label space construction — how can we discover the labels, especially the long-tail ones? Therefore, our primary focus is on developing a novel method to address this challenge; we didn't propose any new zero-shot multi-label text classifier, since it is beyond the scope of this paper. Given that our proposed X-MLClass starts with a subset of documents, its efficacy may be limited for extremely long-tail labels (e.g., those occurring less frequently than 0.0001% of the documents). Alternatively, a considerably large subset would be required, potentially incurring significant computational costs from LLM. While our evaluation includes a diverse set of datasets, there is potential for further extension to more challenging datasets with an exceptionally large label space (e.g., over 1000 different labels are expected).

Ethical Considerations

This paper uses open-source datasets and models for training and evaluation, which is reproducible. We will also release the code upon acceptance. It is important to note that the keyphrases generated by LLMs may vary with each run, potentially leading to minor differences in results compared to those presented. Additionally, our evaluation toolkit uses OpenAI models, which may impact reproducibility.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Charu C Aggarwal, Alexander Hinneburg, and Daniel A Keim. 2001. On the surprising behavior of distance metrics in high dimensional space. In *Database Theory—ICDT 2001: 8th International Conference London, UK, January 4–6, 2001 Proceedings 8*, pages 420–434. Springer.

Dimo Angelov. 2020. Top2vec: Distributed representations of topics. *arXiv preprint arXiv:2008.09470*.

Kush Bhatia, Himanshu Jain, Purushottam Kar, Prateek Jain, and Manik Varma. 2015. Locally non-linear embeddings for extreme multi-label learning. *arXiv preprint arXiv:1507.02743*.

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

Florian Boudin. 2016. Pke: an open source python-based keyphrase extraction toolkit. In *Proceedings of COLING 2016, the 26th international conference on computational linguistics: system demonstrations*, pages 69–73.

Marc-Etienne Brunet, Ashton Anderson, and Richard Zemel. 2023. Icl markup: Structuring in-context learning using soft-token tags. *arXiv preprint arXiv:2312.07405*.

Maarten De Raedt, Fréderic Godin, Thomas Demeester, and Chris Develder. 2023. Idas: Intent discovery with abstractive summarization. *arXiv preprint arXiv:2305.19783*.

Franca Debole and Fabrizio Sebastiani. 2005. An analysis of the relative hardness of reuters-21578 subsets. *Journal of the American Society for Information Science and technology*, 56(6):584–596.

Cédric Févotte and Jérôme Idier. 2011. Algorithms for nonnegative matrix factorization with the β-divergence. *Neural computation*, 23(9):2421–2456.

Ariel Gera, Alon Halfon, Eyal Shnarch, Yotam Perlitz, Liat Ein-Dor, and Noam Slonim. 2022. Zero-shot text classification with self-training. *arXiv preprint arXiv:2210.17541*.

Maarten Grootendorst. 2022. Bertopic: Neural topic modeling with a class-based tf-idf procedure. *arXiv preprint arXiv:2203.05794*.

Pengcheng He, Jianfeng Gao, and Weizhu Chen. 2021. Debertav3: Improving deberta using electra-style pretraining with gradient-disentangled embedding sharing.

Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick Van Kleef, Sören Auer, et al. 2015. Dbpedia–a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic web*, 6(2):167–195.

David D Lewis, Yiming Yang, Tony Russell-Rose, and Fan Li. 2004. Rcv1: A new benchmark collection for text categorization research. *Journal of machine learning research*, 5(Apr):361–397.

Jingzhou Liu, Wei-Cheng Chang, Yuexin Wu, and Yiming Yang. 2017. Deep learning for extreme multi-label text classification. In *Proceedings of the 40th international ACM SIGIR conference on research and development in information retrieval*, pages 115–124.

Ziwen Liu, Josep Grau-Bove, and Scott Allan Orr. 2022. Bert-flow-vae: A weakly-supervised model for multi-label text classification.

Julian McAuley and Jure Leskovec. 2013. Hidden factors and hidden topics: understanding rating dimensions with review text. *Proceedings of the 7th ACM conference on Recommender systems*.

Leland McInnes, John Healy, and James Melville. 2018. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*.

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.

Marc Pàmies, Joan Llop, Francesco Multari, Nicolau Duran-Silva, César Parra-Rojas, Aitor González-Agirre, Francesco Alessandro Massucci, and Marta Villegas. 2023. A weakly supervised textual entailment approach to zero-shot text classification. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 286–296.

Chau Minh Pham, Alexander Hoyle, Simeng Sun, and Mohit Iyyer. 2023. Topicgpt: A prompt-based topic modeling framework. *arXiv preprint arXiv:2311.01449*.

Juan Ramos et al. 2003. Using tf-idf to determine word relevance in document queries. In *Proceedings of the first instructional conference on machine learning*, volume 242, pages 29–48. Citeseer.

Anthony Rios and Ramakanth Kavuluru. 2018. Few-shot and zero-shot multi-label learning for structured label spaces. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing. Conference on Empirical Methods in Natural Language Processing*, volume 2018, page 3132. NIH Public Access.

Jiaming Shen, Wenda Qiu, Yu Meng, Jingbo Shang, Xiang Ren, and Jiawei Han. 2021. Taxoclass: Hierarchical multi-label text classification using only class names. In *NAAC'21: Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies,{NAACL-HLT} 2021*, volume 2021.

Hongjin Su, Weijia Shi, Jungo Kasai, Yizhong Wang, Yushi Hu, Mari Ostendorf, Wen-tau Yih, Noah A Smith, Luke Zettlemoyer, and Tao Yu. 2022. One embedder, any task: Instruction-finetuned text embeddings. *arXiv preprint arXiv:2212.09741*.

Tianle Wang, Zihan Wang, Weitang Liu, and Jingbo Shang. 2023a. Wot-class: Weakly supervised open-world text classification. *arXiv preprint arXiv:2305.12401*.

Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. 2020a. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. *Advances in Neural Information Processing Systems*, 33:5776–5788.

Zihan Wang, Dheeraj Mekala, and Jingbo Shang. 2020b. X-class: Text classification with extremely weak supervision. *arXiv preprint arXiv:2010.12794*.

Zihan Wang, Jingbo Shang, and Ruiqi Zhong. 2023b. Goal-driven explainable clustering via language descriptions. *arXiv preprint arXiv:2305.13749*.

Yuanhao Xiong, Wei-Cheng Chang, Cho-Jui Hsieh, Hsiang-Fu Yu, and Inderjit Dhillon. 2021. Extreme zero-shot learning for extreme text classification. *arXiv preprint arXiv:2112.08652*.

Pengcheng Yang, Xu Sun, Wei Li, Shuming Ma, Wei Wu, and Houfeng Wang. 2018. Sgm: sequence generation model for multi-label classification. *arXiv preprint arXiv:1806.04822*.

Wenpeng Yin, Jamaal Hay, and Dan Roth. 2019. Benchmarking zero-shot text classification: Datasets, evaluation and entailment approach. *arXiv preprint arXiv:1909.00161*.

Yuwei Zhang, Zihan Wang, and Jingbo Shang. 2023. ClusterLLM: Large language models as a guide for text clustering. pages 13903–13920. Association for Computational Linguistics.

Yuwei Zhang, Haode Zhang, Li-Ming Zhan, Xiao-Ming Wu, and Albert Lam. 2022. New intent discovery with pre-training and contrastive learning. *arXiv preprint arXiv:2205.12914*.

Xueling Zhu, Jiuxin Cao, Jian Liu, Dongqi Tang, Furong Xu, Weijia Liu, Jiawei Ge, Bo Liu, Qingpei Guo, and Tianyi Zhang. 2023. Text as image: Learning transferable adapter for multi-label classification. *ArXiv*, abs/2312.04160.

A Prompt Templates for Dominant Label

Code 1 is the prompt we use to find the dominant label for the selected document.

B Prompt Templates for Generating Keyphrases

Code 2 provides an example of the prompt used to generate keyphrases for a selected chunk of the Amazon-531 dataset. Users can help us define the objective with examples. For example, the coarse-grained objectives look like "games" and "animals", while the corresponding fine-grained objectives are "trading_card_games" and "reptiles".

C Label Space refinement with human involvement

Human experts play a crucial role in refining the label space generated by LLM. For instance, when the cosine similarity score between two labels falls between 0.50 and 0.75, indicating a certain degree of semantic similarity, human intervention is preferred to determine whether these labels are synonymous. Synonyms need to be identified and treated accordingly, with one of them being removed from the label space. To specify, we show them with an instruction "*Do label pairs have similar meanings in the text classification problem? If Yes, please output the label that we should delete*." However, there is also the case that these two labels may represent concepts from different scopes; for example, "health_care" and "health_personal_care." In such instances, human judgment is necessary to detect and treat them as separate labels.

Furthermore, some predicted labels may contain multiple meanings, necessitating human intervention to split them into distinct labels. For instance, if a predicted label is "computer vision and machine learning," it is evident that the label should be divided into two separate labels. These judgments require human expertise for accurate and context-aware decisions.

D Datasets Detailed Information

- **AAPD** (Yang et al., 2018) contains computer science papers. The labels are research topics.
- **Reuters**-21578 (Debole and Sebastiani, 2005) is a collection of news articles from the Reuters financial newswire service in 1987. The labels are the news topics.
- **RCV1-V2** (Lewis et al., 2004) contains categorized newswire articles by Reuters Ltd. The labels are the news topics.
- **DBPedia**-298 (Lehmann et al., 2015) are extracted from Wikipedia articles. The labels are the article categories.
- **Amazon**-531 (McAuley and Leskovec, 2013) encompasses product reviews and associated metadata. The labels are the product tags.

E GPT Instructions for Verifying Matching Pairs

Code 3 is the instruction we used to verify matching pairs.

F Time and Computational Resources Required for Model Training and Prediction

The number of training examples we select varies based on the scope of each dataset. On average, we prompt LLM on approximately 18,000 chunks for each dataset. We use a single 80GB A100 for LLM prompting, and the process consumes less than 35GB of GPU memory. Each prompting only takes less than 1 second to execute. Thus, on average, each dataset requires less than 4 hours for LLM prompting, making it a manageable cost.

During the text classification and label space improvement step, the majority of the computation cost arises from calculating the textual entailment score for each chunk. However, the entailment model is lightweight, occupying only around 3GB of GPU memory. When dealing with large label spaces, nearest-neighbor techniques can be applied to trim down label candidates, reducing the time required for calculating entailment scores. On average, we require approximately 2 hours to complete one iteration. However, users have the flexibility to choose the number of iterations needed to update the label space. Given that our model already outperforms the baseline with the initial label space, only a few iterations are necessary, making both time and cost manageable.

```
sys_prompt = "You are a poetic assistant, skilled in explaining complex programming concepts with creative flair."
user_prompt = "
    Which label in the label space {true_label_array} is the dominant label that covers more than 50% of the below content?
    {documents}

    Please output the dominant label only if exist or output \'NO\' if there are no dominant labels.
    "
prompt = "{sys_prompt} \n {user_prompt}"
```

Code 1: Prompt to find the dominant label for the selected document

```
#Amazon-531 keyphrases generation template
sys_prompt = "<s>[INST] <<SYS>>
    You are a helpful, respectful and honest assistant for labeling topics.
    <</SYS>>."
example_prompt = "
    The following is a customer review of a product bought from Amazon.
    [documents]
    Based on the topic information mentioned above, the coarse-grained keyphrases are formatted as {user classification
      objective}, while the fine-grained keyphrases are formatted as {user classification objective}.
    "
main_prompt = "
    [INST]
    Based on the information about the topic above, please find two coarse-grained and two fine-grained keyphrases for the
      example.
    [DOCUMENTS]
    Please only return the keyphrases in one line using the format below:
    [/INST] [keyphrase] and [/keyphrase].
    "
prompt = "{sys_prompt} \n {example_prompt} \n {main_prompt}"
```

Code 2: Prompt templates to generate keyphrases

```
sys_prompt = "You are an expert in text classification, with specialized skills in discerning matching pairs for labels."
user_prompt = "
    Given that we have established matching pairs such as
    "\'Machine learning\' and \'artificial intelligence\'",
    "\'Computational Geometry\' and \'Algebraic Geometry\'",
    "\'Physics and Society\' and \'Physics\'",       //optional
    "\'teether\' and \'baby_dental_care\'",          //optional
    when using util.dot_score to measure semantic similarity between tokens, would you consider {ground_truth} and {prediction}
      as a matching pair in a text classification problem?

    Please respond with Yes or No.
    "
prompt = "{sys_prompt} \n {user_prompt}"
```

Code 3: Prompt templates for verifying match pairs