# PINA: Leveraging Side Information in eXtreme Multi-label Classification via Predicted Instance Neighborhood Aggregation

**Eli Chien** [1 *] **Jiong Zhang** [2] **Cho-Jui Hsieh** [3] **Jyun-Yu Jiang** [2] **Wei-Cheng Chang** [2] **Olgica Milenkovic** [1] **Hsiang-Fu Yu** [2]

## Abstract

The eXtreme Multi-label Classification (XMC) problem seeks to find relevant labels from an exceptionally large label space. Most of the existing XMC learners focus on the extraction of semantic features from input query text. However, conventional XMC studies usually neglect the side information of instances and labels, which can be of use in many real-world applications such as recommendation systems and e-commerce product search. We propose Predicted Instance Neighborhood Aggregation (PINA), a data enhancement method for the general XMC problem that leverages beneficial side information. Unlike most existing XMC frameworks that treat labels and input instances as featureless indicators and independent entries, PINA extracts information from the label metadata and the correlations among training instances. Extensive experimental results demonstrate the consistent gain of PINA on various XMC tasks compared to the state-of-the-art methods: PINA offers a gain in accuracy compared to standard XR-Transformers on five public benchmark datasets. Moreover, PINA achieves a $\sim$ 5% gain in accuracy on the largest dataset LF-AmazonTitles-1.3M. Our implementation is publicly available https://github.com/amzn/pecos/tree/mainline/examples/pina.

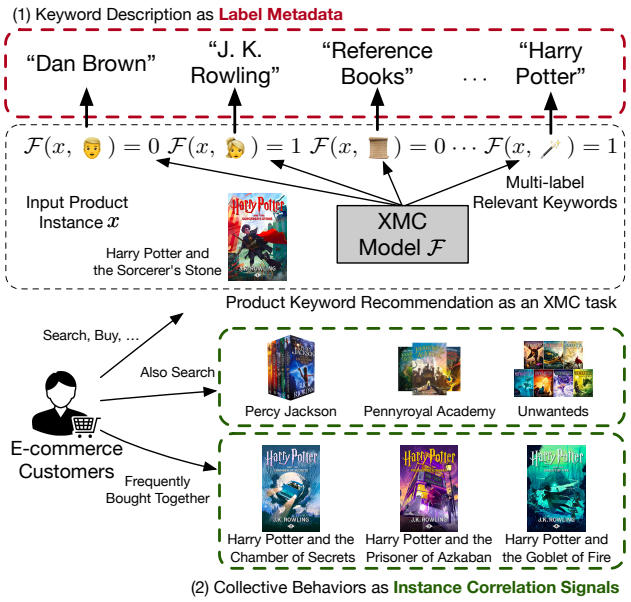


*Figure 1.* Illustration of two types of side information, including (1) label metadata and (2) instance correlation signals, based on an example XMC task that recommends relevant keywords (labels) for input products (instances) in E-commerce. Specifically, the text descriptions of keywords serve as label metadata while customer behaviors collectively provide instance correlation signals.

## 1. Introduction

Many real-world applications, such as e-commerce dynamic search advertising (Prabhu & Varma, 2014; Prabhu et al., 2018), semantic matching (Chang et al., 2021), and open-domain question answering (Chang et al., 2020a; Lee et al., 2019), can be formulated as eXtreme Multi-label Classification (XMC) problems. Given a text input, XMC aims to predict relevant labels from a label collection of extremely large size $L$. The scale of $L$, which is often of the order of millions, makes designing accurate and efficient XMC models arduous.

Despite the progress in tackling the XMC problem, most XMC solvers still only take instance features as inputs for

*This work was done during Eli Chien's internship at Amazon, USA. [1]Department of ECE, University of Illinois Urbana-Champaign, USA [2]Amazon, USA [3]Department of CS, University of California, Los Angeles, USA. Correspondence to: Eli Chien <ichien3@illinois.edu>.

*Proceedings of the $40^{th}$ International Conference on Machine Learning*, Honolulu, Hawaii, USA. PMLR 202, 2023. 

prediction. Even though side information, such as label metadata (i.e., label text) and instance correlation signals, may be highly beneficial for the learning task, it cannot be leveraged directly. Taking product keyword recommendation as an example, Figure 1 illustrates two types of side information. For label metadata, the standard XMC formulation treats labels as identifiers and ignores their text descriptions (You et al., 2019; Babbar & Schölkopf, 2019). More precisely, while recent XMC solutions such as XR-Linear (Yu et al., 2022) and XR-Transformer (Zhang et al., 2021a) have exploited the correlations among labels to generate label partitions or hierarchical label trees, they do not use label text features. Instead, they construct label embeddings via aggregation of positive instance features. Recent works (Mittal et al., 2021a; Dahiya et al., 2021a) have also demonstrated that using label text features is beneficial for the XMC problem, leading to state-of-the-art results on datasets containing label text information. Moreover, instance correlation signals based on the collective behaviors of customers are also ignored in the standard XMC formulation. For example, the co-purchase signal from Amazon is now used as a benchmark graph dataset for node classification problems (Chiang et al., 2019; Hu et al., 2020). Beyond e-commerce, the idea of leveraging side information is universal and can be applied to XMC tasks in diverse fields, such as disease descriptions and cross-disease statistics in medical diagnosis (Almagro et al., 2020). Hence, it is of critical importance and expected to be widely impactful to enable side information inclusion into XMC models and thereby enhance prediction quality.

In the recent graph learning literature, Chien et al. 2021a have bridged the gap between XMC and neighborhood prediction. Intuitively, the XMC label matrix can be described as a biadjacency matrix of a bipartite graph connecting instances and labels. As shown in Figure 3, the XMC task leads to the problem of predicting the neighborhood of each instance, which is termed the neighborhood prediction task (Chien et al., 2021a). This work clearly illustrates the point that graph learning techniques can be useful in addressing XMC tasks. One standard operation to enhance the performance of graph learning methods is graph convolution (Kipf & Welling, 2017), or message passing (Gilmer et al., 2017). The idea is to aggregate the neighborhood features, which implicitly encode the graph topological information. The graph convolution operation has by now been successfully used in various graph learning methods, including generalized PageRank (Li et al., 2019), Graph Neural Networks (GNNs) (Hamilton et al., 2017; Veličković et al., 2018; Chien et al., 2020; 2021a) and hypergraph learning (Chien et al., 2019; 2021b;c). This work asserts that aggregating neighborhood features can also be beneficial for XMC.

Motivated by the connection between XMC and neighborhood prediction, we propose Predicted Instance Neighborhood Aggregation, PINA, to allow XMC methods such as XR-Transformers to leverage the aforementioned side information in a data enhancement manner. Our contributions can be summarized as follows:

1. We introduce PINA, a data enhancement method that allows XMC models to leverage two types of side information, label metadata and instance correlation signal in a unified manner.

2. On five public benchmark datasets where the side information is label metadata, we compare PINA with the state-of-the-art XMC model, XR-Transformer. PINA consistently beats classical XR-Transformers and achieves roughly a $5\%$ gain in accuracy on the largest dataset LF-AmazonTitles-1.3M. Moreover, XR-Transformer enhanced by the PINA technique is shown to outperform all previous published results.

3. We test PINA on the industrial scale proprietary dataset containing millions of instances and labels, where the side information is of the form of instance correlation signals. PINA provides a $3.5\%$ relative improvement in accuracy compared to the baseline XMC method.

In summary, our approach consistently improves XR-Transformer on public benchmark datasets (Bhatia et al., 2016) when the side information is label text. We achieve new state-of-the-art results on the public benchmark datasets with a significant gain, and also observe performance gains brought forth by PINA on proprietary datasets when the side information is in the form of instance correlation signals.

## 2. Related Work

### 2.1. Extreme multi-label classification

Pioneering works on XMC adopt static input text representations and focus on the handling of extremely large label space. Treating labels as being binary, OVA architectures such as DiSMEC (Babbar & Schölkopf, 2017) and PPDSparse (Yen et al., 2017) require carefully designed parallel training algorithms to handle an enormously large number of labels. Even though these methods encourage model sparsity through weight truncation, the linear inference time with respect to the output space would still make them impractical to handle millions of labels. To address this issue, some works have focused on shortlisting candidate labels to achieve sub-linear training and inference complexity. One line of study focuses on partitioning label spaces. Tree-based methods (Choromanska & Langford, 2015; Daumé III et al., 2017) divide the label space recursively into hierarchical label trees and therefore come with logarithmic inference times. More recent works such as Parabel (Prabhu et al., 2018), Xtext (Wydmuch et al., 2018), Bonsai (Khandagale

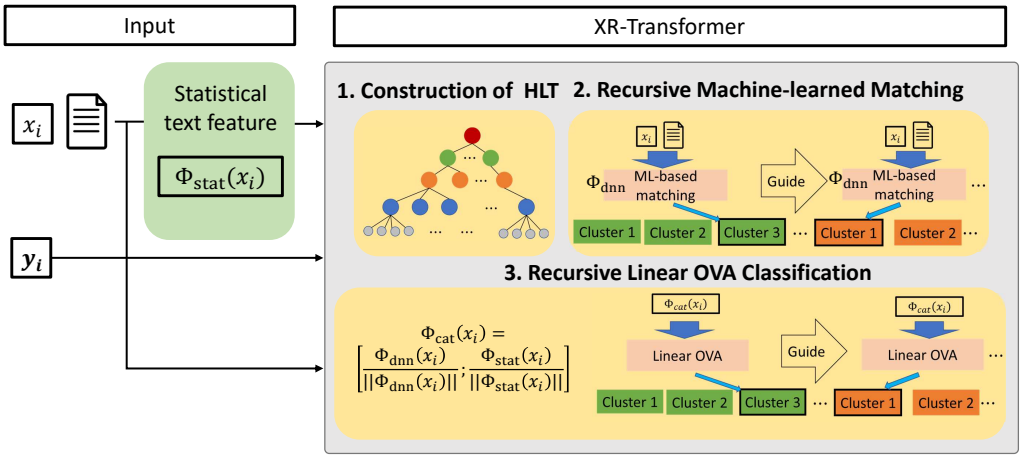


*Figure 2.* Illustration of the simplified XR-Transformer model. First, the model uses statistical text features (i.e. BoW or TF-IDF) and training labels to build the hierarchical label tree (HLT). Note that each layer of the HLT itself represents an XMC problem. Second, it trains a transformer $\Phi_{\text{dnn}}$ from the root to the leaves in a recursive manner. Third, it concatenates the statistical text feature $\Phi_{\text{stat}}(x)$ and transformer feature $\Phi_{\text{dnn}}(x)$ for learning linear one-versus-all (OVA) classifiers recursively.

et al., 2020), NapkinXC (Jasinska-Kobus et al., 2020; 2021) and XR-Linear (Yu et al., 2022) use the tree-partitioning architecture. Approximate nearest neighbor search (ANNS) is another method that adopts shortlisting the candidate labels in XMC. Through pre-built label indexing (Malkov, 2018) or product quantization (Johnson et al., 2019), ANNS can greatly reduce the search computation with relatively small reduction in recall compared with exact KNN search. Methods like AnnexML (Tagami, 2017), SLICE (Jain et al., 2019) leverage these methods to shortlist candidate label set for any given query.

### 2.2. Deep learning based methods

Recent works on deep learning based XMC models adopt different neural network architectures to extract semantic features and have demonstrated better performance than methods using statistical features only, such as bag-of-words (BoW) and Term Frequency-Inverse Document Frequency (TF-IDF). Methods that use shallow networks such as XML-CNN (Liu et al., 2017) and AttentionXML (You et al., 2019) employ CNN and BiLSTM to directly extract semantic representation from input text. On the other hand, token embedding based methods (Medini et al., 2019; Dahiya et al., 2021b; Mittal et al., 2021a; Saini et al., 2021; Mittal et al., 2021b) use shallow networks to combine pre-trained token embeddings into input sequence representations. Despite having limited capacity to capture semantic meanings, token embedding based methods still offer good performance on short-text applications (search queries, product titles, and document keywords).

With the development of Transformer models (Devlin et al., 2019; Liu et al., 2019; Yang et al., 2019), new state of the art results have been established on XMC benchmarks through fine-tuning the Transformer encoders on the downstream XMC tasks (Chang et al., 2020c; Ye et al., 2020; Jiang et al., 2021). X-Transformer (Chang et al., 2020c) and LightXML (Jiang et al., 2021) fine-tune the transformer encoders on a simplified XMC problem, where each new label is induced by a cluster of the original labels. XR-Transformer (Zhang et al., 2021a) adopts the tree-based label partitioning approach and fine-tunes the transformer encoder on multi-resolution objectives.

### 2.3. XMC with label features

While most traditional XMC architectures treat labels as featureless identifiers, a recent study shows that taking label textural descriptions into consideration enhances the performance of XMC models (Dahiya et al., 2021b). Following this line of thought, methods such as GalaXC (Saini et al., 2021), ECLARE (Mittal et al., 2021b) and SiameseXML (Dahiya et al., 2021a) were put forward. While these methods obtained reasonable performance improvement by using label text especially when input texts are short, most of them make the assumption that the instance label bipartite graph is homophilic by using bi-encoders for candidate set retrieval (i.e. similar nodes are likely to have edges). While this is true for most XMC benchmark datasets, it does not hold in many real-world applications. For instance, complementary product recommendations in e-commerce would prefer to recommend accessories to a

user who just bought a smartphone rather than yet another smartphone. Also, none of these works consider the instance correlation signals as our work.

## 3. Preliminaries

Assume that we are given a training set $\{x_i, \mathbf{y}_i\}_{i=1}^N$ where $x_i \in \mathcal{D}$ is the $i^{th}$ input instance text feature and $\mathbf{y}_i \in \{0,1\}^L$ is the one hot label vector with $y_{i,l} = 1$ indicating that label $l$ is relevant to instance $i$. The standard goal of XMC is to learn a function $f : \mathcal{D} \times [L] \mapsto \mathbb{R}$, such that $f(x, l)$ indicates the "mutual relevance" between $x$ and $l$. The standard way to compute this relevance is to use the one-versus-all (OVA) strategy:

$$f(x, l) = \mathbf{w}_l^T \Phi(x);\ l \in [L], \tag{1}$$

where $\mathbf{W} = [\mathbf{w}_1, \ldots, \mathbf{w}_L] \in \mathbb{R}^{d \times L}$ are learnable weight vectors and $\Phi : \mathcal{D} \mapsto \mathbb{R}^d$ is the text vectorizer. The function $\Phi(\cdot)$ can be obtained by either statistical methods such as BoW and TF-IDF models, or through the use of deep learning models with learnable weights. In practice, directly training with OVA is prohibitive when $L$ is large. This is due to not only the underlying $O(L)$ time complexity, but also due to severe label sparsity issues inherent to long-tailed label distributions (Chang et al., 2020b; Zhang et al., 2021a).

**XR-Transformers.** We start by briefly introducing the state-of-the-art XMC method: XR-Transformers (Zhang et al., 2021a). A simplified illustration of it is given in Figure 2. The first step is to leverage the statistical text vectorizer $\Phi_{\text{stat}}(x)$ and training labels $\mathbf{y}$ to construct the label representation $\mathbf{Z} \in \mathbb{R}^{L \times d}$ (which should not be confused with the label text feature $\{z_l\}_{l=1}^L$). XR-Transformers adopt the Predicted Instance Feature Aggregation (PIFA) strategy for label representations, which is further used to construct the hierarchical label tree (HLT) via hierarchical $k$-means clustering,

$$\text{(PIFA)} \quad \mathbf{Z}_l = \frac{\sum_{i:y_{il}=1} \Phi_{\text{stat}}(x_i)}{\| \sum_{i:y_{il}=1} \Phi_{\text{stat}}(x_i) \|} \ \forall l \in [L]. \tag{2}$$

Note that each level of the HLT gives rise to an XMC problem. The second step is to train the Transformer models $\Phi_{\text{dnn}}$, such as BERT (Devlin et al., 2019), recursively from root to leaves. In the third step, the XR-Transformer concatenates both the statistical feature $\Phi_{\text{stat}}(x)$ and Transformer embedding $\Phi_{\text{dnn}}(x)$ to arrive at the final feature $\Phi_{\text{cat}}(x)$. It also trains linear OVA classifiers (1) based on HLT recursively to generate the final prediction. Through the use of HLT, one can not only reduce the time complexity from $O(L)$ to $O(\log(L))$, but also alleviate the label sparsity issue (Zhang et al., 2021a).

**XMC with label text.** Consider the scenario where the label text $\{z_l\}_{l=1}^L$ is available as side-information, where $z_l \in \mathcal{D}$ is the label text of label $l$. One can observe that standard XMC approaches, such as XR-Transformers, cannot leverage this information directly. While it is possible to use the label text to improve the construction of the HLT, the learnable text vectorizer $\Phi_{\text{dnn}}$ itself cannot leverage the label text information. PINA, as we show, enables XMC learners to leverage label text information in a data enhancement manner, where the learnable text vectorizer $\Phi_{\text{dnn}}$ can also perform training with the label texts.

**XMC with instance correlation signal.** In keyword recommendation problems, researchers aim to predict the most relevant keywords for each product. Keyword recommendation is an example of an XMC problem. In this scenario, instances (products) correlation signals are also available from the customer behavioral data, such as those pertaining to the "frequently bought together" category. This type of side information provides us with beneficial information about the instances. Unfortunately, it is not clear how to leverage the instances correlation signals within the standard XMC problem solvers. PINA makes use of this side information in a data enhancement way similar to what is done with the label text.

### 3.1. The XMC problem and the neighborhood prediction problem

To understand the key idea behind our approach, we have to describe the relationship between the XMC problem and the neighborhood prediction problem first described in the graph learning literature. Recently, Chien et al. 2021a revealed the equivalence of the XMC problem and the neighborhood prediction problem in graph learning. Let $G = (V_{\text{in}}, V_{\text{out}}, E)$ be a directed bipartite graph, where $V_{\text{in}} = [N]$ and $V_{\text{out}} = [L]$ are the input and output node sets, respectively, while $E \subseteq V_{\text{in}} \times V_{\text{out}}$ is the edge set. A common way to characterize the edge relations is to use a biadjacency matrix $\mathbf{B} \in \{0,1\}^{N \times L}$, where $B_{ij} = 1$ if and only if $(i, j) \in E$. The goal of the neighborhood prediction problem is to predict the $i^{th}$ row of $\mathbf{B}$ via the node attributes of node $i$. Since the $i^{th}$ row of $\mathbf{B}$ is just a vector in $\{0,1\}^{1 \times L}$ (i.e., a binary vector), it can also be viewed as a multi-label $\mathbf{y}_i$. See Figure 3 for a pictorial illustration.

One standard operation in graph learning is graph convolution (Kipf & Welling, 2017), where the key idea is to gather the attributes of the neighborhood of a node to enhance its ego node features. It has been proven to be effective for many graph tasks, including node classification (Kipf & Welling, 2017; Hamilton et al., 2017; Veličković et al., 2018; Chien et al., 2020), link prediction (Zhang et al., 2021b; Zhang & Chen, 2018) and graph classification (Xu et al., 2019; Zhang & Li, 2021). Our proposed method –

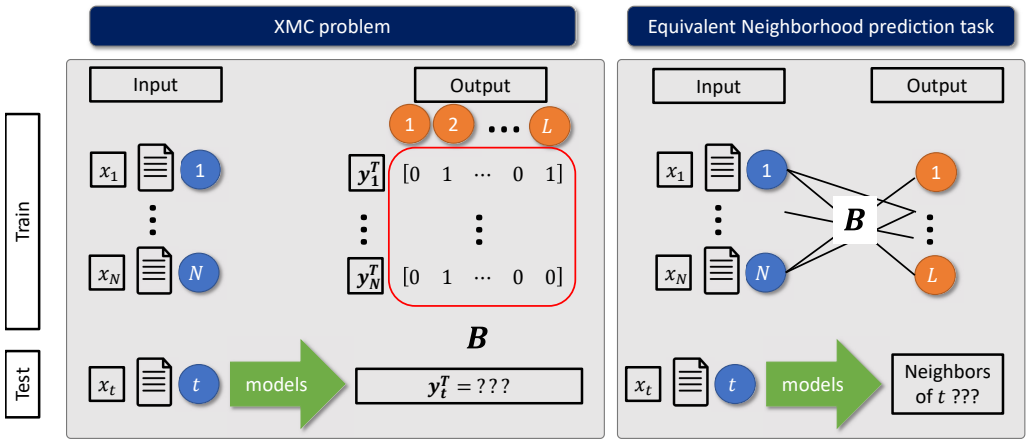


*Figure 3.* Equivalence of the XMC problem and neighborhood prediction problem. Blue nodes correspond to instances and orange nodes correspond to labels. Note that the multi-label vectors $\{\mathbf{y}_i\}$ can be viewed as the rows of biadjacency matrix $\mathbf{B}$, which characterize the edges in the graphs on the right. Hence, predicting the multi-label $\mathbf{y}_i$ is equivalent to predicting the neighborhood of blue node $i$ in the graph on the right.

PINA – is motivated by the connection to the neighborhood prediction task and the graph convolution operation, which we describe in the next section.

## 4. PINA: Predicted Instance Neighborhood Aggregation

For simplicity, we mostly focus on the setting where side information is of the form of label text. The case of side information being of the form of instance correlation signals can be treated similarly. A detailed discussion regarding how to apply PINA with instance correlation signals is available in Section 4.2.

We propose PINA to allow XMC learners such as XR-Transformers to make use of label text information in a *data enhancement* manner. Due to the equivalence of XMC and neighborhood prediction, a naive way of including label attributes is via neighborhood aggregation. However, there are several issues preventing us from applying this idea directly. First, one can only apply the average operation on numerical features instead of raw text. Ideally, we have to fine-tune the text vectorizer $\Phi_{\text{dnn}}$ with both instance and label text features during the training phase. However, the XMC formalism (Figure 3) does not treat label text as an input, which is suboptimal. Second, the neighborhood relation is defined using labels $\mathbf{y}_i$, which are *unknown for test instances*. See Figure 3 for an illustration. Thus, we cannot apply neighborhood aggregation directly even though we are equipped with the bipartite graph underlying the XMC problem. We describe next the high-level ideas how to resolve these issues.

**Lack of knowledge about neighborhoods for test instances.** In order to resolve this issue, we propose to pretrain a neighborhood predictor $g$. Instead of using the exact neighbors (i.e. ground truth multi-labels), we generate predicted neighbors via $g$. This allows us to generate neighbors for both *train and test* instances. Note that pretraining $g$ only leverages training data (which includes both labels and instances).

**The transformer text vectorizer $\Phi_{\text{dnn}}$ does not involve label text.** In order to resolve this issue, we propose a pretraining XMC task that also takes label text as input. More specifically, the input and output space of our pretraining task contains both instances and labels. See the illustration of the proposed pretrained XMC in Figure 4. Hence, our pretrained text vectorizer $\Phi_{\text{pre}}$ is trained with both instance text and label text. This resolves the issue of not being able to include the label text in standard XMCs.

### 4.1. A detailed description of PINA

We implemented PINA as a two-stage method, described in Figure 4. The pseudo-code of the PINA augmentation and pretraining process are listed in the Appendix J. The first stage is the pretraining phase, where we design a pretraining task to learn a neighbor predictor $g(\cdot, \Phi_{\text{stat}})$ via a base XMC learner (e.g., XR-transformer). Note that the pretraining task is also an XMC problem, but both instances and label text are treated as inputs and both the input and output space contain instance and label nodes. The edges are defined

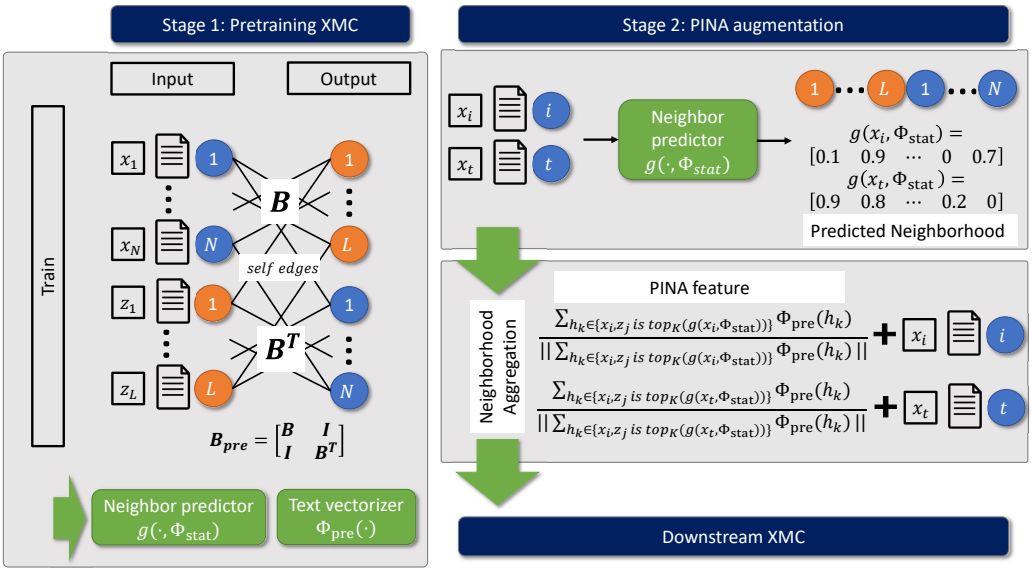


*Figure 4.* Illustration of the two-stage PINA method. At stage 1), we construct a pretraining biadjacency matrix $\mathbf{B}_{pre}$ using only the *training data*. Since we still have an XMC problem, we can train an XMC learner as the neighbor predictor $g(\cdot, \Phi_{\text{stat}})$ and obtain its corresponding text vectorizer $\Phi_{\text{pre}}(\cdot)$ as well. At stage 2), we first use the pretrained neighbor predictor $g(\cdot, \Phi_{\text{stat}})$ to extract the most relevant (top $K$) nodes among the output space of the pretraining task. Then we apply the pretrained text vectorizer $\Phi_{\text{pre}}(\cdot)$ to obtain the numerical features for both instances and labels. Finally, we perform normalized neighborhood aggregation to obtain the PINA augmented features, which can then be used in downstream XMC.

by multi-label relations $\{\mathbf{y_i}\}_{i=1}^N$ in an undirected manner. We also add edges from all instances and label nodes in the input space to their output space counterpart. More specifically, we construct $\mathbf{B}_{pre}$ as described in Figure 4. Recall that $\mathbf{B} \in \{0,1\}^{N\times L}$ is obtained by training the multi-labels $\{\mathbf{y}_i\}_{i=1}^N$ and $\mathbf{I}$ represents an identity matrix of appropriate dimensions. Hence, in our pretraining XMC problem, we aim to predict the $i^{th}$ row of $\mathbf{B}_{pre}$ using $x_i$ when $i \in [N]$ and $z_{i-N}$ when $i = N+1, N+2, \ldots, N+L$. This allows both the label and instance text to be observed during the pretraining phase. We consequently obtain the corresponding text vectorizer $\Phi_{\text{pre}}$ to generate numerical features for both the labels and instances.

The second stage is the PINA augmentation phase, during which we leverage the pretrained neighborhood predictor $g(\cdot, \Phi_{\text{stat}})$ and text vectorizer $\Phi_{\text{pre}}(\cdot)$ to augment the instance features. We first predict the most relevant nodes among the output space of the pretraining stage as neighbors for both training and *test* instances via our pretrained neighbor predictor $g(\cdot, \Phi_{\text{stat}})$. More specifically, we obtain the neighborhood prediction vector $g(x_i, \Phi_{\text{stat}}) \in [0,1]^{1\times L}$ and zero out all but the top $K$ largest values $\mathbf{P}_i = top_K(g(x_i, \Phi_{\text{stat}}))$. Then we perform neighborhood aggregation on the numerical features obtained from $\Phi_{\text{pre}}$ accordingly, which results in PINA features. See lines $3-8$ in Algorithm 1 for PINA feature extraction of each instance. The augmented features are fed to the next XMC learner for solving the downstream XMC task.

### 4.2. Applying PINA to instance correlation signals

We describe next how to apply PINA to instance correlation signals. In this case, the instance correlation can be formulated as an instance-to-instance graph (i.e., I2I graph). Similarly to the construction rules of the Amazon co-purchase graph benchmarking dataset known from the graph learning literature, a link $(i, j)$ exists if and only if the instance correlation signal between $i$ and $j$ is larger than a threshold. We thus capture the instance correlation signal by the (bi)adjacency matrix $\mathbf{B}'$.

We then use $\mathbf{B}'$ directly to formulate our pretraining XMC. More specifically, one can choose $\mathbf{B}_{pre} = \mathbf{B}'$ in Stage 1 of Figure 4 to obtain the neighbor predictor and text vectorizer. Note that the set of instances considered in the downstream application need not be identical to instances in the I2I graph. We only require their text domains to be the same (i.e., all texts associated with instances are product descriptions). This is due to the fact that we can still obtain the predicted neighbors among instances in the I2I graph for each instance in the downstream application (and similar to the reason why PINA applies to test data). The remainder of the PINA pipeline is as described in Figure 4.

## 5. Experimental Results

We demonstrate the effectiveness of PINA on both public benchmark datasets with side information of the form of

*Table 1.* Public benchmark dataset statistics: $N_{train}, N_{test}$ refer to the number of instances in the training and test sets, respectively; $L$: the number of labels. $\bar{L}$: the average number of positive labels per instance; $\bar{n}$: average number of instances per label; $d_{BoW}$: the Bag-of-Word feature dimension.

| | $d_{BoW}$ | $L$ | $N_{train}$ | $N_{test}$ | $\bar{n}$ | $\bar{L}$ |
|---|---|---|---|---|---|---|
| LF-Amazon-131K | 80,000 | 131,073 | 294,805 | 134,835 | 5.15 | 2.29 |
| LF-WikiSeeAlso-320K | 80,000 | 312,330 | 693,082 | 177,515 | 4.67 | 2.11 |
| LF-Wikipedia-500K | 500,000 | 501,070 | 1,813,391 | 783,743 | 24.75 | 4.77 |
| LF-Amazon-1.3M | 128,000 | 1,305,265 | 2,248,619 | 970,273 | 28.24 | 22.20 |

label text and proprietary datasets with side information of the form of instance correlation signals. We report the precision at $k$ (P@$k$) and recall at $k$ (R@$k$) as our evaluation metrics. Their definition can be found in Appendix F.

### 5.1. Label text benchmark datasets

**Datasets.** We consider the public benchmark long-text datasets for product-to-product recommendations as well as predicting related Wikipedia articles, taken from (Bhatia et al., 2016). The data statistics can be found in Table 1. For a fair comparison with previous works, we adopt the provided BoW features as our statistical text feature in the experiments. For LF-Amazon-1.3M, where BOW features are only constructed using title text, we still use the provided BoW features (Bhatia et al., 2016) and leverage the full text only as input to transformer encoders.

**Baseline methods.** We not only compare PINA with plain XR-Transformers (Zhang et al., 2021a), but also with other previously published XMC methods that achieve state-of-the-art results for the label text XMC problem. These include ECLARE (Mittal et al., 2021b), DECAF (Mittal et al., 2021a), AttentionXML (You et al., 2019) and SiameseXML (Dahiya et al., 2021a). For methods other than XR-Transformer, we directly take the reported numbers from the DECAF paper (Mittal et al., 2021a) with superscript $^\dagger$ and SiameseXML paper (Dahiya et al., 2021a) with superscript $^\star$. For PINA, we use XR-Transformers as our baseline XMC learners.

**Results.** The results are summarized in Table 2. Compared to XR-Transformers, PINA consistently improves the performance with a significant gain across all datasets. This demonstrates the effectiveness of PINA as a data enhancement approach for leveraging label text side information. As an example, PINA improves XR-Transformer with a gain of 1-2%. Compared to previously published state-of-the-art methods, XR-Transformer + PINA outperforms SiameseXML by around 2% and 2.5% on LF-Amazon-131K and LF-WikiSeeAlso-320K, respectively. At the same time, XR-Transformer + PINA achieves roughly the same performance as AttentionXML on LF-Wikipedia-500K. AttentionXML is outperformed by XR-Transformer + PINA with a 4%-4.5% margin on LF-Amazon-131K and LF-WikiSeeAlso-320K. Moreover, AttentionXML exhibits at least a 2.5 times larger training time compared to XR-Transformer + PINA. These results again demonstrate the superiority of the proposed PINA method in leveraging label text side information in the XMC problem. More experiment details such as significant tests are included in the Appendix F.

Note that none of the previously reported methods were tested on the (long text) LF-Amazon-1.3M dataset. Some where tested on the short text version LF-AmazonTitles-1.3M, which exclusively uses instance titles as text features. We have included the result on the original LF-AmazonTitles-1.3M dataset in Table 3, which show that PINA outperforms published state-of-the-art methods by a large margin.

**Ablation study.** In Section 4 we introduced two major problems in implementing the neighborhood aggregation idea. For the problem of lacking neighborhoods for test instances, it is straightforward to understand why using predicted neighbors resolves the issue. In the ablation study that follows, we test whether letting the transformer text vectorizer $\Phi_{\text{dnn}}$ observe the label text influences the performance of the learner. Instead of pretraining with $\mathbf{B}_{pre}$, one may also pretrain the neighbor predictor with $\mathbf{B}$. Our results are listed in Table 4. We can see that letting the transformer text vectorizer $\Phi_{\text{dnn}}$ see the label text is indeed crucial, which verifies our intuition described in Section 4. Moreover, we find that pretraining neighborhood predictors with $\mathbf{B}$ can often lead to worse results compared to applying XR-Transformer only. This further highlights the necessity of our design for PINA. Finally, we provide a qualitative analysis in Appendix G which further validate the necessity of the design of PINA.

### 5.2. Proprietary datasets with instance correlation signal

**Datasets.** We also conduct our experiment on a proprietary dataset pertaining to tasks similar to those illustrated in Figure 1. This proprietary dataset consists of millions of instances and labels. Our side information is of the form of instance correlation signals among roughly millions of instances. The instance correlation signal is aggregated similar to the case described in the Introduction.

**Settings.** Due to the large scale of the data, the requirements for small inference times and daily model updates, we choose XR-Linear (Yu et al., 2022) as our downstream XMC model. Nevertheless, since the pretraining step in PINA can be performed beforehand and requires less frequent updates (i.e. monthly), we can still use the XR-Transformer as our neighborhood predictor during PINA augmentation. The

*Table 2.* Main result on label text XMC benchmark datasets. Bold font refers to the best result. Superscripts † and ⋆ indicate the results are taken from DECAF paper (Mittal et al., 2021a) and SiameseXML (Dahiya et al., 2021a) respectively.

| Methods | P@1 | P@3 | P@5 | Train Time (hrs) | P@1 | P@3 | P@5 | Train Time (hrs) |
|---|---|---|---|---|---|---|---|---|
| | LF-Amazon-131K | | | | LF-WikiSeeAlso-320K | | | |
| DECAF† | 42.94 | 28.79 | 21 | 1.8 | 41.36 | 28.04 | 21.38 | 4.84 |
| AttentionXML† | 42.9 | 28.96 | 20.97 | 50.17 | 40.5 | 26.43 | 21.38 | 90.37 |
| SiameseXML⋆ | 44.81 | 30.19 | 21.94 | 1.18 | 42.16 | 28.14 | 21.35 | 2.33 |
| ECLARE⋆ | 43.56 | 29.65 | 21.57 | 2.15 | 40.58 | 26.86 | 20.14 | 9.40 |
| XR-Transformer | 45.61 | 30.85 | 22.32 | 7.9 | 42.57 | 28.24 | 21.30 | 22.1 |
| XR-Transformer + PINA | **46.76** | **31.88** | **23.20** | 9.8 | **44.54** | **30.11** | **22.92** | 28.3 |
| | LF-Wikipedia-500K | | | | LF-Amazon-1.3M | | | |
| DECAF† | 73.96 | 54.17 | 42.43 | 44.23 | - | - | - | - |
| AttentionXML† | 82.73 | **63.75** | **50.41** | 221.6 | - | - | - | - |
| SiameseXML⋆ | 67.26 | 44.82 | 33.73 | 7.31 | - | - | - | - |
| ECLARE⋆ | 68.04 | 46.44 | 35.74 | 86.57 | - | - | - | - |
| XR-Transformer | 81.62 | 61.38 | 47.85 | 41.0 | 54.67 | 47.87 | 42.93 | 28.2 |
| XR-Transformer + PINA | **82.83** | 63.14 | 50.11 | 85.0 | **58.33** | **51.06** | **46.04** | 39.1 |

*Table 3.* Study of PINA on LF-AmazonTitle-1.3M dataset. Bold font numbers indicate the best results.

| Methods | P@1 | P@3 | P@5 |
|---|---|---|---|
| | LF-AmazonTitle-1.3M | | |
| DECAF† | 50.67 | 44.49 | 40.35 |
| AttentionXML† | 45.04 | 39.71 | 36.25 |
| SiameseXML⋆ | 49.02 | 42.72 | 38.52 |
| ECLARE⋆ | 50.14 | 44.09 | 40.00 |
| XR-Transformer | 50.98 | 44.49 | 40.05 |
| XR-Transformer + PINA | **55.76** | **48.70** | **43.88** |

*Table 4.* Ablation study of PINA on the LF-Amazon-131K dataset. Bold font indicates the best results. For PINA-naive, we use only $\mathbf{B}$ as our pretraining target in Stage 1 of PINA.

| LF-Amazon-131K | P@1 | P@3 | P@5 |
|---|---|---|---|
| XR-Transformer | 45.61 | 30.85 | 22.32 |
| XR-Transformer + PINA-naïve | 43.89 | 30.43 | 22.67 |
| XR-Transformer + PINA | **46.76** | **31.88** | **23.20** |

high-level idea of the XR-Linear model can be understood with the help of Figure 2, where XR-Linear does not include Step 2 (i.e., machine learned matching). It directly trains a linear OVA classifier recursively on the HLT with input statistical text features such as BoW or TF-IDF. Besides applying PINA with XR-Linear, we also conduct an ablation study. We test if our performance gain is merely a consequence of concatenating features from pretrained transformer text vectorizers or if neighborhood aggregation also plays an important role.

*Table 5.* Results on the proprietary dataset. The baseline corresponds to applying XR-Linear directly and we report the relative results compared to it in %. Bold fonts indicate the best results.

| | P@1 | P@10 | R@1 | R@10 |
|---|---|---|---|---|
| Use pretrained text vectorizer only | +0 | -1.51 | +1.67 | -1.14 |
| PINA | **+3.45** | +0 | **+3.93** | **+0.23** |

**Results.** We report the relative performance compared to the plain XR-Linear model. Our results are listed in Table 5. One can observe that PINA once again consistently improves the performance of downstream XMC models under all reported metrics. Furthermore, our ablation study shows that the performance gain of PINA does not merely come from concatenating pretrained text features. Our neighborhood aggregation mechanism is indeed important. Notably, merely using pretrained text features can lead to worse performance in P@10 and R@10.

## 6. Conclusion

We proposed Predicted Instance Neighborhood Aggregation (PINA), a data enhancement framework that allows traditional XMC models to leverage various forms of side information, such as label metadata and instance correlation signals. Motivated by the neighborhood prediction problem from the graph learning literature, PINA enriches the instance features via neighborhood aggregation similar to what graph convolutions and message passing operations do in many graph learning tasks. We conducted experiments on both public benchmark datasets and a proprietary dataset. PINA offers consistent gains when compared to its

backbone XMC models such as XR-Transformers and XR-Linear. The combination of PINA and a XR-Transformer also outperforms published state-of-the-art methods specialized for label text on all the benchmarking datasets.

## Acknowledgements

The authors thank the support from Amazon and the Amazon Conference Grant. Part of this work was funded by the NSF CIF Grant 1956384. Cho-Jui Hsieh is supported in part by NSF IIS-2008173 and IIS-2048280.

## References

Almagro, M., Unanue, R. M., Fresno, V., and Montalvo, S. Icd-10 coding of spanish electronic discharge summaries: an extreme classification problem. *IEEE Access*, 8:100073–100083, 2020.

Babbar, R. and Schölkopf, B. DiSMEC: distributed sparse machines for extreme multi-label classification. In *WSDM*, 2017.

Babbar, R. and Schölkopf, B. Data scarcity, robustness and extreme multi-label classification. *Machine Learning*, pp. 1–23, 2019.

Barkan, O., Caciularu, A., Rejwan, I., Katz, O., Weill, J., Malkiel, I., and Koenigstein, N. Cold item recommendations via hierarchical item2vec. In *2020 IEEE International Conference on Data Mining (ICDM)*, pp. 912–917. IEEE, 2020.

Bhatia, K., Dahiya, K., Jain, H., Kar, P., Mittal, A., Prabhu, Y., and Varma, M. The extreme classification repository: Multi-label datasets and code, 2016. URL http://manikvarma.org/downloads/XC/XMLRepository.html.

Chang, W.-C., Yu, F. X., Chang, Y.-W., Yang, Y., and Kumar, S. Pre-training tasks for embedding-based large-scale retrieval. In *International Conference on Learning Representations*, 2020a. URL https://openreview.net/forum?id=rkg-mA4FDr.

Chang, W.-C., Yu, H.-F., Zhong, K., Yang, Y., and Dhillon, I. S. Taming pretrained transformers for extreme multi-label text classification. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 3163–3171, 2020b.

Chang, W.-C., Yu, H.-F., Zhong, K., Yang, Y., and Dhillon, I. S. Taming pretrained transformers for extreme multi-label text classification. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 3163–3171, 2020c.

Chang, W.-C., Jiang, D., Yu, H.-F., Teo, C. H., Zhang, J., Zhong, K., Kolluri, K., Hu, Q., Shandilya, N., Ievgrafov, V., et al. Extreme multi-label learning for semantic matching in product search. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pp. 2643–2651, 2021.

Chiang, W.-L., Liu, X., Si, S., Li, Y., Bengio, S., and Hsieh, C.-J. Cluster-gcn: An efficient algorithm for training deep and large graph convolutional networks. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 257–266, 2019.

Chien, E., Peng, J., Li, P., and Milenkovic, O. Adaptive universal generalized pagerank graph neural network. In *International Conference on Learning Representations*, 2020.

Chien, E., Chang, W.-C., Hsieh, C.-J., Yu, H.-F., Zhang, J., Milenkovic, O., and Dhillon, I. S. Node feature extraction by self-supervised multi-scale neighborhood prediction. In *International Conference on Learning Representations*, 2021a.

Chien, E., Li, P., and Milenkovic, O. Landing probabilities of random walks for seed-set expansion in hypergraphs. In *2021 IEEE Information Theory Workshop (ITW)*, pp. 1–6. IEEE, 2021b.

Chien, E., Pan, C., Peng, J., and Milenkovic, O. You are allset: A multiset function framework for hypergraph neural networks. In *International Conference on Learning Representations*, 2021c.

Chien, I. E., Zhou, H., and Li, P. $hs^2$: Active learning over hypergraphs with pointwise and pairwise queries. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pp. 2466–2475. PMLR, 2019.

Choromanska, A. E. and Langford, J. Logarithmic time online multiclass prediction. *Advances in Neural Information Processing Systems*, 28:55–63, 2015.

Dahiya, K., Agarwal, A., Saini, D., Gururaj, K., Jiao, J., Singh, A., Agarwal, S., Kar, P., and Varma, M. Siamesexml: Siamese networks meet extreme classifiers with 100m labels. In *International Conference on Machine Learning*, pp. 2330–2340. PMLR, 2021a.

Dahiya, K., Saini, D., Mittal, A., Shaw, A., Dave, K., Soni, A., Jain, H., Agarwal, S., and Varma, M. DeepXML: A deep extreme multi-label learning framework applied to short text documents. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, pp. 31–39, 2021b.

Daumé III, H., Karampatziakis, N., Langford, J., and Mineiro, P. Logarithmic time one-against-some. In *International Conference on Machine Learning*, pp. 923–932. PMLR, 2017.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, 2019.

Gilmer, J., Schoenholz, S. S., Riley, P. F., Vinyals, O., and Dahl, G. E. Neural message passing for quantum chemistry. In *International conference on machine learning*, pp. 1263–1272. PMLR, 2017.

Hamilton, W., Ying, Z., and Leskovec, J. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30, 2017.

Hu, W., Fey, M., Zitnik, M., Dong, Y., Ren, H., Liu, B., Catasta, M., and Leskovec, J. Open graph benchmark: Datasets for machine learning on graphs. *Advances in neural information processing systems*, 33:22118–22133, 2020.

Jain, H., Balasubramanian, V., Chunduri, B., and Varma, M. SLICE: Scalable linear extreme classifiers trained on 100 million labels for related searches. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, pp. 528–536. ACM, 2019.

Jasinska-Kobus, K., Wydmuch, M., Dembczynski, K., Kuznetsov, M., and Busa-Fekete, R. Probabilistic label trees for extreme multi-label classification. *arXiv preprint arXiv:2009.11218*, 2020.

Jasinska-Kobus, K., Wydmuch, M., Thiruvenkatachari, D., and Dembczynski, K. Online probabilistic label trees. In *International Conference on Artificial Intelligence and Statistics*, pp. 1801–1809. PMLR, 2021.

Jiang, T., Wang, D., Sun, L., Yang, H., Zhao, Z., and Zhuang, F. LightXML: Transformer with dynamic negative sampling for high-performance extreme multi-label text classification. In *AAAI*, 2021.

Johnson, J., Douze, M., and Jégou, H. Billion-scale similarity search with gpus. *IEEE Transactions on Big Data*, 7 (3):535–547, 2019.

Khandagale, S., Xiao, H., and Babbar, R. Bonsai: diverse and shallow trees for extreme multi-label classification. *Machine Learning*, 109(11):2099–2119, 2020.

Kipf, T. N. and Welling, M. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=SJU4ayYgl.

Lee, K., Chang, M.-W., and Toutanova, K. Latent retrieval for weakly supervised open domain question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 6086–6096, 2019.

Li, P., Chien, I., and Milenkovic, O. Optimizing generalized pagerank methods for seed-expansion community detection. *Advances in Neural Information Processing Systems*, 32, 2019.

Liu, C., Li, X., Cai, G., Dong, Z., Zhu, H., and Shang, L. Noninvasive self-attention for side information fusion in sequential recommendation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 4249–4256, 2021.

Liu, J., Chang, W.-C., Wu, Y., and Yang, Y. Deep learning for extreme multi-label text classification. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 115–124. ACM, 2017.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. RoBERTa: A robustly optimized BERT pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

Malkov, Y. A. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. *IEEE transactions on pattern analysis and machine intelligence*, 42(4):824–836, 2018.

Medini, T. K. R., Huang, Q., Wang, Y., Mohan, V., and Shrivastava, A. Extreme classification in log memory using count-min sketch: A case study of amazon search with 50m products. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper/2019/file/69cd21a0e0b7d5f05dc88a0be36950c7-Paper.pdf.

Mittal, A., Dahiya, K., Agrawal, S., Saini, D., Agarwal, S., Kar, P., and Varma, M. Decaf: Deep extreme classification with label features. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, pp. 49–57, 2021a.

Mittal, A., Sachdeva, N., Agrawal, S., Agarwal, S., Kar, P., and Varma, M. ECLARE: Extreme classification with label graph correlations. In *Proceedings of The ACM International World Wide Web Conference*, April 2021b.

Prabhu, Y. and Varma, M. Fastxml: A fast, accurate and stable tree-classifier for extreme multi-label learning. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 263–272, 2014.

Prabhu, Y., Kag, A., Harsola, S., Agrawal, R., and Varma, M. Parabel: Partitioned label trees for extreme classification with application to dynamic search advertising. In *Proceedings of the 2018 World Wide Web Conference*, pp. 993–1002, 2018.

Saini, D., Jain, A., Dave, K., Jiao, J., Singh, A., Zhang, R., and Varma, M. GalaXC: Graph neural networks with labelwise attention for extreme classification. In *Proceedings of The Web Conference*, April 2021.

Tagami, Y. AnnexML: Approximate nearest neighbor search for extreme multi-label classification. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 455–464, 2017.

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., and Bengio, Y. Graph attention networks. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=rJXMpikCZ.

Wang, J., Huang, P., Zhao, H., Zhang, Z., Zhao, B., and Lee, D. L. Billion-scale commodity embedding for e-commerce recommendation in alibaba. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 839–848, 2018.

Wydmuch, M., Jasinska, K., Kuznetsov, M., Busa-Fekete, R., and Dembczynski, K. A no-regret generalization of hierarchical softmax to extreme multi-label classification. In *NIPS*, 2018.

Xu, K., Hu, W., Leskovec, J., and Jegelka, S. How powerful are graph neural networks? In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=ryGs6iA5Km.

Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R., and Le, Q. V. XLNet: Generalized autoregressive pretraining for language understanding. In *NIPS*, 2019.

Ye, H., Chen, Z., Wang, D.-H., and Davison, B. Pretrained generalized autoregressive model with adaptive probabilistic label clusters for extreme multi-label text classification. In *International Conference on Machine Learning*, pp. 10809–10819. PMLR, 2020.

Yen, I. E., Huang, X., Dai, W., Ravikumar, P., Dhillon, I., and Xing, E. PPDsparse: A parallel primal-dual sparse method for extreme classification. In *KDD*. ACM, 2017.

You, R., Zhang, Z., Wang, Z., Dai, S., Mamitsuka, H., and Zhu, S. Attentionxml: Label tree-based attention-aware deep model for high-performance extreme multi-label text classification. *Advances in Neural Information Processing Systems*, 32, 2019.

Yu, H.-F., Zhong, K., Zhang, J., Chang, W.-C., and Dhillon, I. S. Pecos: Prediction for enormous and correlated output spaces. *Journal of Machine Learning Research*, 2022.

Zhang, J., Chang, W.-C., Yu, H.-F., and Dhillon, I. S. Fast multi-resolution transformer fine-tuning for extreme multi-label text classification. volume 34, pp. 7267–7280, 2021a.

Zhang, M. and Chen, Y. Link prediction based on graph neural networks. *Advances in neural information processing systems*, 31, 2018.

Zhang, M. and Li, P. Nested graph neural networks. *Advances in Neural Information Processing Systems*, 34: 15734–15747, 2021.

Zhang, M., Li, P., Xia, Y., Wang, K., and Jin, L. Labeling trick: A theory of using graph neural networks for multi-node representation learning. *Advances in Neural Information Processing Systems*, 34:9061–9073, 2021b.

## A. Future directions

Although we focused on combining PINA with XR-Transformers, it is worth pointing out that PINA can also be used in conjunction with other XMC methods. Recall that the neighborhood predictor $g$ in PINA can be modeled by another XMC method and that one can make use of the pre-trained text vectorizer $\Phi_{\text{pre}}$ therein. It is also an interesting question to test the performance of PINA combined with other XMC methods.

## B. Potential Social Impact

Our method allows XMC solutions such as XR-Transformer to leverage various side information, including label text and instance correlation signals. This can have positive social impacts such as improving search results for minorities, where XMC solvers usually have worse performance due to less data abundance. Since our method is a general approach to the XMC problem, we do not aware of any negative social impact of our work.

## C. Additional details on proprietary datasets

We subsampled roughly 2 million instances and 3 million labels. The label matrix contains roughly 7 million positives. The underlying task is e-commerce keyword recommendation and the side information is similar to the co-purchase signal that we mentioned in our paper.

## D. Computational Environment

All experiments are conducted on the AWS p3dn.24xlarge instance, consisting of 96 Intel Xeon CPUs with 768 GB of RAM and 8 Nvidia V100 GPUs with 32 GB of memory each.

## E. Hyperparameters of PINA and XR-Transformers

For the PINA augmentation stage, we aggregate the top-5 neighbors (with respect to the highest prediction score) according to the output of the neighborhood predictor $g$. We directly use the normalized prediction score as edge weights for the neighborhood aggregation process. For the pretrained neighborhood predictor $g$ and downstream XR-Transformer, we adopt the default hyperparameters provided by the original XR-Transformer repository [*] with some slight modifications. More specifically, we first choose the hyperparameters from datasets that are "closest" to our datasets. For example, we adopt the hyperparameters used for "amazon-670k" from Zhang et al. 2021a for our "LF-Amazon-131K" dataset. We adopt the hyperparameters used for "wiki-500k" from Zhang et al. 2021a for our "LF-WikiSeeAlso-320K" and "LF-Wikipedia-500K" datasets. We adopt the hyperparameters used for "amazon-3m" from Zhang et al. 2021a for our "LF-Amazon-1.3M" dataset. For pretraining the neighborhood predictor $g$, we only change the HLT-related parameters in order to accommodate different sizes of the output space. All hyperparameters are provided in our code for reproducibility.

[*] https://github.com/amzn/pecos/tree/mainline/examples/xr-transformer-neurips21

## F. Evaluation Details

Note that we follow previous works such as ECLARE (Mittal et al., 2021b) and SiameseXML (Dahiya et al., 2021a) to post-process the test data and predictions. We use the code from ECLARE for "Reciprocal-pair Removal". This procedure removes instance-label pairs that are reciprocal, meaning that the two instances are also each others labels. See the XMC repository website [†] for more details.

**Definition of precision and recall at $k$.** We report the precision at $k$ (P@$k$) and recall at $k$ (R@$k$) as our evaluation metrics. Their formal definitions are as follows:

$$\text{P@k} = \frac{1}{k}\sum_{l=1}^{k} y_{rank(l)} \tag{3}$$

$$\text{R@k} = \frac{1}{\sum_{i=1}^{L} y_i}\sum_{l=1}^{k} y_{rank(l)} \tag{4}$$

where $y \in \{0,1\}^L$ is the ground truth label and $rank(l)$ is the index of the $l$-th highest predicted label.

**Significant Tests.** We conduct significant tests to verify the improvements by leveraging PINA. In particular, we calculated the test instance-wise $P@1,3,5$ for XR-Transformer and XR-Transformer+PINA and perform t-test on each one of these metrics. As shown in table 6, the gain of XR-Transformer+PINA over XR-Transformer is indeed significant with p-value less than $8.5 \times 10^{-4}$ for all precision metrics in table 2.

*Table 6.* P-values of significant test between XR-Transformer and XR-Transformer+PINA on instance-wise metrics.

| | p-value:P@1 | p-value:P@3 | p-value:P@5 |
|---|---|---|---|
| LF-Amazon-131K | $8.5 \times 10^{-4}$ | $1.8 \times 10^{-8}$ | $1.8 \times 10^{-14}$ |
| LF-WikiSeeAlso-320K | $3.8 \times 10^{-15}$ | $1.2 \times 10^{-34}$ | $1.7 \times 10^{-57}$ |
| LF-Wikipedia-500K | $6.1 \times 10^{-28}$ | $1.0 \times 10^{-113}$ | $1.4 \times 10^{-179}$ |

*Table 7.* Qualitative analysis of predicted results on the LF-Amazon-131K dataset. Bold fonts indicate correct predictions while italic fonts represent the "helpful" neighbors predicted by the neighborhood predictor $g$ of PINA.

| Instance title: | Himalayan Cats (Barron's Complete Pet Owner's Manuals) | | |
|---|---|---|---|
| Ground Truth (unorderly) | Guide to Owning Himalayan Cat; Guide to Owning a Persian Cat; The Complete Book of Cat Breeding | | |
| Top K recommendation | XR-Transformer | Neighbor predictor $g$ of PINA | XR-Transformer + PINA |
| 1 | The Art of Keeping Snakes (Advanced Vivarium Systems) | *Guide to Owning Himalayan Cat* | The Art of Keeping Snakes (Advanced Vivarium Systems) |
| 2 | Taming/Training Budgerigars | *Persian Cats: Everything About Purchase, Care, Nutrition, Disease, and Behavior (Special Chapter : Understanding Persian Cats)* | **Guide to Owning a Persian Cat** |
| 3 | Reptiles (1-year) | The Art of Keeping Snakes (Advanced Vivarium Systems) | **Guide to Owning Himalayan Cat** |
| 4 | A Gardener's Handbook of Plant Names: Their Meanings and Origins | Chameleons (Barron's Complete Pet Owner's Manuals) | Reptiles (1-year) |
| 5 | Little Dogs: Training Your Pint-Sized Companion | Boas (Barron's Complete Pet Owner's Manuals) | **The Complete Book of Cat Breeding** |

*Table 8.* Predicted results of XR-Transformer + PINA-naive on LF-Amazon-131K dataset. The setting is the same as the one used in Table 7. None of the predicted labels match the ground truth.

| Instance title: | Himalayan Cats (Barron's Complete Pet Owner's Manuals) |
|---|---|
| Top K recommendation | XR-Transformer + PINA-naive |
| 1 | Meditation and Its Practice |
| 2 | Science of Breath |
| 3 | Himalayan Crystal Salt Coarse Granulated - 1000 g - Salt |
| 4 | Original Himalayan Crystal Salt Inhaler |
| 5 | Himalayan Crystal Salt Fine Granulated - 1000 g - Salt |

## G. Qualitative analysis

**Qualitative analysis.** We further examine the predicted results of XR-Transformers and XR-Transformers combined with PINA (see Table 7). The considered instance is a book about caring for Himalayan cats and the ground truth labels are books about caring for cats. Labels returned by the XR-Transformer are all related to taking care of animals or plants. However, none of them relates to cats, which is the key concept of the ground truth labels. On the other hand, XR-Transformer + PINA successfully returns all three ground truth labels within its top 5 predictions. For further verification, we examine the predicted neighbors that are used in PINA augmentation. The top 2 neighbors are related to keeping not only Himalayan cats but also Persian Cats, which illustrates how PINA augmentation helps the XR-Transformer to provide better predictions. Note that the PINA neighborhood predictor $g$ is trained only on plain training data, while this example was chosen from the test data. Hence, the PINA neighborhood predictor $g$ saw neither the instance title nor its associated ground truth labels. Nevertheless, it did see the label text such as "Guide to Owning Himalayan Cat", which is critical for training a better text vectorizer as we mentioned in the previous section. As a comparison, none of the predicted labels from XR-Transformer + PINA-naive matches the ground truth ones (Table 8). This further demonstrates the importance of letting the text vectorizer observe the label text.

[†] http://manikvarma.org/downloads/XC/XMLRepository.html#ba-pair

## H. Additional Experiment results

We also examine the trained neighbor predictor $g$ on LF-Amazon131K. We find that 26% of the predicted result from $g$ are labels (74% are instances), and the corresponding P@1 (filtering out instances) is 33.72%. Recall that the vanilla XR-Transformer can achieve P@1 = 45.61%. This result shows that the gain of XR-Transformer+PINA is not merely due to being able to predict the correct labels, but the predicted instances also play an important role.

## I. Relation between PINA and the other methods

**Comparison with Barkan et al. 2020.** In the setting of Barkan et al. 2020, the authors assume that there is a given hierarchical label tree (HLT). That is, they assume the label (tags in Barkan et al. 2020) naturally has a viable hierarchical structure. This is similar to the XR-Transformer, instead, the HLT in the XR-Transformer is built based on hierarchical k-means instead of directly given in the dataset. The key idea in Barkan et al. 2020 is to "aggregate" the parent tag embeddings for all nodes in the HLT (for both items

(instances) and tags) and learn them simultaneously. Indeed, Barkan et al. 2020 shares the similar idea of "aggregating" embeddings from others as our PINA approach. However, we would like to emphasize that the main difference between Barkan et al. 2020 and our work is that Barkan et al. 2020 does not aggregate predicted neighbor embeddings. Indeed, from the example given in Barkan et al. 2020 (Figure 1), we can see that the "cold item" is already provided the tag information in the data. That is, we already know the parent tags of "QT8" in Figure 1-(a) and "Radio", "Diva" in Figure 1-(b) in the case of Barkan et al. 2020. Thus, the "cold item" in Barkan et al. 2020 can "aggregate" the parent tags representations. However, for a "test item" that does not have tags information (which is what we need to predict), the method in Barkan et al. 2020 can only predict the relevant tags for it based on its own embeddings. No neighborhood aggregation can be applied as the relevant tags are not included in the dataset (like test labels).

**Comparison to Wang et al. 2018.** Note that in [1], the side information is naturally provided in the form of a per-item fashion. For instance, see Section 2.4 in Wang et al. 2018, where $W_v^0$ is the original item embedding of item $v$ and $W_v^i, i \geq 1$ are the side information embedding of item $v$. Thus, it is easier to infuse this side information for each item, where the authors of Wang et al. 2018 take a weighted average (or concatenation) of them (see equations (6), (7), and Figure 3 in Wang et al. 2018). In contrast, both types of side information considered in our paper, label text and instance correlation signal, are not in a per-item fashion. It is impossible to directly add label text embedding to each instance without the help of our PINA formulation. Similarly, the instance correlation signal is not in the form of a per-instance (per-item) fashion, as it is of the form (bi)adjacency matrix (i.e. the instance-to-instance graph mentioned in Section 4.2). Also, note that it is prohibitive to directly concatenate the rows of the (bi)adjacency matrix to the instance features, as we potentially do not have the corresponding co-purchase information for the new instances. Thus, learning the neighbor predictor $g$ as in our PINA is crucial. Finally, while it might be possible to leverage the DeepWalk type of self-supervised loss for accounting the instance correlation signal, we would like to emphasize that it does not work for label text information. In contrast, our PINA method provides a unified way to leverage both types of side information (label text and instance correlation signal) under the XMC framework. That is, the pretraining objective is still an XMC problem so we can use the same XMC model for pretraining. This is the unique advantage of PINA in the XMC problem compared to not only Wang et al. 2018 but also prior works such as SiameseXML (Dahiya et al., 2021a).

**Comparison to Liu et al. 2021.** Similar to our discussion above, we note that the side information considered in Liu et al. 2021 is also in a per-user or per-item fashion. We would like to emphasize that the problem considered in Liu et al. 2021 is predicting the next item's ID that a user $u$ would buy based on the historical action of the user . Roughly speaking, the problem in Liu et al. 2021 wants to predict a user-to-item relation and thus the user in Liu et al. 2021 has a similar role of instance and the item in Liu et al. 2021 has a similar role of the label as in the XMC problem formulation. Thus, the approach in Liu et al. 2021 cannot take the instance correlation signal into account (i.e., user-to-user graph in the scenario of Liu et al. 2021). While both Liu et al. 2021 and our work consider the per-label side information, we would like to emphasize that the base model considered in Liu et al. 2021 and the XMC models are different. As we mentioned in section 2.3, most traditional XMC architectures treat labels as featureless identifiers. Thus, it is impossible to concatenate/append label text to labels as in Liu et al. 2021 for XMC models such as XR-Transformer.

## J. Pseudo Codes

**Algorithm 1** PINA augmentation

**Input:** neighbor predictor $g$, pretrained text vectorizer $\Phi_{\text{pre}}$, number of aggregated neighbors $K$, statistical text vectorizer $\Phi_{\text{stat}}$, instance text $\{x_i\}_{i\in[N+N_{test}]}$, label text $\{z_l\}_{l\in[L]}$.
**for** $i \in [N + N_{test}]$ **do**
  Construct ego feature $\mathbf{V}_i^{(0)} \leftarrow \Phi_{\text{pre}}(x_i)$
  Get predicted top-$K$ neighborhood prediction vector: $\mathbf{P}_i \leftarrow top_K(g(x_i, \Phi_{\text{stat}})) \in [0, 1]^{1\times(L+N)}$.
  Construct aggregated neighborhood feature: $\mathbf{V}_i^{(1)} \leftarrow \sum_{j:\mathbf{P}_{ij}>0} \mathbf{P}_{ij}\Phi_{\text{pre}}(h_j)$, where $h_j = z_j$ for $j \in [L]$ and $h_j = x_{j-L}$ for $j > L$.
  Normalization: $\mathbf{V}_i^{(0)} \leftarrow \frac{\mathbf{V}_i^{(0)}}{\|\mathbf{V}_i^{(0)}\|}$, $\mathbf{V}_i^{(1)} \leftarrow \frac{\mathbf{V}_i^{(1)}}{\|\mathbf{V}_i^{(1)}\|}$.
  Concatenate ego and neighborhood feature: $\mathbf{V}_i \leftarrow \left[\mathbf{V}_i^{(0)}, \mathbf{V}_i^{(1)}\right]$.
  Normalization: $\mathbf{V}_i \leftarrow \frac{\mathbf{V}_i}{\|\mathbf{V}_i\|}$.
**end for**
**Return** PINA feature: $\mathbf{V}$.

**Algorithm 2** Pretraining XMC: obtaining $g$ and $\Phi_{\text{pre}}$

**Input:** statistical text vectorizer $\Phi_{\text{stat}}$, train instance text $\{x_i\}_{i\in[N]}$, label text $\{z_l\}_{l\in[L]}$, train multi-label $\{\mathbf{y}_i\}_{i\in[N]}$.
Convert train multi-label to corresponding biadjacency matrix $\mathbf{B} \in \{0,1\}^{N\times L}$: $\mathbf{B}_i \leftarrow \mathbf{y}_i$.
Construct pretraining biadjacency matrix $\mathbf{B}_{pre} = \left[\mathbf{B}, \mathbf{I}; \mathbf{I}, \mathbf{B}^T\right]$.
Organize pretraining instances: $x_i^{(\text{pre})} \leftarrow x_i \ \forall i \in [N]$, $x_{N+l}^{(\text{pre})} \leftarrow z_l \ \forall l \in [L]$.
Organize pretraining labels: $\mathbf{y}_i^{(\text{pre})} \leftarrow \mathbf{B}_{pre,i} \ \forall i \in [N+L]$.
Train XR-Transformer with $\{x_i^{(\text{pre})}\}_{i=1}^{N+L}$, $\{\Phi_{\text{stat}}(x_i^{(\text{pre})})\}_{i=1}^{N+L}$ and $\{\mathbf{y}_i^{(\text{pre})}\}_{i=1}^{N+L}$, resulting model $g$ and its corresponding text vectorizer $\Phi_{\text{dnn}}$.
Concatenate dense and sparse text vectorizer: $\Phi_{\text{pre}}(\cdot) \leftarrow [\Phi_{\text{dnn}}(\cdot); \Phi_{\text{stat}}(\cdot)]$.
**Return** Neighbor predictor $g$, pretrained text vectorizer $\Phi_{\text{pre}}$.