Gandalf : Data Augmentation is all you need for Extreme Classification

Anonymous ACL submission

Abstract

Extreme Multi-label Text Classification (XMC) involves learning a classifier that can assign an input with a subset of most relevant labels from millions of label choices. Recent works in this domain are increasingly focusing on the problem setting with (i) short-text input data, and (ii) labels endowed with meta-data in the form of textual descriptions. Short-text XMC with label features has found numerous applications in areas such as prediction of Related Searches, product recommendation based on titles, and bid-phrase suggestion, amongst others.

In this work, by exploiting the problem characteristics of short-text XMC, we develop postulates stating the desired invariances, and propose two data augmentation techniques to achieve them. One, *LabelMix*, which performs data augmentation by concatenating an annotating label to the data-point; and the other, *Gandalf*, which generates additional data-points by considering labels as legitimate data-points. The efficacy of the proposed augmentation methods is demonstrated by showing upto 30% relative improvement when applied to a range of existing algorithms, and proposing an algorithmic framework, *InceptionXML-LF*, which furthers state-of-the-art on benchmark datasets.

1 Introduction

Related Searches, product recommendation and bid-phrase matching tasks require predicting the most relevant results that are either highly correlated or frequently co-occur with the given input query/product. *Extreme Multilabel Classification* (XMC) has found multiple applications in these domains where the problem is modelled as a short-text classification task over millions of possible searches/products/ad-phrases considered as labels. Real world data from these domains, when modelled as an XMC problem, is highly imbalanced towards some popular or trending ad-phrases/products and notoriously exhibits fit to Zipf's law. Here, most labels in the extremely large output space, are tail labels i.e, those which have very few (≤ 5) instances in the training set (Babbar and Schölkopf, 2019). Similarly, the words in queries also follow a long-tailed distribution. While there exists insufficient training data for these tail labels/words, the short-text nature of these queries makes it no simpler for the models to learn meaningful, non-overfitting embeddings and encoded representations for tail words and labels.

Due to the increasing requirement of scalable and low latency models in these domains, there has been a surge in works that model recommendations tasks like related searches, query-to-product and document-to-document recommendation as a short-text XMC problem using only the search query, product name or document title. Hence, most of these works are focused on building lightweight and frugal architectures that can predict in milliseconds and scale up to millions of labels. Despite being frugal in terms of number of layers/parameters in the network, these models can learn the training data well enough. Hence, creating deeper models for better representation learning is perhaps not the most optimal solution under this setting.

Many of the recent works, thus, make architectural improvements to leverage "label features" in order to imbue strong inductive biases in their models. A label feature is the text associated with labels, which spans the same vocabulary universe as query text. Label features, when encoded in the same embedding space as query texts, help mitigate the difficulty in learning efficient representations for tail labels by enabling joint query-label representation learning in common embedding space (Dahiya et al., 2021a; Mittal et al., 2021a,b); thereby improving the prediction performance. However, even after improved representation learning by leveraging label features, a significant generalization gap is noticeable in these approaches (Figure 4, Appendix B).

Contributions In this work, we take a data-centric approach, and aim at answering *"Can we extend mixup to feature-label extrapolation to guarantee a robust model behavior far away from the training data?"*, a question posed in Zhang et al. (2018). To this end, we (i) propose *LabelMix* augmentation and motivate it through the *Vicinal Risk Minimization* (VRM) (Chapelle et al., 2000) principle, which is achieved by explicating the desired invariance properties and leveraging them for augmentation purposes. To the best of our knowledge, this is the first work that attempts to further the application of vicinal risk minimization to an embedding space where data instances and their label features co-exist in a shared embedding space.

We further (ii) discuss self and soft-annotation properties of label features and propose *Gandalf* - **G**r**A**ph i**N**duced **D**ata **A**ugmentation based on **L**abel **F**eatures - to efficiently leverage label features as valid training instances. (iii) As an algorithmic contribution, we propose an extension to INCEPTIONXML (Kharbanda et al., 2021), to accommodate label features, as an efficient alternative framework to current short-text XMC pipelines (See Appendix A). (iv) We demonstrate the generality and effectiveness of the proposed data augmentations, by showing upto 30% relative improvements in multiple state-of-the-art extreme classifiers on public benchmark datasets. Our experiments reflect that strong inductive biases that are currently imbued into models through complicated training pipelines and architectural modifications can also be induced with simple data augmentation techniques as proposed in the paper.

2 Related Work

Earlier works in XMC have focused on the problem of tagging long text documents consisting of hundreds of tokens. These are broadly categorized based on their algorithmic characteristics as follows: (i) Label-tree methods (Jasinska et al., 2016; Prabhu et al., 2018; Khandagale et al., 2020), (ii) Decision tree-based methods (Prabhu and Varma, 2014; Choromanska and Langford, 2015; Agrawal et al., 2013) (iii) Label-embedding methods (Bhatia et al., 2015; Yu et al., 2014; Tagami, 2017), (iv) One-vs-rest methods (Babbar and Schölkopf, 2017; Yen et al., 2017), and (v) Deep learning methods (Liu et al., 2017; You et al., 2019). Of late, works aimed towards scaling up transformer encoders for XMC have dominated the research landscape in this domain (Chang et al., 2020; Ye et al., 2020; Zhang et al., 2021).

XMC with Label Features: More recent works have shifted their focus to short-text XMC to keep up with the increasing requirements of low-latency models in recommendation tasks. These works can be split into two categories: (i) those which inherently do not leverage label features like ASTEC (Dahiya et al., 2021b) and INCEPTIONXML (Kharbanda et al., 2021), and (ii) those which do heavy architectural modifications or employ complicated training strategies in order to leverage label features along with short-text instances to induce strong inductive bias into their models. For example, SIAMESEXML (Dahiya et al., 2021a) employs a siamese constrastive learning stage between instance and its label features through a modified negative log-likelihood loss and, DECAF (Mittal et al., 2021a) and ECLARE (Mittal et al., 2021b) extend the DEEPXML (Dahiya et al., 2021b) pipeline by augmenting the ASTEC encoder with one or two extra ASTEC-like encoders for label-text and graph-augmented label text and as a result end up taking ~3x the time as compared to ASTEC.

Data Augmentation Apart from the architectural design choices, data augmentation methodologies have been successful in providing much needed inductive biases leading to better performance of machine learning models on unseen data. While these have been highly popular for computer vision tasks inspired by recent works (Zhang et al., 2018; Verma et al., 2019; Yun et al., 2019), augmentation techniques remain relatively under-explored in Natural Language Processing. However, recent works have shown their efficacy in NLP tasks such as machine translation (Gao et al., 2019), common-sense reasoning (Yang et al., 2020), semantic parsing (Guo et al., 2020), text classification (Zhao et al., 2022; Wei and Zou, 2019), and for achieving adversarial robustness (Li et al., 2017). Further details on tasks specific techniques for data augmentation in NLP can be found in Feng et al. (2021). For (short-text) XMC, which is the focus of this paper, there have been no works which have leveraged data augmentation or studied its implications.

3 Background & Notation

For training, we have available a dataset $\mathcal{D} = \{\{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}^{N}, \{\mathbf{z}_l\}_{l=1}^{L}\}$ of N pairs of input data-points $\mathbf{x}_i$, their corresponding labels $\mathbf{y}_i$ and a label

Datasets	N	L	APpL	ALpP	AWpP
LF-AmazonTitles-131K	294,805	131,073	5.15	2.29	6.92
LF-WikiSeeAlsoTitles-320K	693,082	312,330	4.67	2.11	3.01
LF-WikiTitles-500K	1,813,391	501,070	17.15	4.74	3.10

Table 1: Characteristics of short-text benchmark datasets with label features. Here, APpL stands for avg. points per label, ALpP stands for avg. labels per point and AWpP is length i.e. avg. words per point.

feature $\mathbf{z}_l$ associated with each label. In short-text setting, typically the data-points (in the form of queries or titles) and label features both comprise of very few words on average (Table : 1). Further, $\mathbf{x}_i, \mathbf{z}_l \in \mathcal{X}$ for some input space $\mathcal{X} = \bigoplus_{m=1}^{\infty} \mathcal{V}^m$, where $\mathcal{V}$ denotes a common vocabulary universe, and $\bigoplus$ represents the operation for aggregating $\mathcal{V}^m$ i.e. text sequences of length m, into a set.

When posed a short-text XMC problem with L labels, we are interested in finding a learning function $f := \{\Phi, \Psi\}$ which maps $\mathbf{x}_i$ to a subset of labels $\mathrm{y} \subset [L]$ out of the L available labels identified through the integers $[L] \coloneqq \{1, \dots, L\}$. Usually, we identify the labels through a binary vector $\mathbf{y} \in \{0,1\}^L$, where $\mathbf{y}_l = 1 \Leftrightarrow l \in \mathrm{y}$. A common strategy for implementing the classifier component Ψ in the XMC pipelines is via the one-vs-all (OVA) scheme, such that for each $\mathbf{x}_i$, a score $s_l(\mathbf{x}_i)$ is calculated for each label $l \in [L]$. In practice, the sum over all labels is very expensive, and is therefore often approximated e.g. by a label shortlisting procedure (Jain et al., 2019; Jiang et al., 2021).

In the OVA paradigm, scores are typically calculated by projecting both the instances and the labels to some common Euclidean space $\mathcal{E} = \mathbb{R}^d$, and then taking their inner product. The mapping of instances to embeddings is realized through a feature extractor $\Phi : \mathcal{X} \longrightarrow \mathcal{E}$. All labels are decoded through $\Psi \stackrel{\text{def}}{=} \{\mathbf{w}_l\}_{l=1}^{L}$, where $\mathbf{w}_l \in \mathcal{E}$ is the label's *decoding representation*. In this notation, we have $s_l(\mathbf{x}) = \langle \Phi(\mathbf{x}), \Psi(l) \rangle$.

The short-text XMC problem is characterized by two random variables $X \in \mathcal{X}$ and $Y \in \{0,1\}^L$ jointly distributed according to $\mathbb{P}$. Here, the labels are sparse, $\mathbb{E}[\|Y\|_1] = \bar{y} \ll L$, and follow a long-tailed distribution. Also, the instances are short-text i.e. $\mathbb{E}[\mathrm{len}(X)] = \bar{m}$, with $\bar{m}$ in the range of about 3 to 8 tokens[1] as shown in Table 1.

[1]We do not place a strict upper-bound on the number of tokens, because this complicates the concatenation arguments

4 Invariances in Short-text XMC & Vicinal Risk Minimization

The extreme scarcity of training data for tail labels in XMC implies that a good classifier for these labels can only be learned if, in addition to the few training examples, strong inductive biases are employed during training. Even though there exist problem-agnostic regularizations such as limiting the magnitude of the parameters (via L_1 or L_2 regularization), implicit regularization through SGD dynamics, or dropout, it is beneficial to use domain knowledge for more efficient inductive biases.

As discussed in section 2, many recent XMC baselines leverage label features in order to imbue strong inductive biases in their models either through computationally expensive architectural additions or complicated training procedures. If similar inductive biases could be achieved through data augmentation, then these would not be restricted to a single architecture, but benefit most current and future short-text XMC methods. Thus, our goal is to identify underlying properties of the short-text XMC problem, and use these to derive new data augmentation techniques.

Data augmentation can be seen as a form of *vicinal risk minimization*, the idea that one minimizes risk over the empirical distribution

$$\mathrm{d}\mathbb{P}_{\mathcal{D}}(\mathbf{x}, \mathbf{y}) = \frac{1}{n} \sum_{i=1}^{n} \delta_{\mathbf{x}_i}(\mathbf{x})\, \delta_{\mathbf{y}_i}(\mathbf{y}), \qquad (1)$$

but instead over a smoothed out version $\mathbb{P}_{\mathrm{v}}$. That means that each data point $\mathbf{x}$ in the input, corresponding to a peak $\delta_{\mathbf{x}}$ in the empirical distribution, instead is turned into a smooth distribution that has nonzero density in the vicinity of $\mathbf{x}$. The key task is then to determine what constitutes the vicinity of a data point in this setting.

Symmetries and transformation laws have long been a fruitful source for inductive biases in machine learning. For example, in computer vision the underlying symmetries are, for example, translation, rotation, and flips. Invariance under small translations is typically achieved through the network architecture, by using convolution layers which are covariant[2] and pooling layers which are invariant to small shifts. For more complex trans-

[2]A note on terminology: We categorize possible transformation behaviour into three groups: Covariant (sometimes called equivariant), where the output transforms in the same manner as the input, contravariant, where the output transforms in the opposite way as the input, and invariant, where the transformation in the input leaves the output unchanged.

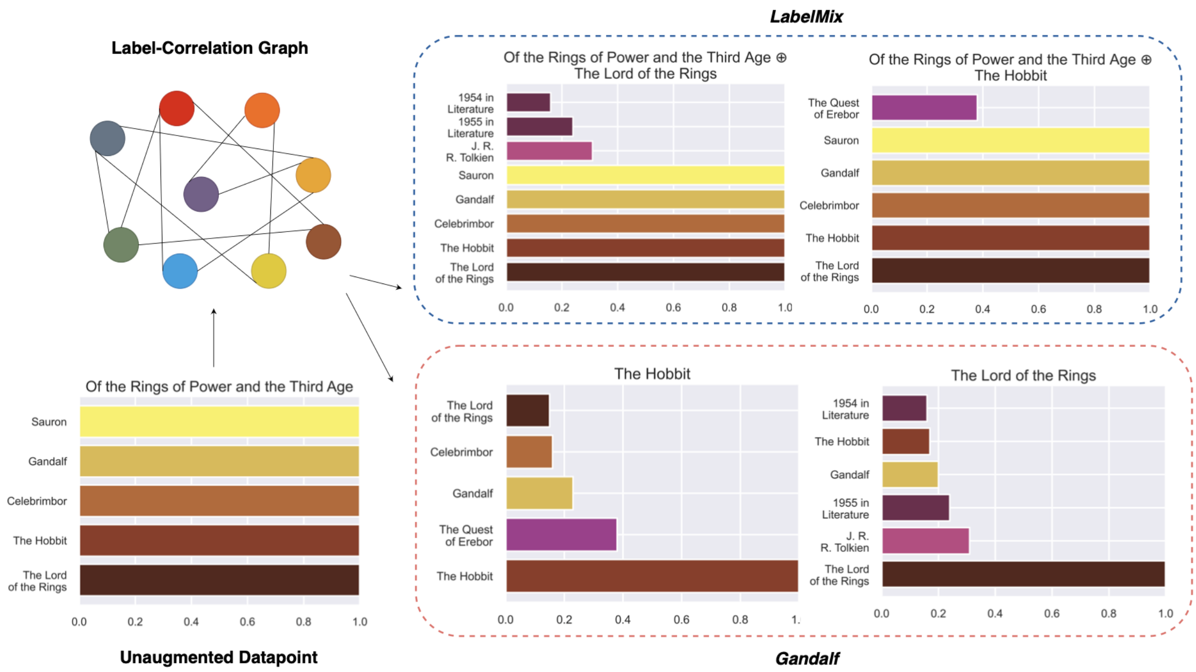

Figure 1: A pictorial representation of the proposed data augmentation strategies. The title of each plot denotes the data point, the y-axis its labels and the x-axis their soft targets. We demonstrate our augmentations on the data point *"Of the Rings of Power and the Third Age"*, which is the final book in the Lord of the Rings(LOTR) series along with labels *"The Hobbit"* and *"The Lord of the Rings"*. The augmentations are formed as per 1 and 2. Notably, the labels found through soft targets through the LCG are all related to the LOTR universe - *"J. R. R. Tolkien"* is the author of the LOTR books, *"The Quest of Erebor"* is a central plotline and *"Celebrimbor"* and *"Gandalf"* (not to be confused with our data augmentation strategy) are major characters. Beyond this, the soft targets also cover generic labels like *"1954 in Literature"* and *"1955 in Literature"*, which is the correct timeline for when the books were released.

formations, such as the rotations and flips, however, data augmentation is required. For continuous transformations, like rotations, translations, or scaling, one can easily postulate that the classification should remain invariant if the change is very small.

Due to the discrete nature of text input in language tasks, however, there are no continuous transformations available. It has been shown in recent works that simple discrete transformations of input data such as replacement with synonym, introduction of typos, and swapping of neighboring texts can improve the performance to a certain extent (Xie et al., 2017; Coulombe, 2018; Wei and Zou, 2019; Niu and Bansal, 2018), however, and such transformations can lead to semantic inconsistency and illegibility. Therefore, we have to look deeper into the actual properties of the short-text data to find transformations with predictable behaviour.

Considerations for input concatenations in short-text XMC For textual data, combining data-points via direct concatenation of input texts of other data-points, can lead to significant changes in their intended meaning. In such a setting, one might assume that if two inputs are joined together, the resulting labels would be the union of the labels of the two data points. Especially for longer text, this could be seen as a sensible approach, e.g. if a Wikipedia article consists of two sections, then the tags for that article could be the union of the tags for each section. This can be encoded as follows :

Hypothesis 1 (Concatenation Covariance). *For two (long-text) inputs $(\mathbf{x}_i, \mathrm{y}_i)$ and $(\mathbf{x}_j, \mathrm{y}_j)$, where $\mathrm{y}_i, \mathrm{y}_j \subset [L]$ are represented as sets, concatenation of inputs corresponds to union of sets*

$$\Phi(\mathbf{x}_i \oplus \mathbf{x}_j) = \mathrm{y}_i \cup \mathrm{y}_j. \qquad (2)$$

However, for short text, one could argue that the opposite is true. If a user adds additional words to a search query, a Wikipedia page, or a product name, then these words are often meant to filter the results further. For example, changing the search query from *"Boat Wireless headphones"* to *"Boat Wireless headphones with microphone"* would lead to a filtered result. This leads to the opposite hypothesis

Hypothesis 2 (Concatenation Contravariance). *For two input queries $(\mathbf{x}_i, \mathrm{y}_i)$ and $(\mathbf{x}_j, \mathrm{y}_j)$, where*

$y_i, y_j \subset [L]$ *are represented as sets, concatenation of inputs corresponds to intersection of sets*

$$\Phi(\mathbf{x}_i \oplus \mathbf{x}_j) = y_i \cap y_j. \tag{3}$$

Further, we have to concede that for two arbitrary short-text queries/product names, it is very difficult to predict the exact meaning of their concatenation. One could argue to use combine the queries in the manifold space (Verma et al., 2019), but as shown in Figure 4, while Manifold Mixup does help reduce the overfitting, it still does not imbue enough inductive bias into the model.

In the case where label features are also short-text data, we can at least identify a subset of concatenations that should leave the classification unchanged. That is, if the second input text can be known beforehand to only reaffirm the content of the first text. Therefore, choosing the second text to be one of the relevant label's text should result in an invariance. From examples in section 5 and Figure 5, one can observe that concatenating a query with one of its label feature only reaffirms the content of the query. This brings us to

Postulate 1 (Label-Affirming Concatenations). *Let* $(\mathbf{x}, y)$ *be a training data point in* $\mathcal{D}$*, and* $j \in y$ *be a label relevant to* $\mathbf{x}$*. Then the classifier should be invariant under concatenation with* $\mathbf{z}_j$

$$\Phi(\mathbf{x} \oplus \mathbf{z}_j) = \Phi(\mathbf{x}). \tag{4}$$

This is corroborated by Figure 2, where queries concatenated with label features in the input space $\Phi(\mathbf{x} \oplus \mathbf{z}_l)$ have their encoded representations in the vicinity (indicated by high cosine similarity) of the encoded representation of only queries as input $\Phi(\mathbf{x})$. Thus, this postulate enables us to specify the vicinal distribution. Given a datapoint $(\mathbf{x}, y) \in \mathcal{D}$, its vicinity is given by $V(\mathbf{x}) \coloneqq \{\mathbf{x} \oplus \mathbf{z}_j : j \in y\}$.

The straightforward way to define the vicinal distribution would be to sample uniformly on $V(\mathbf{x})$. However, as the main goal of the augmentation is to improve the generalization on tail labels, it can be beneficial to allow for weighted distributions. Using an instance-independent weight vector $\mathbf{r} \in \mathbb{R}^L$, the probability of choosing $\mathbf{x} \oplus \mathbf{z}_j$ as the augmented label is given by $y_j r_j / \langle \mathbf{y}, \mathbf{r} \rangle$, where the first term ensures that j is actually a relevant label, the second term is the weighting factor, and the third the normalization. Averaging over the entire dataset thus leads to the vicinal distribution:

$$\mathrm{d}\mathbb{P}_{\mathrm{v}}(\mathbf{x}, \mathbf{y}) = \frac{1}{n} \sum_{i=1}^{n} \frac{\delta_{\mathbf{y}_i}(\mathbf{y})}{\langle \mathbf{y}_i, \mathbf{r} \rangle} \sum_{j=1}^{L} y_{ij} r_j \delta_{\mathbf{x}_i \oplus z_j}(\mathbf{x}).$$

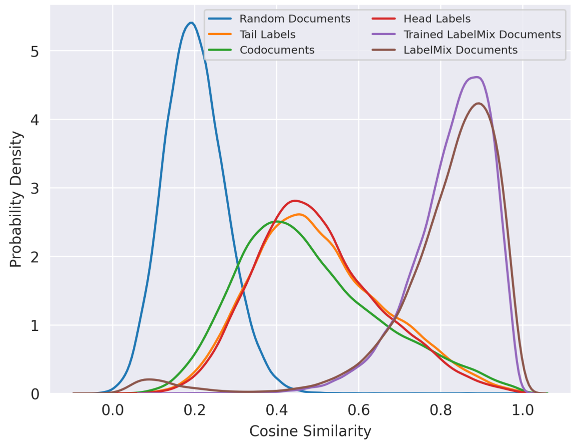

Figure 2: To obtain this distribution, we take 50,000 input queries from LF-AmazonTitles-131K dataset and evaluate their cosine similarity with other queries which are either sampled randomly (Random Documents), or those which have one common label (Codocuments), and also with label features of one of it's labels which belong to either Head Labels or Tail Labels. To demonstrate the efficacy of *LabelMix*, we also evaluate cosine similarity between input queries obtained with the said augmentation and LabelMix documents - denoted by "Trained LabelMix Documents".

5 Label Features in Short-text XMC

5.1 What exactly are Label Features?

To explain label features, we show examples from our datasets (i) LF-WikiTitles-500K, where given the title of wikipedia page, the model needs to predict the relevant categories, and (ii) LF-AmazonTitles-131K, where given a product's title, one need to recommend related products.

Example 1: Similarly, for the wikipedia page *"2022 French presidential election"*, we have the available categories *April 2022 events in France* | *2022 French presidential election* | *2022 elections in France* | *Presidential elections in France*. Further, a google search of the same query, leads to the following related searches - *French election 2022 - The Economist* | *French presidential election coverage on FRANCE 24* | *Presidential Election 2022: A Euroclash Between a "Liberal...* | *French polls, trends and election news for France - POLITICO.eu*, amongst others.

Example 2: For a product on Amazon, *"Mario Kart: Double Dash!! with Bonus Disc"*, we have available *Super Smash Bros Melee* | *Super Mario Sunshine* | *Mario Party 7* | *Super Mario*

Strikers as the recommended products.

Observations In view of these examples, one can affirm two important observations: (i) the problem indeed requires recommending similar items which are either highly correlated or co-occur frequently with the queried item, and (ii) the data instance and the corresponding label-features (approximately) form an equivalence class. For example, a valid news headline search on a search engine should result in a page mentioning the same headline and similar re-phrased headlines from other news media outlets (see Example 1). As a result, we can conclude that data instances are *exchangable* with their labels features.

The above observations are unique to the short-text XMC problem setting dealt in this paper. This is in contrast to a conventional text-classification problem such as *Amazon Reviews for Sentiment Analysis*[3] or *20 Newsgroups* dataset[4], where the model needs to classify news articles, consisting of hundreds of tokens, into 20 categories like politics, sports, electronics etc. Here, the role of the data instances and labels is significantly asymmetric and hence *non-exchangeable*.

5.2 Label Features as Input Queries

The key insight that we leverage from previous observations is that label features are, in fact, valid related searches or products and hence legitimate data instances themselves. Further, we observe that in the XMC settings, where the label space is formed by assigning numeric integers to these related items i.e. label features, every valid query should, ideally, have itself as a label as it is relevant to itself (see example 1). However, this might not be possible for most training instances as not all training instances may exist as a label in a limited training dataset $\mathcal{D}$. However, every label feature $\mathbf{z}_l$ when posed as an input search query (data instance) should fulfill the following postulate:

Postulate 2 (Self-Annotation). *If the features $\mathbf{z}_l$ of a label are interpreted as a data instance, then this instance will be annotated with the corresponding label l. This means that we assume the underlying probability distribution to fulfill*

$$\mathbb{P}[Y_l = 1 \mid X = \mathbf{z}_l] = 1 \qquad (5)$$

[3]https://jmcauley.ucsd.edu/data/amazon/

[4]http://qwone.com/ jason/20Newsgroups/

The above postulate suggests a methodology to use label features as data instances with the self-annotation property. One natural question arises regarding the labels for the thus created data instances (i.e., via self-annotation) : *In a label space $[L]$ comprising of hundreds of thousands or millions of labels, what are the suitable labels $l' \neq l$ for $\mathbf{z}_l$, when posed an a data instance*? According to the observation (i) in Section 5, the labels that are highly correlated to or the ones that frequently co-occur with a label l should, ideally, also make up as the label-set for the label feature $\mathbf{z}_l$, when searched as a query or posed as a data instance.

One way to approximate label correlations or co-occurrences is to use the Label Correlation Graph, as proposed in ECLARE, which gives a smooth approximation of label-occurrences purposely skewed in favor of tail labels. Since the entries in LCG are normalized, these can be interpreted regularized variants of the label co-occurrence matrix. As argued in ECLARE, for each label, the LCG finds a set of semantically similar labels that either share tokens with the label, or are used in the same context. This can be further seen in Figure 5 where the degree of correlation of a label with its first order neighbours in the LCG has been plotted. While ECLARE uses the LCG efficiently to either re-weight logits as a post processing step, or to augment labels' decoding representations $\mathbf{w}_l$, we propose to leverage the graph weights (with an additional row-wise normalization to get values in range [0, 1]) as probabilistic soft labels for $\mathbf{z}_l$ as input data instance. We thereby propose to use the following postulate:

Postulate 3 (Soft-annotations via LCG). *If the features $\mathbf{z}_l$ of a label are interpreted as a data instance; then beyond self-annotation, $\mathbf{z}_l$ can be soft-annotated with the other labels $l' \neq l$ either highly correlated to or that frequently co-occur with l. Therefore, we require the following :*

$$\mathbb{P}[Y_{l'} = 1 \mid X = \mathbf{z}_l] = LCG[l, l'] \qquad (6)$$

6 Proposed Augmentations in Practice

***LabelMix* Augmentation** Through LabelMix, we propose a way of performing label-affirming concatenations where, a data point is chosen with a uniform probability of 0.4 and is concatenated with one of it's label features and the corresponding label space is formed by the concatenation of the label set of the data point and the soft annotations of the label feature (Postulate 2). To efficiently learn representations for tail labels, when sampling a label feature from the label set of the data point, we assign the labels weights according to a skewed label distribution as also found to be suitable in Dahiya et al. (2021a); Mikolov et al. (2013). Further, to restrict the impact of noisy correlations, we empirically find it beneficial to threshold the soft labels obtained from LCG at 0.1.

Algorithm 1: *LabelMix* Augmentation

```
# x - input instance tokens
# y - ground truth label vector
# Z - label feature token matrix
# r - label wise sampling weight vector
def LabelMix(x, y, r, Z, LCG):
  aug_prob = numpy.random.uniform()
  if aug_prob > 0.4:
    return (x, y)
  p_a = numpy.multipy(y, r)
  j = numpy.random.choice(y, p=(p_a/sum(p_a)))
  y_lm = LCG[j, :] / LCG[j, j]
  y_a = numpy.minimum(y + y_lm, 1.0)
  y_a = numpy.where(y_a > 0.1, y_a, 0)
  x_a = numpy.append(x, Z[j])
  return (x_a, y_a)
```

***Gandalf* Augmentation** Through *Gandalf*, we propose a methodology to add label features as data points in the training set. We annotate the label feature (as a data point) with its self and soft annotations (derived from LCG), as described in postulates 2 and 3 in the previous section. The *Gandalf* data augmentation as an algorithmic procedure is shown in Algorithm 1 below :

Algorithm 2: *Gandalf* Augmentation

```
# j - sampling index
# Z - label feature token matrix
def Gandalf(j, Z, LCG):
  x = Z[j]
  y = LCG[j, :] / LCG[j, j]
  y = numpy.where(y > 0.1, y, 0)
  return (x, y)
```

7 Experiments & Discussion

Benchmarks, Baselines & Metrics : We benchmark our experiments on 3 standard public datasets LF-AmazonTitles-131K, LF-WikiSeeAlsoTitles-350K, and LF-WikiTitles-500K. We test the generality and effectiveness of our proposed *LabelMix*[5] and *Gandalf* augmentations across multiple state-of-the-art short-text extreme classifiers, the details of which have been discussed in section 2. Similarly, to test the effectiveness of the INCEPTIONXML encoder on these datasets, we extend the model to leverage label features and call it INCEPTIONXML-LF. For this, we augment it with additional label-text and graph-augmented label text classifiers as done in Mittal et al. (2021b). We further create a 2-stage training strategy as compared to a single one in INCEPTIONXML. The implementation details and training strategy can be found in Appendix A. We measure the models' performance using standard metrics precision@k, denoted P@k, and its propensity-scored version PSP@k (Jain et al., 2016).

7.1 Results

We can make some key observations and develop strong insights not only about the short-text XMC problem with label features but also about specific dataset properties from Table 2. For example, while the effect of data augmentations can be observed strongly in the first two datasets, only limited improvement are noticeable in LF-WikiTitles-500K dataset. This can be attributed to LF-WikiTitles-500K having three times as many data points, on average, per label (Table 1), and thus not requiring as much inductive bias as the other two datasets.

LabelMix *LabelMix* produces synthetic data points in the vicinity of training data and hence, while being effective in capturing correlations between data instances and tail labels, can only imbue limited additional inductive bias into the model. ECLARE, on the other hand, is able to better capture these correlations through its LCG-augmented classifier and only gains trivially from *LabelMix*. DECAF, though, gains non-trivially as it only encodes label text in its classifier which leaves out the scope to capture query-tail label correlations further. Similarly, INCEPTIONXML stands to gain significantly more from *LabelMix* as compared to it's LF-counterpart which, similar to ECLARE, also employs a LCG-augmented classifier. Notably, *LabelMix* works much better on INCEPTIONXML(-LF) because of their dynamic negative mining pipeline, which enables the augmentation to work more effectively.

Gandalf We witness exceptional increase in prediction performance over multiple state-of-the-art extreme classifiers with the *Gandalf* augmentation,

[5]Effectively applying LabelMix on DEEPXML-based methods like ASTEC, DECAF and ECLARE is challenging given their multi-stage pipelines where, in some stages, the word embeddings are frozen.

Method	P@1	P@3	P@5	PSP@1	PSP@3	PSP@5
	LF-AmazonTitles-131K					
AttentionXML	32.25	21.70	15.61	23.97	28.60	32.57
GalaXC	39.17	26.85	19.49	32.50	38.79	43.95
SiameseXML++	41.42	**30.19**	21.21	35.80	40.96	46.19
Astec	37.12	25.20	18.24	29.22	34.64	39.49
+ *LabelMix*	37.95	25.65	18.59	29.91	35.58	40.63
+ *Gandalf*	43.95	29.66	21.39	37.40	43.03	48.31
DecaF	38.4	25.84	18.65	30.85	36.44	41.42
+ *LabelMix*	39.30	26.60	19.23	31.81	37.67	42.83
+ *Gandalf*	42.43	28.96	20.90	35.22	42.12	47.61
Eclare	40.46	27.54	19.63	33.18	39.55	44.10
+ *LabelMix*	40.34	27.54	19.96	33.48	39.74	45.11
+ *Gandalf*	42.51	28.89	20.81	35.72	42.19	47.46
InceptionXML	36.79	24.94	17.95	28.50	34.15	38.79
+ *LabelMix*	40.41	27.45	19.82	32.12	38.54	43.81
+ *Gandalf*	**44.67**	30.00	**21.50**	37.98	43.83	48.93
InceptionXML-LF	40.74	27.24	19.57	34.52	39.40	44.13
+ *LabelMix*	41.90	28.20	20.35	35.60	41.07	46.20
+ *Gandalf*	43.84	29.59	21.30	**38.22**	**43.90**	**49.03**
	LF-WikiSeeAlsoTitles-320K					
AttentionXML	17.56	11.34	8.52	9.45.	10.63	11.73
GalaXC	27.87	18.75	14.30	19.77	22.25	24.47
SiameseXML++	31.97	21.43	16.24	**26.82**	28.42	30.36
Astec	22.72	15.12	11.43	13.69	15.81	17.50
+ *LabelMix*	22.91	15.79	12.02	13.99	16.57	18.04
+ *Gandalf*	31.10	21.54	16.53	23.60	26.48	28.80
DecaF	25.14	16.90	12.86	16.73	18.99	21.01
+ *LabelMix*	26.55	18.04	13.75	17.86	20.46	22.61
+ *Gandalf*	31.10	21.60	16.53	23.81	26.69	29.09
Eclare	29.35	19.83	15.05	22.01	24.23	26.27
+ *LabelMix*	29.42	19.94	15.17	22.05	24.36	26.46
+ *Gandalf*	31.33	21.40	16.31	24.83	27.18	29.29
InceptionXML	23.10	15.54	11.52	14.15	16.71	17.39
+ *LabelMix*	25.16	17.03	12.97	16.11	18.72	20.76
+ *Gandalf*	32.54	22.15	16.86	25.27	27.76	30.03
InceptionXML-LF	28.99	19.53	14.79	21.45	23.65	25.65
+ *LabelMix*	29.68	20.16	15.32	22.24	24.69	26.80
+ *Gandalf*	**33.12**	**22.70**	**17.29**	26.68	**29.03**	**31.27**
	LF-WikiTitles-500K					
AttentionXML	40.90	21.55	15.05	14.80	13.97	13.88
SiameseXML++	42.08	22.80	16.01	23.53	21.64	21.41
Astec	44.40	24.69	17.49	18.31	18.25	18.56
+ *LabelMix*	44.63	24.91	18.35	19.21	19.53	19.32
+ *Gandalf*	45.24	25.45	18.57	21.72	20.99	21.16
DecaF	44.21	24.64	17.36	19.29	19.82	19.96
+ *LabelMix*	45.33	25.44	18.51	23.01	21.66	21.93
+ *Gandalf*	45.76	25.76	18.90	24.08	22.14	22.78
Eclare	44.36	24.29	16.91	21.58	20.39	19.84
+ *LabelMix*	44.55	24.42	17.22	21.70	20.54	19.98
+ *Gandalf*	44.82	24.52	17.48	22.11	20.44	20.10
InceptionXML	44.61	24.79	19.52	18.65	18.70	18.94
+ *LabelMix*	44.85	24.91	19.73	19.37	18.98	19.56
+ *Gandalf*	45.93	25.81	**20.36**	21.89	21.54	22.56
InceptionXML-LF	44.89	25.71	18.23	23.88	22.58	22.50
+ *LabelMix*	45.64	26.35	18.78	24.09	22.98	23.00
+ *Gandalf*	**47.13**	**26.87**	19.03	**24.12**	**23.92**	**23.82**

Table 2: Effect of adding *LabelMix* and *Gandalf* data augmentations on state-of-the-art extreme classifiers in terms of P@k and PSP@k public benchmark datasets. The best-performing approach is in **bold**.

Method	P@1	P@3	P@5	PSP@1	PSP@3	PSP@5
	LF-AmazonTitles-131K					
InceptionXML	35.62	24.13	17.35	27.53	33.06	37.50
+*LabelMix w/o SA*	37.25	25.02	17.98	29.25	34.58	39.09
+*LabelMix w SA*	39.05	26.52	19.15	30.98	37.20	42.26
+*Gandalf w/o SA*	37.59	25.25	18.18	30.75	35.54	40.06
+*Gandalf w SA*	43.52	29.23	20.92	36.96	42.71	47.64
	LF-WikiSeeAlsoTitles-320K					
InceptionXML	21.53	14.19	10.66	13.06	14.87	16.33
+*LabelMix w/o SA*	22.61	14.98	11.30	14.02	15.95	17.55
+*LabelMix w SA*	23.90	16.10	12.28	15.20	17.60	19.56
+*Gandalf w/o SA*	24.43	16.16	12.15	16.89	18.45	20.02
+*Gandalf w SA*	31.31	21.38	16.22	24.31	26.79	28.83

Table 3: Ablation results demonstrating the effectiveness of using soft-annotations (denoted *SA*) obtained from the LCG on a single InceptionXML model. Notably, soft annotations play an a very important role in learning label-label correlations.

with gains up to 30% being observed in case of Astec and InceptionXML, which inherently do not leverage label features. We believe the models benefit from *Gandalf* in two ways: (i) from Figure 2 it is evident that $\Phi(\mathbf{z}_l)$ does not exist in the vicinity of $\Phi(\mathbf{x}_i)$, where $l \in \mathrm{y}_i$, for either head or tail labels. Thus, *Gandalf* essentially expands the dataset by adding label features as data points which are far from training instances in $\mathcal{D}$ and, (ii) as shown in the ablation Table 3, the soft-annotations play an important role in enabling the encoder to inherently learn the label-label correlations. Notably, *Gandalf*-augmented baselines do not need to make any architectural modifications or employ complicated training pipelines to learn strong inductive biases. For instance, Astec and InceptionXML beat their LF-counterparts DecaF and Eclare, and InceptionXML-LF respectively on LF-AmazonTitles-131K, while performing at par with them on other two datasets. Further visualizations depicting differences in predictions obtained by our proposed augmentations can be found in Table 4(Appendix B).

8 Conclusion

In this paper, we proposed two data augmentation methods which are particularly suited for short-text extreme classification. These augmentations not only eliminate the need for complicated training procedures in order to imbue inductive biases, but dramatic increase in prediction performance of state-of-the-art methods in this domain. It is expected that our treatment towards studying invariances in this domain will spur further research.

References

R. Agrawal, A. Gupta, Y. Prabhu, and M. Varma. 2013. Multi-label learning with millions of labels: Recommending advertiser bid phrases for web pages. In *WWW*.

R. Babbar and B. Schölkopf. 2017. DiSMEC: Distributed Sparse Machines for Extreme Multi-label Classification. In *WSDM*.

R. Babbar and B. Schölkopf. 2019. Data scarcity, robustness and extreme multi-label classification. *Machine Learning*, 108:1329–1351.

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

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

Olivier Chapelle, Jason Weston, Léon Bottou, and Vladimir Vapnik. 2000. Vicinal risk minimization. In *Advances in Neural Information Processing Systems*, volume 13. MIT Press.

A. Choromanska and J. Langford. 2015. Logarithmic time online multiclass prediction. *NIPS*.

Claude Coulombe. 2018. Text data augmentation made simple by leveraging nlp cloud apis. arxiv. *Retrieved June*, 15:2020.

K. Dahiya, A. Agarwal, D. Saini, K. Gururaj, J. Jiao, A. Singh, S. Agarwal, P. Kar, and M. Varma. 2021a. Siamesexml: Siamese networks meet extreme classifiers with 100m labels. In *ICML*.

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

Steven Y Feng, Varun Gangal, Jason Wei, Sarath Chandar, Soroush Vosoughi, Teruko Mitamura, and Eduard Hovy. 2021. A survey of data augmentation approaches for nlp. *arXiv preprint arXiv:2105.03075*.

Fei Gao, Jinhua Zhu, Lijun Wu, Yingce Xia, Tao Qin, Xueqi Cheng, Wengang Zhou, and Tie-Yan Liu. 2019. Soft contextual data augmentation for neural machine translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5539–5544.

Demi Guo, Yoon Kim, and Alexander M Rush. 2020. Sequence-level mixed sample data augmentation. *arXiv preprint arXiv:2011.09039*.

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

Himanshu Jain, Yashoteja Prabhu, and Manik Varma. 2016. Extreme multi-label loss functions for recommendation, tagging, ranking & other missing label applications. In *KDD*, pages 935–944.

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

Ting Jiang, Deqing Wang, Leilei Sun, Huayi Yang, Zhengyang Zhao, and Fuzhen Zhuang. 2021. 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, pages 7987–7994.

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

Siddhant Kharbanda, Atmadeep Banerjee, Akash Palrecha, Devaansh Gupta, and Rohit Babbar. 2021. Inceptionxml : A lightweight framework with synchronized negative sampling for short text extreme classification. *arXiv preprint arXiv:2109.07319*.

Yitong Li, Trevor Cohn, and Timothy Baldwin. 2017. Robust training under linguistic adversity. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 21–27.

J. Liu, W. Chang, Y. Wu, and Y. Yang. 2017. Deep Learning for Extreme Multi-label Text Classification. In *SIGIR*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc.

A. Mittal, K. Dahiya, S. Agrawal, D. Saini, S. Agarwal, P. Kar, and M. Varma. 2021a. Decaf: Deep extreme classification with label features. In *WSDM*.

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

Tong Niu and Mohit Bansal. 2018. Adversarial oversensitivity and over-stability strategies for dialogue models. *arXiv preprint arXiv:1809.02079*.

Jeffrey Pennington, R. Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *EMNLP*.

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

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

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

Vikas Verma, Alex Lamb, Christopher Beckham, Amir Najafi, Ioannis Mitliagkas, David Lopez-Paz, and Yoshua Bengio. 2019. Manifold mixup: Better representations by interpolating hidden states. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 6438–6447. PMLR.

Jason Wei and Kai Zou. 2019. Eda: Easy data augmentation techniques for boosting performance on text classification tasks. *arXiv preprint arXiv:1901.11196*.

Ziang Xie, Sida I Wang, Jiwei Li, Daniel Lévy, Aiming Nie, Dan Jurafsky, and Andrew Y Ng. 2017. Data noising as smoothing in neural network language models. *arXiv preprint arXiv:1703.02573*.

Brandon Yang, Gabriel Bender, Quoc V Le, and Jiquan Ngiam. 2019. Condconv: Conditionally parameterized convolutions for efficient inference. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Yiben Yang, Chaitanya Malaviya, Jared Fernandez, Swabha Swayamdipta, Ronan Le Bras, Ji-Ping Wang, Chandra Bhagavatula, Yejin Choi, and Doug Downey. 2020. Generative data augmentation for commonsense reasoning. *arXiv preprint arXiv:2004.11546*.

H. Ye, Z. Chen, D.-H. Wang, and Davison B. D. 2020. Pretrained Generalized Autoregressive Model with Adaptive Probabilistic Label Clusters for Extreme Multi-label Text Classification. In *ICML*.

E.H. I. Yen, X. Huang, W. Dai, I. Ravikumar, P.and Dhillon, and E. Xing. 2017. PPDSparse: A Parallel Primal-Dual Sparse Method for Extreme Classification. In *KDD*.

R. You, Z. Zhang, Z. Wang, S. Dai, H. Mamitsuka, and S. Zhu. 2019. Attentionxml: Label tree-based attention-aware deep model for high-performance extreme multi-label text classification. In *NeurIPS*.

H. Yu, P. Jain, P. Kar, and I. S. Dhillon. 2014. Large-scale Multi-label Learning with Missing Labels. In *ICML*.

Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Young Joon Yoo. 2019. Cutmix: Regularization strategy to train strong classifiers with localizable features. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 6022–6031.

Hongyi Zhang, Moustapha Cisse, Yann N. Dauphin, and David Lopez-Paz. 2018. mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations*.

Jiong Zhang, Wei-cheng Chang, Hsiang-fu Yu, and Inderjit Dhillon. 2021. Fast multi-resolution transformer fine-tuning for extreme multi-label text classification. *Advances in Neural Information Processing Systems*, 34.

Minyi Zhao, Lu Zhang, Yi Xu, Jiandong Ding, Jihong Guan, and Shuigeng Zhou. 2022. Epida: An easy plug-in data augmentation framework for high performance text classification. *arXiv preprint arXiv:2204.11205*.

A InceptionXML-LF

Model Outlook: Short-text queries are encoded by a modified InceptionXML encoder, which encodes an input query $\mathbf{x}_i$ using an encoder $\Phi_q := (E, \theta)$ parameterised by E and θ, where E denotes a D-dimensional embedding layer of $\mathbb{R}^{\mathcal{V} \times D}$ for vocabulary tokens $\mathcal{V} = [t_1, t_2, \ldots, t_V]$ and θ denotes the parameters of the embedding enhancement and the inception module respectively. Alongside Φ_q, InceptionXML-LF learns two frugal Astec-like (Dahiya et al., 2021b) encoders, one each as a label-text encoder $\Phi_l := \{E, \mathcal{R}\}$ and a graph augmented encoder $\Phi_g := \{E, \mathcal{R}\}$. Here, $\mathcal{R}$ denotes the parameters of a fully connected layer bounded by a spectral norm and the embedding layer E is shared between all Φ_q, Φ_l and Φ_g for joint query-label word embedding learning. Further, an attention module $\mathcal{A}$, meta-classifier $\mathcal{W}_m$ and an extreme classifier $\mathcal{W}_e$ are also learnt together with the encoders. Next, we specify the details of all components of InceptionXML-LF.

A.1 Instance-Attention in Query Encoder

We make two improvements to the inception module InceptionXML for better efficiency. Firstly, in the inception module, the activation maps from the first convolution layer are concatenated before passing them onto the second convolution layer. To make this more computationally efficient, we replace this "inception-like" setting with a "mixture of expert" setting (Yang et al., 2019). Specifically, a route function is added that produces dynamic weights for each instance to perform a dynamic element-wise weighted sum of activation maps of each filter. Along with the three convolutional experts, we also add an average pool as a down sampling residual connection to ensure better gradient flow across the encoder.
Second, we decouple the second convolution layer to have one each for the meta and extreme classification tasks.

A.2 Dynamic Hard Negative Mining

Training one-vs-all (OvA) label classifiers becomes infeasible in the XMC setting where we have hundreds of thousands or even millions of labels. To mitigate this problem, the final prediction or loss calculation is done on a shortlist of size $\sqrt{L}$ comprising of only hard-negatives label. This mechanism helps reduce complexity of XMC from an intractable $O(NDL)$ to a computationally feasible $O(ND\sqrt{L})$ problem. InceptionXML-LF inherits the synchronized hard negative mining framework as used in the InceptionXML. Specifically, the encoded meta representation is passed through the meta-classifier which predicts the top-K relevant label clusters per input query. All labels present in the top-K shortlisted label clusters then form the hard negative label shortlist for the extreme task. This allows for progressively harder labels to get shortlisted per short-text query as the training proceeds and the encoder learns better representations.

A.3 Label-text and LCG Augmented Classifiers

InceptionXML-LF's extreme classifier weight vectors $\mathcal{W}_e$ comprise of 3 weights, as in Mittal et al. (2021b). Specifically, the weight vectors are a result of an attention-based sum of (i) label-text embeddings, created through Φ_l, (ii) graph augmented label embeddings, created through graph encoder Φ_g and, (iii) randomly initialized per-label independent weights $\mathbf{w}_l$.
As shown in Figure 3, we first obtain label-text embeddings as $\mathbf{z}_l^1 = E \cdot \mathbf{z}_l^0$, where $\mathbf{z}_l^0$ are the TF-IDF weights of label feature corresponding to label l. Next, we use the label correlation graph $\mathbf{G}$ to create the graph-weighted label-text embeddings $\mathbf{z}_l^2 = \sum_{m \in [L]} \mathbf{G}_{lm} \cdot \mathbf{z}_l^0$ to capture higher order query-tail label correlations. $\mathbf{z}_l^1$ and $\mathbf{z}_l^2$ are then passed into the frugal encoders Φ_l and Φ_g respectively. These encoders comprise only of a residual connection across a fully connected layer as $\alpha \cdot \mathcal{R} \cdot \mathcal{G}(\tilde{z}_l) + \beta \cdot \tilde{z}_l$, where $\tilde{z}_l = \{\mathbf{z}_l^1, \mathbf{z}_l^2\}$, $\mathcal{G}$ represents GELU activation and α and β are learned weights. Finally, the per-label weight vectors for the extreme task are obtained as

$$\mathcal{W}_{e,l} = \mathcal{A}(\mathbf{z}_l^1, \mathbf{z}_l^2, \mathbf{w}_l) = \alpha^1 \cdot \mathbf{z}_l^1 + \alpha^2 \cdot \mathbf{z}_l^2 + \alpha^3 \cdot \mathbf{w}_l$$

where $\mathcal{A}$ is the attention block and $\alpha^{\{1,2,3\}}$ are the dynamic attention weights produced by the attention block.

A.4 Two-phased Training

Motivation: We find there to be a mismatch in the training objectives in DeepXML-based approaches like Astec, Decaf and Eclare which first train their word embeddings on meta-labels in Phase I and then transfer these learnt embeddings for classification over extreme fine-grained labels in Phase III (Dahiya et al., 2021b). Thus,

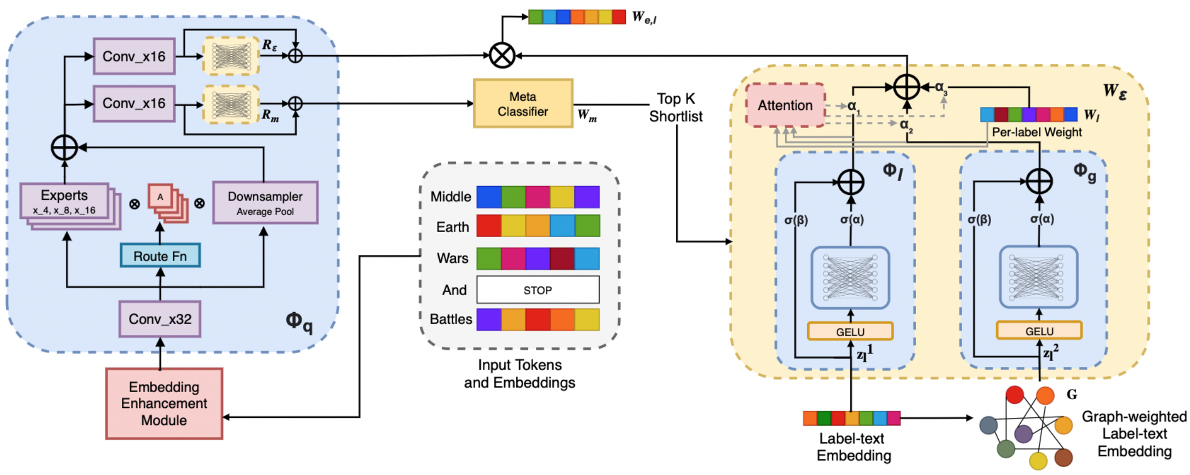

Figure 3: INCEPTIONXML-LF. The improved Inception Module along with instance attention is shown in detail. Changes to the INCEPTIONXML framework using the ECLARE classifier are also shown.

in our two-phased training for INCEPTIONXML-LF, we keep our training objective same for both phases. Note that, in INCEPTIONXML-LF the word embeddings are always learnt on labels instead of meta-labels or label clusters and we only augment our extreme classifier weight vectors $\mathcal{W}_e$ with label-text embeddings and LCG weighted label embeddings. We keep the meta-classifier $\mathcal{W}_m$ as a standard randomly initialized classification layer.

Phase I: In the first phase, we initialize the embedding layer E with pre-trained GloVe embeddings (Pennington et al., 2014), the residual layer $\mathcal{R}$ in Φ_l and Φ_g is initialized to identity and the rest of the model comprising of Φ_q, $\mathcal{W}_m$ and $\mathcal{A}$ is randomly initialized. The model is then trained end-to-end but without using free weight vectors $\mathbf{w}_l$ in the extreme classifier $\mathcal{W}_e$. This set up implies that $\mathcal{W}_e$ only consists of weights tied to E through Φ_l and Φ_g which allows for efficient joint learning of query-label word embeddings (Mittal et al., 2021a) in the absence of free weight vectors. Model training in this phase follows the INCEPTIONXML+ pipeline as described in Kharbanda et al. (2021) without detaching any gradients to the extreme classifier for the first few epochs. In this phase, the final per-label score is given by:

$$P_l = \mathcal{A}(\Phi_l(\mathbf{z}_l^1),\ \Phi_g(\mathbf{z}_l^2)) \cdot \Phi_q(x)$$

Phase II: In this phase, we first refine our clusters based on the jointly learnt word embeddings. Specifically, we recluster the labels using the dense $\mathbf{z}_l^1$ representations instead of using their sparse PIFA representations (Chang et al., 2020) and consequently reinitialize $\mathcal{W}_m$. We repeat the Phase I training, but this time the formulation of $\mathcal{W}_e$ also includes $\mathbf{w}_l$ which are initialised with the updated $\mathbf{z}_l^1$ as well. Here, the final per-label score is given by:

$$P_l = \mathcal{A}(\Phi_l(\mathbf{z}_l^1),\ \Phi_g(\mathbf{z}_l^2),\ \mathbf{w}_l) \cdot \Phi_q(x)$$

B Visualizations and Extra Results

Additional visualizations capturing the label correlations and their first order-neighbors are shown in Figure 5. The relative comparison of outputs generated by vanilla model, and those as a result of the proposed augmentations is shown in Table 3.

C Limitations

Our work is limited to the extreme classification problem setting in which the labels are endowed with textual descriptions, and the input data-points are short-text instances as those encountered in Search and recommendation problems based on product titles.

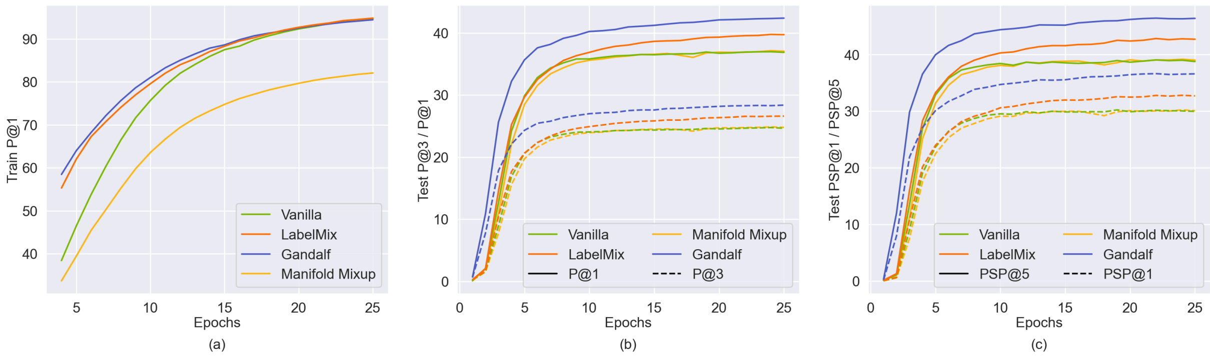

Figure 4: Effect of different data augmentations on INCEPTIONXML-LF. Remarkable improvements can be noted as a result of using the proposed data augmentations *LabelMix* and *Gandalf*. However, from (a) and (b), a significant generalization can be observed between Train and Test P@1. While manifold mixup is effective in reducing overfitting, it only makes trivial improvements to the prediction performance.

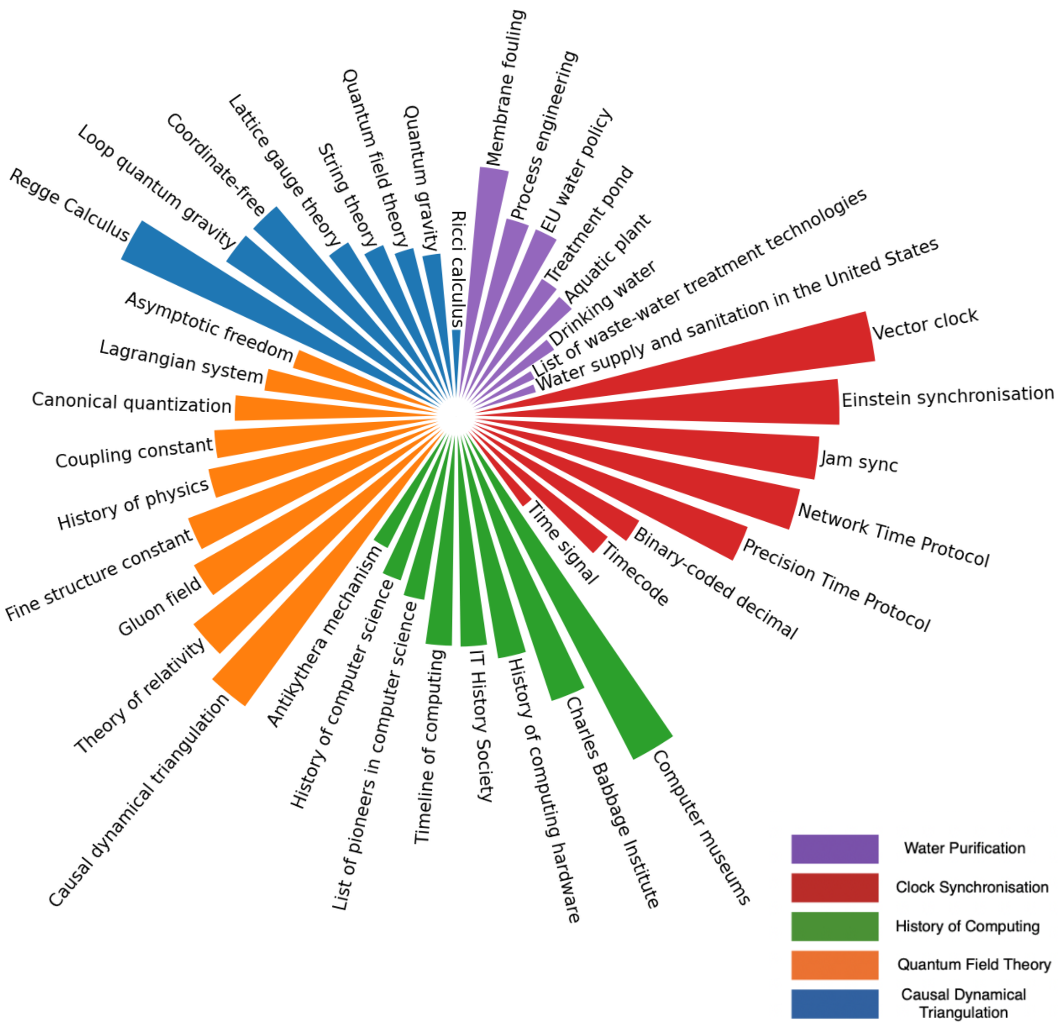

Figure 5: Correlations between labels and their first-order neighbours, as found by the LCG on the LF-WikiTitles-500K dataset. The legend shows the label in question, the bar chart shows the degree of correlation with its neighbouring labels. Correlated labels often share tokens with each other and/or may be used in the same context.

Method	**Datapoint**	**Vanilla Predictions**	***LabelMix* Predictions**	***Gandalf* Predictions**
InceptionXML-LF	Topological group	Pontryagin duality, Topological order, Topological quantum field theory, Topological quantum number, Quantum topology	Topological order, Algebraic group, Topological ring, Topological quantum field theory, Topological quantum number	Compact group, Haar measure, Lie group, Algebraic group, Topological ring
DecaF		Topological quantum computer, Topological order, Topological quantum field theory, Topological quantum number, Quantum topology	Topological order, Algebraic group, Topological ring, Topological quantum field theory, Topological quantum number	Compact group, Haar measure, Lie group, Algebraic group, Topological ring
ECLARE		Topological quantum computer, Topological order, Topological quantum field theory, Topological quantum number, Quantum topology	Topological order, Topological ring, Topological quantum field theory, Topological quantum number, Quantum topology	Compact group, Topological order, Lie group, Algebraic group, Topological ring
InceptionXML-LF	Oat	List of lighthouses in Scotland, List of Northern Lighthouse Board lighthouses, Oatcake, Communes of the Finistere department, Communes of the Cotes-d'Armor department	Oatcake, Oat milk, Rolled oats, List of oat diseases, Goboat	Oatcake, Oatmeal, Oat milk, Porridge, Rolled oats
DecaF		Oatcake, Oatmeal, Design for All (in ICT), Oatley Point Reserve, Oatley Pleasure Grounds	Oatcake, Oatmeal, Oat milk, Oatley Point Reserve, Oatley Pleasure Grounds	Oatcake, Oatmeal, Oat milk, Porridge, Rolled oats
ECLARE		Oatmeal, Oat milk, Parks in Sydney, Oatley Point Reserve, Oatley Pleasure Grounds	Oatmeal, Rolled oats, McCann's Steel Cut Irish Oatmeal, Oatley Point Reserve, Oatley Pleasure Grounds	Oatcake, Porridge, Rolled oats, Oatley Point Reserve, Oatley Pleasure Grounds
InceptionXML-LF	Grand Lake, Colorado	Colorado metropolitan areas, Front Range Urban Corridor, Outline of Colorado, Index of Colorado-related articles, State of Colorado	Colorado metropolitan areas, Outline of Colorado, Index of Colorado-related articles, State of Colorado, Colorado counties	Colorado metropolitan areas, Outline of Colorado, Index of Colorado-related articles, Colorado cities and towns, Colorado counties
DecaF		Colorado metropolitan areas, Front Range Urban Corridor, State of Colorado, Colorado municipalities, National Register of Historic Places listings in Grand County, Colorado	Front Range Urban Corridor, Index of Colorado-related articles, National Register of Historic Places listings in Grand County, Colorado, Grand County, Colorado, List of lakes in Colorado	Outline of Colorado, State of Colorado, Colorado cities and towns, Colorado municipalities, Colorado counties
ECLARE		State of Colorado, Colorado cities and towns, Colorado counties, National Register of Historic Places listings in Grand County, Colorado, Grand County, Colorado	Colorado metropolitan areas, State of Colorado, Colorado cities and towns, Colorado counties, Colorado census designated places	Outline of Colorado, Index of Colorado-related articles, State of Colorado, Colorado cities and towns, Colorado counties
InceptionXML-LF	Lunar Orbiter program	Lunar Orbiter Image Recovery Project, Lunar Orbiter 3, Lunar Orbiter 5, Chinese Lunar Exploration Program, List of future lunar missions	Exploration of the Moon, List of missions to the Moon, Lunar Orbiter Image Recovery Project, Lunar Orbiter 3, Lunar Orbiter 5	Surveyor program, Luna programme, Lunar Orbiter Image Recovery Project, Lunar Orbiter 3, Lunar Orbiter 5
DecaF		Exploration of the Moon, List of man-made objects on the Moon, Lunar Orbiter Image Recovery Project, Lunar Orbiter 3, Lunar Orbiter 5	Exploration of the Moon, Lunar Orbiter program, Lunar Orbiter Image Recovery Project, Lunar Orbiter 3, Lunar Orbiter 5	Exploration of the Moon, Apollo program, Surveyor program, Luna programme, Lunar Orbiter program
ECLARE		Exploration of the Moon, Lunar Orbiter program, Lunar Orbiter Image Recovery Project, Lunar Orbiter 3, Lunar Orbiter 5	Exploration of the Moon, Lunar Orbiter program, Lunar Orbiter Image Recovery Project, Lunar Orbiter 3, Lunar Orbiter 5	Exploration of the Moon, Pioneer program, Surveyor program, Luna programme, Lunar Orbiter program
InceptionXML-LF	Armed Forces of Saudi Arabia	Royal Saudi Air Defense, Royal Saudi Strategic Missile Force, Saudi Royal Guard Regiment, Terrorism in Saudi Arabia, Capital punishment in Saudi Arabia	Saudi-led intervention in Bahrain, Royal Saudi Navy, Royal Saudi Air Defense, Royal Saudi Strategic Missile Force, Saudi Royal Guard Regiment	Military of Saudi Arabia, Royal Saudi Air Force, Royal Saudi Air Defense, Royal Saudi Strategic Missile Force, King Khalid Military City
DecaF		Saudi Arabian-led intervention in Yemen, Saudi-led intervention in Bahrain, Human rights in Saudi Arabia, Legal system of Saudi Arabia, Joint Chiefs of Staff (Saudi Arabia)	Saudi-led intervention in Bahrain, Saudi Arabia, Military of Saudi Arabia, Royal Saudi Strategic Missile Force, Saudi Arabian National Guard	Royal Saudi Air Force, Royal Saudi Navy, Royal Saudi Air Defense, Royal Saudi Strategic Missile Force, Saudi Arabian National Guard
ECLARE		List of armed groups in the Syrian Civil War, Military of Saudi Arabia, Royal Saudi Strategic Missile Force, King Khalid Military City, Joint Chiefs of Staff (Saudi Arabia)	Military of Saudi Arabia, Royal Saudi Air Defense, King Khalid Military City, Saudi Royal Guard Regiment, List of rulers of Saudi Arabia	Military of Saudi Arabia, Royal Saudi Air Defense, Royal Saudi Strategic Missile Force, King Khalid Military City, Saudi Royal Guard Regiment

Table 4: Prediction examples of different datapoints from the LF-WikiSeeAlsoTitles-320K dataset. Labels indicate mispredictions. It may be noted that queries with even just a single word, like *"Oat"*, which has random labels in the case of a Vanilla Prediction, gets all the labels right with the addition of *Gandalf*. Furthermore, even mispredictions get closer when our data augmentation strategy is introduced.