Prototypical Extreme Multi-label Classification with a Dynamic Margin Loss

Kunal Dahiya[*†]
IIT Delhi
kunalsdahiya@gmail.com

Diego Ortego[*]
NielsenIQ
diego.ortego@nielseniq.com

David Jiménez
NielsenIQ
david.jimenez@nielseniq.com

Abstract

Extreme Multi-label Classification (XMC) methods predict relevant labels for a given query in an extremely large label space. Recent works in XMC address this problem using deep encoders that project text descriptions to an embedding space suitable for recovering the closest labels. However, learning deep models can be computationally expensive in large output spaces, resulting in a trade-off between high performing brute-force approaches and efficient solutions. In this paper, we propose PRIME, a XMC method that employs a novel prototypical contrastive learning technique to reconcile efficiency and performance surpassing brute-force approaches. We frame XMC as a data-to-prototype prediction task where label prototypes aggregate information from related queries. More precisely, we use a shallow transformer encoder that we coin as a Label Prototype Network, which enriches label representations by aggregating text-based embeddings, label centroids, and learnable free vectors. We jointly train a deep encoder and the Label Prototype Network using an adaptive triplet loss objective that better adapts to the high granularity and ambiguity of extreme label spaces. PRIME achieves state-of-the-art results in several public benchmarks of different sizes and domains, while keeping the model efficient. PRIME's code is available at: https://github.com/kunaldahiya/prime

1 Introduction

Extreme Multi-label Classification (XMC) is the task of predicting the most relevant subset of labels in an extremely large label space (potentially millions of labels) for a given query data point. XMC methods find applications in various real-world problems, including product recommendation in e-commerce, document tagging, and sponsored search. The label is typically endowed with a short description in such applications. For instance, in a product label space, *"Kerplunk!: Stories"* and *"The Good Samaritan Strikes Again"* are relevant labels for the query *"The Grasshopper Trap"*. In this example, relevancy is defined by products that were seen or bought together, while other applications might define different relevancy relations. The complete example is presented in Table 1. Note that all the relevant relations are often not available in the ground truth in such a large space, *i.e.*, there are missing labels.

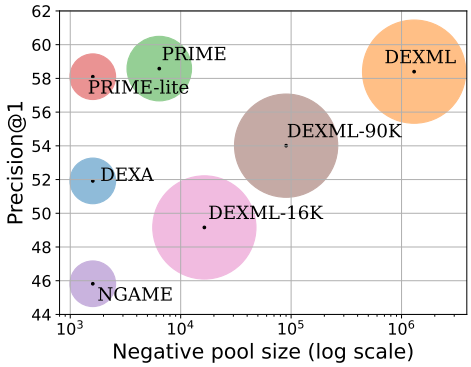

Figure 1: Performance vs efficiency comparison for several encoder-only methods in LF-AmazonTitles-1.3M dataset and our PRIME proposal. Blob size represents the models' batch size, y-axis performance, and x-axis number of negatives per query. Note that different versions of DEXML and PRIME vary the negative pool and the batch size, which dominate the method's efficiency.

XMC algorithms typically learn an encoder and extreme classifiers either jointly (Jain et al., 2023) or separately by following a modular approach (Dahiya et al., 2021b; Zhang et al., 2021b; Yadav et al., 2024). This modular approach allows for effortless utilization of label metadata in the form of text (Dahiya et al., 2021a), images (Mittal et al., 2022), or graphs (Saini et al., 2021), thereby enhancing encoder's accuracy. Therefore, learning a robust encoder has become a cornerstone for XMC methods.

[*]Equal contribution
[†]Work done as an intern at NielsenIQ

Query	Positive labels (×7)
"The Grasshopper Trap"	"How I Got This Way", "Circles in the Snow: A Bo Tully Mystery", "With Recipes and Commentaries", "Real Ponies Don't Go Oink!", "The Good Samaritan Strikes Again", "Kerplunk!: Stories.", "The Bear in the Attic"
DEXA predictions (×5)	
"The Ant and the Grasshopper: An Aesop's Fable", "Ant and Grasshopper", "Grasshopper" *(ID1)*, "Grasshopper" *(ID2)*, "Grasshopper" *(ID3)*.	
PRIME predictions (×5)	
"Kerplunk!: Stories", "The Good Samaritan Strikes Again" "The Tamarack Murders: A Bo Tully Mystery", "How I Got This Way", "The Bear in the Attic".	

Table 1: Qualitative example from LF-AmazonTitles-1.3M dataset. PRIME recovers semantically related products, while DEXA predicts irrelevant ones containing some query words. Red and green indicate incorrect and correct predictions, respectively.

Several recent works in the XMC literature focus on learning robust deep encoders using metric learning style of training (Dahiya et al., 2023a,b; Gupta et al., 2024; Mohan et al., 2024). Most of these methods follow a data-to-data metric learning approach, meaning that query and label sentence embeddings are directly obtained from their text descriptions. Although it is possible to obtain top performance using a data-to-data brute-force approach (Gupta et al., 2024) that incorporates all negative labels per query, this strategy is prohibitive in most real-world settings. In particular, this approach incurs a cost of $\mathcal{O}(KL)$ to compute query or label embeddings, where L is the number of labels, and K is the complexity of encoding the sentence embeddings (please see section E in the Appendix for further details). We stress that XMC algorithms should be designed both to achieve top-performance and scale well to large datasets.

In this paper, we introduce a novel method called PRIME, which efficiently learns a robust encoder by leveraging data-to-prototype relations. We are inspired by works demonstrating that embedding representations from groups of data points can narrow down metric learning complexity (Snell et al., 2017; Dopierre et al., 2021; Kim et al., 2020). In particular, we compute label embeddings using a novel Label Prototype Network that learns to aggregate text-based embeddings, label centroids, and learnable free vectors. We refer to the proposed aggregated representation as a label prototype as multiple queries are used for its estimation. This architectural design is a key difference with similar works such as NGAME (Dahiya et al., 2023a), DEXA (Dahiya et al., 2023b) or DEXML (Gupta et al., 2024), which either do not exploit any additional information than label text (Dahiya et al., 2023a; Gupta et al., 2024) or do not use queries nor a learned aggregation for computing label representations.

Additionally, extreme multi-label tasks have an intrinsic high granularity and ambiguity due to the variable hardness of positive and negative labels in such a large space. Then, for better adaptation to extreme label spaces, we propose to incorporate a dynamic margin in our training loss objectives. Note that related works based on similar objectives (Dahiya et al., 2023a,b; Mohan et al., 2024) do not account for this situation. We empirically show that this margin is a simple yet effective mechanism to boost XMC performance and theoretically demonstrate its effect (see Appendix A). More precisely, the proposed dynamic margin enables positive and negative labels to be projected closer in the embedding space when they are semantically related, while also considering uncertain cases with ambiguous samples or missing labels (a well-known limitation in extreme label spaces). As a result, PRIME can recover semantically related products more accurately than related methods (we refer the reader to Table 1 for a visual example). Furthermore, our experiments demonstrate that PRIME achieves state-of-the-art results within a single GPU budget and outperforms brute-force approaches defined in DEXML (Gupta et al., 2024) (encoder-only) and Renée (Jain et al., 2023) (extreme classifier-based). Note that solutions like DEXML use up to 16×A100 GPUs in some configurations.

Our main contributions are as follows:

- We propose a high-performing yet efficient XMC encoder-based method based on label prototypes that aggregate information from multiple queries.
- Our multi-objective optimization incorporates a novel adaptive triplet loss formulation that accounts for high granularity and uncertainty, inherent to extremely large label spaces.
- PRIME yields large performance improvements when compared with methods requiring comparable resources (see Figure 1). PRIME notably outperforms brute-force approaches in all experiments and for most metrics, while maintaining efficiency.

2 Related work

Extreme Multi-label Classification (XMC). Deep XMC methods (You et al., 2019; Kharbanda et al., 2022; Chien et al., 2023; Dahiya et al., 2023b; Gupta et al., 2024) have demonstrated that learning task-specific embeddings can yield significantly more accurate results as compared to traditional methods (see Appendix G for details). It is well established that jointly learning the encoder and the extreme classifiers can be expensive with a large number of labels (Dahiya et al., 2021b). In particular, the brute-force approach employed by Renée (Jain et al., 2023) can lead to accurate results, however, it is expensive to scale beyond a million labels even with multiple GPUs. Negative sampling approaches, *i.e.*, ANCE (Xiong et al., 2021) and NGAME (Dahiya et al., 2023a) narrow down the training complexity by selecting a small subset of hard negative labels. Moreover, several methods use a modular strategy (Dahiya et al., 2021b; Zhang et al., 2021b) that decouples the training of encoders and classifiers to improve scalability. ECLARE (Mittal et al., 2021), PINA (Chien et al., 2023), and XR-Transformers (Zhang et al., 2021a) train the encoder by learning meta-classifiers, *i.e.*, classifiers are learned for groups of labels. Conversely, NGAME (Dahiya et al., 2023b) proposed to learn a shared deep encoder in a Siamese fashion, where labels are represented via label embeddings. Building on NGAME, DEXA (Dahiya et al., 2023b) introduces auxiliary parameters to bridge the semantic gap, *i.e.*, the lack of information to accurately represent labels when their descriptions are short. On the other hand, DEXML (Gupta et al., 2024) demonstrates that Siamese deep encoders alone can theoretically yield state-of-the-art results using a brute-force approach with large batch sizes and complete label sets as negatives.

Deep metric learning (DML). Contrastive learning using the vanilla contrastive loss (Chopra et al., 2005), triplet loss (Schroff et al., 2015) or InfoNCE losses (Van den Oord et al., 2018) is the core of DML across NLP and computer vision. In NLP, they are widely used to learn robust sentence representations (Gao et al., 2021; Zhang et al., 2021a) or, more generally, multi-purpose embeddings with encoders (Wang et al., 2022) or decoder-only architectures (Muennighoff et al., 2024; Wang et al., 2024). As for XMC, DML has successfully been applied using deep encoders (Dahiya et al., 2023a,b; Mohan et al., 2024; Gupta et al., 2024). However, this approaches are based on data-to-data comparisons, which in XMC becomes specially complex, as demonstrated by the great benefits of increasing batch-wise comparisons (Gupta et al., 2024). However, the DML community has proposed alternatives to narrow down such complexity exploiting data-to-prototype (Zeng et al., 2022) or data-to-proxy (Kim et al., 2020) relations. Prototypes and proxies are both representative embeddings for groups of semantically similar instances (Kim et al., 2020; Zeng et al., 2022). The difference is that proxies are usually learned as network parameters (Kim et al., 2020; Ren et al., 2024), while prototypes are computed averaging embeddings of related instances (Snell et al., 2017; Dopierre et al., 2021; Song et al., 2022; Zeng et al., 2022). Therefore, a few set of proxies or prototypes can capture the global structure of an embedding space and replace data-to-data comparisons for reduced training complexity (Kim et al., 2020).

Adaptive triplet loss. The triplet loss (Schroff et al., 2015) is a widely used alternative for metric learning across many fields (Schroff et al., 2015; Liu et al., 2021a; Nguyen et al., 2022). This loss function ensures that anchor-positive label relations have higher similarity than anchor-negative label ones, while including a margin to force a sufficiently high similarity gap. Despite methods in XMC using a fixed margin that lacks flexibility to address ambiguous situations (Dahiya et al., 2023a,b; Mohan et al., 2024), some works in the wider metric learning community have proposed dynamic margins to tackle this limitation. For example, (Liu et al., 2021b) propose a margin function for measuring more detailed similarities between taxonomy paths within the context of self-supervised taxonomy expansion. In forensic medical image matching, the authors in (Nguyen et al., 2022) propose an auto-margin strategy based on exploiting similarity statistics over time. Similarly, (So et al., 2023) work improves the triplet loss by adapting the margin based on the interpolation strength of mixed samples.

Decoder models. Decoder-only models provide significant benefits over encoder-only ones in multiple natural language problems including entity retrieval (De Cao et al., 2021), natural question answering (Wang et al., 2022) or the MTEB benchmark (Muennighoff et al., 2024; Wang et al., 2024). XLGen (Jung et al., 2023) and QUEST (Zhou et al., 2024) have applied decoder-only models in extreme classification. However, their perfor-

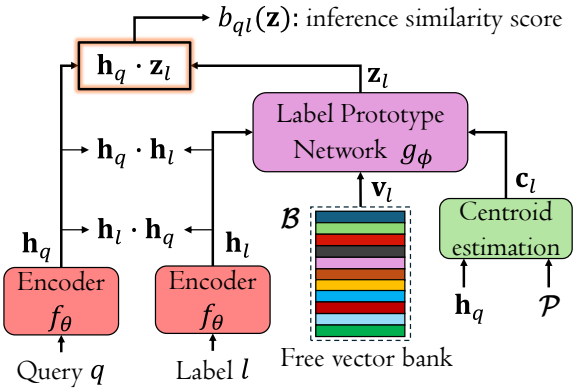

Figure 2: Overview of PRIME - PRototypIcal extreME multi-label classification. Our method exploits query-to-prototype ($\mathbf{h}_q$ to $\mathbf{z}_l$), query-to-label ($\mathbf{h}_q$ to $\mathbf{h}_l$) and label-to-query ($\mathbf{h}_l$ to $\mathbf{h}_q$) embedding relations for robust encoder training. The Label Prototype Network g_ϕ computes enhanced label embeddings $\mathbf{z}_l$ (label prototypes) by aggregating label-text embeddings $\mathbf{h}_l$, label centroids $\mathbf{c}_l$ and learnable free vectors $\mathbf{v}_l$. As a result, the query-to-prototype similarity score $b_{ql}(\mathbf{z})$ accurately predicts the relevant labels for a given query.

mance is still far from current state-of-the-art models in XMC. A detailed investigation is required to understand the underlying reasons for this limitation. Nonetheless, decoder-only models remain a promising research direction to follow in XMC.

3 Method

We propose PRIME, a novel method for XMC based on label prototypes rather than the usual text-based label embeddings from related works (Dahiya et al., 2023a; Gupta et al., 2024; Mohan et al., 2024). PRIME computes data-to-prototype relations and uses a dynamic margin in its triplet loss objectives, achieving state-of-the-art results on multiple benchmarks. The overview of PRIME's architecture is presented in Figure 2.

Setup: Consider $\mathcal{D} = \{q_i, \mathcal{P}_i\}_{i=1}^{Q}$ be a training set with a set of Q queries $\mathcal{Q} = \{q_i\}_{i=1}^{Q}$, where each query q_i has a positive label set $\mathcal{P}_i$. We define the space of L labels $\mathcal{Y} = \{l_r\}_{r=1}^{L}$ for any q_i to be a super-set of the positive $\mathcal{P}_i = \{p_j\}_{j=1}^{P_i}$ and the negative sets $\mathcal{N}_i = \{n_k\}_{k=1}^{N_i}$. P_i and N_i are the number of positive and negative labels associated to q_i, respectively. In practice, we differentiate two types of approaches depending on the definition of $\mathcal{N}_i$: (i) brute-force, which use the complete set of negatives for every query (Gupta et al., 2024); (ii) negative mining, which sample a small subset of hard negative labels during training to strike a balance between accuracy and efficiency (Xiong et al., 2021; Dahiya et al., 2023a).

We pose XMC as a maximum inner product search task between query and label embeddings to accurately predict the positive labels for every query q_i. To accomplish this, we aim to learn a function $f_\theta : (\mathcal{Q}, \mathcal{Y}) \rightarrow \mathbb{R}^d$, where θ denotes the parameters of a neural network model that encodes the given textual representation into a d-dimensional sentence embedding. This function is used to encode every query q_i and label l_r into $\mathbf{h}_q^i$ and $\mathbf{h}_l^r$. Note that $\mathbf{h}_l$ can be defined as $\mathbf{h}_p$ and $\mathbf{h}_n$ to distinguish between positive and negative text-based label embeddings, respectively.

Existing Siamese training for XMC (Dahiya et al., 2023a; Mohan et al., 2024) directly imposes a data-to-data triplet loss objective as follows:

$$\mathcal{L}_T = \sum_{i=1}^{B} \sum_{\substack{j \in \mathcal{P}_i \\ k \in \mathcal{N}_i}} \max(\mathbf{h}_q^i \cdot \mathbf{h}_n^k - \mathbf{h}_q^i \cdot \mathbf{h}_p^j + m, 0), \quad (1)$$

where B is the number of batch queries, m is a fixed margin and $\mathbf{h}$ refers to L2-normalized embeddings. We simplify the notation in Eq. 1 and refer to $\mathcal{L}_T(\mathbf{h}_q, \mathbf{h}_l)$ as $\mathcal{L}_T$. Also, Eq. 1 introduces abuse of notation when using the set of negative labels $\mathcal{N}_i$ as the subset of negatives used for each query.

For simplicity, in the remaining of the paper we remove the superscript for query and label indexes and re-define $\mathcal{L}_T$ using similarity scores as:

$$\begin{aligned} \mathcal{L}_T &= \max\left(s_{qn} - s_{qp} + m, 0\right) \\ &= \begin{cases} \Delta s_q^{n-p} + m, & \text{if } \Delta s_q^{p-n} \leq m \\ 0, & \text{otherwise} \end{cases}, \end{aligned} \quad (2)$$

where $s_{qn} = \mathbf{h}_q \cdot \mathbf{h}_n$ and $s_{qp} = \mathbf{h}_q \cdot \mathbf{h}_p$ denote the query-negative and the query-positive cosine similarities, respectively. We denote $\Delta s_q^{n-p} = s_{qn} - s_{qp}$ and $\Delta s_q^{p-n} = s_{qp} - s_{qn}$ as the differences of the cosine similarities between the query-positive and query-negative tuples, respectively.

3.1 Dynamic margin

The traditional triplet loss formulation presented in Eq. 2 imposes a fixed margin m during training such that all negatives are pushed apart from the query, while positives are pulled close given the condition $\Delta s_q^{p-n} \leq m$. Despite being possible to define margins that work well in practice, not all triplets are equally hard, thus suggesting that

fixed values might be suboptimal. To overcome this limitation, we propose a dynamic margin that provides adaptation to each triplet hardness.

Proposition 1. *Consider the non-differentiable piece-wise linear function defining the adaptive margin to be* $m\left(s_{ap}, s_{an}\right) = |s_{ap} - s_{an}|$. *Adding* $m\left(s_{ap}, s_{an}\right)$ *into Eq. 8 expands the support of the function by relaxing the margin constraint of the original triplet formulation, inducing a modulation effect that allows semantically similar representations to be projected closer in the embedding space.*

Intuitively, this modulation of the margin enables a more informative embedding space by accounting for degrees of similarities between positives and negatives, *i.e.*, the higher the semantic similarity, the closer they will project. Conversely, for high differences where negatives exhibit a much higher similarity than positives, we impose a large penalty.

Proposition 2. *Clipping the values of the dynamic margin partitions the triplet loss landscape allowing positives and negatives in uncertain settings to be projected nearby in the embedding space.*

Consider $\text{clip}(x)$ be the clipping function,

$$\text{clip}(x) = \begin{cases} x, & \text{if} \quad \gamma_{\text{min}} \leq x \leq \gamma_{\text{max}} \\ \gamma_{\text{min}}, & \text{if} \quad x < \gamma_{\text{min}} \\ \gamma_{\text{max}}, & \text{if} \quad x > \gamma_{\text{max}} \end{cases},$$

and consider the conditions for query-to-label similarities $\mathcal{C}_p : s_{qp} > s_{qn}$ and $\mathcal{C}_n : s_{qn} > s_{qp}$ be the inequality conditions of the dynamic margin defined in Proposition 1. Then, the triplet loss with clipped dynamic margin $\mathcal{L}_T^{cd}$ becomes:

$$\mathcal{L}_T^{cd} = \begin{cases} 0, & \text{if } \mathcal{C}_p, \ \Delta_q^{p-n} \geq \gamma_{\text{min}} \\ \Delta_q^{p-n} + \gamma_{min}, & \text{if } \mathcal{C}_p, \ \Delta_q^{p-n} < \gamma_{\text{min}} \\ \Delta_q^{n-p} + \text{clip}\left(\Delta s_q^{n-p}\right), & \text{if } \mathcal{C}_n \end{cases}. \tag{3}$$

In Eq. 3 we show that the addition of the dynamic margin removes the fixed margin constraint of Eq. 1 enabling the model to learn from any observation that satisfies condition $\mathcal{C}_n$, where negatives are closer to the query than positives. Besides, the proposed formulation introduces a new learning region where the gradient direction is inverted: $\mathcal{C}_p, \ \Delta_q^{p-n} < \gamma_{\text{min}}$. We argue that this region, where positives and negatives are nearly equally distant to the query, covers cases with high uncertainty such as fine-grain differences or missing labels, which represent well-known challenges in extreme multi-label setups. For these triplets, negatives and positives could be, respectively, missing labels and false positives. Therefore, by reverting the learning process in this small margin, the network does not take risks and separates positives, while pulling negatives close to the query. Note that if the negative had higher similarity than the positive, this behaviour would not continue. We refer the reader to Appendix A for a detailed analysis with proofs of the propositions.

3.2 Label prototypes

We propose a prototypical contrastive learning method aiming at narrowing down data-to-data complexity during training, where we build label prototypes as enhanced label embeddings. We do so by using a Label Prototype Network g_ϕ that aggregates three sources of information: label-text embeddings $\mathbf{h}_l$, label centroids $\mathbf{c}_l$ and learnable free vectors $\mathbf{v}_l$ (see Figure 2 for reference). The resulting aggregation produces the label prototype,

$$\mathbf{z}_l = g_\phi\left(\mathbf{h}_l, \mathbf{c}_l, \mathbf{v}_l\right).$$

In particular, we use a transformer encoder block layer (Vaswani et al., 2017) for g_ϕ to make the three sources interact via self-attention and build the label prototype via mean pooling of the resulting contextualized embeddings. These label prototypes reduce the complexity of data-to-data relations, as they are estimated from centroids and free vectors, which contain an aggregated information from queries and labels not present when solely using text-based label embeddings.

We compute label centroids using a momentum-based approach that updates its values using the batch queries containing the label at hand by:

$$\mathbf{c}_p \leftarrow \alpha\, \mathbf{c}_p + (1 - \alpha)\, \mathbf{h}_q^i, \tag{4}$$

where α is the momentum coefficient and $\mathbf{h}_q^i$ is the i-th query embedding such that $p \in \mathcal{P}_i$. We smoothly update the centroids using a high α value.

Learnable free or auxiliary vectors are normally used in the literature to model side-information that can be complementary to text descriptions, e.g. for modeling attributes in dense retrieval (Kong et al., 2022; Sun et al., 2024) and when modeling groups of labels (Dahiya et al., 2023b) in XMC. We adopt the latter strategy and define a bank of free vectors $\mathcal{B}$ where every free vector $\mathbf{v}_l \in \mathcal{B}$ is shared across a cluster $\mathcal{T}_c = \{l_o\}_{o=1}^{|\mathcal{T}_c|}$ of semantically similar labels,

i.e., the free vector $\mathbf{v}_l$ is the same $\forall l \in \mathcal{T}_c$. The clustering assignment is conducted before training by finding a partition $\mathcal{J}$ of the label space $\mathcal{Y}$ such that $\mathcal{J} = \{\mathcal{T}_c\}_{c=1}^{|\mathcal{J}|}$. In particular, we follow (Dahiya et al., 2023b) and use a balanced hierarchical 2-means clustering of label text representations $\mathbf{h}_l$ obtained with the pre-trained encoder. Note that the number of free vectors $|\mathcal{B}| << L$ to ensure scalability. These free vectors are learned during training as gradients flow through query-to-prototype relations, thus providing an additional degree of freedom that does not depend directly on the text of queries or labels. Note that the authors in (Dahiya et al., 2023b) directly add $\mathbf{v}_l$ to $\mathbf{h}_l$, limiting the contribution of free vectors to a linear combination. Conversely, we integrate free vectors as one of the building components used by our Label Prototype Network, thus learning how to incorporate free vectors into the label prototypes for better performance.

3.3 Encoder and prototype network training

We jointly train the encoder and the Prototype Label Network using a multi-objective optimization. We use the proposed dynamic triplet loss function $\mathcal{L}_T^{cd}(\mathbf{h}_q, \mathbf{z}_l)$ to optimize query-to-prototype relations, which is how we compute our similarity scores at inference time as shown in Figure 2. This objective does not directly focus on the optimization of text-based embeddings $\mathbf{h}$, which is important for learning a strong backbone. Therefore, we propose to include two additional objectives, $\mathcal{L}_T^{cd}(\mathbf{h}_q, \mathbf{h}_l)$ and $\mathcal{L}_T^{cd}(\mathbf{h}_l, \mathbf{h}_q)$, that promote query-to-label and label-to-query relations in the text-based embedding space, respectively. This type of backbone optimization has proved to be effective for dense retrieval (Li et al., 2021; Mesquita et al., 2022) as it increases the data-to-data relations exploited during training and learns a better embedding space. Our final loss is then defined as:

$$\mathcal{L} = \mathcal{L}_T^{cd}(\mathbf{h}_q, \mathbf{z}_l) + \mathcal{L}_T^{cd}(\mathbf{h}_q, \mathbf{h}_l) + \mathcal{L}_T^{cd}(\mathbf{h}_l, \mathbf{h}_q) + \lambda\mathcal{R}, \tag{5}$$

where λ is the contribution of the regularization term $\mathcal{R}$. We bring this regularization from authors in (Mohan et al., 2024), which regularize similarities from a cross-attention module that incorporates meta-data information into query embeddings to be better than similarities produced without the meta-data. We apply the same concept in our approach to enforce query-to-prototype similarities to be better than query-to-label ones, as they are enriched with centroids and free vectors. We define $\mathcal{R}$ as the average of $\mathcal{R}_p$ and $\mathcal{R}_n$:

$$\mathcal{R}_p = \frac{1}{P}\sum_{p\in\mathcal{P}_i} s_{qp} - b_{qp} + m', \tag{6}$$

$$\mathcal{R}_n = \frac{1}{N}\sum_{n\in\mathcal{N}_i} b_{qn} - s_{qn} + m', \tag{7}$$

where $\mathcal{R}_p$ and $\mathcal{R}_n$ are the positive and negative similarity differences, m' is a fixed margin and $b_{ql} = h_q \cdot z_l$ is the query-to-prototype similarity for label l, i.e. a positive p or negative n label. Note that $\mathcal{P}_i$ and $\mathcal{N}_i$ are the number of positive and negative comparisons in the batch, respectively.

4 Experiments

4.1 Datasets and metrics

We benchmark our method on 6 publicly available XMC datasets (Bhatia et al., 2016) (details in Table 7 of Appendix B) covering: product recommendation from LF-AmazonTitles-131K, LF-Amazon-131K and LF-AmazonTitles-1.3M; the prediction of relevant Wikipedia categories in LF-WikiTitles-500K and LF-Wikipedia-500K; and predicting similar Wikipedia articles in LF-WikiSeeAlso-320K. Note that these datasets include both short and long-text settings, demonstrating the applicability of PRIME in diverse contexts.

Our main focus is to compare with encoder-only approaches: DPR (Karpukhin et al., 2020), ANCE (Xiong et al., 2021), SiameseXML (Dahiya et al., 2021a), GraphFormer (Yang et al., 2021), NGAME (Dahiya et al., 2023a), DEXA (Dahiya et al., 2023b) and DEXML (Gupta et al., 2024). However, we also benchmark PRIME with classifier methods: XR-Transformer (Zhang et al., 2021b), LightXML (Jiang et al., 2021), ELIAS (Gupta et al., 2022), CascadeXML (Kharbanda et al., 2022), PINA (Chien et al., 2023) and Renée (Jain et al., 2023). We mainly adopt precision and propensity-scored precision (P@k and PSP@k) metrics defined at (Bhatia et al., 2016) for evaluation.

4.2 Implementation details

Hard negative and positive sampling. Triplet selection involving hard negatives and positives can have a significant impact on performance. We balance the visualization of positive labels by sampling with a distribution over propensity scores of labels (see Appendix J for more details). Moreover,

we sample two positives per query, unless otherwise stated. Please, note that the memory and compute requirements increase with the number of sampled positives (see Appendix E and F for more details on complexity). On the other hand, we adopt the efficient NGAME (Dahiya et al., 2023a) negative sampling strategy, which constructs batches in such a way that inexpensive in-batch sampling offers semi-hard negatives.

Training and inference. We adopt common practices and configurations from previous works. To keep the comparison fair, we adopt the 66M parameter DistilBERT (Reimers and Gurevych, 2019) encoder used in other papers and report results for a single run. We compute sentence embeddings by mean pooling token representations. We use maximum inner product search between queries and label prototype embeddings at inference time. We refer the reader to Appendix H and I for further details. Please, note that despite using triplet loss objectives in our loss terms, PRIME could also be optimized using InfoNCE type of losses as done in (Gupta et al., 2024), however, we leave this study for future work. Finally, we set all PRIME configurations to fit in a single 80 GB A100 GPU.

4.3 Results

In tables 2 and 3 we present a comparative evaluation of PRIME against most relevant encoder-based XMC algorithms on Amazon and Wikipedia datasets, respectively. Our results demonstrate that PRIME outperforms recent works (ANCE, DEXA and DEXML) in all experiments and for most metrics. In particular, PRIME achieves state-of-the-art performance in the largest and most challenging dataset LF-AmazonTitles-1.3M, outperforming the brute-force approach DEXML using $16\times$ more computational resources, and surpassing by more than 6 points in P@1 the rest of the methods. We extend our comparison with brute force approaches in Subsection 4.4 (see Figure 1 for a visual comparison).

PRIME's primary objective revolves around efficiency in XMC models and thus our focus on classifier-free encoder-based approaches. Nevertheless, learning classifiers on top of PRIME's embeddings (we follow NGAME's (Dahiya et al., 2023a) One-vs-All approach) might provide moderate gains over the encoder-only model. For completeness, we present results using extreme classifiers in Table 4 on the largest Amazon and Wikipedia datasets, demonstrating the superior performance of our PRIME proposal. Note that Renée can be seen as a counterpart of DEXML as it learns the extreme classifiers in a brute-force fashion, *i.e.*, without negative sampling. The results on all datasets are included in Appendix C.

Method	P@1	P@3	P@5	PSP@5
LF-AmazonTitles-1.3M				
DPR	44.64	39.05	34.83	36.72
ANCE	46.44	41.48	37.59	37.25
SiameseXML	43.80	38.60	34.94	28.48
NGAME	45.82	39.94	35.48	36.80
DEXA	51.92	44.01	38.86	<u>37.31</u>
DEXML ✦	<u>58.40</u>	-	**45.46**	36.58
PRIME	**58.58**	**50.83**	<u>45.44</u>	**39.07**
LF-AmazonTitles-131K				
GraphFormers	20.84	13.57	10.06	24.93
DPR	41.85	28.71	20.88	49.45
ANCE	42.67	29.05	20.98	49.03
SiameseXML	41.42	27.92	<u>21.21</u>	46.19
NGAME	42.61	28.86	20.69	48.71
DEXA	<u>44.76</u>	<u>29.72</u>	21.18	<u>49.50</u>
DEXML ✦	42.52	-	20.64	47.40
PRIME	**44.87**	**30.06**	**21.53**	**49.73**
LF-Amazon-131K				
DPR	43.30	29.74	21.90	<u>51.52</u>
ANCE	44.87	30.31	21.89	50.12
NGAME	45.35	29.89	21.35	49.32
DEXA	<u>46.64</u>	<u>30.93</u>	<u>22.06</u>	50.38
DEXML ✦	-	-	-	-
PRIME	**48.09**	**32.39**	**23.34**	**53.43**

Table 2: Classifier-free evaluation on product recommendation.✦ denotes brute-force algorithm. Bold and underlined numbers denote best and second best results, respectively. '-' indicates results not available.

4.4 PRIME vs brute-force DEXML

DEXML (Gupta et al., 2024) has demonstrated that, in the absence of classifiers, it is possible to boost XMC performance when using the brute-force approach of pairing every query with all negative labels in the dataset. However, this setup requires substantially more resources than the rest of the algorithms compared in Subsection 4.3.

In Table 5, we show that PRIME outperforms DEXML for all relaxed setups, while requiring substantially less resources. Notably, PRIME achieves better (Amazon data) or competitive (Wikipedia data) performance than the most computationally demanding configuration using $\sim$2.5 orders of magnitude more negatives and $2.5-4\times$ bigger

Method	P@1	P@3	P@5	PSP@5
LF-Wikipedia-500K				
GraphFormers	31.10	-	14.00	21.83
DPR	65.23	45.85	35.23	49.90
ANCE	63.33	43.35	33.12	39.71
SiameseXML	50.33	32.81	24.86	32.51
NGAME	77.92	54.87	40.95	57.33
DEXA	79.99	57.08	42.52	57.68
DEXML ✦	**85.78**	-	50.53	**58.97**
PRIME	85.37	**65.56**	**50.92**	57.74
LF-WikiTitles-500K				
GraphFormers	24.53	-	11.33	19.53
ANCE	29.68	-	12.51	21.18
NGAME	29.68	18.06	12.51	21.18
DEXA	34.76	20.88	14.39	23.83
DEXML ✦	-	-	-	-
PRIME	**46.33**	**25.98**	**18.09**	**24.03**
LF-WikiSeeAlso-320K				
DPR	41.66	27.16	20.66	36.25
ANCE	44.35	29.15	21.99	37.15
SiameseXML	40.70	27.16	20.74	35.67
NGAME	43.58	28.01	20.86	36.03
DEXA	46.57	29.92	22.26	38.27
DEXML ✦	-	-	-	-
PRIME	**48.00**	**31.41**	**23.53**	**40.20**

Table 3: Classifier-free evaluation on Wikipedia category and similar page prediction.✦ denotes brute-force algorithm and bold and underlined best and second best. '-' indicates that results are not available.

Method	P@1	P@5	PSP@1	PSP@5
LF-AmazonTitles-1.3M				
XR-Transf. (**W**)	50.14	39.98	20.06	27.79
CascadeXML (**W**)	47.82	38.31	17.17	24.76
NGAME (**W**)	54.69	42.80	30.01	35.29
DEXA (**W**)	55.76	42.96	29.12	34.86
Renée (**W**) ✦	56.04	45.32	28.54	36.14
PRIME (θ)	58.58	45.44	**32.14**	**39.07**
PRIME (**W**)	**59.62**	**46.75**	31.20	38.64
LF-Wikipedia-500K				
CascadeXML(**W**)	81.13	49.12	32.12	49.37
LightXML(**W**)	81.59	47.64	31.99	46.53
ELIAS(**W**)	81.26	48.82	35.02	51.13
PINA (**W**)	82.83	50.11	-	-
NGAME (**W**)	84.01	49.97	41.25	57.04
DEXA (**W**)	84.92	50.51	42.59	58.33
Renée (**W**) ✦	84.95	**51.68**	39.89	56.70
OAK (θ, **W**) ✧	85.23	50.79	**45.28**	**60.80**
PRIME (θ)	85.37	50.92	40.60	57.74
PRIME (**W**)	**85.75**	51.58	40.29	58.61

Table 4: Comparative evaluation against XMC methods using OVA classifiers (**W**). Note that (θ, **W**) methods ensemble the encoder and classifier predictions. ✦ denotes brute-force algorithm and ✧ algorithm using extra meta-data information. Bold and underlined denote best and second best.

batch sizes. Furthermore, as we can observe from Table 6 (PRIME-lite vs PRIME) it is reasonable to think that increasing PRIME's batch size and number of positives would improve performance, while still using much less resources than DEXML. However, our focus in this work is to propose a high performing method that can run in a single GPU, aiming at keeping the scaling capabilities needed in XMC.

4.5 Understanding PRIME's components

In this section we analyze key components of our Label Prototype Network and the loss function. We run a light configuration of PRIME (lite) for faster experimentation (one positive and half batch size).

Figure 3 depicts the performance of PRIME-lite compared to variations when ablating free vectors (w/o v_l) and centroids (w/o c_l). While using free vectors clearly boosts final performance, the addition of centroids have a remarkable impact on faster convergence reaching DEXA's final performance $5\times$ faster. We further analyze the impact of the number of free vectors in Appendix D. As expected, we observe a positive trend in performance when we increase the number of free vectors.

We analyze in Table 6 the impact of the different terms in PRIME's loss function (Equation 5). Results obtained demonstrate that: 1) although we use label prototypes at inference time, optimizing text-based query-to-label and label-to-query relations is crucial for higher encoder quality; and 2) adopting the proposed dynamic margin m_d and the regularization consistently improve the results. Finally, increasing the batch size and adding one positive during training achieves best performance.

5 Conclusion

This paper proposes PRIME, a novel prototypical extreme multi-label classification method demonstrating that encoder-based models do not need to sacrifice efficiency to deliver state-of-the-art performance. PRIME's key contributions revolve around two key concepts, leveraging dynamic margin-based contrastive learning and label prototypes. The proposed Label Prototype Network learns to efficiently aggregate information from text-based em-

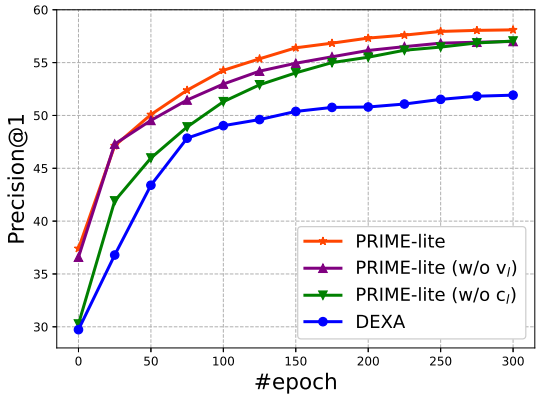

Figure 3: Impact of key components of the Label Prototype Network, *i.e.*, centroids ($\mathbf{c}_l$) and free vectors ($\mathbf{v}_l$) in LF-AmazonTitles-1.3M dataset. For convenience, we report results for PRIME-lite, single positive and half batch size variant of PRIME.

Method	Batch size	Negative pool size	P@1	PSP@5
LF-AmazonTitles-1.3M				
DEXML	8192	$\sim$ 1.3M	58.40	36.58
		$\sim$ 90K	54.01	-
		$\sim$ 16K	49.16	-
PRIME	3200	$\sim$ 6.4K	58.58	39.07
LF-Wikipedia-500K				
DEXML	2048	$\sim$ 501K	85.78	58.97
		$\sim$ 22K	84.77	-
		$\sim$ 4K	82.85	-
PRIME	512	$\sim$ 1K	85.37	57.74

Table 5: Comparison of PRIME and DEXML demonstrating the superiority of PRIME under similar budget.

PRIME-lite variations	P@1	PSP@1	R@100
w/o $[\mathcal{L}_T^{cd}(\mathbf{h}_q, \mathbf{h}_l), m_d, \mathcal{L}_T^{cd}(\mathbf{h}_l, \mathbf{h}_q), \mathcal{R}]$	51.71	26.27	57.02
w/o $[m_d, \mathcal{L}_T^{cd}(\mathbf{h}_l, \mathbf{h}_q), \mathcal{R}]$	56.34	31.23	60.79
w/o $[\mathcal{L}_T^{cd}(\mathbf{h}_l, \mathbf{h}_q), \mathcal{R}]$	57.40	29.69	62.30
w/o $[\mathcal{R}]$	58.25	30.83	62.78
PRIME-lite	58.10	31.68	63.45
PRIME	**58.58**	**32.14**	**63.56**

Table 6: Ablation study in LF-AmazonTitles-1.3M over various components of PRIME's loss function.

beddings, label centroids and learnable free vectors resulting in highly informative label prototypes for improved results and faster convergence to high performances. Additionally, by equipping our multi-objective contrastive optimization with PRIME's novel dynamic margin loss, we demonstrate better adaptability to the high granularity and ambiguity posed by extreme label spaces. Our experiments show that PRIME outperforms previous works and the leading brute-force approaches in all experiments and for most metrics.

6 Limitations

Our PRIME approach is based on encoder-only architectures, however, decoder-only models have recently demonstrated to be powerful embedding extractors (Muennighoff et al., 2024; Wang et al., 2024) and might have the potential to boost XMC performance. However, based on the results of decoder-only methods for XMC (Jung et al., 2023; Zhou et al., 2024), it is still an open problem how to achieve comparable performance to encoder-only models. In addition, XMC revolves around efficiency and, even if this decoder-only methods can benefit from efficient fine-tuning, both training and inference time would require higher computational demands.

Furthermore, while PRIME demonstrates top performance on standard XMC baselines, testing it on larger datasets would be interesting to understand scaling limitations when going beyond 1.3M labels, *e.g.*, how to define the size of the free vector bank. Additionally, as most related work do, we overlook dealing with unseen labels and we see it as a limitation that future work should address.

Acknowledgements

The authors want to thank Alejandro Barón for the fruitful discussions during the development of this work. Grateful for the invaluable guidance, inspiration, and friendship of Dr. Kevin McGuinness, Diego Ortego wants to honour his legacy. His memory continues to inspire those who were fortunate to know him. Kevin, we miss you.

References

R. Babbar and B. Schölkopf. 2017. DiSMEC: Distributed Sparse Machines for Extreme Multi-label Classification. In *ACM International Conference on Web Search and Data Mining (WSDM)*.

R. Babbar and B. Schölkopf. 2019. Data scarcity, robustness and extreme multi-label classification. *Machine Learning (ML)*.

K. Bhatia, K. Dahiya, H. Jain, P. Kar, A. Mittal, Y. Prabhu, and M. Varma. 2016. The Extreme Classification Repository: Multi-label Datasets & Code.

K. Bhatia, H. Jain, P. Kar, M. Varma, and P. Jain. 2015. Sparse Local Embeddings for Extreme Multi-label

Classification. In *Advances in Neural Information Processing Systems (NeurIPS)*.

E. Chien, C.-J. Zhang, J. Hsieh, J.-Y. Jiang, W.-C. Chang, O. Milenkovic, and H.-F. Yu. 2023. Pina: Leveraging side information in extreme multi-label classification via predicted instance neighborhood aggregation. In *International Conference on Machine Learning (ICML)*.

S. Chopra, R. Hadsell, and Y. LeCun. 2005. Learning a similarity metric discriminatively, with application to face verification. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*.

K. Dahiya, A. Agarwal, D. Saini, K. Gururaj, J. Jiao, A. Singh, S. Agarwal, P. Kar, and M. Varma. 2021a. SiameseXML: Siamese Networks meet Extreme Classifiers with 100M Labels. In *International Conference on Machine Learning (ICML)*.

K. Dahiya, N. Gupta, D. Saini, A. Soni, Y. Wang, K. Dave, J. Jiao, K. Gururaj, P. Dey, A. Singh, D. Hada, V. Jain, B. Paliwal, A. Mittal, S. Mehta, R. Ramjee, S. Agarwal, P. Kar, and M. Varma. 2023a. Ngame: Negative mining-aware mini-batching for extreme classification. In *ACM International Conference on Web Search and Data Mining (WSDM)*.

K. Dahiya, D. Saini, A. Mittal, A. Shaw, K. Dave, A. Soni, H. Jain, S. Agarwal, and M. Varma. 2021b. DeepXML: A Deep Extreme Multi-Label Learning Framework Applied to Short Text Documents. In *ACM International Conference on Web Search and Data Mining (WSDM)*.

K. Dahiya, S. Yadav, S. Sondhi, D. Saini, S. Mehta, J. Jiao, S. Agarwal, P. Kar, and M. Varma. 2023b. Deep encoders with auxiliary parameters for extreme classification. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*.

N. De Cao, G. Izacard, S. Riedel, and F. Petroni. 2021. Autoregressive entity retrieval. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*.

T. Dopierre, C. Gravier, and W. Logerais. 2021. ProtAugment: Unsupervised diverse short-texts paraphrasing for intent detection meta-learning. In *Annual Meeting of the Association for Computational Linguistics (ACL)*.

T. Gao, X. Yao, and D. Chen. 2021. SimCSE: Simple Contrastive Learning of Sentence Embeddings. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

N. Gupta, P. H. Chen, H.-F. Yu, Cho-J. Hsieh, and I. S. Dhillon. 2022. Elias: End-to-end learning to index and search in large output spaces. In *Advances in Neural Information Processing Systems (NeurIPS)*.

N. Gupta, D. Khatri, A.S. Rawat, S. Bhojanapalli, P. Jain, and I. Dhillon. 2024. Dual-Encoders for Extreme Multi-Label Classification. In *International Conference on Learning Representations (ICLR)*.

H. Jain, V. Balasubramanian, B. Chunduri, and M. Varma. 2019. Slice: Scalable Linear Extreme Classifiers trained on 100 Million Labels for Related Searches. In *ACM International Conference on Web Search and Data Mining (WSDM)*.

H. Jain, Y. Prabhu, and M. Varma. 2016. Extreme Multi-label Loss Functions for Recommendation, Tagging, Ranking and Other Missing Label Applications. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*.

V. Jain, J. Prakash, D. Saini, J. Jiao, R. Ramjee, and M. Varma. 2023. Renée: End-to-end training of extreme classification models. In *Conference on Machine Learning and Systems (MLSys)*.

T. Jiang, D. Wang, L. Sun, H. Yang, Z. Zhao, and F. Zhuang. 2021. LightXML: Transformer with Dynamic Negative Sampling for High-Performance Extreme Multi-label Text Classification. In *Association for the Advancement of Artificial Intelligence Conference on Artificial Intelligence (AAAI)*.

J. Johnson, M. Douze, and H. Jégou. 2019. Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data*.

T. Jung, J.-k. Kim, S. Lee, and D. Kang. 2023. Cluster-guided label generation in extreme multi-label classification. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*.

V. Karpukhin, B. Oguz, S. Min, P. Lewis, L. Wu, S. Edunov, D. Chen, and W.-T. Yih. 2020. Dense passage retrieval for open-domain question answering. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

S. Kharbanda, A. Banerjee, E. Schultheis, and R. Babbar. 2022. Cascadexml: Rethinking transformers for end-to-end multi-resolution training in extreme multi-label classification. In *Advances in Neural Information Processing Systems (NeurIPS)*.

S. Kim, D. Kim, M. Cho, and S. Kwak. 2020. Proxy Anchor Loss for Deep Metric Learning. In *IEEE/CVF Computer Vision and Pattern Recognition Conference (CVPR)*.

W. Kong, S. Khadanga, C. Li, S. Gupta, M. Zhang, W. Xu, and M. Bendersky. 2022. Multi-aspect dense retrieval. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*.

Y. Li, Z. Liu, C. Xiong, and Z. Liu. 2021. More Robust Dense Retrieval with Contrastive Dual Learning. In *ACM SIGIR International Conference on Theory of Information Retrieval (ICTIR)*.

H. Liu, Z. Dai, D.-R. So, and Q.-V. Le. 2021a. Pay attention to MLPs. *arXiv preprint arXiv:2105.08050*.

Z. Liu, H. Xu, Y. Wen, N. Jiang, H. Wu, and X. Yuan. 2021b. TEMP: Taxonomy Expansion with Dynamic Margin Loss through Taxonomy-Paths. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

A. Y. Malkov and D. A. Yashunin. 2020. Efficient and robust approximate nearest neighbor search using Hierarchical Navigable Small World graphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*.

T. Mesquita, B. Martins, and M. Almeida. 2022. Dense Template Retrieval for Customer Support. In *International Conference on Computational Linguistics (COLING)*.

A. Mittal, K. Dahiya, S. Malani, J. Ramaswamy, S. Kuruvilla, J. Ajmera, K. Chang, S. Agrawal, P. Kar, and M. Varma. 2022. Multimodal extreme classification. In *IEEE/CVF Computer Vision and Pattern Recognition Conference (CVPR)*.

A. Mittal, N. Sachdeva, S. Agrawal, S. Agarwal, P. Kar, and M. Varma. 2021. ECLARE: Extreme Classification with Label Graph Correlations. In *International Conference on World Wide Web (WWW)*.

S. Mohan, D. Saini, A. Mittal, S.R. Chowdhury, B. Paliwal, J. Jiao, M. Gupta, and M. Varma. 2024. Oak: Enriching document representations using auxiliary knowledge for extreme classification. In *International Conference on Machine Learning (ICML)*.

N. Muennighoff, H. Su, L. Wang, N. Yang, F. Wei, T. Yu, A. Singh, and D. Kiela. 2024. Generative representational instruction tuning. *arXiv preprint arXiv:2402.09906*.

K. Nguyen, H. H. Nguyen, and A. Tiulpin. 2022. Ada-Triplet: Adaptive Gradient Triplet Loss with Automatic Margin Learning for Forensic Medical Image Matching. In *International Conference on Medical Image Computing and Computer Assisted Intervention (MICCAI)*.

Y. Prabhu, A. Kag, S. Gopinath, K. Dahiya, S. Harsola, R. Agrawal, and M. Varma. 2018a. Extreme multi-label learning with label features for warm-start tagging, ranking and recommendation. In *ACM International Conference on Web Search and Data Mining (WSDM)*.

Y. Prabhu, A. Kag, S. Harsola, R. Agrawal, and M. Varma. 2018b. Parabel: Partitioned label trees for extreme classification with application to dynamic search advertising. In *International Conference on World Wide Web (WWW)*.

Y. Prabhu and M. Varma. 2014. FastXML: A Fast, Accurate and Stable Tree-classifier for eXtreme Multi-label Learning. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*.

N. Reimers and I. Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

L. Ren, C. Chen, L. Wang, and K. Hua. 2024. Towards Improved Proxy-based Deep Metric Learning via Data-Augmented Domain Adaptation. In *Association for the Advancement of Artificial Intelligence Conference on Artificial Intelligence (AAAI)*.

D. Saini, A.K. Jain, K. Dave, J. Jiao, A. Singh, R. Zhang, and M. Varma. 2021. GalaXC: Graph Neural Networks with Labelwise Attention for Extreme Classification. In *International Conference on World Wide Web (WWW)*.

F. Schroff, D. Kalenichenko, and J. Philbin. 2015. Facenet: A unified embedding for face recognition and clustering. In *IEEE/CVF Computer Vision and Pattern Recognition Conference (CVPR)*.

J. Snell, K. Swersky, and R.S. Zemel. 2017. Prototypical Networks for Few-shot Learning. In *Advances in Neural Information Processing Systems (NeurIPS)*.

J. So, Y. Lim, Y. Kim, C. Oh, and K. Song. 2023. Robust contrastive learning with dynamic mixed margin. *IEEE Access*.

X. Song, L. Huang, H. Xue, and S. Hu. 2022. Supervised Prototypical Contrastive Learning for Emotion Recognition in Conversation. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

X. Sun, K. Bi, J. Guo, S. Yang, Q. Zhang, Z. Liu, G. Zhang, and X. Cheng. 2024. A multi-granularity-aware aspect learning model for multi-aspect dense retrieval. In *ACM International Conference on Web Search and Data Mining (WSDM)*.

Y. Tagami. 2017. AnnexML: Approximate Nearest Neighbor Search for Extreme Multi-label Classification. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*.

A. Van den Oord, Y. Li, and O. Vinyals. 2018. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*.

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, Ł. Kaiser, and I. Polosukhin. 2017. Attention Is All You Need. In *Advances in Neural Information Processing Systems (NeurIPS)*.

L. Wang, N. Yang, X. Huang, B. Jiao, L. Yang, D. Jiang, R. Majumder, and F. Wei. 2022. Text embeddings by weakly-supervised contrastive pre-training. *arXiv preprint arXiv:2212.03533*.

L. Wang, N. Yang, X. Huang, L. Yang, R. Majumder, and F. Wei. 2024. Improving Text Embeddings with Large Language Models. In *Annual Meeting of the Association for Computational Linguistics (ACL)*.

L. Xiong, C. Xiong, Y. Li, K.-F. Tang, J. Liu, P. Bennett, J. Ahmed, and A. Overwijk. 2021. Approximate nearest neighbor negative contrastive learning for dense text retrieval. In *International Conference on Learning Representations (ICLR)*.

S. Yadav, D. Saini, A. Buvanesh, B. Paliwal, K. Dahiya, S. Asokan, Y. Prabhu, J. Jiao, and M. Varma. 2024. Extreme meta-classification for large-scale zero-shot retrieval. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*.

J. Yang, Z. Liu, S. Xiao, C. Li, D. Lian, S. Agrawal, A. Singh, G. Sun, and X. Xie. 2021. Graphformers: Gnn-nested transformers for representation learning on textual graph. In *Advances in Neural Information Processing Systems (NeurIPS)*.

E. H. I. Yen, X. Huang, W. Dai, P. Ravikumar, I. Dhillon, and E. Xing. 2017. PPDSparse: A Parallel Primal-Dual Sparse Method for Extreme Classification. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*.

R. You, S. Dai, Z. Zhang, H. Mamitsuka, and S. Zhu. 2019. AttentionXML: Extreme Multi-Label Text Classification with Multi-Label Attention Based Recurrent Neural Networks. In *Advances in Neural Information Processing Systems (NeurIPS)*.

J. Zeng, Y. Yin, Y. Jiang, S. Wu, and Y. Cao. 2022. Contrastive Learning with Prompt-derived Virtual Semantic Prototypes for Unsupervised Sentence Embedding. In *Conference on Empirical Methods in Natural Language Processing, Findings Track (EMNLP-F)*.

D. Zhang, S.-W. Li, W. Xiao, H. Zhu, R. Nallapati, A.O. Arnold, and B. Xiang. 2021a. Pairwise Supervised Contrastive Learning of Sentence Representations. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

J. Zhang, W. C. Chang, H. F. Yu, and I. Dhillon. 2021b. Fast multi-resolution transformer fine-tuning for extreme multi-label text classification. In *Advances in Neural Information Processing Systems (NeurIPS)*.

C. Zhou, J. Dong, X. Huang, Z. Liu, K. Zhou, and Z. Xu. 2024. QUEST: Efficient extreme multi-label text classification with large language models on commodity hardware. In *Findings of the Association for Computational Linguistics: EMNLP 2024*.

Appendices

A Triplet loss with clipped dynamic margin. Propositions and proofs.

Consider $\mathcal{L}_T$ in Eq. 8 be the simplified version of the original triplet loss (Schroff et al., 2015) in the angular domain,

$$\begin{aligned}\mathcal{L}_T &= \max\left(s_{an} - s_{ap} + m, 0\right) \\ &= \begin{cases}\Delta s_a^{n-p} + m, & \text{if } \Delta s_a^{p-n} \leq m \\ 0, & \text{otherwise}\end{cases},\end{aligned} \tag{8}$$

where s_{an} and s_{ap} denote the anchor-negative and the anchor-positive cosine similarities, respectively. For convenience, we denote $\Delta s_a^{n-p} = s_{an} - s_{ap}$ and $\Delta s_a^{p-n} = s_{ap} - s_{an}$ as the differences of the cosine similarities between the tuples anchor-positive and anchor-negative, respectively. Assuming normalized feature vectors, Eq. 8 can take the values $m \in [0, 2)$.

The traditional triplet loss formulation imposes a fixed margin during training such that all negatives, regardless the degree of similarity to the positive, are pushed apart from the anchor while positives are pulled close to the anchor given condition $\Delta s_a^{p-n} \leq m$. The partial derivatives with respect to s_{ap} and s_{an} look as follows:

$$\left(\frac{\partial \mathcal{L}_T}{\partial s_{ap}}, \frac{\partial \mathcal{L}_T}{\partial s_{an}}\right) = \begin{cases}(-1, 1), & \text{if } \Delta s_a^{p-n} \leq m \\ (\ 0, 0), & \text{otherwise}\end{cases}. \tag{9}$$

A.1 Proof of proposition 1

Proposition 1. *Consider the non-differentiable piece-wise linear function defining the adaptive margin to be* $m\left(s_{ap}, s_{an}\right) = |s_{ap} - s_{an}|$. *Adding* $m\left(s_{ap}, s_{an}\right)$ *into Eq. 8 expands the support of the function by relaxing the margin constraint of the original triplet formulation, inducing a modulation effect that allows semantically similar representations to be projected closer in the embedding space.*

Proof. Consider $m\left(s_{ap}, s_{an}\right)$ be the non-differentiable piece-wise linear margin function defined as,

$$m\left(s_{ap}, s_{an}\right) = \begin{cases}s_{ap} - s_{an}, & \text{if } s_{ap} > s_{an} \\ s_{an} - s_{ap}, & \text{if } s_{ap} \leq s_{an}\end{cases}, \tag{10}$$

and consider as well $(x)_\dagger$ be the operator that detaches x from the computational graph, thus avoiding the gradients to back-propagate. Then, applying the dynamic margin in Eq. 8, the triplet loss becomes:

$$\mathcal{L}_T^d = \begin{cases}s_{an} - s_{ap} + (s_{an} - s_{ap})_\dagger, & \text{if } \Delta s_a^{p-n} \leq 0 \\ 0, & \text{otherwise}\end{cases} \tag{11}$$

In Eq. 11 we show that adding the dynamic margin removes the fixed margin constraint of the original triplet loss formulation enabling the model to learn from any observation that satisfies the inequality $s_{ap} \leq s_{an}$ where negatives are closer to the anchor than positives. Besides, we observe that the new function modulates the margin depending on the hardness of the triplets, allowing semantically similar observations ($s_{ap} \approx s_{an}$ with $s_{ap} \leq s_{an}$) to be closer in the embedding space. Conversely, semantically different observations are pushed apart with higher strength.

□

A.2 Proof of Proposition 2

Proposition 2. *Clipping the values of the dynamic margin partitions the triplet loss landscape allowing positives and negatives in uncertain settings to be projected nearby in the embedding space.*

Proof. Consider $\text{clip}(x)$ be the clipping function in Eq. 12,

$$\text{clip}(x) = \begin{cases} x, & \text{if} \quad \gamma_{\text{min}} \leq x \leq \gamma_{\text{max}} \\ \gamma_{\text{min}}, & \text{if} \quad x < \gamma_{\text{min}} \\ \gamma_{\text{max}}, & \text{if} \quad x > \gamma_{\text{max}} \end{cases}, \tag{12}$$

then, the triplet loss with clipped dynamic margin becomes:

$$\begin{aligned} \mathcal{L}_T^{cd} &= \begin{cases} \max\left(\Delta s_a^{n-p} + \text{clip}(\Delta s_a^{p-n})_\dagger, 0\right), & \text{if } s_{ap} > s_{an} \\ \max\left(\Delta s_a^{n-p} + \text{clip}(\Delta s_a^{n-p})_\dagger, 0\right), & \text{if } s_{ap} \leq s_{an} \end{cases} \\ &= \begin{cases} 0, & \text{if } s_{ap} > s_{an}, \ \Delta s_a^{p-n} > \gamma_{\text{min}} \\ \Delta s_a^{p-n} + \gamma_{min}, & \text{if } s_{ap} > s_{an}, \ \Delta s_a^{p-n} < \gamma_{\text{min}} \\ \Delta s_a^{n-p} + \text{clip}\left(\Delta s_a^{n-p}\right)_\dagger, & \text{if } s_{ap} \leq s_{an} \end{cases}, \end{aligned} \tag{13}$$

From Eq. 13 one can easily observe that the norm of the partial derivatives when $s_{ap} \leq s_{an}$ might increase by a multiplicative factor of 2 if we do not detach the dynamic margin. We empirically observe that increasing the gradient norm for those cases is preventing the model to converge properly. We argue that doubling the norm of the gradients may generate instabilities during training that impact the inter-class separation and the uniformity of the embedding space. By detaching the dynamic margin from the computational graph, the partial derivatives become:

$$\left(\frac{\partial \mathcal{L}_T^d}{\partial s_{ap}}, \frac{\partial \mathcal{L}_T^d}{\partial s_{an}}\right) = \begin{cases} (\ \ 0, \ \ 0), & \text{if } s_{ap} > s_{an}, \left(\Delta_a^{p-n} > \gamma_{min}\right) \\ (\ \ 1, -1), & \text{if } s_{ap} > s_{an}, \left(\Delta_a^{p-n} < \gamma_{min}\right) \\ (-1, \ \ 1), & \text{if } s_{ap} \leq s_{an} \end{cases}. \tag{14}$$

The new formulation introduces a new learning region ($s_{ap} > s_{an}, \ \Delta_a^{p-n} < \gamma_{min}$) to the dynamic triplet loss where the gradient direction is inverted, *i.e.*, negatives are pulled closer to the anchor while positives are pushed apart until $s_{ap} \leq s_{an}$. We argue that this region, where positives and negatives are nearly equally distant to the anchor, accounts for those cases with a high degree of uncertainty such as ambiguous observations or missing labels, a well known problem in extreme multi-label settings. By reverting the learning process in this small margin, the network is capable of handling these scenarios.

In summary, we can observe the following cases depending on the hardness of the triplet construction:

- *Easy cases* $\rightarrow s_{ap} > s_{an}, \ \Delta_a^{p-n} > \gamma_{min}$. Positives are closer to the anchor than negatives and positives are reasonably separated from negatives. Similarly to $\mathcal{L}_T$, we do not back-propagate those observations.

- *Hard cases* $\rightarrow s_{ap} \leq s_{an}$. We keep the same gradients (norm and direction) as the original triplet loss for hard cases where negatives are closer to the anchor than positives while keeping the modulation effect described in Proposition 1.

- *Ambiguity and missing labels* $\rightarrow s_{ap} > s_{an}, \ \Delta_a^{p-n} < \gamma_{min}$. The proposed dynamic margin relaxes the original $\mathcal{L}_T$ by allowing positives and negatives to live in a region where both representations are closer to each other. Indeed, the direction of the partial derivatives are inverted, allowing an inversion of the natural order of positives and negatives with respect to the anchor that accounts for high uncertain observations.

□

Dataset	#q train	#l	#q test	#q/l	#l/q
LF-AmazonTitles-131K	294.8K	131K	134.8K	5.15	2.29
LF-Amazon-131K	294.8K	131K	134.8K	5.15	2.29
LF-AmazonTitles-1.3M	2.2M	1.3M	0.97M	38.24	22.20
LF-Wikipedia-500K	1.8M	0.5M	0.8M	24.75	4.77
LF-WikiTitles-500K	1.8M	0.5M	0.8M	17.15	4.74
LF-WikiSeeAlso-320K	693.1K	312.3K	177.5K	2.11	4.68

Table 7: Dataset statistics for benchmark datasets. Key: #q (number of queries), #l (number of labels), #q/l (number of queries per label), #l/q (number of labels per query).

B Datasets details

In Table 7 we present the details of the datasets used in Section 4. The datasets consider multiple real-world applications. In particular, LF-Wikipedia-500K, and LF-Wikipedia-500K involve predicting the relevant categories given the title or the full Wikipedia page, respectively. The LF-WikiSeeAlso-320K dataset addresses the prediction of similar Wikipedia articles. On the other hand, LF-AmazonTitles-131K and LF-AmazonTitles-1.3M involve recommending similar products using the product title. The LF-Amazon-131K dataset is similar to LF-AmazonTitles-131K, but adding long product descriptions to the product titles. Finally, the label is endowed with a short description for all of these datastes.

C Comparison against methods using classifiers

Many works in the XMC literature design efficient strategies to train a One-vs-All (OVA) classifier $\mathbf{W}$. On the one hand, some methods just train the classifier and do not focus on training backbone encoders that can be used to compute predictions, as it is the case for XR-Transformer (Zhang et al., 2021b), LightXML (Jiang et al., 2021), ELIAS (Gupta et al., 2022), CascadeXML (Kharbanda et al., 2022), PINA (Chien et al., 2023) and Renée (Jain et al., 2023). On the other hand, methods such as NGAME (Dahiya et al., 2023a), DEXA (Dahiya et al., 2023b) and OAK (Mohan et al., 2024) train the OVA classifiers after encoder training and, then, ensemble the scores predicted by each of the stages, which increases inference complexity (two ANN search structures are required). Note that OAK also introduces external information to enrich the queries, e.g. hyper-link information in Wikipedia datasets. The results presented in Table 8 demonstrate that PRIME with and without extreme classifiers is competitive and sometimes provides better performance than classifier-based methods. PRIME achieves top P@1 in 3 out of 6 datasets (4 if we exclude OAK, which exploits additional meta-data information).

Method	P@1	P@5	PSP@1	PSP@5	P@1	P@5	PSP@1	PSP@5	P@1	P@5	PSP@1	PSP@5
Datasets $\longrightarrow$	**LF-Amazon-131K**				**LF-AmazonTitles-1.3M**				**LF-AmazonTitles-131K**			
XR-Tranf. $(\mathbf{W})$	45.61	22.32	34.93	49.24	50.14	39.98	20.06	27.79	38.10	18.32	28.86	39.59
ELIAS $(\mathbf{W})$	-	-	-	-	-	-	-	-	40.13	19.54	31.05	42.88
CascadeXML $(\mathbf{W})$	-	-	-	-	47.82	38.31	17.17	24.76	35.96	18.15	-	-
NGAME $(\mathbf{W})$	46.53	22.02	38.53	50.45	54.69	42.80	28.23	34.48	44.95	21.20	38.25	48.42
NGAME $(\theta, \mathbf{W})$	46.65	22.03	38.67	50.12	56.75	44.09	29.18	35.36	46.01	21.47	38.81	49.43
DEXA $(\mathbf{W})$	47.12	22.35	38.86	50.59	55.76	42.95	30.01	35.29	45.78	21.29	38.57	48.56
DEXA $(\theta, \mathbf{W})$	47.16	22.42	38.70	50.97	56.63	43.90	29.12	34.86	**46.42**	21.59	39.11	49.65
PINA $(\mathbf{W})$	46.76	23.20	-	-	-	-	-	-	-	-	-	-
Renée $(\mathbf{W})$ ✦	48.05	23.26	40.11	**53.67**	56.04	45.32	28.54	36.14	46.05	**22.04**	39.08	**50.48**
PRIME (θ)	48.09	**23.34**	**40.48**	53.43	58.58	45.44	**32.14**	**39.07**	44.87	21.53	**39.59**	49.73
PRIME $(\mathbf{W})$	**48.20**	23.28	40.16	53.22	**59.62**	**46.75**	31.20	38.64	45.26	21.48	39.29	49.44
Datasets $\longrightarrow$	**LF-WikiTitles-500K**				**LF-Wikipedia-500K**				**LF-WikiSeeAlso-320K**			
LightXML$(\mathbf{W})$	-	-	-	-	81.57	47.64	31.99	46.53	34.50	16.83	17.85	24.16
XR-Transf.$(\mathbf{W})$	-	-	-	-	81.62	47.85	33.58	47.81	42.57	21.30	25.18	33.79
CascadeXML$(\mathbf{W})$	47.29	**19.00**	19.19	19.75	81.13	49.12	32.12	49.37	-	-	-	-
NGAME $(\mathbf{W})$	-	-	-	-	84.01	49.97	41.25	57.04	45.72	22.06	-	-
NGAME $(\theta, \mathbf{W})$	39.04	16.08	23.12	23.03	84.01	49.97	41.25	57.04	47.65	23.68	33.83	**41.03**
DEXA $(\mathbf{W})$	-	-	-	-	84.92	50.51	42.59	58.33	-	-	-	-
DEXA $(\theta, \mathbf{W})$	**47.41**	17.62	25.27	24.03	83.52	50.85	42.15	57.38	47.11	22.71	31.82	38.78
Renée $(\mathbf{W})$ ✦	-	-	-	-	84.95	**51.68**	39.89	56.70	47.86	**24.05**	32.02	40.90
PINA $(\mathbf{W})$	-	-	-	-	82.83	50.11	-	-	44.54	22.92	-	-
OAK $(\theta, \mathbf{W})$ ✧	44.82	17.67	**25.79**	**24.90**	85.23	50.79	**45.28**	**60.80**	**48.57**	23.28	**33.92**	40.44
PRIME (θ)	46.33	18.09	25.36	24.03	85.37	50.92	40.60	57.74	48.00	23.53	32.88	40.20
PRIME $(\mathbf{W})$	46.75	18.44	25.13	24.31	**85.75**	51.58	40.29	58.61	48.14	23.61	32.80	40.24

Table 8: Comparative evaluation against XMC methods using OVA classifiers ($\mathbf{W}$). Note that $(\theta, \mathbf{W})$ methods ensemble the encoder and classifier predictions. ✦ denotes brute-force algorithm and ✧ algorithm using extra meta-data information. Bold and underlined denote best and second best.

# Auxiliary Vectors	P@1	P@3	P@5
-	57.00	49.67	44.50
1,024	57.09	49.70	44.51
4,096	57.13	49.75	44.53
16,384	57.47	49.97	44.73
32768	57.63	50.09	44.83
65,536	57.87	50.31	45.03
131,072	**58.10**	**50.39**	**45.05**

Table 9: Performance in LF-AmazonTitles-1.3M when varying the number of free vectors used in PRIME-lite.

D Effect of the number of free vectors

We analyze the impact of the number of free vectors used in Table 9. More free vectors exhibit better performance, leading to around 1% boost in P@1. Although this behaviour is aligned with observations made in DEXA (Dahiya et al., 2023b) work, free vectors are less important for boosting performance for our method.

E Computational complexity calculations

An encoder-based method will need to compute the embeddings of the queries and labels at every mini-batch. Consider B and $\mathcal{S}$ be the batch size and the subset of labels considered in a mini-batch, respectively. Moreover, K is the complexity of computing the embedding of a single query or label item and L is the total number of labels. The computational complexity of different components is defined as follows:

1. $\mathcal{O}\left(KB\right)$ for embedding B queries.
2. $\mathcal{O}\left(K|\mathcal{S}|\right)$ for embedding the labels considered in the mini-batch.
3. $\mathcal{O}\left(Bd|\mathcal{S}|\right)$ for computing the loss. Recall that d refers to the dimensionality of the final embeddings computed using the encoder.

This results in $\mathcal{O}\left(BK + KL + BdL\right) = \mathcal{O}\left(KL + BdL\right)$ for a brute-force approach as $\mathcal{S} = L$ and $L \gg B$. On the other hand, it translates to $\mathcal{O}\left(BK + B^2d\right)$ as $\mathcal{S} \approx B$ for PRIME as it uses a small pool of labels.

F Memory complexity calculations

PRIME's memory requirements can be analyzed in terms of static and dynamic components. PRIME needs to store model parameters and gradients constantly on the GPU. The model parameters include the encoder parameters (θ), prototype network parameters (ϕ) and free vectors ($\mathcal{B}d$). Consider B and $\mathcal{S}$ be the batch size and the subset of labels considered in a mini-batch, respectively. Then dynamic memory requirements include:

1. $\mathcal{O}\left((|\mathcal{S}| + B)(t^2 + td)\right)$ for embedding queries queries and labels. Moreover, t is the max sequence length for the text in the mini-batch.
2. $\mathcal{O}\left(|\mathcal{S}|d\right)$ for auxiliary vectors and label centroids.
3. $\mathcal{O}\left(d|\mathcal{S}|\right)$ for the label prototype network.

G Additional related works

Early XMC methods focused on efficient learning of classifiers albeit with fixed sparse (Babbar and Schölkopf, 2017; Jain et al., 2016; Yen et al., 2017; Tagami, 2017) or dense features (Jain et al., 2019). These methods attempt to learn a tree (Prabhu and Varma, 2014; Jain et al., 2016; Prabhu et al., 2018a),

Dataset	Number of Free Vectors	Batch Size	γ_{min}	γ_{max}	epochs	Learning rate lr	Maximum seq. len t_{max}
LF-AmazonTitles-131K	65,536	3200	0.1	0.3	300	0.0003	32
LF-Amazon-131K	65,536	512	0.1	0.3	300	0.0003	128
LF-AmazonTitles-1.3M	131,072	3200	0.1	0.3	300	0.0003	32
LF-WikiTitles-500K	65,536	3200	0.1	0.3	300	0.0002	32
LF-Wikipedia-500K	65,536	512	0.1	0.3	50	0.0001	256
LF-WikiSeeAlso-320K	65,536	1024	0.1	0.3	200	0.0001	128

Table 10: PRIME's hyper-parameter values on benchmark datasets for reproducibility. Most hyper-parameters were set to their default values across all datasets. PRIME samples a single positive per data point. Whereas, PRIME++ makes use of multiple positives and double the batch size of PRIME when possible.

embedding (Tagami, 2017; Bhatia et al., 2015) or a brute-force classifier (Babbar and Schölkopf, 2017, 2019). Furthermore, Slice (Jain et al., 2019), Parabel (Prabhu et al., 2018b), and PPD-Sparse (Yen et al., 2017) discuss negative sampling in the context of fixed pre-trained features.

H Hyper-parameters

We set PRIME configurations to fit a single 80 GB A100 and it trains in around 60 hours on LF-AmazonTitles-1.3M dataset. PRIME-lite, instead, can be trained in around 42 hours in the same dataset.

We do not conduct a fine-grain tuning of PRIME on every specific dataset aiming at demonstrating the generality of PRIME's hyper-parameters (see Table 10 for main hyper-parameters). We set the batch size to maximize memory utilization in a single GPU, thus using a larger batch size of 3200 on the titles datasets and reducing it for full-text datasets. Learning rate (lr) and number of epochs are set as proposed by (Dahiya et al., 2023a). The remaining hyper-parameters $\alpha = 0.95$ (for smooth centroid updates), regularization weight $\lambda = 0.1$, regularization margin $m' = 0.1$, and γ_{min} and γ_{max} for the loss function are selected on the basis of performance (P@1) on a validation set of the smallest dataset and, then, we keep them fixed across all datasets. It is worth noting that we explored other strategies for centroid estimation by: 1) using multiple queries for each label, which led to minor gains (0.1-0.2%) at the cost of increased computation due to having to encode more query textual embeddings batch-wise; 2) using the exact label centroids in the inference phase, which led to comparable performance.

Additionally, we use AdamW optimizer and a weight decay of 0.01 that is removed from Bias, LayerNorm and the bank of free vectors $\mathcal{B}$. Moreover, for the Prototype Label Network we use a Transformer Encoder layer (1 attention head, 1024 internal dimensionality, and 0.1 dropout). We tried increased dimensionality and/or number of attention heads observing marginal gains, thus decided to keep it simple. As for negative sampling, we use the NGAME sampler to select a subset of negatives and adopt the default values for its cluster size. Finally, note that we always report model performance for its weights in the last training epoch, *i.e.*, we do not conduct any custom checkpoint selection based on validation metrics.

I Inference

PRIME computes and saves the final label prototypes, *i.e.*, $\{\mathbf{z}_l\}_{r=1}^{L}$, once the training is finished. Given a new test query q, we follow the steps below:

1. Encode the query q as $\mathbf{h}_q$.
2. Compute the scores $\{\mathbf{h}_q^\top \mathbf{z}_l\}_{r=1}^{L}$. Note that $\mathbf{h}_q$ and $\{\mathbf{z}_l\}_{r=1}^{L}$ are L2-normalized.
3. Sort the label indices on the basis of scores and return top-k most relevant labels.

In practice, we use IndexFlatIP provided by the Faiss (Johnson et al., 2019) library. PRIME can make predictions within a few milli-seconds on the LF-AmazonTitles-1.3M dataset on a single GPU. It should be noted that the inference complexity of PRIME is similar to other XMC algorithms that make

Positive Sampling	P@1	P@3	P@5	PSP@1	PSP@3	PSP@5	R@100
Uniform	57.48	49.85	44.54	29.65	34.30	36.86	62.50
Proposed	58.10	50.39	45.05	31.68	36.12	38.55	63.45

Table 11: Impact of positive sampling on the PRIME-lite algorithm on the LF-AmazonTitles-1.3M dataset. The proposed sampling results in prominent gains in the propensity score metrics (PSP@k).

predictions on the basis of text embeddings (Gupta et al., 2024) or classifiers (Jain et al., 2023). Finally, an Approximate Nearest Neighbor (ANNS) index (Malkov and Yashunin, 2020) can be readily integrated for datasets where label prototypes do not fit on a single GPU.

J Positive sampling

PRIME samples its positives from a distribution where the probability of sampling a positive is proportional to its inverse propensity score (Jain et al., 2016). In particular, the probabilities for the i-th query q_i are defined as follows:

$$p_{il} = \frac{\gamma_l}{\sum_{j \in \mathcal{P}_i} \gamma_j}.$$

Here γ_l is the inverse propensity score for the label l (please refer to (Jain et al., 2016) for more details) and $\mathcal{P}_i$ is the set of positives for the i-th query data point. Such a positive sampling strategy leads to gains across all evaluation metrics (please see Table 11 for results on LF-AmazonTitles-1.3M dataset). It should be noted that the gains are more significant in the propensity score metrics (PSP@k), *i.e.*, more benefits are found on the tail labels.