Unified Dual Encoder and Classifier Training for Extreme Multi-label Classification

Siddhant Kharbanda [*1,2] **Devaansh Gupta** [*3] **Gururaj K** [2] **Pankaj Malhotra** [2] **Cho-jui Hsieh** [1] **Rohit Babbar** [4,5]

Abstract

Extreme Multi-label Classification (XMC) involves predicting a subset of relevant labels from an extremely large label space, given an input query and labels with textual features. Models developed for this problem have conventionally used modular approach with (i) a Dual Encoder (DE) to embed the queries and label texts, (ii) a One-vs-All classifier to rerank the shortlisted labels mined through meta-classifier training. While such methods have shown empirical success, we observe two key uncharted aspects, (i) DE training typically uses only a single positive relation even for datasets which offer more, (ii) existing approaches fixate on using only OvA reduction of the multi-label problem. This work aims to explore these aspects by proposing UniDEC, a novel end-to-end trainable framework which trains the dual encoder and classifier in together in a unified fashion using a multi-class loss. For the choice of multi-class loss, the work proposes a novel *pick-some-label (PSL)* reduction of the multi-label problem with leverages multiple (in come cases, all) positives. The proposed framework achieves state-of-the-art results on a single GPU, while achieving on par results with respect to multi-GPU SOTA methods on various XML benchmark datasets, all while using $4 - 16\times$ lesser compute and being practically scalable even beyond million label scale datasets.

1. Introduction

Extreme Multi-label Classification (XMC) is described as the task of identifying i.e. retrieving a subset, comprising one or more labels, that are most relevant to the given data point from an extremely large label space potentially consisting of millions of possible choices. Typically, *long-text XMC* approaches are leveraged for tasks of document tagging and product recommendation and *short-text XMC* approaches are targeted at tasks such as query-ad keyword matching and related query recommendation. Notably, in the real world manifestations of these use cases, the distribution of instances among labels exhibits fit to Zipf's law (Adamic & Huberman, 2002). This implies, the vast label space ($L \approx 10^6$) is skewed and is characterized by the existence of *head*, *torso* and *tail* labels (Schultheis et al., 2022). For example, in query-ad keyword matching for search engine like Bing, Google etc head keywords are often exact match or phrase extensions of popularly searched queries while tail keywords are often targeted at specific niche queries. Typically, we can characterize head, torso and tail keywords as having > 100, 10 - 100, and 1 - 10 annotations, respectively.

Contemporary algorithms for building state-of-the-art (SOTA) XMC models typically consist of an encoder and a high-capacity One-vs-All (OvA) linear classifier learnt for each label. These modules can either be trained jointly or in a modular fashion, each having their own pros and cons. In general, it is often a trade-off between scalability and performance. With the availability of label metadata in the form of label-text, recent SOTA methods (Dahiya et al., 2023a;b) focus on developing a two-stage modular solution. First stage involves training a Dual Encoder (DE) via contrastive learning. This is followed by a fine-tuning stage for OvA classifiers. It has been shown in these works that even though DE alone performs reasonably well on XMC tasks, leveraging classifiers significantly enhances the empirical performance of these models.

1.1. Dual Encoder Training

DE models are a popular choice for dense retrieval (DR) and Open-Domain Question Answering (ODQA) tasks which are predominantly *few* and *zero-shot* scenarios. In contrast, XMC task covers a broader range of scenarios (see Appendix B). Consequently, the XMC task, modelled as a retrieval problem, is tantamount to training a DE simultaneously on *many*, *few* and *one-shot* scenarios.

As observed in ODQA literature, the performance of DE is positively correlated with the batch size and the number of

[*]Equal contribution [1]University of California, Los Angeles [2]Microsoft, India [3]Nanyang Technological University, Singapore [4]University of Bath [5]Aalto University, Finland. Correspondence to: Siddhant Kharbanda <skharbanda17@g.ucla.edu>.

Preprint. Under Review.

hard negatives (Karpukhin et al., 2020; Xiong et al., 2020). However, increasing either can make the training computationally expensive. Hence, to make training tractable and mine harder negatives, recent XMC approaches perform hierarchical k-means clustering over queries leading to the creation of negative mining-aware training batches (Dahiya et al., 2023a;b). As a result, explicit sampling of a single positive label for each query implicitly leads to the accumulation of multiple positives for most queries (See Figure 2). This makes the direct application of InfoNCE loss infeasible, hence these works find it suitable to replace InfoNCE loss with a triplet loss, as it can be potentially applied over multiple positives and hard negatives (equation 1 in (Dahiya et al., 2023a)). Interestingly, both contrastive DR methods and XMC approaches either (i) calculate loss over only a single positive for faster convergence, choosing to ignore other positive signals in the batch (e.g. (Dahiya et al., 2023a)) or, (ii) decouple the softmax operation over positive labels (Gupta et al., 2023). In this work, we investigate the use of a multi-class loss which can leverage multiple positives in contrastive learning framework for DE training of XMC. To the best of our knowledge, ours is the first work that aims to empirically study a multi-class reduction paradigm for classifier training over millions of labels.

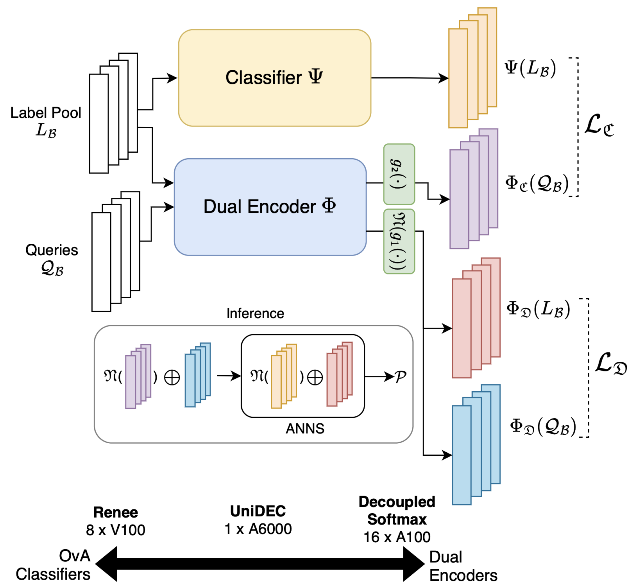


Figure 1. The architecture for the UNIDEC framework, denoting the the classifiers and DE trained in parallel, along with the loss functions used. The inference pipeline is shown in the rectangular box. UNIDEC strikes a sweet spot between two ends of extreme classification requiring only 1xA6000 GPU.

1.2. Classifier Training

In XMC, classifiers are conventionally trained with a binary OvA loss (e.g. BCE loss, hinge/squared-hinge loss) either over entire label space (in an end-to-end fashion (Jain et al., 2023)), or over hard-negative label shortlist (via a modular approach (Dahiya et al., 2023a) because of scalability concerns (Chang et al., 2020)). Another common strategy for partitioning the label space is to build a label tree and exploit its structure to create dynamic label shortlist (Jiang et al., 2021; Jasinska et al., 2016; Wydmuch et al., 2018; You et al., 2019). We discuss the shortlisting strategies and their shortcomings in § 2.2.

What remains common across all classifier-training algorithms targeted at solving XMC is the advocacy of OvA reduction for the multi-label problem and its sufficiency to achieve SOTA prediction performance (Dahiya et al., 2021a). The alternative approach of using a multi-class loss via pick-all-label (PAL) reduction has neither been well-studied nor successfully leveraged to train the classifiers in XMC for two reasons (Menon et al., 2019) : (i) such losses require considering all labels per instance, however, this is prohibitively expensive given the extremely large label space (ii) the label-label correlations in XMC have been thought to be quite sparse, thus eliminating the need for multi-class losses (Kharbanda et al., 2023b). Contrarily, we see that, (i) training on a shortlist comprising of hard negatives serves as a good approximation for OVA loss, however, the same has not been studied for PAL multi-label reduction and, (ii) recent approaches have successfully been able to capture label-label correlations in XMC (Mittal et al., 2021a;b), especially with the availability of label-text. These observations motivate us towards investigating an efficient implementation of a *PAL* loss for classifier training.

1.3. Novelty and Contributions

At face value, PAL reduction for DE training should be made tractable by optimizing it over the label shortlist, however in practice, it does not scale to larger datasets due to the higher number of positives per label. For instance, for LF-AmazonTitles-1.3M a batch consisting of 1,000 queries will need an inordinately large label pool of size $\sim$ 22.2K (considering in-batch negatives) to effectively train a DE with the PAL loss. To alleviate the issue, we propose ***"pick-some-labels"*** *(PSL)* approximation to the PAL reduction for the multi-label classification problem which enables scaling to large datasets ($\sim 10^6$ labels). Here, instead of trying to sample all labels for instances in a batch, we propose to randomly sample at max β positive labels per instance.

To this end, we propose a novel end to end training framework - UNIDEC - which leverages the proposed PSL loss to **Uni**fy the **D**ual **E**ncoder and **C**lassifier training for XMC. As shown in Figure 1, in a single pass, UNIDEC computes PSL loss over two heads:- (i) between query's DE head embeddings and sampled label-text embeddings from the encoder, (ii) between query's CLF head embeddings and sampled classifier weights. Training both a DE and classifier head together in an end-to-end manner offers multiple benefits. (i) It helps us do away with a meta-classifier (Jiang et al., 2021; Kharbanda et al., 2022; 2023a) and modular training (Mittal et al., 2021a;b; Dahiya et al., 2021a; 2023a), (ii) It dynamically provides progressively harder negatives with lower GPU VRAM consumption. (iii) It equips the model with both zero-shot (through DE) and many-shot (through classifiers) capabilities. (iv) Additionally, with an Approximate Nearest Neighbour Search (ANNS), it can leverage explicitly mined hard negative labels added to the in-batch negatives. Overall, UNIDEC significantly reduces the training computational cost while also providing the model with hard negatives, updated every few epochs, for efficient training on **a single NVIDIA A6000 GPU.** As shown in Figure 1, UNIDEC strikes a sweet spot between the two compute-heavy ends of XMC (i) *Renee* (Jain et al., 2023), that provides the empirical upper bound for OVA classifier performance and, (ii) *Decoupled Softmax* (Gupta et al., 2023), that proves the sufficiency of DE for XMC.

2. Related Work & Preliminaries

For training, we have available a multi-label dataset $\mathcal{D} = \{\{\mathbf{x}_i, \mathcal{P}_i\}_{i=1}^N, \{\mathbf{z}_l\}_{l=1}^L\}$ comprising of N data points and L labels. Each $\mathbf{x}_i$ is associated with a small ground truth label set $\mathcal{P}_i \subset [L]$ out of $L \sim 10^6$ possible labels. Further, $\mathbf{x}_i, \mathbf{z}_l \in \mathcal{X}$ denote the textual descriptions of the data point i and the label l respectively, which, in this setting, derive from the same vocabulary universe $\mathcal{V}$ (Dahiya et al., 2021a). The goal is to learn a parameterized function f which maps each instance $\mathbf{x}_i$ to the vector of its true labels $\mathbf{y}_i \in [0,1]^L$ where $\mathbf{y}_{i,l} = 1 \Leftrightarrow l \in \mathcal{P}_i$.

2.1. Dual Encoder Training

A common approach for DR models, for dealing with above problems is via a two-tower a.k.a DE models which consists of the query encoder Φ_q, and a label encoder Φ_l. In the context of XMC, existing approaches, such as NGAME and DEXA (Dahiya et al., 2023a;b), remodel the two-towers to encode either long-text data (e.g product description) or short-text data (e.g product title) using Φ_q and the label text (e.g related product title) using Φ_l, which project $\mathbf{x}_i$ and label $\mathbf{z}_l$, respectively, into a shared d-dimensional unit hypersphere $\mathcal{S}^{d-1}$. For each instance $\mathbf{x}_i$, its similarity with label $\mathbf{z}_l$ is then computed via an inner product i.e., $s_{i,l} = \langle \Phi_q(\mathbf{x}_i), \Phi_l(\mathbf{z}_l)\rangle$ to produce a ranked list of top-k relevant labels. Following the common practice in DE (Dahiya et al., 2023a;b; Gupta et al., 2023), the parameters for Φ_q and Φ_l are shared, and thus we will simply represent it as Φ.

Training two-tower algorithms for XMC at scale is made possible by creating negative-mining aware training batches $\mathfrak{B}$, where the instance encoder embeddings, $\{\Phi(\mathbf{x}_i)\}_{i=1}^N$, are recursively split (say, via a hierarchical clustering strategy) into clusters. This is done until the desired cluster-size $N/|\mathfrak{B}|$ is reached, for a pre-defined number of clusters $|\mathfrak{B}|$. These clusters serve as negative-mining aware training batches $\mathfrak{B} = \bigcup_{\mathcal{B}=1}^{|\mathfrak{B}|}\{Q_\mathcal{B}, L_\mathcal{B}\}$. The batch is characterised by a set of instance indices $Q_\mathcal{B} = \{i \mid i \in [N]\},\ s.t.\ |Q_\mathcal{B}| = N/|\mathfrak{B}|$, and the corresponding collated set of (typically one per instance) sampled positive labels $p \in \mathcal{P}_i$, defined as $L_\mathcal{B} = \{p \mid p \in \mathcal{P}_i \text{ and } i \in Q_\mathcal{B}\}$. As per the in-batch negative sampling strategy common across existing works (Dahiya et al., 2023a;b), the negative label pool then is made up of the positive labels sampled for other instances in the batch i.e. $\mathcal{N}_i = \{L_\mathcal{B} - \mathcal{P}_i\}$. Dahiya et al. (2023a) posit that the negative-mining aware batching ensures the “hardness” of in-batch negatives. Triplet loss is then applied with $\mathbf{x}_i$ as the anchor and optimized with z_p and z_n, chosen as the positive and negative labels.

$$\mathcal{L}_{\text{Triplet}} = \sum_{i=1}^{N} \sum_{\substack{p\in\mathcal{P}_i \\ n\in\mathcal{N}_i}} [\, \langle \Phi(\mathbf{x}_i), \Phi(\mathbf{z}_n)\rangle - \langle \Phi(\mathbf{x}_i), \Phi(\mathbf{z}_p)\rangle + \gamma \,]_+$$

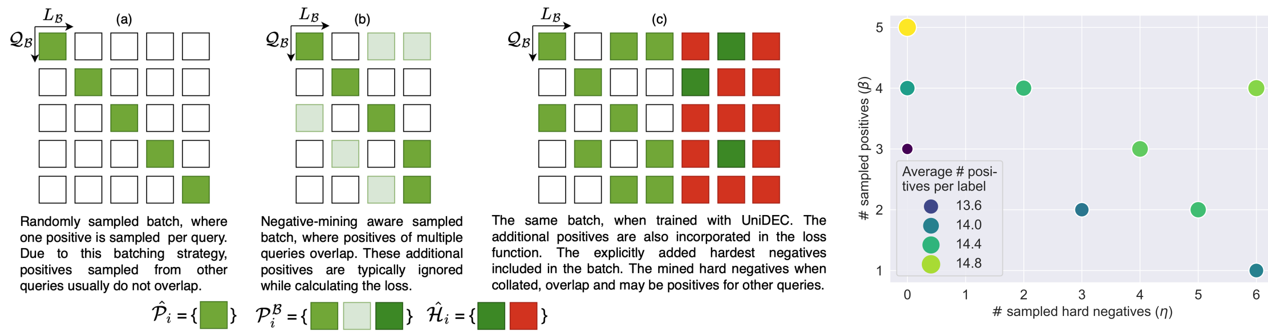
Note that, (Dahiya et al., 2023a) find that convergence is faster if only single positive is used, instead of all sampled in the batch (see Figure 2). A concurrent work (Gupta et al., 2023) studies the efficacy of DE for XMC tasks by proposing a decoupled variant of the softmax/InfoNCE loss function to handle multiple labels, which they call *Decoupled Softmax*. They find that the performance of such a DE algorithm for XMC can be pushed to the limit, provided that the loss is computed over the entire label space, which leads to very large batch sizes. As a result, training with their approach is computationally expensive; for instance, Decoupled Softmax requires 16 × A100s to run on LF-AmazonTitles-1.3M, the largest XMC public benchmark dataset with label features.

2.2. Classifiers for Extreme Classification

The traditional XMC set-up considers labels as featureless integer identifiers which replace the encoder representation of labels $\Phi(\mathbf{z}_l)$ with learnable classifier embeddings $\Psi_{l=1}^L \in \mathbb{R}^{L\times d}$ (You et al., 2019; Chang et al., 2020; Zhang et al., 2021). The relevance of a label l to an instance is scored using an inner product, $s_{i,l} = \langle \Phi(\mathbf{x}_i), \Psi_l\rangle$ to select the k highest-scoring labels. However, since modelling all possible label combinations is prohibitively difficult at the extreme scale, *OvA* reduction, which is optimal for precision metrics (Menon et al., 2019), is typically employed to train XMC classifiers (Babbar & Schölkopf, 2017; 2019).

Under this paradigm, each label is independently treated with a binary loss function ℓ_{BC} applied to each entry in the score vector. The OvA reduction can be expressed as,

$$\mathcal{L}_{\text{OVA}} = \sum_{l=1}^{L} \{y_l \cdot \ell_{BC}(1, s_{i,l}) + (1 - y_l) \cdot \ell_{BC}(0, s_{i,l})\}$$



(a) (b)

Figure 2. (a) Visualizing the batching strategy for UNIDEC. Such a framework naturally leads to higher number of positives per query, enabling us to scale without increasing the batch size significantly. (b) Scatter plot showing the average number of positive labels per query, when we sample β positives and η hard negatives in the batch. Note that, even with $\beta = 3$ and $\eta = 0$, avg$(|P|) = 13.6$.

Further reduction of computational complexity, from linear (under the OVA reduction) to logarithmic, can be achieved if the loss computation is performed only on a subset of shortlisted labels with cardinality $\mathcal{O}(\log(L))$. This shortlist can be created in one of the two ways : (i) by training a meta classifier on coarser levels of a probabilistic label tree, created by hierarchically clustering the label space. The leaf nodes of the top-k nodes constitute the shortlist (Jiang et al., 2021; Kharbanda et al., 2022; 2023a) (ii) by retrieving the top-k labels for a query from an ANNS built on the label representations from a contrastively trained DE (Dahiya et al., 2021a). Both these methods have different trade-offs. The meta-classifier based approach has a higher memory footprint due to the presence of additional meta classifier ($\sim \mathbb{R}^{L \times d}$ in size) along with the extreme classifier, but it gives enhanced performance since this provides progressively harder negatives in a dynamic shortlist, varying every epoch (Jiang et al., 2021; Kharbanda et al., 2022; 2023a). The shortlisting based on ANNS requires training the model in multiple stages, which has low memory usage, but needs longer training schedules and uses a static shortlist for training extreme classifiers (Mittal et al., 2021a;b; Dahiya et al., 2021b; 2023a).

3. Method : UNIDEC

In this work, we propose a novel multi-task learning framework which, in an end-to-end manner, trains both the DE head and the classifier head in parallel. The framework eliminates the need of a meta classifier for a dynamic in-batch shortlist. Further, it provides the encoder with the capability to explicitly mine hard-negatives by sampling them from a set of hard negatives, obtained by querying an ANNS, created over $\{\Phi(\mathbf{z}_l)\}_{l=1}^{L}$, which is refreshed every ε epochs.

The DE head is denoted by $\Phi_{\mathfrak{D}}(\cdot) = \mathfrak{N}(g_1(\Phi(\cdot)))$ and the classifier head by $\Phi_{\mathfrak{C}}(\cdot) = g_2(\Phi(\cdot))$, where $\mathfrak{N}$ represents the L2 normalization operator and $g_1(\cdot)$ and $g_2(\cdot)$ represent separate nonlinear projections. Unlike DE, and as is standard practice for OvA classifiers, we train them without an additional normalization operator (Dahiya et al., 2021a;b).

Training is carried out with the proposed *PSL* reduction, which is an approximation of the PAL-N reduction (Menon et al., 2019), formulated as :

$$\mathcal{L}_{\text{PAL-N}} = \frac{1}{\sum_{j=1}^{L} y_j} \sum_{l=1}^{L} y_l \cdot \ell_{MC}(1, \langle \Phi(\mathbf{x}), \Psi_l \rangle)$$

Choosing ℓ_{MC} as cross entropy loss with temperature τ,

$$\mathcal{L}_{\text{PAL-N}} = \frac{-1}{\sum_{j=1}^{L} y_j} \sum_{l=1}^{L} y_l \cdot \log \frac{\exp(\langle \Phi(\mathbf{x}), \Psi_l \rangle / \tau)}{\sum_{k=1}^{L} \exp(\langle \Phi(\mathbf{x}), \Psi_k \rangle / \tau)}$$

The above equation provides a generic framework of defining loss function for training classifier as well as DE. Replacing $\Phi(\mathbf{x})$ with $\Phi_{\mathfrak{D}}(\mathbf{x})$ and Ψ_l with $\Phi_{\mathfrak{D}}(z_l)$ gives rise to DE training (Dahiya et al., 2021a). Similarly, substituting $\Phi(\mathbf{x})$ with $\Phi_{\mathfrak{C}}(\mathbf{x})$ and retaining Ψ_l leads to classifier training. Combining both these losses in an end-to-end framework achieves the unification of dual encoder and classifier training. More details have been provided in § 3.2.

3.1. Dual Encoder Training with *Pick-some-Labels*

As with other reductions, $\mathcal{L}_{\text{PAL-N}}$ is computationally intractable to evaluate over the entire label space. Hence, we propose a more efficient approximation of this reduction, which we call *pick-some-labels (PSL)*. This approximation enables employing PAL-N over a minibatch $Q_{\mathcal{B}}$ by sampling a subset of positive labels $\hat{\mathcal{P}}_i \subseteq \mathcal{P}_i$ *s.t.* $|\hat{\mathcal{P}}_i| \leq \beta$. Typical value for β can be found in Appendix C. The collated label pool, considering in-batch negative mining, can be defined as $L_{\mathcal{B}} = \{\bigcup_{i \in Q_{\mathcal{B}}} \hat{\mathcal{P}}_i\}$ and the *PSL* loss is formulated as:

$$\mathcal{L}_{\mathfrak{D},q2l} = \sum_{i \in Q_{\mathcal{B}}} \frac{-1}{|\mathcal{P}_i^{\mathcal{B}}|} \sum_{p \in \mathcal{P}_i^{\mathcal{B}}} \log \frac{\exp(\langle \Phi_{\mathfrak{D}}(\mathbf{x}_i), \Phi_{\mathfrak{D}}(\mathbf{z}_p) \rangle / \tau)}{\sum_{l \in L_{\mathcal{B}}} \exp(\langle \Phi_{\mathfrak{D}}(\mathbf{x}_i), \Phi_{\mathfrak{D}}(\mathbf{z}_l) \rangle / \tau)}$$

Here, $\mathcal{P}_i^{\mathcal{B}} = \{\mathcal{P}_i \cap L_{\mathcal{B}}\}$ denotes all the in-batch positives for an instance $\mathbf{x}_i$, i.e., the green, pale green and dark green squares in Figure 2. Note that, $\mathcal{P}_i^{\mathcal{B}}$ consists not only of the sampled positives $\hat{\mathcal{P}}_i$ but also those non-sampled positives that exist in the batch as sampled positives of other instances i.e. $\mathcal{P}_i^{\mathcal{B}} = \hat{\mathcal{P}}_i \cup \{\bigcup_{j \in \{Q_{\mathcal{B}} - \{i\}\}} \hat{\mathcal{P}}_j \cap \mathcal{P}_i\}$. We find the cardinality of the second term to be non-zero for most instances having $|\mathcal{P}_i| > \beta$ due to a high overlap of sampled positive labels in negative-mining aware batches, leading to a more optimal batch size. Thus, although we sample $|\hat{\mathcal{P}}_i| \leq \beta \; \forall \; i \in Q_{\mathcal{B}}$, $\exists \; i \in Q_{\mathcal{B}} \; s.t. \; |\mathcal{P}_i^{\mathcal{B}}| \geq \beta$. As per our observations, $\mathcal{P}_i^{\mathcal{B}} = \mathcal{P}_i$ for most tail and torso queries. For e.g., even if $\beta = 1$ for LF-AmazonTitles-1.3M, for $|Q_{\mathcal{B}}| = 10^3$, $\text{Avg}(|\hat{\mathcal{P}}_i|) = [12, 14]$ (see Appendix C, Table 6). Thus, making *PSL* reduction same as PAL for torso and tail labels and only taking form of *PSL* for head queries.

Dynamic ANNS Hard-Negative Mining While the above strategy leads to collation of hard negatives in a batch, it might not mine hardest-to-classify negatives (i.e. the limiting case when cluster size is same as number of instances) (Dahiya et al., 2023a). We explicitly add them by querying an ANNS created over $\{\Phi_{\mathfrak{D}}(\mathbf{z}_l)\}_{l=1}^{L}$ for all $\{\Phi_{\mathfrak{D}}(\mathbf{x}_i)\}_{i=1}^{N}$. More specifically, for each instance, we create a list of hard negatives $\mathcal{H}_i = \text{top}_{\text{k}}(\text{ANNS}(\Phi_{\mathfrak{D}}(\mathbf{x}_i)|_{i=1}^{N}, \; \Phi_{\mathfrak{D}}(\mathbf{z}_l)|_{l=1}^{L})) \; s.t. \; \mathcal{H}_i \cap \mathcal{P}_i = \phi$ (denoted by red in Figure 2). Every iteration, we uniformly sample a η-sized hard-negative label subset $\hat{\mathcal{H}}_i \subset \mathcal{H}_i$ alongside $\hat{\mathcal{P}}_i \; \forall \; \mathbf{x}_i \in Q_{\mathcal{B}}$. More formally, the new batch label pool can be denoted as $L_{\mathcal{B}} = \{\bigcup_{i \in Q_{\mathcal{B}}} \hat{\mathcal{P}}_i \cup \hat{\mathcal{H}}_i\}$.

Interestingly, due to the multi-positive nature of XMC, sampled hard-negatives for $\mathbf{x}_i$ might turn out to be an unsampled positive label for $\mathbf{x}_j$. More formally, $\exists \; j \in Q_{\mathcal{B}} \; s.t. \; \{\hat{\mathcal{H}}_i \cap \mathcal{P}_j \neq \phi, \; \hat{\mathcal{H}}_i \cap \mathcal{P}_j^{\mathcal{B}} = \phi\}$. This requires altering the definition of $\mathcal{P}_i^{\mathcal{B}}$ to accommodate these *extra* positives (represented by the dark green square in Figure 2) as $\mathcal{P}_i^{\mathcal{B}} = \{\hat{\mathcal{P}}_i \cup \{\bigcup_{j \in \{Q_{\mathcal{B}} - \{i\}\}} \{\hat{\mathcal{P}}_j \cup \hat{\mathcal{H}}_j\} \cap \mathcal{P}_i\}\}$. Query clustering for batching and dynamic ANNS hard-negative mining strategies complement each other, since the presence of similar queries leads to a higher overlap in their positives *and hard negatives*, enabling us to scale the batch size by reducing the effective size of the label pool. Further, to provide $\Phi_{\mathfrak{D}}$ and $\Phi_{\mathfrak{C}}$ with progressively harder negatives, the ANNS is refreshed every τ epochs. To uniformly sample hard negatives from $\mathcal{H}$, we find it beneficial to have $|\mathcal{H}| = \eta \times \tau$.

Note that $L_{\mathfrak{D},q2l}$ denotes the multi-class loss between $\mathbf{x}_i$ and $\mathbf{z}_l \; \forall \; l \in L_{\mathcal{B}}$. As the data points and labels in XMC tasks belong to the same vocabulary universe (such as product recommendation), we find it beneficial to optimize $L_{\mathfrak{D},l2q}$ alongside $L_{\mathfrak{D},q2l}$, making $L_{\mathfrak{D}}$ a symmetric loss. Since (Radford et al., 2021), a plethora of works have leveraged symmetric optimizations in the vision-language retrieval pre-training domain. For XMC, the interchangability of $Q_{\mathcal{B}}$ and $L_{\mathcal{B}}$ in the symmetric objective can be viewed equivalent to (i) feeding more data relations in a batch, and (ii) bridging missing relations in the dataset (Kharbanda et al., 2023b). Further, we formulate XMC as a symmetric problem from $L_{\mathcal{B}}$ to $Q_{\mathcal{B}}$, enabling us to calculate the multi-class loss between $\mathbf{z}_l$ and $\mathbf{x}_i \; \forall \; i \in Q_{\mathcal{B}}$ given by:

$$\mathcal{L}_{\mathfrak{D},l2q} = \sum_{l \in L_{\mathcal{B}}} \frac{-1}{|\mathcal{P}_l^{\mathcal{B}}|} \sum_{p \in \mathcal{P}_l^{\mathcal{B}}} \log \frac{\exp(\Phi_{\mathfrak{D}}(\mathbf{z}_l) \cdot \Phi_{\mathfrak{D}}(\mathbf{x}_p)/\tau)}{\sum_{q \in Q_{\mathcal{B}}} \exp(\Phi_{\mathfrak{D}}(\mathbf{z}_l) \cdot \Phi_{\mathfrak{D}}(\mathbf{x}_q))/\tau)}$$

Note that, here, $\mathcal{P}_l^{\mathcal{B}}$ represents the in-batch instances positively annotated by l. More formally, $\mathcal{P}_l^{\mathcal{B}} = \{i \,|\, i \in Q_{\mathcal{B}}, \; l \in L_{\mathcal{B}}, \; \mathbf{y}_{i,l} = 1\}$. The total DE contrastive loss can thus be written as as:

$$\mathcal{L}_{\mathfrak{D}} = \lambda_{\mathfrak{D}} \cdot \mathcal{L}_{\mathfrak{D},q2l} + (1 - \lambda_{\mathfrak{D}}) \cdot \mathcal{L}_{\mathfrak{D},l2q}$$

For simplicity and to remove dependence on hyperparameters, we use $\lambda_{\mathfrak{D}} = 0.5$, which works well in practice.

Algorithm 1 Training step in UNIDEC

```
Input: instance x, label features z, positive labels 𝒫, encoder Φ, classifier lookup-table Ψ, non-linear transformations g1(·) and g2(·)
Φ_𝔇(·), Φ_ℭ(·) := 𝔑(g_1(Φ(·))), g_2(Φ(·))
for e in 1..ϵ do
    if e % τ is 0 then
        𝒞 ← CLUSTER(Φ_𝔇(x_i)|_{i=0}^N)
        𝔅 ← IN-BATCH(𝒞)
        ℋ ← topk(ANNS(Φ_𝔇(x_i)|_{i=0}^N, Φ_𝔇(z_l)|_{l=0}^L))
    for Q_ℬ in 𝔅 do
        for i in Q_ℬ do
            𝒫̂_i, ℋ̂_i ← sample(𝒫_i, β), sample(ℋ_i − 𝒫_i, η)
        L_ℬ = {⋃_{i∈Q_ℬ} 𝒫̂_i ∪ ℋ̂_i}
        𝒫^ℬ ← {{𝒫_i ∩ L_ℬ}|_{i∈Q_ℬ}}
        𝒫^L ← {{i | i ∈ Q_ℬ, 𝒫_{i,l} = 1}|_{l∈L_ℬ}}
        ℒ_{𝔇,q2l} ← LOSS(Φ_𝔇(x_i)|_{i∈Q_ℬ}, Φ_𝔇(z_l)|_{l∈L_ℬ}, 𝒫^ℬ)
        ℒ_{𝔇,l2q} ← LOSS(Φ_𝔇(z_l)|_{l∈L_ℬ}, Φ_𝔇(x_i)|_{i∈Q_ℬ}, 𝒫^L)
        ℒ_𝔇 ← λ_𝔇 · ℒ_{𝔇,q2l} + (1 − λ_𝔇) · ℒ_{𝔇,l2q}
        ℒ_{ℭ,q2l} ← LOSS(Φ_ℭ(x_i)|_{i∈Q_ℬ}, Ψ(l)|_{l∈L_ℬ}, 𝒫^ℬ)
        ℒ_{ℭ,l2q} ← LOSS(Ψ(l)|_{l∈L_ℬ}, Φ_ℭ(x_i)|_{i∈Q_ℬ}, 𝒫^L)
        ℒ_ℭ ← λ_ℭ · ℒ_{ℭ,q2l} + (1 − λ_ℭ) · ℒ_{ℭ,l2q}
        ℒ ← λ · ℒ_𝔇 + (1 − λ) · ℒ_ℭ
    adjust Φ, g_1(·), g_2(·) and Ψ to reduce loss ℒ.
```

3.2. Unified Classifier Training with *Pick-some-Labels*

XMC classifiers are typically trained on a shortlist consisting of *all* positive and $\mathcal{O}(Log(L))$ hard negative labels (Dahiya et al., 2021b). As the reader can observe from Figure 1 and algorithm 1, the document and label embedding computation and batch pool is shared between $\Phi_{\mathfrak{D}}$ and $\Phi_{\mathfrak{C}}$. We simply unify the classifier training with that of DE by leveraging the same PSL reduction used for contrastive learning, with only minor changes: $\Phi_{\mathfrak{C}}(\mathbf{x}_i)|_{i \in Q_{\mathcal{B}}}$ replaces $\Phi_{\mathfrak{D}}(\mathbf{x}_i)|_{i \in Q_{\mathcal{B}}}$ and, the label embeddings $\Phi_{\mathfrak{D}}(\mathbf{z}_l)|_{l \in L_{\mathcal{B}}}$ are replaced by $\Psi_l|_{l \in L_{\mathcal{B}}}$. Formally, the multi-class *PSL* loss for classifier training $L_{\mathfrak{C},q2l}$ can be defined as:

$$\mathcal{L}_{\mathfrak{C},q2l} = \sum_{i \in Q_{\mathcal{B}}} \frac{-1}{|\mathcal{P}_i^{\mathcal{B}}|} \sum_{p \in \mathcal{P}_i^{\mathcal{B}}} \log \frac{\exp(\langle \Phi_{\mathfrak{C}}(\mathbf{x}_i), \Psi_p \rangle / \tau)}{\sum_{l \in L_{\mathcal{B}}} \exp(\langle \Phi_{\mathfrak{C}}(\mathbf{x}_i), \Psi_l \rangle / \tau)}$$

Similar to DE training, we find it beneficial to employ a symmetric loss for classifier training as well. The symmetric classifier loss can be defined (with $\lambda_{\mathfrak{C}} = 0.5$) as:

$$\mathcal{L}_{\mathfrak{C}} = \lambda_{\mathfrak{C}} \cdot \mathcal{L}_{\mathfrak{C},q2l} + (1 - \lambda_{\mathfrak{C}}) \cdot \mathcal{L}_{\mathfrak{C},l2q}$$

Finally, we combine the two losses and train together in an end-to-end fashion, thereby achieving Unification of DE and classifier training for XMC.

$$\mathcal{L} = \lambda \mathcal{L}_{\mathfrak{D}} + (1 - \lambda) \mathcal{L}_{\mathfrak{C}}$$

3.3. Inference

For ANNS inference, the label graph can either be created over the encoded label embeddings $\{\Phi_{\mathfrak{D}}(z_l)\}_{l=1}^{L}$ or the label classifier embeddings $\{\mathfrak{N}(\Psi(l))\}_{l=1}^{L}$, which are queried by $\{\Phi_{\mathfrak{D}}(\mathbf{x}_i)\}_{i=1}^{N}$ or $\{\mathfrak{N}(\Phi_{\mathfrak{C}}(\mathbf{x}_i))\}_{i=1}^{N}$ respectively. Even though we train the classifiers over an un-normalized embedding space, we find it empirically beneficial to perform ANNS search over the unit normalized embedding space (Khosla et al., 2020; Gunel et al., 2021). Interestingly, the concatenation of these two embeddings leads to a much more efficient retrieval. More specifically, we create the ANNS retrieval graph over the concatenated label representation $\{\Phi_{\mathfrak{D}}(z_l) \oplus \mathfrak{N}(\Psi(l))\}|_{l=0}^{L}$, which is queried by the concatenated document representations $\{\Phi_{\mathfrak{D}}(\mathbf{x}_i) \oplus \mathfrak{N}(\Phi_{\mathfrak{C}}(\mathbf{x}_i))\}|_{i=0}^{N}$. Intuitively, this is a straightforward way to ensemble the similarity scores from both the embedding spaces.

4. Experiments

Datasets: We benchmark our experiments on 4 standard datasets, comprising of both long-text inputs (LF-Amazon-131K, LF-WikiSeeAlso-320K) and short-text inputs (LF-AmazonTitles-131K, LF-AmazonTitles-1.3M). Details of these datasets can be found at (Bhatia et al., 2016).

Baselines: We compare against two classes of Baselines namely, (i) DE APPROACHES (Φ) consisting of only an encoder (Karpukhin et al., 2020; Xiong et al., 2020; Gupta et al., 2023) and, (ii) CLASSIFIER BASED APPROACHES (Ψ) which use linear classifiers, with or without the encoder (Dahiya et al., 2023a;b; Jain et al., 2023).

Evaluation Metrics: We use popular metrics such as Precision@K (P@K, K $\in \{1, 3, 5\}$) and Propensity-scored Precision@K (PSP@K, K $\in \{1, 3, 5\}$). The Definitions of these metrics can be found at (Bhatia et al., 2016).

Implementation Details and Ablation Study We initialize UNIDEC with a pre-trained 6L-DISTILBERT and train the Φ, $g(\cdot)$ and Ψ with a learning rate of 1e-4, 2e-4 and 1e-3 respectively for 150 epochs, refreshed every $\tau = 5$ epochs, for all datasets. An ablation over best suited β, η, #Epochs, search dimensionality d, and effect of leveraging all positive labels for classifier training, along with choice of $g_1(\cdot)$ and $g_2(\cdot)$, can be found in Appendix C.

SUPCONDE As UNIDEC is the unified training framework, we name our DE approach SUPCONDE, as the *pick-some-labels* loss ends up taking form of *Supcon* loss, as proposed in (Khosla et al., 2020). However, with some key differences as Supcon loss is more popularly used for vision (Kalantidis et al., 2020; Caron et al., 2020) and vision-language (Yang et al., 2022) pre-training tasks.

4.1. Evaluation on XMC Datasets

In these settings, we evaluate SUPCONDE and UNIDEC against both DE and XMC baselines. UNIDEC differs from these baselines in the following ways, (i) on training objective, UNIDEC uses the proposed *PSL* approximation of PAL reduction for both DE and CLF training. (ii) UNIDEC does away with the need of modular training to make training tractable by unifying DE and classifier training. (iii) finally, UNIDEC framework adds explicitly mined hard negatives to the negative mining-aware batches. We discuss impact of each of these aspects in detail in Appendix C. Note that direct comparison against some of the baselines may not be justifiable due to reasons mentioned below.

UNIDEC VS NGAME/DEXA: Table 1 depicts that UNIDEC(Φ + Ψ) consistently outperforms both (i) NGAME(Ψ) (it's direct comparison baseline), where we see gains of $2 - 6\%$ in P@K and, (ii) DEXA(Ψ) (note that it is equipped with auxiliary parameters (γ) during DE training), with $1 - 5\%$ gains in P@K, across all benchmarks.

SUPCONDE VS NGAME/DEXA: SUPCONDE(Φ) yields mixed results in comparison with NGAME(Φ) as we observe P@K to be consistently better where as PSP@K tends to be worse. However, leveraging auxiliary parameters to train the encoder in DEXA($\Phi + \gamma$) helps the model outperform both NGAME(Φ) and SUPCONDE(Φ), which do not lever-

Method	d	P@1	P@3	P@5	PSP@1	PSP@3	PSP@5	d	P@1	P@3	P@5	PSP@1	PSP@3	PSP@5
Long-text →		**LF-Amazon-131K**						**LF-WikiSeeAlso-320K**						
NGAME (Φ)	768	42.61	28.86	20.69	38.27	43.75	48.71	768	43.58	28.01	20.86	30.59	33.29	36.03
DEXA ($\Phi + \gamma$)	768	$\underline{46.64}$	$\underline{30.93}$	$\underline{22.06}$	$\underline{38.83}$	$\underline{44.98}$	$\underline{50.38}$	768	$\underline{46.57}$	$\underline{29.92}$	$\underline{22.26}$	$\underline{32.38}$	$\underline{35.34}$	$\underline{38.27}$
SUPCONDE (Φ)	384	44.91	30.14	21.77	37.42	43.58	49.33	384	44.42	28.58	21.27	30.15	32.72	35.45
SIAMESEXML (Ψ)	300	44.81	-	21.94	37.56	43.69	49.75	300	42.16	-	21.35	29.01	32.68	36.03
NGAME (Ψ)	768	46.95	30.95	22.03	38.67	44.85	50.12	768	45.74	29.61	22.07	30.38	33.89	36.95
DEXA (Ψ)	768	47.12	31.35	22.35	38.86	45.17	50.59	768	47.11	30.24	22.45	32.51	35.37	38.24
UNIDEC ($\Phi \oplus \Psi$)	768	**48.00**	**32.34**	**23.38**	**40.43**	**47.08**	**53.27**	768	**47.74**	**31.15**	**23.34**	**34.08**	**37.52**	**40.85**
Short-text →		**LF-AmazonTitles-131K**							**LF-AmazonTitles-1.3M**					
DPR(Φ)	768	41.85	28.71	20.88	38.17	43.93	49.45	768	44.64	39.05	34.83	32.62	35.37	36.72
ANCE (Φ)	768	42.67	29.05	20.98	38.16	43.78	49.03	768	46.44	41.48	37.59	31.91	35.31	37.25
NGAME (Φ)	768	42.61	28.86	20.69	38.27	43.75	48.71	768	45.82	39.94	35.48	$\underline{\textbf{33.03}}$	35.63	36.80
DEXA ($\Phi + \gamma$)	768	$\underline{44.76}$	$\underline{29.72}$	$\underline{21.18}$	**39.29**	$\underline{44.58}$	$\underline{49.51}$	768	$\underline{51.92}$	44.01	38.85	32.96	$\underline{35.86}$	$\underline{37.31}$
SUPCONDE (Φ)	384	41.77	28.34	20.50	36.89	42.28	47.41	384	51.03	$\underline{45.38}$	$\underline{40.86}$	29.60	33.25	35.25
SIAMESEXML (Ψ)	300	41.42	30.19	21.21	35.80	40.96	46.19	300	49.02	42.72	38.52	27.12	30.43	32.52
NGAME (Ψ)	768	44.95	29.87	21.20	38.25	43.75	48.42	768	54.69	47.76	42.80	28.23	32.26	34.48
DEXA (Ψ)	768	**45.78**	30.13	21.29	38.57	43.95	48.56	768	**55.76**	48.07	42.95	30.01	33.37	35.29
UNIDEC ($\Phi \oplus \Psi$)	768	44.83	**30.28**	**21.76**	39.12	**44.82**	**50.08**	512	53.79	**48.23**	**43.83**	32.19	**35.86**	**37.96**

Table 1. Experimental results showing the effectiveness of **UNIDEC** and **SUPCONDE** against state-of-the-art extreme classifiers. The best-performing results are put in **bold** while the best among DE results are underlined.

age these auxiliary parameters to explicitly capture label correlations. Interestingly, however, SUPCONDE performs on par with DEXA (even increasing P@K by $\sim 2\%$ at the expense of PSP@K) on LF-AmazonTitles-1.3M (largest XMC dataset). This suggests that the proposed *PSL* approximation is more suited for training DE methods on denser, head-heavy datasets (see Appendix B).

SUPCONDE VS DPR/ANCE: Empirical performance of DPR demonstrates the limits of a DE model trained with InfoNCE loss and random in-batch negatives (popular in DR methods). Evidently, ANCE improves over DPR in the P@K metrics, which can be observed as the impact of explicitly mining hard-negative labels per instance instead of solely relying on the random in-batch negatives. Even though, these approaches use 12L-BERT-BASE instead of 6L-DISTILBERT common in XMC methods, ANCE only shows marginal gains over NGAME on both datasets.

Our proposed DE method, SUPCONDE (DISTILBERT), despite using half the # Layers and half the search embedding dimension, is able to perform on par, infact, even surpassing for P@K metrics over LF-AmazonTitles-1.3M dataset.

Search Dimensionality As mentioned before, SUPCONDE outperforms NGAME on P@K metrics across benchmarks. Notably, SUPCONDE does so by projecting (using $g_1(\cdot)$) and training the encoder embeddings in a low-dimension space of $d = 384$. Similarly, for UNIDEC, inference is carried out by concatenating $\mathfrak{N}(\Phi_{\mathfrak{C}})$ and $\Phi_{\mathfrak{D}}$ embeddings. Here, both $g_1(\cdot)$ and $g_2(\cdot)$ consist of linear layers projecting $\Phi(\cdot)$ into a low-dimensional space of $d = 256$ or $d = 384$. On the other hand, all aforementioned baselines use a higher dimension of 768 for both DE and CLF evaluations. For the proprietary Query2Bid-450M dataset, we use final dimension of 64 for all the methods necessitated by constraints of Online serving (see Appendix A).

4.2. Comparison with Spectrum of XMC methods

In this section, we provide a comprehensive comparison (refer Table 2) of our proposed DE SUPCONDE and unified XMC framework UNIDEC with two ends of extreme classification spectrum (refer Figure 1): (i) RENEE, which is initialized with pre-trained NGAME(Φ), does joint end-to-end training of classifiers by backpropagating the gradient calculated over the entire label space instead of approximating the loss over a hard-negative label shortlist and, (ii) DECOUPLED SOFTMAX DE which advocates the efficacy of DE by proposing a novel loss function *decoupled softmax* and compute the contrastive gradient over the entire label space instead of the labels in a mini-batch. Therefore, they can be considered as the upper bound of empirical performance of OvA and DE methods, respectively, on XMC tasks. However, what remains common across both these extreme ends of XMC is the need to compute loss over the entire label space for superior empirical performance.

Table 2 shows that similar, and perhaps better, performance is possible by computing the proposed *PSL* loss over a label shortlist and that our proposed framework UNIDEC strikes a sweet spot in this trade-off.

UNIDEC vs RENEE : RENEE claims that end-to-end classifier training over entire label space can achieve significantly better performance over methods which utilize hard-negative label shortlists and goes on to provide a scalable mechanism to achieve the same. However, we observe that UNIDEC delivers matching performance over P@K and PSP@K metrics by utilizing shortlist that is $86\times - 212\times$

Method	P@1	P@5	PSP@1	PSP@5	$\|Q_{\mathcal{B}}\|$	$\|L_{\mathcal{B}}\|$	GPU
			LF-Amazon-131K				
RENEE (Ψ)	48.05	23.26	39.32	53.51	512	131K	4 x V100
UNIDEC ($\Phi \oplus \Psi$)	48.00	23.38	40.43	53.27	576	**1500**	**1xA6000**
			LF-WikiSeeAlso-320K				
RENEE (Ψ)	47.70	23.82	31.13	40.37	2048	320K	4xV100
UNIDEC ($\Phi \oplus \Psi$)	47.74	23.34	34.20	40.85	677	**1500**	**1xA6000**
			LF-AmazonTitles-1.3M				
DESOFTMAX (Φ)	52.04	40.74	-	-	8192	49152	4xA100
DESOFTMAX (Φ)	42.15	32.97	-	-	8192	8192	4xA100
SUPCONDE (Φ)	51.03	40.86	29.60	35.25	1098	**4000**	**1xA6000**
DESOFTMAX (Φ)	**58.40**	**45.46**	31.36	36.58	8192	1.3M	16xA100
RENEE (Ψ)	56.04	45.32	28.56	36.14	1024	1.3M	8xV100
DESOFTMAX (Φ)	54.01	42.08	28.64	33.58	8192	90112	4xA100
UNIDEC ($\Phi \oplus \Psi$)	53.44	43.55	**32.83**	**38.39**	1098	**3000**	**1xA6000**

Table 2. Experimental results showing the effectiveness of SUPCONDE and UNIDEC against the two ends of XMC spectrum. $|Q_{\mathcal{B}}|$ denotes batch size and $|L_{\mathcal{B}}|$ denotes label pool size. The best-performing approach are in **bold**.

smaller than label space. Additionally, it achieves this feat by consuming significantly lower compute (1 x A6000 against 4 x V100). On LF-AmazonTitles-1.3M however, we observe a trade-off in P@K and PSP@K between the two methods. Detailed discussion about this is provided in **P vs PSP trade-off**. These observations imply that while BCE loss reaches it's empirical limits by scaling over entire label space, we can re-establish the efficacy of using a hard-negative label shortlist by leveraging multi-class based loss as proposed in the UNIDEC framework.

SUPCONDE vs DECOUPLED SOFTMAX DE (with shortlist) : The key difference in these approaches lies in the objective function. While SUPCONDE leverages the proposed *PSL* approximation of the PAL-N loss, the latter propose and train using a novel loss function *decoupled softmax*. As evident in Table 2, (i) For a comparable label pool size (4500 vs 8192), SUPCONDE significantly outperforms DECOUPLED SOFTMAX DE by $\sim$20% in P@K metrics. (ii) To achieve similar performance as SUPCONDE, DECOUPLED SOFTMAX DE need to use an effective label pool size of 49×10^3. However in the same setting, SUPCONDE needs only $1/8^{th}$ batch size and $1/10^{th}$ label pool size. These observations empirically imply that by leveraging multiple positives in the loss enables the batch size and shortlist size to be considerably smaller in-order to capture the same level of data relations information.

UNIDEC vs DECOUPLED SOFTMAX DE : DECOUPLED SOFTMAX DE claims that in-order to establish the empirical limit of DE methods, the *decoupled softmax* loss computation was done over the entire label space which achieved state of the art results on XMC datasets (Gupta et al., 2023). However, this requires large computation cost (16 × A100 GPUs) and can be computationally impractical to scale beyond million labels scale datasets. UNIDEC, on the other hand, scales to a million labels on a single A6000 GPU using a label shortlist of only 3000 labels. Despite this, Table 2 indicates that UNIDEC is better than DECOUPLED SOFTMAX DE in PSP@K metrics although at the trade-off of having lower P@K metrics.

P vs PSP trade-off : We observe that there exists a trade-off between P@K and PSP@K metrics for the methods under comparison especially on LF-AmazonTitles-1.3M. In general, PSP@K metrics are the unbiased variants of P@K metrics (Jain et al., 2016). Intuitively, we believe that using multiple positives leads to more informed and unbiased gradients at every step when compared to using a single positive. This explains the empirical observation as to why UNIDEC is better than DECOUPLED SOFTMAX DE and RENEE at PSP@K metric. We conduct ablations on the effect of the number of positives considered in the *PSL* objective (see Appendix C) which supports the above intuition empirically.Also, in another ablation where we remove explicit ANNS hard negatives, we observe that PSP@K improves further, indicating the ampleness of in-batch negatives for *PSL* loss with respect to PSP@K metrics. Infact, one could argue that PSP@K metrics are a better choice to measure the overall performance (especially when performance on tail labels is crucial) (Bhatia et al., 2016). Therefore, this suggests that UNIDEC is able to achieve a sweet spot between the two compute heavy ends of XMC.

Applicability to Real-World Data : Finally, we also demonstrate the applicability to real-world sponsored search dataset, Query2Bid-450M in Appendix A. SUPCONDE is observed to be $1.15-1.83\%$ better in P@K than leading DR & XMC methods in sponsored search. Additionally, SUPCONDE was deployed on Live Search engine where A/B tests indicated that it was able to improve popular metrics such as IY, CY, CTR, QC over ensemble of DR and XMC techniques by 0.87%, 0.66%, 0.21% and 1.11% respectively.

5. Conclusion

In this paper, we present a new XMC framework, UNIDEC, that to the best of our knowledge, is the first work that successfully trains XMC classifiers with a multiclass loss, rather than an OvA loss by leveraging multiple positives. Furthermore, it eliminates the meta-classifier by unifying the DE and classifier training, which allows us to reuse the in-batch shortlist created for the DE to train the classifier. Such a framework leads to an extremely lightweight SOTA that can be trained on a single GPU, achieving on par performance, if not better, with respect to previous works that trained on 16 GPUs. Further, we show the efficacy of our pure DE approach SUPCONDE against SOTA DE approaches. We hope our work inspires future works to study the proposed *PSL* reduction for multilabel problems as a compute-efficient means to further eliminate the need of high-capacity classifiers in XMC, bringing the scope of this problem closer to the more general dense retrieval regime.

References

Adamic, L. A. and Huberman, B. A. Zipf's law and the internet. *Glottometrics*, 3(1):143–150, 2002.

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*, 108:1329–1351, 2019.

Bhatia, K., Dahiya, K., Jain, H., 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.

Caron, M., Misra, I., Mairal, J., Goyal, P., Bojanowski, P., and Joulin, A. Unsupervised learning of visual features by contrasting cluster assignments. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H. (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 9912–9924. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/70feb62b69f16e0238f741fab228fec2-Paper.pdf.

Chang, W.-C., Yu, H.-F., Zhong, K., Yang, Y., and Dhillon, I. Taming Pretrained Transformers for Extreme Multi-label Text Classification. In *KDD*, 2020.

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 *Conference on Web Search and Data Mining (WSDM'21)*, 2021b.

Dahiya, K., Gupta, N., Saini, D., Soni, A., Wang, Y., Dave, K., Jiao, J., K, G., Dey, P., Singh, A., et al. Ngame: Negative mining-aware mini-batching for extreme classification. In *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining*, pp. 258–266, 2023a.

Dahiya, K., Yadav, S., Sondhi, S., Saini, D., Mehta, S., Jiao, J., Agarwal, S., Kar, P., and Varma, M. Deep encoders with auxiliary parameters for extreme classification. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 358–367, 2023b.

Gao, T., Yao, X., and Chen, D. Simcse: Simple contrastive learning of sentence embeddings. *CoRR*, abs/2104.08821, 2021. URL https://arxiv.org/abs/2104.08821.

Gunel, B., Du, J., Conneau, A., and Stoyanov, V. Supervised contrastive learning for pre-trained language model fine-tuning, 2021.

Gupta, N., Khatri, D., Rawat, A. S., Bhojanapalli, S., Jain, P., and Dhillon, I. S. Efficacy of dual-encoders for extreme multi-label classification, 2023.

Jain, H., Prabhu, Y., and Varma, M. Extreme multi-label loss functions for recommendation, tagging, ranking & other missing label applications. In *KDD*, pp. 935–944, 2016.

Jain, V., Prakash, J., Saini, D., Jiao, J., Ramjee, R., and Varma, M. Renee: End-to-end training of extreme classification models. *Proceedings of Machine Learning and Systems*, 2023.

Jasinska, K., Dembczynski, K., Busa-Fekete, R., Pfannschmidt, K., Klerx, T., and Hullermeier, E. Extreme F-measure Maximization using Sparse Probability Estimates. In *ICML*, June 2016.

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 *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 7987–7994, 2021.

Kalantidis, Y., Sariyildiz, M. B., Pion, N., Weinzaepfel, P., and Larlus, D. Hard negative mixing for contrastive learning. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS'20, Red Hook, NY, USA, 2020. Curran Associates Inc. ISBN 9781713829546.

Karpukhin, V., Oguz, B., Min, S., Lewis, P., Wu, L., Edunov, S., Chen, D., and Yih, W.-t. Dense passage retrieval for open-domain question answering. In Webber, B., Cohn, T., He, Y., and Liu, Y. (eds.), *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 6769–6781, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.550. URL https://aclanthology.org/2020.emnlp-main.550.

Kharbanda, S., Banerjee, A., Schultheis, E., and Babbar, R. Cascadexml: Rethinking transformers for end-to-end multi-resolution training in extreme multi-label classification. *Advances in Neural Information Processing Systems*, 35:2074–2087, 2022.

Kharbanda, S., Banerjee, A., Gupta, D., Palrecha, A., and Babbar, R. Inceptionxml: A lightweight framework with synchronized negative sampling for short text extreme classification. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 760–769, 2023a.

Kharbanda, S., Gupta, D., Schultheis, E., Banerjee, A., Verma, V., and Babbar, R. Gandalf : Data augmentation is all you need for extreme classification, 2023b. URL https://openreview.net/forum?id=05ff9BRSMzE.

Khosla, P., Teterwak, P., Wang, C., Sarna, A., Tian, Y., Isola, P., Maschinot, A., Liu, C., and Krishnan, D. Supervised contrastive learning. *Advances in neural information processing systems*, 33:18661–18673, 2020.

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.

Malkov, Y. A. and Yashunin, D. A. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42:824–836, 2020.

Menon, A. K., Rawat, A. S., Reddi, S., and Kumar, S. Multilabel reductions: what is my loss optimising? *Advances in Neural Information Processing Systems*, 32, 2019.

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 Web Conference 2021*, pp. 3721–3732, 2021b.

Prabhu, Y. and Varma, M. FastXML: A Fast, Accurate and Stable Tree-classifier for eXtreme Multi-label Learning. In *KDD*, August 2014.

Prabhu, Y., Kag, A., Gopinath, S., Dahiya, K., Harsola, S., Agrawal, R., and Varma, M. Extreme multi-label learning with label features for warm-start tagging, ranking and recommendation. In *WSDM*, 2018a.

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, 2018b.

Qu, Y., Ding, Y., Liu, J., Liu, K., Ren, R., Zhao, W. X., Dong, D., Wu, H., and Wang, H. Rocketqa: An optimized training approach to dense passage retrieval for open-domain question answering. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 5835–5847, 2021.

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., Krueger, G., and Sutskever, I. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, 2021.

Sachan, D., Patwary, M., Shoeybi, M., Kant, N., Ping, W., Hamilton, W. L., and Catanzaro, B. End-to-end training of neural retrievers for open-domain question answering. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp. 6648–6662, 2021.

Schultheis, E., Wydmuch, M., Babbar, R., and Dembczynski, K. On missing labels, long-tails and propensities in extreme multi-label classification. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 1547–1557, 2022.

Tagami, Y. AnnexML: Approximate Nearest Neighbor Search for Extreme Multi-label Classification. In *KDD*, August 2017.

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.

Xiong, L., Xiong, C., Li, Y., Tang, K.-F., Liu, J., Bennett, P., Ahmed, J., and Overwijk, A. Approximate nearest neighbor negative contrastive learning for dense text retrieval. *arXiv preprint arXiv:2007.00808*, 2020.

Yang, J., Li, C., Zhang, P., Xiao, B., Liu, C., Yuan, L., and Gao, J. Unified contrastive learning in image-text-label space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 19163–19173, 2022.

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. In *NeurIPS*, 2019.

Zhang, J., Chang, W.-C., Yu, H.-F., and Dhillon, I. Fast multi-resolution transformer fine-tuning for extreme multi-label text classification. *Advances in Neural Information Processing Systems*, 34:7267–7280, 2021.

A. Offline Evaluation and Live A/B testing on Sponsored Search

To demonstrate the effectiveness of our method on proprietary datasets and real-world scenarios, we do experiments in sponsored search setting. The proposed model was evaluated on proprietary dataset for matching queries to advertiser bid phrases (Query2Bid) consisting of 450M labels. Query2Bid-450M dataset was created by mining the logs from search engine and enhancing it through Data Augmentation techniques using a ensemble of leading (proprietary) algorithms such as Information Retrieval models (IR), Dense retrieval models (DR), Generative Non-Autoregressive models (NAR), Extreme-Multi-label Classification models (XMC) and even GPT Inference techniques.

Experimental Setup : The BERT Encoder is initialized with 6-Layer DistilBERT base architecture. Since the search queries and bid phrases are of short-text in nature, a max-sequence-length of 12 is used. We evaluate SUPCONDE against XMC and DR models deployed in production which could scale to the magnitude of chosen dataset. Training batch-size is set to 2048 and other Hyperparameters are chosen to be same as for public benchmark datasets. Training is carried out on **8 V100 GPUs** and could easily complete within **48 hours**. Performance is measured using popular metrics such as Precision@K (P@K) with $K \in 1, 3, 5, 10$.

Method	P@1	P@3	P@5	P@10
NGAME	86.16	73.07	64.61	51.94
SIMCSE	86.08	73.26	65.27	53.51
SUPCONDE	**87.33**	**74.63**	**66.44**	**54.13**

Table 3. Results on Query2Bid-450M dataset for Sponsored Search

Offline Results : Table 3 shows that on SUPCONDE can be 1.15-1.83% more accurate than the leading DR & XMC methods in Sponsored Search setting. This indicates that leveraging SUPCONDE can yield superior gains in real-world search applications.

Live A/B Testing in a Search Engine: SUPCONDE was deployed on Live Search Engine and A/B tests were performed on real-world traffic. The effect of adding SUPCONDE to the ensemble of existing models in the system was measured through popular metrics such as Impression Yield (IY), Click Yield (CY), Click-Through Rate (CTR) and Query Coverage (QC). Refer (Dahiya et al., 2023a) for definitions and details about these metrics. SUPCONDE was observed to improve IY, CY, CTR and QC by **0.87%, 0.66%, 0.21% and 1.11%** respectively. Gains in IY, CY and CTR establish that SUPCONDE is able to predict previously unmatched relations and the predictions are more relevant to the end user. QC boost indicates that SUPCONDE is able to serve matches for queries to which there were no matches before in the system. This ascertains the zero-shot capabilities of the model.

B. Dataset Statistics

Datasets	Benchmark	N	L	APpL	ALpP	AWpP
MS-MARCO	DR	502,931	8,841,823	-	<u>1.1</u>	56.58
LF-AmazonTitles-131K	XMC	294,805	131,073	5.15	2.29	6.92
LF-Amazon-131K	XMC	294,805	131,073	5.15	2.29	6.92
LF-AmazonTitles-1.3M	XMC	2,248,619	1,305,265	38.24	**22.20**	8.74
LF-WikiSeeAlso-320K	XMC	693,082	312,330	4.67	2.11	3.01
Query2Bid-450M	Search Engine	52,029,024	454,608,650	34.61	3.96	-

Table 4. Details of the benchmark datasets with label features. APpL stands for avg. points per label, ALpP stands for avg. labels per point and AWpP is the length i.e. avg. words per point.

MS-MARCO, a representative dataset for DR tasks, has 3.2M documents but on average contains only 1.1 positively annotated answers (label) per question (instance) (Qu et al., 2021). On the other hand, LF-AmazonTitles-1.3M, an XMC dataset which is representative dataset for product recommendation task, has a label space spanning 1.3M Amazon products where each instance (a product title) is annotated (tagged), by $\sim$ 22.2 labels (related product titles) and each label annotates, $\sim$ 38.2 instances. This indicates the broader spectrum of XMC tasks is contrast with zero-shot centric nature of ODQA task.

C. Ablation Study

C.1. UNIDEC Framework

We show the effect of the two individual components $\Phi_{\mathfrak{D}}$ and $\Phi_{\mathfrak{C}}$ of UNIDEC in Table 5. The scores are representative of the evaluation of the respective component of the UniDEC framework, (i) UNIDEC-DE ($\Phi_{\mathfrak{D}}$) performs inference with an ANNS built over $\Phi_{\mathfrak{D}}(\mathbf{z}_l)|_{l=0}^{L}$, (ii) UNIDEC-CLF ($\Phi_{\mathfrak{C}}$) performs inference with an ANNS built over $\mathfrak{N}(\Psi(l))|_{l=0}^{L}$ and (iii) UniDEC uses the ANNS built over the concatenation of both $\{\Phi_{\mathfrak{D}}(\mathbf{z}_l) + \mathfrak{N}(\Psi(l))\}|_{l=0}^{L}$. Notably, concatenation of embeddings leads to a more effective retrieval. We attribute its performance to two aspects, (i) as seen in previous XMC models, independent classifier weights significantly improve the discriminative capabilities of these models and (ii) we hypothesise that normalized and unnormalized spaces learn complementary information which leads to enhanced performance when an ANNS is created on their aggregation. Note that, the individual search dimensions of UNIDEC-DE and UNIDEC-CLF are $d/2$ and searching with a concatenated embedding leads to a fair comparison with other baselines which use a dimensionality of d. Note that for all experiments in the paper, $g_1(\cdot)$, $g_2(\cdot)$ is defined as follows,

$$\begin{aligned}
&\Phi_{\mathfrak{D}}(\cdot),\ \Phi_{\mathfrak{C}}(\cdot) \coloneqq \mathfrak{N}(g_1(\Phi(\cdot))),\ g_2(\Phi(\cdot)) \\
&g_1(\cdot) \coloneqq \mathrm{nn.Sequential}(\mathrm{nn.Linear}(\mathrm{d}_\Phi, \mathrm{d}), \mathrm{nn.Tanh}(), \mathrm{nn.Dropout}(0.1)) \\
&g_2(\cdot) \coloneqq \mathrm{nn.Sequential}(\mathrm{nn.Linear}(\mathrm{d}_\Phi, \mathrm{d}), \mathrm{nn.Dropout}(0.1))
\end{aligned}$$

Method	P@1	P@3	P@5	PSP@1	PSP@3	PSP@5	P@1	P@3	P@5	PSP@1	PSP@3	PSP@5
	LF-AmazonTitles-131K						**w/o Hard Negatives**					
UNIDEC	<u>44.83</u>	<u>30.28</u>	<u>21.76</u>	<u>39.12</u>	<u>44.82</u>	<u>50.08</u>	44.80	30.23	21.71	39.05	44.70	49.90
UNIDEC-DE	41.96	28.56	20.60	37.22	42.75	47.82	43.30	29.09	20.84	37.10	42.40	47.35
UNIDEC-CLF	43.48	29.27	21.05	37.35	42.77	47.92	41.61	28.31	20.45	36.91	42.36	47.46
	LF-Amazon-131K						**w/o Hard Negatives**					
UNIDEC	<u>48.00</u>	<u>32.34</u>	<u>23.38</u>	40.43	47.08	53.27	47.69	32.22	23.29	<u>40.72</u>	<u>47.33</u>	<u>53.51</u>
UNIDEC-DE	44.50	29.86	21.60	37.21	43.21	49.00	47.52	32.04	23.14	40.75	47.07	53.11
UNIDEC-CLF	47.59	32.14	23.22	40.36	46.85	52.95	43.53	29.29	21.33	37.00	42.92	48.90
	LF-WikiSeeAlso-320K						**w/o Hard Negatives**					
UNIDEC	<u>47.91</u>	<u>31.39</u>	<u>23.46</u>	33.29	37.05	40.24	47.64	31.15	23.34	<u>34.18</u>	<u>37.52</u>	<u>40.75</u>
UNIDEC-DE	44.16	28.47	21.26	30.18	32.78	35.60	45.70	29.75	22.16	33.50	36.20	38.94
UNIDEC-CLF	46.47	30.17	22.53	33.09	36.30	39.21	42.72	27.55	20.62	30.02	32.30	35.27
	LF-AmazonTitles-1.3M						**w/o Hard Negatives**					
UNIDEC	<u>53.79</u>	<u>48.23</u>	<u>43.83</u>	32.19	35.85	37.96	48.36	43.81	38.30	<u>34.03</u>	<u>37.25</u>	<u>38.81</u>
UNIDEC-DE	47.49	42.11	37.94	29.04	32.20	34.02	44.71	39.20	34.92	30.17	32.89	34.03
UNIDEC-CLF	46.26	41.77	38.34	30.30	32.82	34.44	41.35	37.07	33.87	32.87	35.31	36.76

Table 5. Experimental results showing the effect of adding ANNS-mined hard negatives while training UNIDEC. Further, the table also shows the scores of inference done using either the DE head embedding $\Phi_{\mathfrak{D}}(\mathbf{x})$ or the normalized CLF head embedding $\mathfrak{N}(\Phi_{\mathfrak{C}}(\mathbf{x}))$, instead of the concatenated vector $\{\Phi_{\mathfrak{D}}(\mathbf{x}) \oplus \mathfrak{N}(\Phi_{\mathfrak{C}}(\mathbf{x}))\}$. The P vs PSP trade-off associated with adding ANNS-mined hard-negatives is clear by observing the <u>underlined</u> values.

C.2. Effect of ANNS-mind Hard Negatives

The effect of explicitly adding ANNS-mined hard negatives is shown via a vis-a-vis comparison with UNIDEC **(w/o Hard Negatives)** in Table 5. More broadly, we observe a P vs PSP trade-off in this ablation as well. We find that not including hard negatives in the shortlist not only trains faster, but also performs better on PSP@K metrics. Consequently, adding (typically $\eta = 6$) hard negatives generally increases performance on P@K metrics, while compromising on PSP@K metrics. While smaller datasets show some improvements by explicitly adding hard negatives, these effects are much more pronounced in the larger dataset, proving its necessity in the pipeline.

C.3. Number of Positive Labels in a Batch ($|\mathcal{P}_i^{\mathcal{B}}|$)

We perform and extensive ablation study of UniDEC on the LF-Amazon-131K and LF-AmazonTitles-1.3M dataset, and discuss the results below. One key difference to note lies in the sparsity of these two datasets. LF-Amazon-131K is sparser dataset as compared to LF-AmazonTitles-1.3M as the former has only 2.29 labels per document, while the latter is significantly more head-heavy i.e. has 22.20 labels per document implying the existence of more "head" documents with a lot of annotating labels. Even though we sample $\beta = 3$, the $\text{Avg}(|\mathcal{P}_i^{\mathcal{B}}|)$ still is 2.0. On the other hand, while we only sample $\beta = 1$, the batch still accumulates a lot of overlapping positives by the virtue of negative-aware batching strategy and the $\text{Avg}(|\mathcal{P}_i^{\mathcal{B}}|)$ gathers to $\sim$ 14 on average for a batch size of 1098.

These stats imply that we pick all labels for a majority of tail and torso queries. However, in order to keep the training tractable, we approximate the loss for head queries over the positives and hard negatives accumulated in a batch.

#	Method	Epochs	Avg($\|\mathcal{P}_i^{\mathcal{B}}\|$)	β	η	**P@1**	**P@3**	**P@5**	**PSP@1**	**PSP@3**	**PSP@5**
1.	SUPCONDE	150	13.9	1	4	51.04	45.39	40.87	29.56	33.21	35.19
2.	SUPCONDE	150	14.2	1	6	51.03	45.38	40.86	29.60	33.25	35.25
3.	SUPCONDE	150	14.8	4	6	50.31	44.76	40.34	30.24	33.90	35.92
4.	SUPCONDE-PAL	150	14.2	1	6	49.06	44.15	40.16	28.17	31.92	34.14
5.	SUPCONXML	100	13.7	2	3	52.84	47.33	42.99	32.55	35.98	37.97
6.	SUPCONXML-POS-CLF	100	13.7	2	3	51.65	46.30	42.13	<u>33.14</u>	<u>36.25</u>	<u>38.11</u>
7.	SUPCONXML	100	14.2	1	6	53.32	47.74	43.37	31.89	35.41	37.49
8.	SUPCONXML-PAL	100	14.5	2	5	51.63	46.42	42.50	28.06	31.80	34.16
9.	SUPCONXML	150	14.5	2	5	<u>53.44</u>	<u>47.89</u>	<u>43.55</u>	32.83	**36.33**	**38.39**
10.	SUPCONXML	150	14.2	1	6	**53.79**	**48.23**	**43.83**	32.19	35.86	37.96

Table 6. Ablations results on UNIDEC showing the effect of each individual component on LF-AmazonTitles-1.3M. The best-performing approach are in **bold**, and the second best are <u>underlined</u>.

C.4. Effect of more Positive and Negative Labels in a Batch

To study the effect of the multiple positive labels in the objective function, we switch β and η across runs on LF-AmazonTitles-131K. We find in Table 5 that while the presence of hard negatives helps increase P@K, increasing η from 3 or 4 to 6 barely leads to any empirical gains. This implies that sampling $\eta = 3$ is good enough for SOTA performance on public benchmarks and we don't need to dramatically scale the batch size. We leave scaling the performance with batch size to a future work.

However, trading an extra positive for a hard negative in #9 and #10 in Table 6 leads to increase in PSP@K, at the cost of P@K. This motivates us to increase the number of positives in a batch. As it is computationally feasible for use to include more positives for classifier training, we run an experiment on two datasets (see UNIDEC-POS-CLF in Table 6 and Table 7). In this experiment, instead of using the same sampled label pool $L_{\mathcal{B}}$ for Φ_C and Ψ training, we randomly sample $\beta_{\mathfrak{C}} = 6$ extra positives and add them to the label pool. We note that the effect of the same is minimal on LF-Amazon-131K, where PSL approximation is equal to PAL reduction for most data points. On the other hand, on LF-AmazonTitles-1.3M, we notice a familiar trade-off between P@K and PSP@K metrics with increasing number of positives.

#	Method	d	Avg($\|\mathcal{P}_i^{\mathcal{B}}\|$)	β	η	λ	**P@1**	**P@3**	**P@5**	**PSP@1**	**PSP@3**	**PSP@5**
1.	SUPCONXML	384	2.0	3	6	0.5	47.77	32.25	23.30	40.33	47.05	53.17
2.	SUPCONXML	384	2.0	3	3	0.3	47.76	32.20	23.26	40.32	46.88	53.01
3.	SUPCONXML	384	2.0	3	6	0.3	48.00	32.34	23.38	40.43	47.08	53.27
4.	SUPCONXML-POS-CLF	384	2.0	3	6	0.3	47.88	32.37	23.40	40.36	47.11	53.34
5.	SUPCONXML	256	2.0	3	6	0.3	47.64	32.15	23.28	39.95	46.66	52.94
6.	SUPCONXML-PAL	256	2.0	3	6	0.3	46.53	31.54	22.89	38.65	45.43	51.69

Table 7. Ablations results on UNIDEC showing the effect of each individual component on LF-Amazon-131K.

C.5. Choice of multi-label reduction

As per previous works (Menon et al., 2019), the normalising term $|\mathcal{P}_i^{\mathcal{B}}|$ in PAL-N reduction has twofold benefits over PAL, (i) it ensures that the head labels do not dominate the loss over tail labels and (ii) it leads to optimisation of recall, while eliminating it leads to optimisation of precision. However, our empirical results show that PAL-N also shows much superior P@K and PSP@K values as compared to the PAL loss on both (i) LF-Amazon-131K - sparse, long-text XMC benchmark with 131K labels and, (ii) LF-AmazonTitles-1.3M - dense, short-text XMC benchmark with 1.3M labels . Our empirical results are shown in Table 6 and Table 7 where the UNIDEC-PAL denotes a UNIDEC model trained with PSL approximation of PAL reduction instead of PAL-N reduction of the multi-label loss. Here, evidently, UNIDEC-PAL performs worse than UNIDEC, making the choice of multi-label reduction clear.