Multi-modal Extreme Classification

Anshul Mittal
Kunal Dahiya[*]
IIT Delhi
me@anshulmittal.org
kunalsdahiya@gmail.com

Shreya Malani[*]
Janani Ramaswamy
Seba Kuruvilla
Microsoft
{shreya0510,seba.2312}@gmail.com
jramaswamy@microsoft.com

Jitendra Ajmera
Keng-hao Chang
Microsoft
jiajmera@microsoft.com
kenchan@microsoft.com

Sumeet Agarwal
IIT Delhi
sumeet@iitd.ac.in

Purushottam Kar
IIT Kanpur
purushot@cse.iitk.ac.in

Manik Varma
Microsoft Research, IIT Delhi
manik@microsoft.com

In fond memory of Sh. Vinay Mittal (1963 - 2022)

Abstract

This paper develops the MUFIN technique for extreme classification (XC) tasks with millions of labels where datapoints and labels are endowed with visual and textual descriptors. Applications of MUFIN to product-to-product recommendation and bid query prediction over several millions of products are presented. Contemporary multi-modal methods frequently rely on purely embedding-based methods. On the other hand, XC methods utilize classifier architectures to offer superior accuracies than embedding-only methods but mostly focus on text-based categorization tasks. MUFIN bridges this gap by reformulating multi-modal categorization as an XC problem with several millions of labels. This presents the twin challenges of developing multi-modal architectures that can offer embeddings sufficiently expressive to allow accurate categorization over millions of labels; and training and inference routines that scale logarithmically in the number of labels. MUFIN develops an architecture based on cross-modal attention and trains it in a modular fashion using pre-training and positive and negative mining. A novel product-to-product recommendation dataset MM-AmazonTitles-300K containing over 300K products was curated from publicly available amazon.com listings with each product endowed with a title and multiple images. On the MM-AmazonTitles-300K and Polyvore datasets, and a dataset with over 4 million labels curated from click logs of the Bing search engine, MUFIN offered at least 3% higher accuracy than leading text-based, image-based and multi-modal techniques.

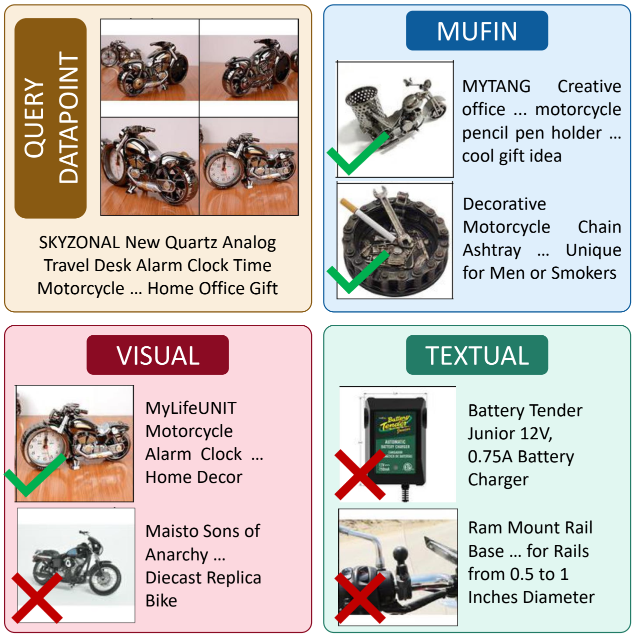

QUERY DATAPOINT
SKYZONAL New Quartz Analog Travel Desk Alarm Clock Time Motorcycle ... Home Office Gift
MUFIN
MYTANG Creative office ... motorcycle pencil pen holder ... cool gift idea
Decorative Motorcycle Chain Ashtray ... Unique for Men or Smokers
VISUAL
MyLifeUNIT Motorcycle Alarm Clock ... Home Decor
Maisto Sons of Anarchy ... Diecast Replica Bike
TEXTUAL
Battery Tender Junior 12V, 0.75A Battery Charger
Ram Mount Rail Base ... for Rails from 0.5 to 1 Inches Diameter

Figure 1. Predictions on the MM-AmazonTitles-300K product-to-product recommendation task illustrate the need for accurate multi-modal retrieval. For a decorative motorcycle-shaped alarm clock as the query product, multi-modal retrieval using MUFIN was able to retrieve visually similar products such as a motorcycle-shaped pencil holder as well as visually dissimilar but related products such as a motorcycle themed ashtray. Recovery using the visual modality alone ignored thematically linked products, instead recovering mostly motorcycle-shaped products. Textual recovery on the other hand fixated on the word "motorcycle" and started recovering accessories for actual motorcycles.

[*] Equal contribution. Author names appear in alphabetical order.

1. Introduction

Extreme Classification (XC). The goal of extreme multi-label classification is to develop architectures to annotate datapoints with the most relevant *subset* of labels from an extremely large set of labels. For instance, given a product purchased by a user, we may wish to recommend to the user, the subset (i.e. one or more) of the most related products from an extremely large inventory of products. In this example, the purchased product is the datapoint and each product in the inventory becomes a potential label for that datapoint. Note that multi-label classification generalizes multi-class classification where the objective is to predict a single mutually exclusive label for a given datapoint. An example of a multi-class problem would be to assign a product to a single exclusive category in a product taxonomy.

Multi-modal XC. An interesting XC application arises when datapoints and labels are endowed with both visual and textual descriptors. Example uses cases include

(1) Product-to-product recommendation [27] with products being represented using their titles and one or more images.

(2) Bid-query prediction [5] where an advertisement with visual and textual descriptions has to be tagged with the list of user queries most likely to lead to a click on that ad.

(3) Identifying compatible outfits where each outfit is described using multiple images and a textual caption [35].

Challenges in Multi-modal XC. Existing multi-modal methods [10, 32, 34, 35] are often *embeddings-only* i.e. categorization is done entirely using embeddings of datapoints and categories obtained from some neural architecture. However, XC research has shown that training classifiers alongside embedding architectures can offer improved results [3, 5, 39]. However, existing XC research focuses mostly on text-based categorization. Bridging this gap requires architectures that offer multi-modal embeddings sufficiently expressive to perform categorization over millions of classes. Also required are routines that can train classifiers over millions of classes and still offer predictions in milliseconds as demanded by real-time applications [3, 5, 12]. This is usually possible only if training and inference scale logarithmically with the number of labels.

Contributions. The MUFIN method targets XC tasks with millions of labels where both datapoints and labels can be endowed with visual and textual descriptors.

(1) MUFIN melds a novel embedding architecture and a novel classifier architecture. The former uses multi-modal attention whereas the latter uses datapoint-label cross attention and high-capacity one-vs-all classifiers.

(2) MUFIN training scales to tasks with several millions of labels by using pre-training and hard-positive and hard-negative mining. MUFIN offers predictions within 3-4 milliseconds per test point even on tasks with millions of labels.

(3) This paper releases the MM-AmazonTitles-300K product-to-product recommendation dataset curated from publicly available amazon.com listings with over 300K products each having a title and multiple images.

(4) MUFIN offers at least 3% higher accuracy than leading text-only, image-only and multi-modal methods on several tasks (MM-AmazonTitles-300K, A2Q-4M) including zero-shot tasks (Polyvore) indicating the superiority of not just MUFIN's classifiers but its embedding model as well.

2. Related Work

Large-scale Visual Categorization. Categorization with a large number of classes has received much attention [9, 11, 22]. Early methods learnt classifiers over hand-crafted or pre-trained features such as HoG [7] ($100K$ classes). Contemporary approaches offer superior accuracies by using task-specific representations obtained from neural architectures. Some of these [9, 22, 30, 36] eschew classifiers entirely and focus on purely embedding-based methods while others train embedding and classifier models jointly using techniques such as hierarchical soft-max, in-batch negative mining [15] and hard-negative mining [41]. However, these works do not consider multi-modal data.

Extreme Classification. XC methods seek to learn classifiers that offer efficient prediction even with millions of labels. Earlier works used fixed or pre-trained features and learnt classifier architectures such as multi-way classification trees [16], one-vs-all classifiers [1, 12] and probabilistic label trees [13]. Recent advances [5, 6, 14, 17, 27, 33, 39] have introduced task-specific neural representations that are jointly learnt alongside the classifiers and offer performance boosts over embedding-only methods. However, these mostly consider tasks with textual descriptions only.

Multi-modal Product Recommendation. The task of recommending related products such as compatible outfits [35] has led to several multi-modal techniques that utilize product images as well as product title or category. ADDE-O [10] learns a disentangled visual representation for outfits so that an outfit with an altered category such as color or size can be recovered simply by appending the query product with a category modifier such as "blue" or "extra large". The Type-aware approach [35] learns product embeddings that respect textual product types but capture product similarity and compatibility. SCE-Net [34] learns image representations that jointly capture multiple aspects of similarity e.g. color, texture without having to learn separate feature spaces for each aspect. SSVR [32] introduces semi- and self-supervised techniques that use textual categories to regularize product image embeddings. S-VAL [18] and CSA-Net [21] perform similarity and compatibility-based retrieval focusing on using the visual modality alone or else using the textual category/type information as a black-box category. Note that none of these methods utilize classifiers and are purely embedding-based methods. Modality fusion techniques have also been explored. Early works adopted late fusion by treating modalities separately till each yielded a score whereas recent works [28] have explored early and *bottle-necked* fusion. MUFIN performs early fusion via its multi-modal attention blocks (see Sec. 3).

Multi-Modal Learning. Methods for multi-modal tasks such as image captioning and associated word prediction [15] have proposed embedding-only solutions (CLIP [30], VisualBERT [19]) as well as classifier architectures (IMRAM [4], M3TR [40]). MUFIN empirically outperforms CLIP and VisualBERT while IMRAM and M3TR could not scale to the datasets used in our experiments.

3. MUFIN MUltimodal extreme classiFIcatioN

Notation. L is the number of labels (*e.g*. number of products available for recommendation, bid queries). N train points are presented as $\{(X_i, \boldsymbol{y}_i)\}_{i=1}^N$. Datapoint i is represented using m_i descriptors (textual e.g. product title and/or visual e.g. product image) as $X_i = \{x_i^1, \ldots, x_i^{m_i}\}$. $\boldsymbol{y}_i \in \{-1, +1\}^L$ is the ground-truth label vector for datapoint i with $y_{il} = +1$ if label $l \in [L]$ is relevant to the datapoint i else $y_{il} = -1$. Each label $l \in [L]$ is represented as $Z_l = \{z_l^1, \ldots, z_l^{m_l}\}$ using m_l textual/visual descriptors.

Motivation for MUFIN's Architecture. MUFIN seeks to obtain an embedding $\hat{\boldsymbol{x}}_i \in \mathbb{R}^D$ for every datapoint X_i and a classifier $\boldsymbol{w}_l \in \mathbb{R}^D$ for every label $l \in [L]$ so that $\boldsymbol{w}_l^\top \hat{\boldsymbol{x}}_i$ is indicative of the relevance of label l to datapoint i. Datapoints and labels each having multiple descriptors i.e. $m_i, m_l \geq 1$ present opportunities to ease this process:
(1) The neural architecture used to obtain datapoint embeddings $\hat{\boldsymbol{x}}_i$ can also be used to obtain label embeddings $\hat{\boldsymbol{z}}_l$ that can serve as a convenient warm start when learning $\boldsymbol{w}_l$ and has been found to accelerate training in XC methods [5,27].
(2) Cross-talk among descriptors of a datapoint and those of a label may make the classifier's job easier by promoting affinity among related datapoint-label pairs. Alignment between descriptors of datapoint i and those of label l can be used to construct an alternate embedding $\hat{\boldsymbol{x}}_i^l$ of the datapoint that is *adapted* to the label l. The goal of this *label-adapted* embedding would be not budge if the label l is irrelevant i.e. $\hat{\boldsymbol{x}}_i^l \approx \hat{\boldsymbol{x}}_i$ if $y_{il} = -1$ but approach the label classifier if the label is relevant i.e. $\hat{\boldsymbol{x}}_i^l \to \hat{\boldsymbol{w}}_l$ if $y_{il} = +1$.
(3) Self-talk among descriptors of the same datapoint/label allow different modalities to interact and produce superior embeddings for that datapoint/label.

MUFIN adopts both bag and vector representations for labels and datapoints to let descriptors retain their identity and allow efficient classification. Attention blocks are used to implement cross-talk and self-talk. Fig. 5 shows that label-adapted embeddings learnt by MUFIN do achieve the objectives stated above by noticing that images of a datapoint appear among images of a relevant label.

Bag Embeddings. A visual architecture $\mathcal{E}_V$ is used to map visual descriptors to $\mathbb{R}^D$ (MUFIN uses ViT-32 [8] with $D = 192$). A textual architecture $\mathcal{E}_T$ (MUFIN uses msmarco-distilbert-base-v4 [31] with $D = 192$) is used to map textual descriptors to $\mathbb{R}^D$. We note that both the ViT and Sentence-BERT models have a native dimensionality of 768. An adaptive maxpool 1D layer was use to project down to obtain 192 dimensional descriptor embeddings. $\mathcal{E}_V, \mathcal{E}_T$ are shared by datapoints and labels. MUFIN maps datapoints and labels to bags of embeddings as shown in Fig. 2(a). A datapoint $X_i = \{x_i^1, \ldots, x_i^{m_i}\}$ is mapped to $\hat{\mathbf{X}}_i^1 = \mathcal{E}(X_i) \in \mathbb{R}^{m_i \times D}$ by first encoding each descriptor of that datapoint using either $\mathcal{E}_V$ or $\mathcal{E}_T$, depending on whether that descriptor is visual or textual, to obtain a bag of pre-embeddings $\hat{\mathbf{X}}_i^0 \in \mathbb{R}^{m_i \times D}$. These are then passed through a self-attention block $\mathcal{A}_S$ (an instantiation of the block depicted in Fig. 2(d)) to obtain $\hat{\mathbf{X}}_i^1 = \mathcal{A}_S(\hat{\mathbf{X}}_i^0) = \mathcal{A}(\hat{\mathbf{X}}_i^0, \hat{\mathbf{X}}_i^0)$. A label $Z_l = \{z_l^1, \ldots, z_l^{m_l}\}$ is similarly mapped to $\hat{\mathbf{Z}}_l^1 = \mathcal{E}(Z_l) \in \mathbb{R}^{m_l \times D}$. We note that the same self-attention block $\mathcal{A}_S$ is used to embed both datapoints and labels.

Vector Embeddings. MUFIN obtains vector embeddings by aggregating and normalizing the bag embeddings offered by $\mathcal{E}$ (see Fig. 2(a)). The vector embedding for a datapoint i is obtained as $\hat{\boldsymbol{x}}_i^1 = \mathfrak{N}(\mathbf{1}^\top \hat{\mathbf{X}}_i^1) \in S^{D-1}$ where $\mathbf{1} \in \mathbb{R}^{m_i}$ is the all ones vector, $\mathfrak{N} : \boldsymbol{v} \mapsto \boldsymbol{v}/\left\|\boldsymbol{v}\right\|_2$ is the normalization operator and S^{D-1} is the unit sphere in D dimensions. Similarly $\hat{\boldsymbol{z}}_l^1 = \mathfrak{N}(\mathbf{1}^\top \hat{\mathbf{Z}}_l^1)$. Given a datapoint i and label l, MUFIN constructs the label-adapted embedding $\hat{\boldsymbol{x}}_i^{2,l}$ for the datapoint as shown in Fig. 2(c). A bag embedding for the datapoint adapted to the label is obtained as $\hat{\mathbf{X}}_i^{2,l} = \mathcal{A}_C(\hat{\mathbf{X}}_i^1, \hat{\mathbf{Z}}_l^1)$ where $\hat{\mathbf{X}}_i^1 = \mathcal{E}(X_i), \hat{\mathbf{Z}}_l^1 = \mathcal{E}(Z_l)$ using a cross-attention block $\mathcal{A}_C$ (Fig. 2(d)) which is vectorized to yield the label-adapted vector embedding $\hat{\boldsymbol{x}}_i^{2,l} = \mathfrak{N}(\mathbf{1}^\top \hat{\mathbf{X}}_i^{2,l})$. Note that $\mathcal{A}_S$ and $\mathcal{A}_C$ do not share parameters.

Scoring Model and Label Classifiers. Given a datapoint i and a label $l \in [L]$, MUFIN assigns a relevance score by taking a dot product of the adapted vector embedding of the datapoint $\hat{\boldsymbol{x}}_i^{2,l}$ with the classifier vector $\boldsymbol{w}_l$ constructed as shown in Fig. 2(b) by linearly combining the (normalized) vector embedding for the label $\hat{\boldsymbol{z}}_l^1$ with a normalized free vector $\mathfrak{N}(\boldsymbol{\eta}_l)$. The free vector $\boldsymbol{\eta}_l$ and the combination weight $\alpha_l \in [0, 1]$ are learnt independently per label.

3.1. Modular Training with MUFIN

Trainable Parameters. The encoder blocks $\mathcal{E}_V, \mathcal{E}_T$, the attention blocks $\mathcal{A}_S, \mathcal{A}_C$, the free vectors and weights $\boldsymbol{\eta}_l, \alpha_l, l \in [L]$ for the label classifiers were trained. MUFIN adopted a training strategy first proposed in the DeepXML paper [6] that breaks training into 4 distinct modules.

Module I: Pre-training. In this module, only the encoders $\mathcal{E}_V, \mathcal{E}_T$ and the self-attention block $\mathcal{A}_S$ were trained in a Siamese fashion. The cross-attention block $\mathcal{A}_C$ was bypassed i.e $\hat{\boldsymbol{x}}_i^{2,l} = \hat{\boldsymbol{x}}_i^1$ and α_l was set to 0 for all $l \in [L]$ so as to also exclude the free vectors. A pretrained ViT-32 model [8] was used to initialize $\mathcal{E}_V$ and its final layer

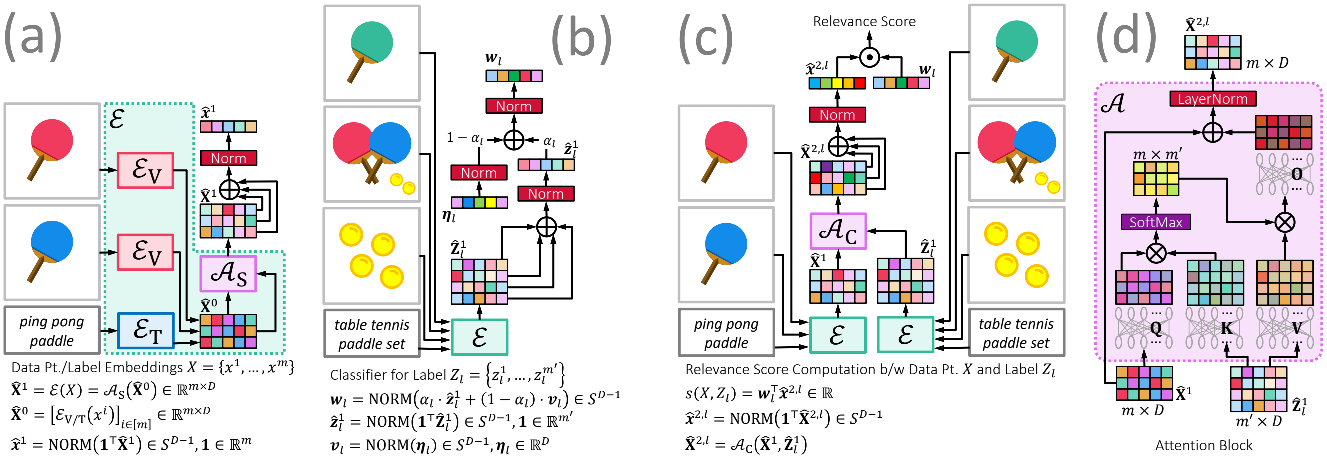

(a)
$\mathcal{E}$
$\mathcal{E}_V$
$\mathcal{E}_V$
$\mathcal{E}_T$
$\mathcal{A}_S$
Norm
$\hat{x}^1$
$\hat{X}^1$
$\hat{X}^0$
ping pong paddle
Data Pt./Label Embeddings $X = \{x^1, \ldots, x^m\}$
$\hat{\mathbf{X}}^1 = \mathcal{E}(X) = \mathcal{A}_S(\hat{\mathbf{X}}^0) \in \mathbb{R}^{m \times D}$
$\hat{\mathbf{X}}^0 = [\mathcal{E}_{V/T}(x^i)]_{i \in [m]} \in \mathbb{R}^{m \times D}$
$\hat{\boldsymbol{x}}^1 = \mathrm{NORM}(\mathbf{1}^\top \hat{\mathbf{X}}^1) \in S^{D-1}, \mathbf{1} \in \mathbb{R}^m$
(b)
w_l
Norm
$1 - \alpha_l$
α_l
$\hat{z}_l^1$
Norm
Norm
η_l
$\hat{Z}_l^1$
$\mathcal{E}$
table tennis paddle set
Classifier for Label $Z_l = \{z_l^1, \ldots, z_l^{m'}\}$
$\boldsymbol{w}_l = \mathrm{NORM}(\alpha_l \cdot \hat{\boldsymbol{z}}_l^1 + (1 - \alpha_l) \cdot \boldsymbol{v}_l) \in S^{D-1}$
$\hat{\boldsymbol{z}}_l^1 = \mathrm{NORM}(\mathbf{1}^\top \hat{\mathbf{Z}}_l^1) \in S^{D-1}, \mathbf{1} \in \mathbb{R}^{m'}$
$\boldsymbol{v}_l = \mathrm{NORM}(\boldsymbol{\eta}_l) \in S^{D-1}, \boldsymbol{\eta}_l \in \mathbb{R}^D$
(c)
Relevance Score
$\hat{x}^{2,l}$
w_l
Norm
$\hat{X}^{2,l}$
$\mathcal{A}_C$
$\hat{X}^1$
$\hat{Z}_l^1$
$\mathcal{E}$
$\mathcal{E}$
ping pong paddle
table tennis paddle set
Relevance Score Computation b/w Data Pt. X and Label Z_l
$s(X, Z_l) = \boldsymbol{w}_l^\top \hat{\boldsymbol{x}}^{2,l} \in \mathbb{R}$
$\hat{\boldsymbol{x}}^{2,l} = \mathrm{NORM}(\mathbf{1}^\top \hat{\mathbf{X}}^{2,l}) \in S^{D-1}$
$\hat{\mathbf{X}}^{2,l} = \mathcal{A}_C(\hat{\mathbf{X}}^1, \hat{\mathbf{Z}}_l^1)$
(d)
$\hat{X}^{2,l}$
$m \times D$
$\mathcal{A}$
LayerNorm
$m \times m'$
O
SoftMax
Q
K
V
$\hat{X}^1$
$m \times D$
$\hat{Z}_l^1$
$m' \times D$
Attention Block

Figure 2. **(a)** The embedding block $\mathcal{E}$ uses a multi-modal self-attention block $\mathcal{A}_S$ to represent a datapoint (or a label) as a bag of embeddings $\hat{\mathbf{X}}^1$ that can be optionally aggregated into a single normalized vector $\hat{\boldsymbol{x}}^1$. **(b)** One-vs-all classifier vectors $\boldsymbol{w}_l$ are learnt for each label by combining the vector representation for that label $\hat{\boldsymbol{z}}_l^1$ with a normalized free vector $\mathfrak{N}(\boldsymbol{\eta}_l)$ ($\mathfrak{N}$ is normalization). **(c)** The relevance score between a datapoint i and label l is computed using the label classifier $\boldsymbol{w}_l$ and a vector representation of the datapoint $\hat{\boldsymbol{x}}_i^{2,l}$ that is adapted to the label l by using the cross-attention block $\mathcal{A}_C$. **(d)** The attention block is instantiated twice, once to implement self-attention as $\mathcal{A}_S : X \mapsto \mathcal{A}(X, X)$ and once to implement cross-attention as $\mathcal{A}_C : (X, Z) \mapsto \mathcal{A}(X, Z)$. The two instantiations use distinct parameters.

was fine-tuned during training. A pre-trained Sentence-BERT model (msmarco-distilbert-base-v4) [31] was used to initialize $\mathcal{E}_T$ and was fine-tuned end-to-end during training. The transformation layers Q, K, V, O in $\mathcal{A}_S$ were initialized to identity. Datapoints and labels were represented by their vector embeddings i.e. $\hat{\boldsymbol{x}}_i^1$ and $\hat{\boldsymbol{z}}_l^1$ respectively. Training encouraged $\hat{\boldsymbol{x}}_i^1$ and $\hat{\boldsymbol{z}}_l^1$ to approach each other for related pairs and repel for unrelated pairs. Mini-batches B were created over labels instead of datapoints by sampling labels randomly. This was observed to improve performance over rare labels [5, 26]. Training with respect to all N datapoints for each label would have resulted in an $\Omega(NL)$ epoch complexity that is infeasible when N, L are both in the millions. Thus, a set $\mathcal{P}_l$ of *hard-positive* datapoints for each label $l \in B$ was chosen among the set $\{i : y_{il} = +1, \langle \hat{\boldsymbol{z}}_l^1, \hat{\boldsymbol{x}}_i^1 \rangle \leq 0.9\}$ since positive datapoints too similar to the label i.e. $\langle \hat{\boldsymbol{z}}_l^1, \hat{\boldsymbol{x}}_i^1 \rangle > 0.9, y_{il} = +1$ would yield vanishing gradients. In-batch negative sampling was also done by selecting for each label $l \in B$, a set $\mathcal{N}_l$ of *hard-negative* datapoints among positive datapoints of other labels in the same minibatch. Hard-positive and negative mining was found to accelerate training by focusing on those label-datapoint pairs that gave most prominent gradients. The following contrastive loss was used to train $\mathcal{E}_V, \mathcal{E}_T$ and $\mathcal{A}_S$ using mini-batches B over labels:

$$\sum_{l \in B} \sum_{i \in \mathcal{P}_l} \sum_{j \in \mathcal{N}_l} \left[\langle \hat{\boldsymbol{z}}_l^1, \hat{\boldsymbol{x}}_j^1 \rangle - \langle \hat{\boldsymbol{z}}_l^1, \hat{\boldsymbol{x}}_i^1 \rangle + \gamma \right]_+$$

MUFIN used $\gamma = 0.2, |\mathcal{P}_l| = 2, |\mathcal{N}_l| = 3$. Capping the sizes of the sets to $|\mathcal{P}_l|, |\mathcal{N}_l| \leq \mathcal{O}(1)$ ensured that an epoch complexity of $\mathcal{O}(L)$ instead of $\Omega(LN)$.

Module II: Augmented Retrieval. The label-wise training strategy adopted by Module I is sympathetic to rare labels but is not aligned to final prediction where labels need to be predicted for datapoints, not the other way round. Moreover, in-batch negative mining is inexpensive but may offer inferior convergence [38]. To accelerate subsequent training, a set of $\mathcal{O}(\log L)$ most promising labels was retrieved for each datapoint. The irrelevant labels in this set would form hard-negatives for subsequent training. MUFIN improved retrieval by exploiting multiple descriptors for each label. After Module I, datapoint vector and label bag embeddings i.e. $\hat{\boldsymbol{x}}_i^1, \hat{\mathbf{Z}}_l^1$ were re-computed. Label centroid vectors [5] were created as $\hat{\boldsymbol{\mu}}_l = \text{mean}\{\hat{\boldsymbol{x}}_i^1 : y_{il} = +1\}$. An ANNS (approximate nearest neighbor search) structure NN^x [24] was created over the set of $\sum_{l \in [L]}(m_l + 1)$ vectors $\bigcup_{l \in [L]} \hat{\mathbf{Z}}_l^1 \cup \{\hat{\boldsymbol{\mu}}_l\}$ with each vector recording the identity of the label to which it belonged. ANNS queries of the form $\text{NN}^x(\hat{\boldsymbol{x}}_i^1)$ were then fired to retrieve for each datapoint i, a set R_i of $\mathcal{O}(\log L) \leq 100$ unique labels. Negative labels in this set i.e. $\{l \in R_i : y_{il} = -1\}$ were well-suited to serve as hard-negative labels for the datapoint i. Ablations in Sec. 4 show that this technique offers superior performance than if the ANNS structure NN^x were to be created over vector representations $\hat{\boldsymbol{z}}_l^1$ of the labels instead. Note that we could have fired bag-queries on the datapoint side as well i.e. fire m_i ANNS queries for datapoint i, one for each element of the datapoint bag $\hat{\mathbf{X}}_i^1 = \mathcal{E}(X_i)$. However that would substantially increase retrieval time by a factor of m_i ($m_i \approx 5$ for MM-Amazon-300K) and was thus avoided. Instead, the approach adopted by MUFIN ensures superior retrieval at the cost of a single ANNS query per datapoint.

Module III: Pre-training to Fine-tuning Transfer. The encoders $\mathcal{E}_V, \mathcal{E}_T$ and the self-attention block $\mathcal{A}_S$ were initialized to their values after Module I training. The transformation layers Q, K, V, O within $\mathcal{A}_C$ are initialized to identity. The free vectors $\boldsymbol{\eta}_l$ were all offered uniform Xavier initialization and $\alpha_l = 0.5$ was initialized for all labels $l \in [L]$.

Module IV: Fine-tuning. $\mathcal{E}_V, \mathcal{E}_T, \mathcal{A}_S$ were further fine-tuned whereas $\mathcal{A}_C, \boldsymbol{\eta}_l, l \in [L]$ and $\alpha_l, l \in [L]$ were trained from scratch. Mini-batches B were created over datapoints to align with the final prediction task. For each $i \in B$, a set $\mathcal{S}_i$ of random positive labels was chosen. A set $\mathcal{T}_i$ of hard-negative labels was chosen among negative labels in the shortlist R_i constructed in Module II. A datapoint i was represented using label-adapted embeddings w.r.t. the positive and hard-negative labels shortlisted for them i.e. $\left\{\hat{\boldsymbol{x}}_i^{2,l} : l \in \mathcal{S}_i \cup \mathcal{T}_i\right\}$. Labels were represented using their classifier vectors $\boldsymbol{w}_l$ (see Fig. 2(b)). The following cosine embedding loss was used to train $\mathcal{E}_V, \mathcal{E}_T, \mathcal{A}_S, \mathcal{A}_C$ and $\boldsymbol{\eta}_l, \alpha_l, l \in [L]$ using mini-batches B of datapoints:

$$\sum_{i \in B}\left\{\sum_{l \in \mathcal{S}_i}\left(1 - \left\langle \hat{\boldsymbol{x}}_i^{2,l}, \boldsymbol{w}_l\right\rangle\right) + \sum_{k \in \mathcal{T}_i}\left[\left\langle \hat{\boldsymbol{x}}_i^{2,k}, \boldsymbol{w}_k\right\rangle - \gamma\right]_+\right\}$$

MUFIN used $\gamma = 0.5, |\mathcal{S}_i| = 2, |\mathcal{T}_i| = 12$. Capping the set sizes to $|\mathcal{S}_i|, |\mathcal{T}_i| \leq \mathcal{O}(\log L)$ ensured that an epoch complexity of $\mathcal{O}(N \log L)$ instead of $\Omega(NL)$.

Prediction with MUFIN. Given a test point X_t with m_t descriptors $X_t = \left\{x_t^1, \ldots, x_t^{m_t}\right\}$, its vector representation $\hat{\boldsymbol{x}}_t^1 = \mathfrak{N}\left(\mathbf{1}^\top \mathcal{E}(X_t)\right)$ is used to query the ANNS structure and perform augmented retrieval of labels to yield a shortlist $R_t = \text{NN}^x(\hat{\boldsymbol{x}}_t^1)$ of $100 \leq \mathcal{O}(\log L)$ labels. For each retrieved label $l \in R_t$, a *similarity* score is assigned as $a_{tl} \overset{\text{def}}{=} \max \left\langle \hat{\boldsymbol{x}}_t^1, \boldsymbol{v}\right\rangle, \boldsymbol{v} \in \hat{\mathbf{Z}}_l^1 \cup \{\hat{\boldsymbol{\mu}}_l\}$ (recall that in augmented retrieval, each label $l \in [L]$ contributes $m_l + 1$ entries to the ANNS structure). Vector representations for X_t adapted to all shortlisted labels i.e. $\left\{\hat{\boldsymbol{x}}_t^{2,l}, l \in R_t\right\}$ are computed and the corresponding label classifiers applied to yield *classifier* scores $c_{il} \overset{\text{def}}{=} \left\langle \boldsymbol{w}_l, \hat{\boldsymbol{x}}_t^{2,l}\right\rangle$ for each $l \in R_t$. The classifier and similarity scores are then combined linearly as $s_{tl} = \beta \cdot c_{tl} + (1 - \beta) \cdot a_{tl}$. A fixed value of $\beta = 0.7$ was used. Final predictions are made in descending order of the scores s_{tl}. The prediction time complexity of MUFIN is derived in App. A in the supplementary.

Handling unseen labels with MUFIN ($\boldsymbol{\alpha} = \mathbf{1}$). A variant dubbed "MUFIN ($\alpha = 1$)" was developed to handle unseen labels (for which supervision was not available during training) by setting $\alpha_l = 1$ for all $l \in [L]$. This causes MUFIN to start using the vector label representation itself as the classifier i.e. $\boldsymbol{w}_l \equiv \hat{\boldsymbol{z}}_l^1$ and give relevance scores of the form $\left\langle \hat{\boldsymbol{z}}_l^1, \hat{\boldsymbol{x}}_t^{2,l}\right\rangle$. Note that the cross-attention block $\mathcal{A}_C$ can still be applied to yield adapted datapoint embeddings $\hat{\boldsymbol{x}}_t^{2,l}$ even w.r.t. unseen labels. The variant MUFIN ($\alpha = 1$) sets $\alpha_l = 1$ for all labels $l \in [L]$ to ensure consistency.

4. Experimental Results

Datasets. Dataset construction details are given in App. B in the supplementary [link]. Tab. 1 presents dataset statistics. The Polyvore FITB task relies on precomputed shortlists for each query and does not present a satisfactory benchmark for multi-modal XC methods where the goal is to retrieve results directly from a catalog of millions of labels. Other multimodal datasets [20, 23] were found similarly lacking. Thus, two other tasks were considered. The A2Q-4M dataset presents a heterogeneous task where datapoints have multi-modal descriptors but labels are purely textual. The MM-AmazonTitles-300K dataset also presents occasional datapoints/labels with either the text or vision modality missing entirely. Thus, these tasks demand that the architecture be resilient to missing modes.

MM-AmazonTitles-300K: An XC product-to-product recommendation dataset was curated from an Amazon click dump [29]. Given a query product, the task is to retrieve the subset of the most relevant products from a catalog of over 300K unique products. Each product is represented by a title and up to 15 images. This dataset has been released at the The Extreme Classification Repository [2] [link].

A2Q-4M: A large bid-query prediction task was mined from the internal click logs of the Bing search engine. Given an ad as a datapoint represented by an image and textual description, the task is to predict the subset of user queries (textual) most likely to lead to a click on that ad.

Polyvore-Disjoint: Polyvore is a popular fashion website where users can create outfit compositions [35]. The Fill-In-The-Blank (FITB) task requires the most compatible outfit to be chosen from a pre-computed shortlist given an incomplete *query* outfit with 4–5 images and short captions.

Baselines. Due to lack of space, a detailed discussion on the baselines is provided in App. C of the supplementary.

MM-AmazonTitles-300K: MUFIN was compared to leading text-based XC methods [5, 16, 25, 27, 37, 39] and leading multi-modal methods CLIP [30] and VisualBert [19] that employ cross-modal pre-training to embed related items (*e.g.* an image and its associated caption) closeby. AttentionXML [39] employs label-specific datapoint representations similar to MUFIN. SiameseXML was augmented to utilize a DistilBERT architecture similar to MUFIN instead of the bag-of-embeddings model used in [5]. The details of the augmentation are given in App. C. CLIP and VisualBert use the ViT and Resnet-101 image encoders respectively. To offer a fair comparison, pre-trained encoders for these methods were injected into MUFIN's training pipeline and afforded the same self-attention, cross attention and classifier architectures. Tab. 2 shows that MUFIN outperformed even augmented versions of these algorithms.

Table 1. Statistics for datasets used to benchmark MUFIN. For Polyvore-Disjoint, a '-' indicates that only unseen labels were available for recommendation at test time. For A2Q-4M, a ‡ indicates numbers redacted for the proprietary dataset.

Dataset	Train Datapoints N	Labels L	Test Instances $N^{'}$	Average Labels per datapoint	Average Tokens per datapoint	Average Images per datapoint
Polyvore-Disjoint	16,995	-	15,145	1	27.31	4
MM-AmazonTitles-300K	586,781	303,296	260,536	8.13	20.41	4.91
A2Q-4M	9,618,490	4,528,191	3,933,149	‡	‡	‡

A2Q-4M: Multi-modal baseline methods struggled to scale to this dataset so comparisons were made only to the leading text-based method SiameseXML [5].

Polyvore-Disjoint: MUFIN was compared to leading methods including ADDE-O [10], CSA-Net [21], Type-aware [35], S-VAL [18], SCE-Net average [34] and SSVR [32]. Since this dataset offers only unseen labels as recommendation candidates at test time, only the MUFIN ($\alpha = 1$) variant was executed for fair comparison.

Evaluation Metrics. Standard XC metrics *e.g.* area under the curve (AUC), precision (P@k), nDCG (N@k), and recall (R@k) were used for the MM-AmazonTitles-300K and A2Q-4M tasks. Classification accuracy was used for the multi-class Polyvore-Disjoint task as is standard [10,21,35].

Hyperparameters. MUFIN uses ViT-32 [8] as the image encoder $\mathcal{E}_V$ with 32 × 32 patches and the msmarco-distilbert-base-v4 architecture [31] as the text encoder $\mathcal{E}_T$. The AdamW optimizer with a one-cycle cosine scheduler with warm start of 1000 iterations was used. MUFIN could train on a 24-core Intel Skylake 2.4 GHz machine with 4 V100 GPUs within 48 hrs on the A2Q-4M dataset. See App. D in the supplementary for hyperparameter details.

4.1. Results and Discussion

MM-AmazonTitles-300K. Tab. 2 shows MUFIN gave 3.6–11% higher P@1 than text-based XC methods. MUFIN is 5.5% better in P@1 than AttentionXML [39] that also employs label-specific datapoint representations. MUFIN is also 3.5% more accurate than SiameseXML that uses a DistilBERT encoder similar to MUFIN. This indicates the benefit of melding multi-modal information with high-capacity classifier architectures. MUFIN's lead is similarly high in terms of other metrics such as P@5 and R@10. MUFIN also gave 3.2-12% higher P@1 than multi-modal methods CLIP and VisualBERT. It is notable that the methods being compared to are variants of CLIP and VisualBERT that were offered MUFIN's attention modules and training strategies. App. E shows that MUFIN's lead over these methods could be as high as 30% if they are not offered these augmentations. This highlights the utility of MUFIN's task-specific pre-training in Module I.

A2Q-4M. MUFIN could train on this dataset with 9M training points within 48 hrs on 4×V100 GPUs. MUFIN achieved 47.56% P@1 compared to 44.46% P@1 by SiameseXML. MUFIN also offered predictions within 4 milliseconds per test datapoint on a single V100 GPU. MUFIN is able to scale to tasks with several millions of labels offering prediction times suitable for real-time applications.

Polyvore-Disjoint. Tab. 3 presents results on the FITB task where MUFIN ($\alpha = 1$) could be 3-4% more accurate than the next-best method. MUFIN is encoder-agnostic and continues to outperform existing methods even if MUFIN replaces its $\mathcal{E}_V$ with Resnet18 (used by ADDE-O [10]).

Table 2. Results on MM-AmazonTitles-300K. MUFIN outperforms state-of-the-art XC methods by 3–11% in P@1 as well as R@10. MUFIN also outperforms state-of-the-art vision+language pre-training strategies by 3–12% in P@1 and 4–13% in R@10. Recall that the multi-modal techniques were augmented with MUFIN's pipeline. App. E shows that MUFIN's lead rises significantly if the methods are not offered these augmentations. The column t_{pred} reports the per-datapoint prediction times in milliseconds for various methods. MUFIN offers millisecond level prediction comparable to or better than existing methods.

Method	**P@1/ N@1**	**P@5**	**N@5**	**R@10**	**AUC**	t_{pred} **(ms)**
MUFIN	**52.3**	**34.76**	**50.46**	**50.63**	**0.60**	1.32
Textual (XC)						
SiameseXML [5]	48.64	32.99	47.46	47.72	0.57	0.82
ECLARE [27]	47.84	32.22	46.04	46.08	0.55	0.08
AttentionXML [39]	46.45	31.17	44.34	43.73	0.53	4.33
Bonsai [16]	47.34	31.65	45.4	45.04	0.55	5.71
MACH [25]	42.22	28.14	40.39	39.7	0.49	0.46
XT [37]	41.45	27.71	39.64	38.91	0.52	5.21
Visual + Textual						
CLIP [30] + MUFIN	40.49	27.38	38.45	37.19	0.485	6.46
VisualBert [19] + MUFIN	49.11	32.35	46.95	46.43	0.58	8.35

Category-wise Analysis. To analyze the gains offered by MUFIN, the performance of various algorithms was considered on the 20 unique categories of 300K products in the MM-AmazonTitles-300K dataset. Fig. 3 shows that MUFIN's multi-modal recommendations are 2–6% more accurate on all popular categories than SiameseXML (that used the same text encoder $\mathcal{E}_T$ as MUFIN). Tab. 6 and Fig. 8 in the supplementary present qualitative results that show that the trend persists and MUFIN offers superior performance than competing methods on almost all categories irrespective of the popularity of the category.

Table 3. Results on the FITB task on Polyvore-Disjoint. MUFIN is 3-4% more accurate compared to the next best method.

Methods	FITB Accuracy
MUFIN ($\alpha = 1$)	**64.17**
MUFIN (Resnet18, $\alpha = 1$)	**61.63**
Visual + Textual	
ADDE-O [10]	60.53
Type-aware [35]	55.65
SCE-Net average [34]	53.67
SSVR [32]	51.5
Visual	
CSA-Net [21]	59.26
S-VAL [18]	54.3

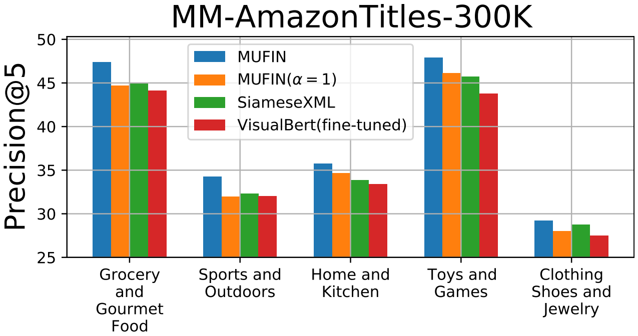


Figure 3. MUFIN outperforms baseline methods on almost all categories. Only the top 5 categories are shown here to avoid clutter. Tab. 6 in the supplementary contains results on all categories.

Label popularity. To analyze MUFIN's performance on rare and popular labels, labels were divided into 5 *equi-voluminous* bins of increasing label frequency such that each bin had an equal number of datapoint-label pairs from the ground truth. Fig. 4 shows that MUFIN and MUFIN ($\alpha = 1$) outperform the baseline methods across all bins.

Impact of Cross Attention ($\mathcal{A}_c$). Fig. 5a shows the cross-attention heat map generated by $\mathcal{A}_C$ between a datapoint i and a relevant label in the retrieved shortlist R_i for that datapoint. MUFIN was able to match a chair in a datapoint image [Image 5] to a similar chair in the background of a label image [Image 3] (magnified in Fig. 5b). Fig. 5c shows that for a given datapoint ($\widehat{\boldsymbol{x}}^1$, ●), cross-attention allowed MUFIN to generate a label-adapted datapoint representation ($\widehat{\boldsymbol{x}}^{2,+}$, ★) that is embedded close to the relevant label ($\boldsymbol{w}_+$, ▲). However, the label-adapted representation of the same datapoint ($\widehat{\boldsymbol{x}}^{2,-}$, ★) is unmoved for an irrelevant label ($\boldsymbol{w}_-$, ▲). Label-adapted representations allow MUFIN to boost the score for relevant labels and rank them higher.

Label semantics. Type-aware methods [10, 21, 35] have demonstrated that explicitly incorporating label categories while training can improve model accuracy. However, in XC settings with millions of labels, label hierarchies are often unavailable or incomplete [2]. Fig. 6 depicts the label classifiers ($\boldsymbol{w}_l$) learnt by MUFIN using t-SNE representations. MUFIN could identify category-based relationships among labels without any explicit feedback on cluster identity. MUFIN's clusters exhibit sub-clustering that can be attributed to the fact that labels belonging to a category can be further grouped into sub-categories, *e.g.* *"Home and Kitchen"* can be further clustered into *"Furniture"* and "*Utensils*". Thus MUFIN draws its gains on diverse label types (Figs. 3 and 4) by deducing label-datapoint relationships from multi-modal information (Figs. 5 and 6).

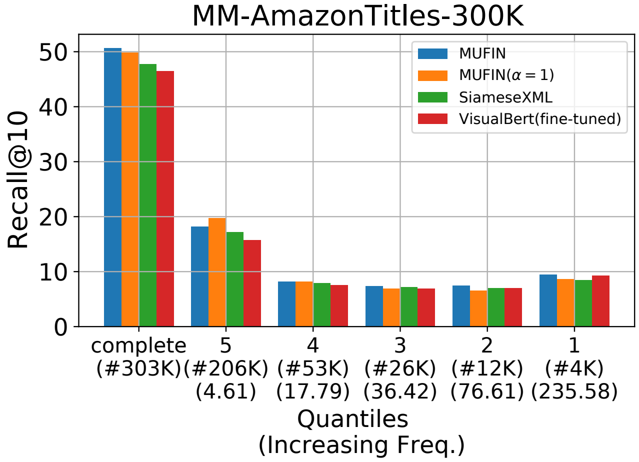


Figure 4. Analyzing the performance of MUFIN and other methods on popular vs. rare labels. Labels were divided into 5 bins in increasing order of popularity (left-to-right). The plots show the overall R@10 of each method (histogram group "complete") and the contribution of each bin to this value. The results indicate that MUFIN's performance on popular labels (histogram group 1) does not come at the cost of performance in rare labels. Other methods seem to exhibit a trade-off between rare and popular labels.

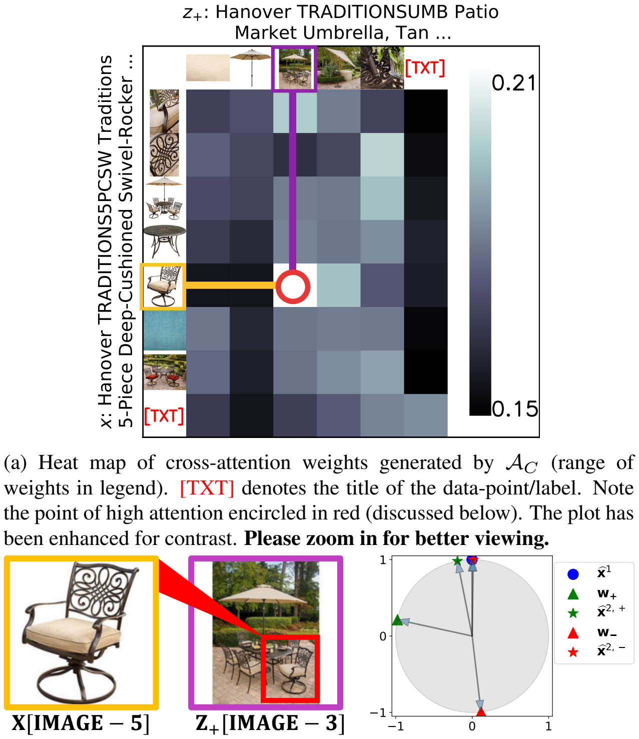
4.2. Ablation

This section investigates design choices made by MUFIN for its key components - sampling, retrieval, representation, and ranker (scorer). Tab. 4 summarizes the ablation results. The ablation experiments have been explained in detail in App. F in the supplementary [link].

Sampling. Removing hard +ve sampling (MUFIN-no +ve) causes a 1% drop in P@5. Removing hard -ve and +ve sampling (MUFIN-no +ve, -ve) leads to a 1.5% drop in P@5.

Retrieval. Recall from Sec. 3 that retrieval of label shortlists R_i could have been done over label embeddings $\hat{z}_l^1$ or bag embeddings $\hat{\mathbf{Z}}_l^1$ of the labels. The augmented retrieval strategy of MUFIN (MUFIN-P-I-bag) can be 0.3% and 1% more accurate in R@10 and P@1 as compared to retrieval based on vector embeddings $\hat{z}_l^1$ alone (MUFIN-P-I-vec).

Representation. MUFIN was 3-4% more accurate than the MUFIN-ConCat variant that concatenated $\hat{\boldsymbol{x}}^1, \hat{z}_l^1$ followed by two feed-forward layers instead of using the cross-



(a) Heat map of cross-attention weights generated by $\mathcal{A}_C$ (range of weights in legend). [TXT] denotes the title of the data-point/label. Note the point of high attention encircled in red (discussed below). The plot has been enhanced for contrast. **Please zoom in for better viewing.**

(b) The high attention weight (highlighted as a red circle in Fig. 5a) was a result of the cross-attention block $\mathcal{A}_C$ being able to align [Image 5] of the datapoint X to [Image 3] if the label Z_+ by observing that the chair depicted in [Image 5] X is closely related to the chair in the background of [Image 3] of Z_+.

(c) Analysing the impact of label-adapted representations. The vector embedding of the data point was adapted to the relevant label Z_+ (the very one referenced in Figs. 5a and 5b) as well as an irrelevant label Z_- (see discussion below).

Figure 5. The impact of multi-modal cross-attention in MUFIN. Fig. 5a shows the cross-attention heat map between the datapoint X and a relevant label Z_+. Fig. 5b shows that cross-attention is able to identify objects in X among objects in the background of the relevant label Z_+. Fig. 5c shows how this helps boost the scores assigned to relevant labels. $\hat{\boldsymbol{x}}^1$ (●) represents the non-adapted vector embedding of the datapoint X. $\boldsymbol{w}_+$(▲) and $\boldsymbol{w}_-$(▲) represent the classifier vectors for the relevant and irrelevant labels Z_+ and Z_- respectively. Similarly, $\hat{\boldsymbol{x}}^{2,+}$(★) and $\hat{\boldsymbol{x}}^{2,-}$(★) represent the vector embedding of the datapoint adapted to Z_+ and Z_- respectively. Notice how adaptation moves $\hat{\boldsymbol{x}}^{2,+}$(★) closer to $\boldsymbol{w}_+$(▲) allowing the relevant label Z_+ to get a higher score. On the other hand, adaptation has no effect when done with respect to an irrelevant label (note that $\hat{\boldsymbol{x}}^1$ (●) and $\hat{\boldsymbol{x}}^{2,-}$(★) do indeed almost overlap). Fig. 5c was plotted by projecting the vectors $\hat{\boldsymbol{x}}^1, \boldsymbol{w}_+, \boldsymbol{w}_-, \hat{\boldsymbol{x}}^{2,+}, \hat{\boldsymbol{x}}^{2,-}$ onto $\mathbb{R}^2$ using a t-SNE embedding.

attention block $\mathcal{A}_C$. Removing the self-attention block from Modules I-IV (MUFIN-no $\mathcal{A}_S$) led to a 1.6% drop in P@5. **Ranker.** MUFIN's novel scoring architecture can be upto 1.5% more accurate in terms of P@5 than variants that either exclude the cross-attention block (MUFIN-no $\mathcal{A}_C$) or the one-vs-all classifiers (MUFIN-($\alpha = 1$)).

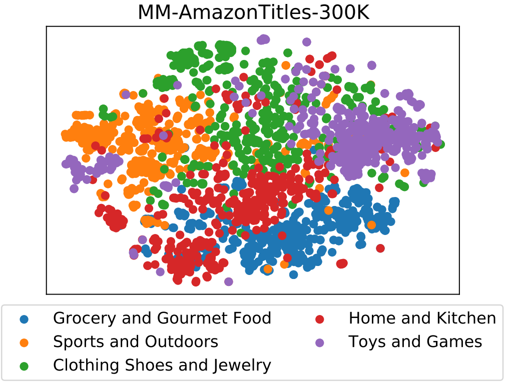


Figure 6. t-SNE representations for classifiers $\boldsymbol{w}_l, l \in [L]$ learnt by MUFIN show that labels belonging to the same category are clustered together. Only the 5 most popular categories are shown.

Table 4. An ablation study exploring alternate architecture and training choices for MUFIN. Choices made by MUFIN could lead to 3-4% gain in P@1 and 1-2% gain in R@10 than alternatives.

Ablation	**P@1/ N@1**	**P@5**	**N@5**	**R@10**
MUFIN	**52.3**	**34.76**	**50.46**	**50.63**
Sampling				
MUFIN-no +ve	50.35	33.71	48.91	49.19
MUFIN-no +ve, -ve	49.69	33.33	47.9	48.76
Retrieval				
MUFIN-P-I-bag	42.72	28.8	42.03	44.49
MUFIN-P-I-vec	41.71	28.26	41.31	44.2
Representation				
MUFIN-ConCat	49.61	32.89	47.87	47.97
MUFIN-no $\mathcal{A}_S$	49.98	33.16	48.11	48.11
Ranker				
MUFIN-no $\mathcal{A}_C$	50.22	33.87	49.03	49.68
MUFIN-($\alpha = 1$)	49.25	33.19	48.53	49.87

The ablations show that MUFIN's design choices with respect to hard +ve, -ve sampling, self- and cross-attention, and one-vs-all classifiers, each offer performance boosts.

Dataset and Supplementary Material. The MM-AmazonTitles-300K dataset can be downloaded at http://manikvarma.org/downloads/XC/XMLRepository.html. MUFIN pseudocode, implementation details, additional results and discussions on limitations of MUFIN, ethical considerations and future work are presented in the supplementary at http://manikvarma.org/pubs/mittal22-supp.pdf.

Acknowledgements

The authors thank the reviewers for helpful comments. The authors are grateful to the High-Performance Computing facility and staff at IIT Delhi. AM is supported by a Google PhD Fellowship. AM, KD, and SA acknowledge funding from a Microsoft Research India grant on Extreme Classification. PK is supported by Microsoft Research India via consultancy grant no MRLIPL/CS/2021296.

References

[1] R. Babbar and B. Schölkopf. Dismec: Distributed sparse machines for extreme multi-label classification. In *WSDM*, 2017. 2

[2] K. Bhatia, K. Dahiya, H. Jain, A. Mittal, Y. Prabhu, and M. Varma. The extreme classification repository: Multi-label datasets and code, 2016. 5, 7, 12

[3] W-C. Chang, H.-F. Yu, K. Zhong, Y. Yang, and I. Dhillon. Taming pretrained transformers for extreme multi-label text classification. In *KDD*, 2020. 2

[4] H. Chen, G. Ding, X. Liu, Z. Lin, J. Liu, and J. Han. IMRAM: Iterative Matching with Recurrent Attention Memory for Cross-Modal Image-Text Retrieval. In *CVPR*, 2020. 3

[5] K. Dahiya, A. Agarwal, D. Saini, K. Gururaj, J. Jiao, A. Singh, S. Agarwal, P. Kar, and M Varma. SiameseXML: Siamese Networks meet Extreme Classifiers with 100M Labels. In *ICML*, 2021. 2, 3, 4, 5, 6, 13, 15

[6] K. Dahiya, D. Saini, A. Mittal, A. Shaw, K. Dave, A. Soni, H. Jain, S. Agarwal, and M. Varma. DeepXML: A Deep Extreme Multi-Label Learning Framework Applied to Short Text Documents. In *WSDM*, 2021. 2, 3

[7] T. Dean, M. A. Ruzon, M. Segal, J. Shlens, S. Vijayanarasimhan, and J. Yagnik. Fast, Accurate Detection of 100,000 Object Classes on a Single Machine. In *CVPR*, 2013. 2

[8] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021. 3, 6, 14

[9] H. Guo and S. Wang. Long-tailed multi-label visual recognition by collaborative training on uniform and re-balanced samplings. In *CVPR*, 2021. 2

[10] Y. Hou, E. Vig, M. Donoser, and L. Bazzani. Learning Attribute-Driven Disentangled Representations for Interactive Fashion Retrieval. In *ICCV*, 2021. 2, 6, 7, 12

[11] D. Huynh and E. Elhamifar. Interactive multi-label cnn learning with partial labels. In *CVPR*, 2020. 2

[12] H. Jain, V. Balasubramanian, B. Chunduri, and M. Varma. Slice: Scalable Linear Extreme Classifiers trained on 100 Million Labels for Related Searches. In *WSDM*, 2019. 2

[13] K. Jasinska, K. Dembczynski, R. Busa-Fekete, K. Pfannschmidt, T. Klerx, and E. Hullermeier. Extreme f-measure maximization using sparse probability estimates. In *ICML*, June 2016. 2

[14] T. Jiang, D. Wang, L. Sun, H. Yang, Z. Zhao, and F. Zhuang. LightXML: Transformer with Dynamic Negative Sampling for High-Performance Extreme Multi-label Text Classification. In *AAAI*, 2021. 2

[15] A. Joulin, L. Maaten, A. Jabri, and N. Vasilache. Learning Visual Features from Large Weakly Supervised Data. In *ECCV*, 2016. 2, 3

[16] S. Khandagale, H. Xiao, and R. Babbar. Bonsai: diverse and shallow trees for extreme multi-label classification. *ML*, 2020. 2, 5, 6, 13

[17] S. Kharbanda, A. Banerjee, A. Palrecha, and R. Babbar. Embedding convolutions for short text extreme classification with millions of labels, 2021. 2

[18] D. Kim, K. Saito, S. Mishra, S. Sclaroff, K. Saenko, and B. A. Plummer. Self-Supervised Visual Attribute Learning for Fashion Compatibility. In *ICCV*, 2021. 2, 6, 7, 12

[19] L. H. Li, M. Yatskar, D. Yin, C. Hsieh, and K. Chang. Visualbert: A simple and performant baseline for vision and language. In *ArXiv*, 2019. 3, 5, 6, 12, 13, 15

[20] T. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014. 5

[21] Y. Lin, S. Tran, and L. S Davis. Fashion outfit complementary item retrieval. In *CVPR*, 2020. 2, 6, 7, 12

[22] L. Liu, W. L. Hamilton, G. Long, J. Jiang, and H. Larochelle. A universal representation transformer layer for few-shot image classification. In *ICLR*, 2021. 2

[23] Z. Liu, P. Luo, S. Qiu, X. Wang, and X. Tang. Deepfashion: Powering robust clothes recognition and retrieval with rich annotations. In *CVPR*, 2016. 5

[24] A. Y. Malkov and D. A. Yashunin. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. *CoRR*, 2016. 4, 11

[25] T. K. R. Medini, Q. Huang, Y. Wang, V. Mohan, and A. Shrivastava. Extreme classification in log memory using count-min sketch: A case study of amazon search with 50m products. In *Neurips*, 2019. 5, 6, 13

[26] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *NIPS*, Dec. 2013. 4

[27] A. Mittal, N. Sachdeva, S. Agrawal, S. Agarwal, P. Kar, and M. Varma. Eclare: Extreme classification with label graph correlations. In *WWW*, April 2021. 2, 3, 5, 6, 11, 13

[28] A. Nagrani, S. Yang, A. Arnab, A. Jansen, C. Schmid, and C. Sun. Attention Bottlenecks for Multimodal Fusion, 2021. arXiv. 3

[29] J. Ni, J. Li, and J. McAuley. Justifying recommendations using distantly-labeled reviews and fine-grained aspects. In *EMNLP-IJCNLP*, 2019. 5, 11, 12

[30] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, et al. Learning transferable visual models from natural language supervision. In *Arxiv*, 2021. 2, 3, 5, 6, 12, 13

[31] N. Reimers and I. Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In *EMNLP*, 2019. 3, 4, 6, 14

[32] A. Revanur, V. Kumar, and D. Sharma. Semi-Supervised Visual Representation Learning for Fashion Compatibility. In *Recommender Systems*, 2021. 2, 6, 7, 12

[33] D. Saini, A.K. Jain, K. Dave, J. Jiao, A. Singh, R. Zhang, and M. Varma. GalaXC: Graph Neural Networks with Labelwise Attention for Extreme Classification. In *WWW*, 2021. 2

[34] R. Tan, M. I. Vasileva, K. Saenko, and B. A. Plummer. Learning Similarity Conditions Without Explicit Supervision. In *ICCV*, 2019. 2, 6, 7, 12

[35] M. I. Vasileva, B. A. Plummer, K. Dusad, S. Rajpal, R. Kumar, and D. Forsyth. Learning Type-Aware Embeddings for Fashion Compatibility. In *ECCV*, 2018. 2, 5, 6, 7, 12

[36] L. Wu, A. Fisch, S. Chopra, K. Adams, A. Bordes, and J. Weston. Starspace: Embed all the things! *CoRR*, 2017. 2

[37] M. Wydmuch, K. Jasinska, M. Kuznetsov, R. Busa-Fekete, and K. Dembczynski. A no-regret generalization of hierarchical softmax to extreme multi-label classification. In *NIPS*, 2018. 5, 6, 13

[38] L. Xiong, C. Xiong, Y. Li, K. Tang, J. Liu, P. Bennett, J. Ahmed, and A. Overwijk. Approximate Nearest Neighbor Negative Contrastive Learning for Dense Text Retrieval. In *ICLR*, 2021. 4

[39] R. You, S. Dai, Z. Zhang, H. Mamitsuka, and S. Zhu. AttentionXML: Extreme Multi-Label Text Classification with Multi-Label Attention Based Recurrent Neural Networks. In *NeurIPS*, 2019. 2, 5, 6, 13

[40] J. Zhao, Y. Zhao, and J. Li. M3TR: Multi-modal Multi-label Recognition with Transformer. In *ACM MM*, 2021. 3

[41] X. Zhu, H. Liu, Z. Lei, H. Shi, F. Yang, D. Yi, G. Qi, and S. Z. Li. Large-Scale Bisample Learning on ID Versus Spot Face Recognition. *IJCV*, 2019. 2

Code and Data Release

Due to large size of the datafiles, the MM-AmazonTitles-300K dataset and the MUFIN model trained on this dataset has been released at the following URL

http://manikvarma.org/pubs/mittal22-supp.pdf

The MUFIN code has been submitted alongside this paper as supplementary material on the submission website. However, it is available on the above mentioned URL as well.

Discussion on Ethical Considerations, Limitations and Future Work

Ethical Considerations: The MM-AmazonTitles-300K dataset was curated from a publicly available crawl [29] and utilizes no personally identifiable or human subject data. The MUFIN approach is itself applied to tasks such as product-to-product recommendation, bid query prediction, and outfit completion that seek to improve user experience when browsing for products. We are unaware of any direct applications of the MUFIN approach that can have negative societal impact.

Limitations and Future Work: We identify two potential avenues for further improving MUFIN's performance. Firstly, Fig. 4 indicates that the MUFIN-($\alpha = 1$) is better at predicting tail/rare labels correctly whereas MUFIN performs better on head/popular labels. Although MUFIN outperforms MUFIN-($\alpha = 1$) overall (see Tab. 4), this indicates towards a possibility for a third variant that scores rare and popular labels differently to get the best of both worlds. Secondly, it is common for products on e-commerce and other portals to be endowed with taxonomies that can give direct information about related products. The use of such tree/graph metadata has been shown in XC literature [27] to positively impact performance. Incorporating such relational metadata at the scale of millions of products is an interesting direction.

A. MUFIN Pseudocode and Outline

Fig. 7 below presents MUFIN's prediction pipeline. The ViT+SentenceBERT encoder $\mathcal{E}$ trained in a Siamese fashion embeds a test point X as a vector $\hat{\boldsymbol{x}}^1$ which is used to shortlist $O(\log L)$ labels using augmented retrieval. For each shortlisted label e.g. label number 42 in the example, a label-adapted representation $\hat{\boldsymbol{x}}^{2,42}$ is generated by applying cross-attention between test point representation $\hat{\mathbf{X}}^1$ and label representation $\hat{\mathbf{Z}}^1_{42}$. The dot product between $\hat{\boldsymbol{x}}^{2,42}$ and the classifier for this label $\boldsymbol{w}_{42}$ gives final score of label 42 for this test point.

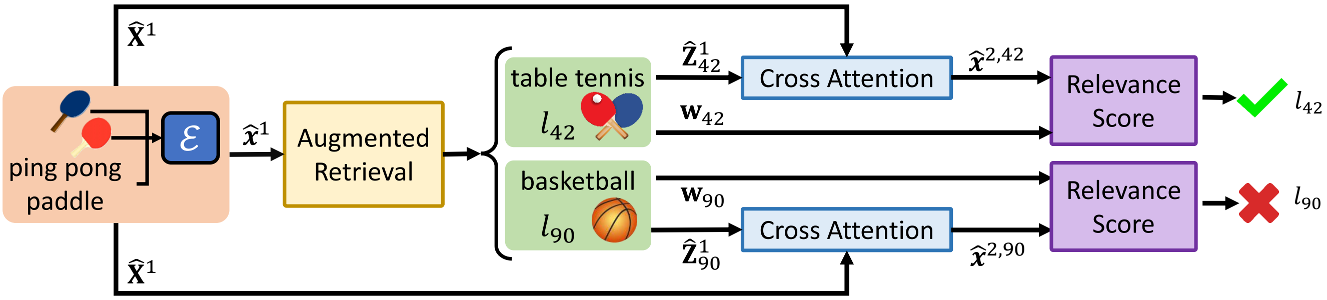


Figure 7. Scalable prediction pipeline deployed by MUFIN

Prediction Time Complexity for MUFIN. For sake of simplicity, let us assume that all labels and datapoints contain the same number m of descriptors. Computing the bag of pre-embeddings $\hat{\mathbf{X}}^0_t$ takes $\mathcal{O}\left(m \cdot (\text{ENC} + D)\right)$ time assuming the encoders $\mathcal{E}_V, \mathcal{E}_T$ take ENC time to encode a descriptor into a D-dimensional vector. Passing $\hat{\mathbf{X}}^0_t$ through the self-attention block $\mathcal{A}_S$ to obtain $\hat{\mathbf{X}}^1_t$ takes $\mathcal{O}\left(mD^2 + m^2 D\right)$ time after which computing the vector representation $\hat{\boldsymbol{x}}^1_t$ takes an additional $\mathcal{O}\left(mD\right)$ time. Querying the ANNS structure NN^x to obtain the shortlist R_t of $\mathcal{O}\left(\log L\right)$ labels takes at most $\mathcal{O}\left(D \log L\right)$ time [24]. Applying the cross-attention block $\mathcal{A}_C$ with respect to each of these shortlisted labels takes a total of $\mathcal{O}\left((mD^2 + m^2 D) \log L\right)$ time. Vectorizing these outputs to obtain $\hat{\boldsymbol{x}}^{2,l}_t$ and applying the label classifiers $\boldsymbol{w}_l$ takes at most $\mathcal{O}\left(mD \log L\right)$ time. This brings the total time complexity of the prediction pipeline per test datapoint to be at most $\mathcal{O}\left(m \cdot \text{ENC} + (m + D) \cdot mD \log L\right)$. In practice, MUFIN offered predictions within a 3-4 milliseconds even on tasks such as A2Q-4M with several millions of labels.

B. Datasets

(Discussion continued from the **Datasets** subsection in Sec. 4)

MM-AmazonTitles-300K.: This dataset was curated from an existing Amazon click dump [29]. Given a product as a datapoint, the aim is to recommend the subset of the most relevant (e.g. frequently bought-together) products from a catalog of over 300K unique products. Each product is represented using multiple descriptors including a product title and up to 15 product images. The dataset released in the supplementary [link] consists of multiple product entries in JSON format. Each product is associated with multiple tags described below:

1. "ASIN": this acts as a unique identifier (UID) for the product
2. "title": this represents a textual title for the product
3. "images": this presents a list of URLs pointing to multiple (up to 15) images of the product
4. "also_buy": this presents a list of UIDs of products that were frequently bought together with the product

Products having no images as well as no title were not included in the dataset. To generate the train test split, guidelines from [2] were closely followed.

C. Baselines and Related work

(Discussion continued from the **Baselines** subsection in Sec. 4)

Baselines for the Polyvore-Disjoint dataset: Performance numbers for all baselines in Tab. 3 were taken directly from published results [10, 18, 21, 32]. For sake of completion, each baseline method for this dataset is described in below in brief.

- **ADDE-O [10]**: Learns attribute-driven disentangled representations.
- **CSA-NET [21]**: Learns a category-based subspace attention network for scalable indexing and retrieval. This work introduced the notion of *outfit ranking loss* that considers the item-relationship of an entire outfit.
- **Type-aware [35]**: Learns an image embedding that respects item type, and jointly learns notions of item similarity and compatibility in an end-to-end manner. This method uses both visual and textual descriptors to represent an outfit.
- **S-VAL [18]**: Learns outfit representation by self-supervision. The self-supervision tasks used in the paper were histogram prediction and learning to discriminate shapeless patches and textures from different images.
- **SCE-Net [34]**: Learns model parameters by optimizing a combination of loss functions. For Polyvore-Disjoint, SCE-Net uses two objective loss function, namely VSE and Sim. The VSE loss requires that image embeddings and textual embeddings of the same outfit be embedded together. The Sim loss encourages images and descriptions of similar products to be embedded close to each other.
- **SSVR-Net [32]**: Learns model parameters using semi-supervised learning. This work uses a Siamese architecture trained using triplet loss that takes an input image, a positive instance (an affine transformation of the image) and a negative instance (an image with color transformations such as random gray scaling, jittering *etc*.).

Baselines for the MM-AmazonTitles-300K dataset: Baselines for this dataset are divided in two sets.

1. **Textual Methods**: Methods in the first set correspond to state-of-the-art extreme classification techniques including, AttentionXML, SiameseXML, ECLARE, Bonsai, MACH and XT. These methods are designed to be text-based and as such use only textual descriptors of a product. Consequently, each product was represented for these methods using its product title alone. Hyperparameters for each method were used as suggested by the respective papers. With the exception of the MACH method, all these methods enjoy an $\mathcal{O}(\log L)$ prediction time complexity similar to MUFIN.
2. **Visual + Textual Methods**: These methods include CLIP [30] and VisualBert [19] that use both visual and textual descriptors of a product.

As before, each baseline method for this dataset is described in below in brief. The details of how CLIP and VisualBERT were augmented and fine-tuned are discussed thereafter.

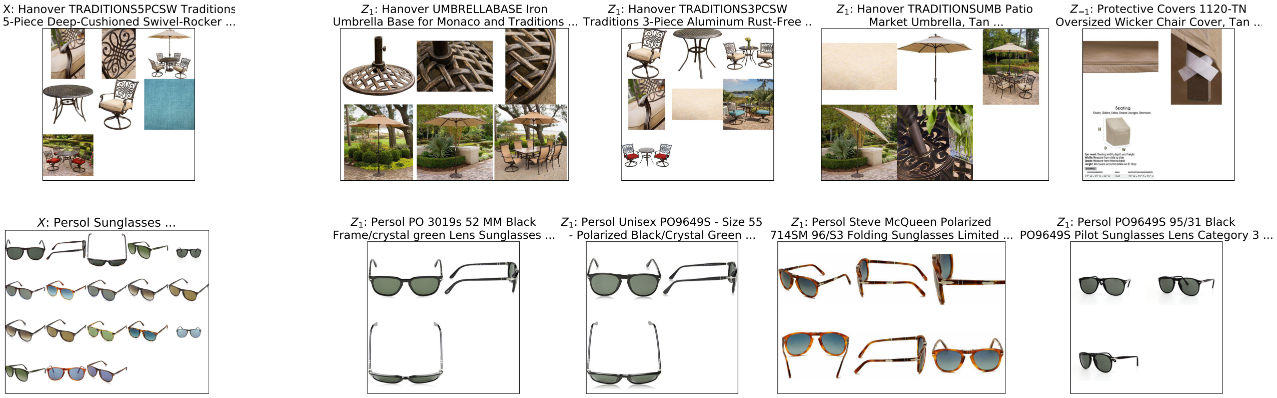
- **SiameseXML [5]**: Melds Siamese networks with one-vs-all classifiers. SiameseXML retrieves label shortlists using multiple ANNS structures unlike MUFIN that uses a single ANNS structure. The shortlisted labels are then ranked according to scores obtained from label-wise one-vs-all classifiers. Add SiameseXML implementation details
- **ECLARE [27]**: Exploits label graphs to obtain superior label representations with an aim to improve performance on rare labels. Since the MM-AmazonTitles-300K dataset does not provide a label graph natively, a label graph was mined by performing random walks using the label vectors as suggested in [27].
- **AttentionXML [39]**: Learns to partition labels using a shallow and wide PLT (depth between 2-3). A context vector is learnt per label that is used to generate label-specific datapoint representations. We note that the cross-attention block $\mathcal{A}_C$ used by MUFIN similarly produces label-adapted datapoint representations.
- **Bonsai [16]**: Implements a scalable tree-based classifier by learning a label hierarchy over the labels by representing each label using its bag-of-words (BoW) centroid vectors.
- **MACH [25]**: Learns an ensemble of 32 learners where each learner randomly partitions labels into several hash bins. Models are learnt to predict the hash bits corresponding to each learner. At prediction time, a majority vote is taken over all the learners to boost confidence. The method offers a prediction time of $\mathcal{O}(L)$.
- **XT [37]**: Generalizes the hierarchical softmax approach popular for multi-class problems to multi-label problems using a probabilistic label tree. Recall from the discussion in Sec. 1 that in multi-class classification, the objective is to predict a single mutually exclusive label for a given datapoint whereas in multi-label classification, the goal is to annotate datapoints with the most relevant *subset* (one or more) of labels.
- **VisualBert [19]**: Uses pre-trained Resnet-101 embeddings to represent images. Given a data point with multiple visual and textual descriptors, the visual descriptors are encoded and fed as tokens into a BERT (large) architecture alongside textual tokens from the textual descriptors. In this sense, VisualBert can be seen as utilizing early fusion.
- **CLIP [30]**: Uses a ViT architecture to encode images and a BERT architecture to encode text. The method pre-trains the architectures to encode relevant image-text pairs together. Subsequently, a late fusion architecture is learnt that fuses an image embedding and a text embedding into a joint representation over which a scoring model can be learnt.

C.1. Adapting CLIP and VisualBERT to MM-AmazonTitles-300K and Fine-tuning Details

Datapoint/label embedding Architecture: As mentioned above, CLIP uses separate encoders to encode images and text that can offer a bag of embeddings $\mathcal{E}_{\text{CP}}(X_i) \in \mathbb{R}^{m_i \times D}$. This bag of embeddings was fed into a fresh instantiation of the self-attention block $\mathcal{A}_S$ to obtain datapoint/label embeddings that we name $\hat{\boldsymbol{x}}^{\text{CP}}$ and $\hat{\boldsymbol{z}}_l^{\text{CP}}$. These are analogous to the $\hat{\boldsymbol{x}}^1$ and $\hat{\boldsymbol{z}}_l^1$ representations used by MUFIN. On the other hand, VisualBert uses a BERT architecture and pre-trained Resnet101 embeddings for images as tokens along with textual tokens to itself encode each datapoint/label as a vector to give embeddings $\hat{\boldsymbol{x}}^{\text{VB}}$ and $\hat{\boldsymbol{z}}_l^{\text{VB}}$. Note that VisualBERT was not offered the self-attention block since it itself performs similar self-attention operations within the layers of its BERT (large) architecture.
Retrieval: For VisualBERT, retrieval was performed using an ANNS structure created over the label embeddings $\hat{\boldsymbol{z}}_l^{\text{VB}}$. For CLIP, since it offers bag embeddings $\mathcal{E}_{\text{CP}}(Z_L) \in \mathbb{R}^{m_l \times D}$ one per descriptor of the label, augmented retrieval was implemented similar to MUFIN.
Scoring Architecture: Both CLIP and VisualBERT were offered independent instantiations of the cross-attention block $\mathcal{A}_C$. For CLIP, it was applied to the bag embeddings $\mathcal{E}_{\text{CP}}(X_i)$ and $\mathcal{E}_{\text{CP}}(Z_l)$. For VisualBERT that does not offer bag embeddings, cross-attention was applied over $\hat{\boldsymbol{x}}_i^{\text{VB}}$ and $\hat{\boldsymbol{z}}_l^{\text{VB}}$ instead. Both methods were offered one-vs-all classifiers $\boldsymbol{w}_l, l \in [L]$.
Training and Fine-Tuning Details: In the first experiment, the encoding architectures of CLIP and VisualBERT were frozen and only the self-/cross-attention architectures were trained in a four module manner identical to MUFIN. The results of this experiment correspond to the rows titled "VisualBert" and "CLIP" in Tab. 2. In the next experiment, the encoders used within CLIP and VisualBERT were additionally trained in Modules I-IV. The results of this experiment correspond to the rows titled "VisualBert-fine-tuned" and "CLIP-fine-tuned" in Tab. 2. Without fine-tuning the encoders, both architectures offered poor performance 30-50% worse than MUFIN in terms of P@1. Fine-tuning significantly improved the performance for both methods but they remained 5-13% worse than MUFIN in terms of P@1.

Table 5. Hyper-parameters used to train MUFIN on the public datasets. MUFIN uses the AdamW optimizer and learning rate scheduler with a warm start of 1000 iterations.

Dataset	Module I			Module IV		
	Epochs	Learning Rate (lr)	Batch Size (B)	Epochs	Learning Rate (lr)	Batch Size (B)
MM-AmazonTitles-300K	200	2e-4	1024	20	5e-5	200
Polyvore-Disjoint	200	2e-4	1024	10	5e-5	200


X: Hanover TRADITIONS5PCSW Traditions 5-Piece Deep-Cushioned Swivel-Rocker ...
Z_1: Hanover UMBRELLABASE Iron Umbrella Base for Monaco and Traditions ...
Z_1: Hanover TRADITIONS3PCSW Traditions 3-Piece Aluminum Rust-Free ...
Z_1: Hanover TRADITIONSUMB Patio Market Umbrella, Tan ...
Z_{-1}: Protective Covers 1120-TN Oversized Wicker Chair Cover, Tan ...
X: Persol Sunglasses ...
Z_1: Persol PO 3019s 52 MM Black Frame/crystal green Lens Sunglasses ...
Z_1: Persol Unisex PO9649S - Size 55 - Polarized Black/Crystal Green ...
Z_1: Persol Steve McQueen Polarized 714SM 96/S3 Folding Sunglasses Limited ...
Z_1: Persol PO9649S 95/31 Black PO9649S Pilot Sunglasses Lens Category 3 ...

(a) Query Datapoint

(b) Retrieved results

Figure 8. Predictions by MUFIN for sample test points in the MM-AmazonTitles-300K dataset. $\mathbf{Z}_1$ (resp. $\mathbf{Z}_{-1}$) in the title indicates that the retrieved product (label) was relevant (resp. irrelevant) for the query datapoint in the ground truth. Not only does MUFIN mostly recommend products relevant according to the ground truth, but the occasional recommendation not in the ground truth list is often a good recommendation nevertheless. For instance, in the first row, the last recommendation does not appear in the ground truth but is a chair cover relevant to the query. (Figure best viewed under magnification)

D. Hyperparameters

(Discussion continued from the **Hyperparameters** subsection in Sec. 4)

MUFIN uses a ViT-32 [8] architecture as the image encoder $\mathcal{E}_V$ with 16×16 patches and a SentenceBert [31] architecture as the text encoder $\mathcal{E}_T$. The AdamW optimizer with a one-cycle cosine scheduler with warm start of 1000 iterations was used. MUFIN was trained on a 24-core Intel Skylake 2.4 GHz machine with 4 V100 GPUs for 48 hrs on the A2Q-4M dataset. To reiterate the main training parameters mentioned in Sec. 3, model parameters were learnt in Module I using the contrastive loss with a margin of $\gamma = 0.2$. In Module IV, these encoders were fine-tuned using the cosine embedding loss with a margin of $\gamma = 0.5$ with $|\mathcal{S}_i| = 2$ randomly sampled positive labels and $|\mathcal{T}_i| = 12$ hard negative labels taken from the shortlist R_i obtained from the augmented retrieval step. For details of other hyper-parameters please refer to Tab. 5.

E. Additional Experimental Results

Fig. 8 presents predictions by MUFIN for a few sample test points in the MM-AmazonTitles-300K dataset. Not only does MUFIN mostly recommend products relevant according to the ground truth, but the occasional recommendation not in the ground truth list is often a good recommendation nevertheless.

Tab. 6 presents MUFIN's performance on various categories of the MM-AmaonTitles-300K dataset of which a snapshot was provided in Fig. 3. For most categories, MUFIN is indeed the best method and MUFIN ($\alpha = 1$) is the second-best method. In particular, MUFIN could give accuracy gains up to 6% on various categories (e.g. Musical Instruments).

F. Ablation

(Discussion continued from **Ablation** Sec. 4.2)

MUFIN makes several design choices with respect to key components in its architecture and training pipeline. its key

Table 6. MUFIN's performance on various categories of the MM-AmaonTitles-300K dataset of which a snapshot was provided in Fig. 3. For each category, the best performance is highlighted in bold black font, the second-best performance is left in normal black font and the third-/fourth- performances are stylized in light gray. Note that for most categories, MUFIN is indeed the best method and MUFIN $(\alpha = 1)$ is the second-best method. In particular, MUFIN could give accuracy gains up to 6% on various categories (e.g. Musical Instruments).

	# Labels	MUFIN / MUFIN $(\alpha = 1)$	SiameseXML [5]	VisualBert [19]
Overall	303,296	**34.76** / 33.19	32.99	32.24
AMAZON FASHION	681	**24.71** / 23.85	23.90	22.98
All Beauty	329	**19.70** / 19.04	18.98	18.80
Appliances	767	**27.51** / 27.17	24.60	25.69
Arts Crafts and Sewing	16,843	**41.62** / 39.18	39.06	38.25
Automotive	18,384	**14.39** / 14.12	12.84	13.40
Cell Phones and Accessories	6,410	**33.32** / 32.81	32.52	32.06
Clothing Shoes and Jewelry	25,379	**29.18** / 28.03	28.75	26.80
Electronics	22,449	**32.65** / 31.07	29.92	29.96
Grocery and Gourmet Food	24,676	**47.41** / 44.72	44.94	44.18
Home and Kitchen	37,111	**35.75** / 34.63	33.88	34.19
Industrial and Scientific	7,333	**36.55** / 34.01	33.77	33.19
Luxury Beauty	3,455	60.74 / 58.98	**61.56**	60.72
Musical Instruments	3,835	**26.18** / 24.49	20.82	23.74
Office Products	16,763	**38.09** / 36.33	36.13	35.17
Patio Lawn and Garden	11,631	**36.33** / 35.03	32.84	34.25
Pet Supplies	12,102	**41.76** / 40.03	37.54	38.47
Prime Pantry	3,328	37.45 / 38.78	**40.21**	38.61
Sports and Outdoors	24,842	**34.24** / 31.99	32.30	30.62
Tools and Home Improvement	23,437	**37.03** / 35.03	34.87	34.11
Toys and Games	43,541	**47.92** / 46.13	45.73	45.55

component- sampling, retrieval, representation and ranker. These ablation experiments attempt to ascertain the differential contribution of each of these design choices to MUFIN's final performance. The results of the ablation experiments are detailed in Tab. 4.

Sampling: Recall that in Module I, MUFIN uses hard negative as well as hard positive sampling to focus training on datapoint-label pairs that offer the most prominent gradients. In the MUFIN-no +ve variant, hard positive sampling was replaced with random positives. In the MUFIN-no +ve, -ve variant, both hard positive and hard negative sampling was replaced with random positive and negatives. As Tab. 4 indicates, "MUFIN-no +ve" and "MUFIN-no +ve, -ve" lead to the loss of 2% and 2.5% in P@1 and 1% and 1.5% in P@5. This points to the need for effective training using hard negatives and positives.

Retrieval: Recall that to perform augmented retrieval, MUFIN represents each label using its corresponding bag-of-embeddings of product images as well as text and creating ANNS structures over an expanded set of $\sum_{l \in [L]} m_l$ vectors, the label l getting represented using m_l vectors, one per descriptor. We refer to this default variant as MUFIN-P-I-bag. The alternate variant MUFIN-P-I-vec represents each label using a single embedding namely $\boldsymbol{z}_l^1$. Results shows that MUFIN-P-I-bag could be 1% and 1.6% more accurate in P@1 and P@5 indicating the benefits of augmented retrieval.

Note about the Retrieval ablation experiment: The performance numbers for MUFIN-P-I-bag and MUFIN-P-I-vec given in Tab. 4 are those of the MUFIN model learnt after Module I i.e. sans the cross-attention layer and one-vs-all classifiers. This is why the absolute numbers for MUFIN-P-I-bag (e.g. 42.72 P@1) are lower than MUFIN (52.3 P@1). The difference can be attributed to the inclusion of the cross-attention block and one-vs-all classifiers.

Representation: MUFIN adapts the representation of a datapoint to a label using its cross attention block $\mathcal{A}_C$. In the MUFIN-ConCat variant, the cross attention block was replaced with a simpler architecture that concatenates the datapoint $(\boldsymbol{x}^1)$ and label $(\boldsymbol{z}_l^1)$ representations and applies two feedforward layers of the form $2D \rightarrow 2D \rightarrow D$ to obtain an alternate label-adapted representation analogous to $\hat{\boldsymbol{x}}^{2,l}$. Results show that MUFIN could be 2.69% and 1.87% more accurate than

MUFIN-ConCat in terms of P@1 and P@5. The MUFIN-no $\mathcal{A}_S$ variant replaces the self-attention block $\mathcal{A}_S$ with a simple feed-forward layer (in Modules I-IV) and observed a loss of 2.3% and 1.6% in terms of P@1 and P@5. The ablations indicate the effectiveness of MUFIN's self- and cross-attention compared to alternatives.

Ranker: MUFIN uses a cross-attention layer and one-vs-all classifiers to perform datapoint-label scoring. In the MUFIN-no $\mathcal{A}_C$ variant, the cross-attention block is completely bypassed, effectively yielding $\hat{\boldsymbol{x}}^{2,l} = \hat{\boldsymbol{x}}^1$ on top of which one-vs-all classifiers are then applied. In the MUFIN-(MUFIN =($\alpha = 1$)) on the other hand, MUFIN completely ignores the one-vs-all classifiers by explicitly setting $\alpha = 1$. Experiments show that MUFIN scoring mechanism employing both a cross-attention block and one-vs-all classifiers can be 2-3% more accurate in P@1 than the MUFIN-no$\mathcal{A}_C$ and MUFIN-($\alpha = 1$) variants.