Generalized Zero-Shot Extreme Multi-label Learning

Nilesh Gupta*
nileshgupta2797@gmail.com

Sakina Bohra†
Sakina.Bohra@microsoft.com

Yashoteja Prabhu‡*
yprabhu@microsoft.com

Saurabh Purohit*
purohit10saurabh@gmail.com

Manik Varma*‡
manik@microsoft.com

ABSTRACT

Extreme Multi-label Learning (XML) involves assigning the *subset* of most relevant labels to a data point from *millions* of label choices. A hitherto unaddressed challenge in XML is that of predicting *unseen* labels with no training points. These form a significant fraction of total labels and contain fresh and personalized information desired by end users. Most existing extreme classifiers are not equipped for zero-shot label prediction and hence fail to leverage unseen labels. As a remedy, this paper proposes a novel approach called ZestXML for the task of Generalized Zero-shot XML (GZXML) where relevant labels have to be chosen from all available seen *and* unseen labels. ZestXML learns to project a data point's features close to the features of its relevant labels through a highly sparsified linear transform. This $L0$-constrained linear map between the two high-dimensional feature vectors is tractably recovered through a novel optimizer based on Hard Thresholding. By effectively leveraging the sparsities in features, labels and the learnt model, ZestXML achieves higher accuracy and smaller model size than existing XML approaches while also promoting efficient training & prediction, real-time label update as well as explainable prediction.

Experiments on large-scale GZXML datasets demonstrated that ZestXML can be up to 14% and 10% more accurate than state-of-the-art extreme classifiers and leading BERT-based dense retrievers respectively, while having 10x smaller model size. ZestXML trains on largest dataset with 31M labels in just 30 hours on a single core of a commodity desktop. When added to an large ensemble of existing models in Bing Sponsored Search Advertising, ZestXML significantly improved click yield of IR based system by 17% and unseen query coverage by 3.4% respectively. ZestXML's source code and benchmark datasets for GZXML will be publically released for research purposes here.

*Microsoft Research India
†Microsoft India
‡Indian Institute of Technology Delhi

KDD '21, August 14–18, 2021, Virtual Event, Singapore
© 2021 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 978-1-4503-8332-5/21/08…$15.00
https://doi.org/10.1145/3447548.3467426

CCS CONCEPTS

• **Computing methodologies → Supervised learning by classification**.

KEYWORDS

Extreme multi-label classification, Sponsored search advertising, Label metadata, Zero-shot learning

ACM Reference Format:
Nilesh Gupta, Sakina Bohra, Yashoteja Prabhu, Saurabh Purohit, and Manik Varma. 2021. Generalized Zero-Shot Extreme Multi-label Learning. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '21), August 14–18, 2021, Virtual Event, Singapore.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3447548.3467426

1 INTRODUCTION

Extreme multi-label learning (XML) involves assigning the *subset* of most relevant labels to a data point from an extremely large set of label choices. This research area has seen a surge in popularity in recent years owing to its numerous applications in document tagging [7], product recommendation [8], computational advertising [45] and language modeling [23]. For instance, the search advertising task can be posed as predicting the subset of all possible search queries (labels) from users for which an ad (a data point) could receive clicks [45]. The XML paradigm aims to provide highly accurate and efficient solutions to large-scale ranking and recommendation tasks.

Unseen labels in XML: This paper addresses a crucial, yet under-explored, problem of *unseen* or zero-shot labels in XML which are absent during training time but are available for prediction. Such labels occur frequently in most XML applications since the label set grows continuously over time with evolving user activities and needs. For example, in Bing advertising, a majority of user queries are new (see Figure 1) and are known to contribute a significant fraction of ad clicks. As part of the long label tail [20], unseen labels also serve fresh, serendipitous and personalized information often desired by the end users. Accurate prediction of relevant but unseen labels is, therefore, critical for maximizing user satisfaction and revenue in XML applications. At the same time, it is also equally important to predict the previously seen relevant labels since they serve the familiar or routine needs of the users.

GZXML paradigm: Motivated by these observations, this paper explores the problem of predicting the relevant label subset for a data point from *millions* of both seen and unseen labels, a paradigm we refer to as *Generalized Zero-shot* Extreme Multi-label Learning (GZXML). Unseen labels have been generally recognized as hard to learn since they lack any representative training points. The existing XML approaches sidestep this challenge by restricting their learning to only the seen labels with enough training points. Consequently, they suffer from degraded model accuracy, need frequent and expensive model re-trainings to mitigate the label churn and are still unable to predict completely new labels. This necessitates new, improved approaches provided by GZXML setting.

Challenges of GZXML: A number of research challenges have to be addressed by an effective solution to GZXML. First, the labels belonging to all data regimes need to be accurately modeled in a seamless manner. These regimes can be roughly divided into many-shot, few-shot and zero-shot labels with > 3, $1 - 3$ and 0 training points respectively. While popular labels might have enough data to allow complex and personalized models, the tail (few-shot and zero-shot) labels require information sharing between labels. A GZXML approach needs to carefully distribute its capacity among the labels to maximize the overall prediction accuracy.

Second, training and prediction should be efficient in terms of both time and memory requirements. This is essential to scale to applications with several million data points, features and labels (see Table 1). A training time of a few hours and a model size of a few GBs enable learning and calibrating models on cheap commodity hardware which maximizes profits in big-data applications and makes these techniques accessible to general practitioners. Furthermore, a prediction latency of just a few milliseconds is essential in most user-facing applications.

Third, since new labels would continually arrive in GZXML applications (*e.g.* queries in Bing Ads), the data structures used for efficient prediction should allow real-time label updates.

Limitations of existing approaches: The current state-of-the-art techniques in XML are 1-vs-All based approaches [2, 11, 63, 66]. These approaches learn a separate personalized classifier for each label seen during training time and thus accurately model the many-shot labels. However, these largely ignore any available label meta-data and are hence incapable of recommending zero-shot labels. Additionally, they also tend to perform poorly on few-shot labels due to classifier over-fitting issues (see Section 5). Recently, several approaches have been proposed which aim to model the few-shot labels more accurately [40, 46]; these, however, do not address the zero-shot prediction problem.

Several zero-shot multi-label learners have also been proposed for the non-extreme setting of up to a few thousand labels [7, 14, 41, 50, 69]. They leverage label meta-data such as label hierarchies or features for transferring the learnt information from seen to unseen labels. Unfortunately, these methods possess linear or super-linear complexities in the number of labels and are prohibitively expensive when applied to millions of labels.

Among the other relevant approaches to GZXML are the dense approximate nearest neighbour search (DANNS) techniques [34, 48, 60] and the generative techniques [62]. The DANNS approaches project the relevant data points and labels close together in a dense shared embedding space and utilize approximate search data structures for efficient label prediction. These methods tend to have inferior performance on head labels due to lack of classifier personalization and often need days to train. In addition, the underlying ANNS data structures [35] do not support real-time label updates in an out-of-the-box manner. The generative approaches, which synthesize the relevant labels for a given test point, tend to have large prediction latencies which makes them unsuitable for real-world tasks with a few milliseconds's latency budget [62].

Proposed algorithm: This paper proposes the ZestXML algorithm for solving the GZXML problem. ZestXML aims to leverage label features effectively in order to generalize well to both seen and unseen labels. During training, it learns to project a data point's features close to the features of its relevant labels through a highly sparsified linear transform. The enforced model sparsity is based on the intuition that most relevant labels can be recognized by the information provided by a small number of discrete feature-feature interactions with a point (see Section 3). The number of such interactions can be really huge since features are often in millions; ZestXML develops a novel optimizer named eXtreme Hard Thresholding Pursuit (XHTP) to address this. ZestXML effectively uses model sparsity, not only as a regularizer, but also for efficient training and prediction.

Due to its use of both seen and unseen labels, ZestXML can be up to 14% and 10% more accurate than state-of-the-art extreme classifiers and leading BERT-based dense retrievers respectively. The extreme sparsity in its model also results in 10x smaller model size than state-of-the-art extreme classifiers. The learnt model also has the advantages of real-time label updates and explainability which are desirable in recommendation and ranking applications.

Contributions: This paper makes the following contributions:

- **GZXML**: Motivates the GZXML paradigm which is better suited for large-scale ranking and recommendation than traditional XML. Stimulates research in GZXML through source code and dataset releases.
- **ZestXML:** Proposes the ZestXML algorithm for GZXML based on a novel XHTP optimization technique for scalably learning sparse and accurate models
- **Bing Ads:** Demonstrates the practical impact of ZestXML through an A/B test in Bing Sponsored Search Ads which resulted in 17% and 3.4% gains in click yield of IR based system and unseen query coverage respectively

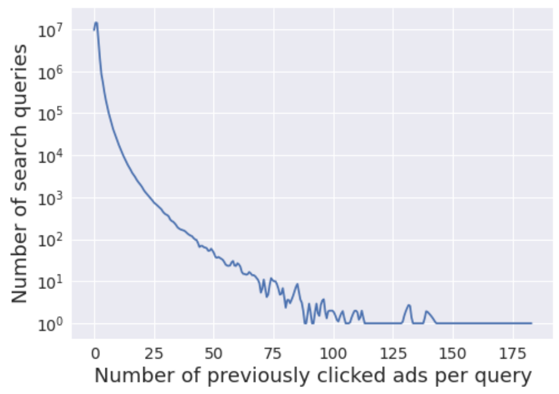

Figure 1: Plot of long query tail in Bing Advertising. Most search queries have few or no previously clicked ads.

2 RELATED WORK

Extreme multi-label learning: A plethora of extreme classifiers have been proposed which can be grouped into tree-based [1, 6, 20, 22, 26, 44–47, 53, 57, 66], embedding-based [4, 16, 21, 31, 39, 56, 67], 1-vs-all based [2, 3, 19, 28, 32, 42, 45, 55, 63, 64, 66] and deep-learning based [23, 32, 37, 37, 66] approaches. Among these, MACH [37], AttentionXML [66], Parabel [45] and Slice [19] are notable for their high prediction accuracy and scalability. MACH, Parabel and Slice maintain logarithmic training and prediction times by using efficient label search data structures based on hashes, trees and graph respectively. AttentionXML achieves state-of-the-art prediction accuracies through BiLSTM-based feature learning but it is slow to train even with high-end GPUs and does not scale to more than a few million labels. Extreme classifiers like PfastreXML [20], XReg [46] and DECAF [40] optimize for few-shot labels where the first two learn for propensity-scored losses which prioritize tail labels and the last incorporates label features for better tail label modeling. Regardless of their nature, none of these perform zero-shot label prediction. ZestXML outperforms all these approaches by up to 14% in prediction accuracy due to better modeling and access to more label choices from unseen ones.

Zero-shot multi-label learning: Although zero-shot classification is a highly researched area, most of the existing work concerns multi-class learning [15, 25, 27, 29, 33, 38, 43, 51, 59] with only a handful of multi-label learning approaches [7, 14, 17, 18, 30, 41, 50, 69]. These existing ZML algorithms are designed for 100s-1000s of labels and don't scale to 1M labels. ZestXML can be more accurate than some of the state-of-the-art ZML methods [7, 50] by up to 7% while predicting both seen and unseen labels (i.e. generalized ZML setting) in a few thousand labels regime. Although it is possible to extend some of the scalable multi-class variants like ConSE [43] to multi-label scenario, such extensions tend to have lesser accuracies.

ANNS on dense embeddings: Dense embeddings have been widely trained in many earlier XML algorithms [4, 21, 39], but they have linearly scaling training time in labels. Several dense-embedding based neural rankers [12, 52, 68] have also been proposed but they don't scale to millions of labels and hence are not pertinant here. A few neural modeling based approaches have also been proposed for prediction with millions of labels [34, 65]. Beside the limitations discussed in the introduction, the performance for dense representations is also known to decrease quickly with more labels as compared to sparse maps [49].

Other approaches: Unsupervised techniques like topic modeling [10] learn relevant topics for labels, index a label against its topics and then map a point on to topics to retrieve its labels. These usually learn and apply around 10K topics which won't generalize well to millions of labels leading to inferior accuracy. Some approaches also synthesize the relevant labels for a point, but are significantly slow to predict and known to be error-prone [62].

3 PROPOSED ALGORITHM: ZESTXML

3.1 Problem Setup

Intuition: Consider the problem of labeling a Wikipedia article on Geoffrey Hinton with relevant Wikipedia categories such as `turing award laureates` out of millions of available categories. A useful observation is that most articles associated with this label contain the phrases `turing award` and `computer scientist` which, in turn, corrrespond to the `turing award` and `laureates` parts of this label, respectively. In many XML problems, the labels can be adequately modeled through a small number of such interactions between data point-label feature pairs. For example, in Bing Ads, the relevant queries can be predicted by semantically matching the product types, brand names and other informative fields between the query and the ad text. ZestXML aims to learn a small number of highly accurate feature pairs in a scalable manner. This results in frugal models which achieve high prediction accuracy while also maintaining logarithmic inference costs and model sizes.

Notations: Given N training points and L labels, for an ith point and lth label respectively, $\mathbf{x}_i \in \mathbb{R}^C, \mathbf{z}_l \in \mathbb{R}^D$ indicate their high dimensional, sparse TF-IDF feature vectors whose sparsities are bounded by $\hat{C}$ and $\hat{D}$. These features can be expressed as $\mathbf{X}, \mathbf{Z}$ which are $C \times N$ and $D \times L$ dimensional matrices with each column representing a separate data point or a label. The ground truth relevances are indicated through binary variables $y_{il} \in \{-1, 1\}$ where $y_{il} = 1$ means that the label l is relevant to the data point i.

Formulation: ZestXML models the relevance between a data point i and a label l through linear feature interactions between $\mathbf{x}_i$ and $\mathbf{z}_l$. The relevance score s_{il} between them is expressed as

$$s_{il} = \mathbf{x}_i^\top \mathbf{W} \mathbf{z}_l \tag{1}$$

where a large (small) s_{il} value means high (low) relevance. For simplicity, the bias terms are absorbed into (1) by appending a constant feature to $\mathbf{x}_i, \mathbf{z}_l$. $\mathbf{W}$ is an $\mathbb{R}^C \times \mathbb{R}^D$-dimensional, highly sparsified, matrix of model parameters. Learning $\mathbf{W}$ involves solving a regularized logistic regression to correctly classify all training point and label pairs:

$$\begin{aligned} &\min_{\mathbf{W}} \frac{1}{2}\|\mathbf{W}\|_F^2 + \lambda \sum_{i=1}^{N} \sum_{l=1}^{L} \log(1 + e^{-y_{il}\mathbf{x}_i^\top \mathbf{W} \mathbf{z}_l}) \\ &\text{s.t.}, \|\mathbf{W}_{i*}\|_0 \le K \ \forall i \in \{1, \cdots, C\} \end{aligned} \tag{2}$$

where K, λ are hyper-parameters of the model and $\|\mathbf{W}_{i*}\|_0$ is the number of non-zeros in the ith row of $\mathbf{W}$. For ease of discussion, the individual entries in $\mathbf{W}$ will be referred to as **voters** and the non-zero entries among them as **active voters**.

In most text-based GZXML applications, N, L, C, D range in millions. The sparsity constraint is therefore enforced to ensure tractable training and prediction. As each row in $\mathbf{W}$ has only a handful of active voters ($K \approx 10$), for a given $\mathbf{x}_i$, only $\hat{C}K$ number of label features get triggered. By leveraging this property, ZestXML filters off most of the irrelevant labels which are semantically unrelated to the point during both training and prediction. Many existing XML approaches perform such negative sampling by using data structures based on hashing [42], trees [22, 45] or graphs [19] for efficient label search. This paper demonstrates that model sparsity can also be a powerful data structure for label search.

Model sparsity has been previously explored in the XML literature to selectively achieve efficient training [63, 64], better prediction accuracy [61] or tractable model size [2]. In contrast, ZestXML effectively leverages model sparsity to jointly attain all these desirable properties. Additionally, ZestXML's predictions are explainable by tracking the voters which are responsible for each label's prediction. The rest of this section presents the ZestXML's training and prediction algorithms and their computational complexities.

3.2 Training

ZestXML training aims to solve (2) highly efficiently in order to scale to millions of data points, labels and features while taking only a few hours with modest computational resources. Towards this goal, a new optimization approach named eXtreme Hard Thresholding Pursuit is proposed.

Existing approaches: The objective (2) with $L0$ constraints is non-convex and NP-hard which forbids any exact solutions which are efficient too. Alternatively, several first order [36, 54] and second order [9, 58] optimization methods have been proposed for approximately solving such problems. The former utilize only the gradient information for optimization whereas the latter methods leverage both the gradient and the hessian. Regardless of their nature, none of these techniques are able to scale to our setting where both the number of instances (NL) and parameters (CD) in (2) range in $O(10^{12})$. Note that exactly computing the gradient for (2) requires a prohibitive $O(10^{16})$ operations even with model sparsity. Moreover, materializing a full hessian or its inverse during training needs terabytes of RAM and hence is expensive.

eXtreme Hard Thresholding Pursuit: XHTP is an extremely efficient second order optimization algorithm which scales only sublinearly with both the number of instances (NL) and parameters (CD). In comparison, the existing second order solutions [9, 58] have at least a linear and quadratic dependence respectively.

The objective (2) can be reformulated as

$$\min_{\mathbf{w}} \frac{1}{2}\|\mathbf{w}\|_2^2 + \lambda \sum_{j=1}^{NL} \log(1 + e^{-b_j \mathbf{w}^\top \mathbf{a}_j}) \tag{3}$$
$$\|\mathbf{w}_{k:k+D}\|_0 \leq K \ \forall K \in \{1, \cdots, C\}$$
$$\mathbf{b} \in \{0,1\}^{NL}, \mathbf{a}_i \in \mathbb{R}^{CD}, \mathbf{w} \in \mathbb{R}^{CD}$$

where $\mathbf{w}, \mathbf{b}$ are flattened $\mathbf{W}, \mathbf{Y}$; $\mathbf{a}_j$ is flattened form of its corresponding outer product matrix $\mathbf{x}_i \mathbf{z}_l^\top$; $\mathbf{w}_{k:k+D}$ is a sub-vector of $\mathbf{w}$.

Similar to other second-order approaches, an iteration of XHTP consists of 2 successive steps: approximation and refinement. The first step approximates the objective in (2) by a quadratic form and minimizes it to obtain a sparsified solution. The second step uses the same obtained sparsity support but refines the values of non-zero parameters to better fit the original objective. The second-order methods are known to converge in only a small number of iterations. By judiciously exploiting a few key assumptions and a favourable starting point, XHTP manages to attain a highly sparsified and accurate model in just one iteration of approximation and refinement steps which further improves its efficiency.

Since the solution to (3) is expected to be extremely sparse, a meaningful starting point for optimization is $\mathbf{w}_0 = \mathbf{0}$.

Approximation step: This step computes the gradient $\mathbf{g}_0$ and the inverse hessian $\mathbf{H}_0^{-1}$ of (3) at the initialized iterate $\mathbf{w}_0$. This is followed by a newton descent step

$$\mathbf{w}_1 = \mathbf{w}_0 - \mathbf{H}_0^{-1}\mathbf{g}_0 \tag{4}$$

to obtain a new iterate $\mathbf{w}_1$. Finally, $\mathbf{w}_1$ is truncated to satisfy the sparsity constraints in (3) by retaining only K parameters with largest magnitude in each sub-vector $\mathbf{w}_{k:k+D}$. The resulting truncated output of this approximation step is denoted by $\mathbf{w}_a$.

To tractably compute $\mathbf{H}_0^{-1}$, the following assumption is made about the underlying data

- Assumption: Each feature in $\mathbf{x}_i$ and $\mathbf{z}_l$, and consequently in $\mathbf{a}_j$, are generated independently of other features.

This is a probabilistic feature independence assumption which is also used in techniques like Naive Bayes classifiers. It decouples the features and thus reduces the computational burden from $O((CD)^2)$ to $O(CD)$. Now, let e_k^+, e_k^- be the expected values of the kth feature over positive and negative instances respectively:

$$e_k^+ = \frac{\sum_{j:b_j=1} a_{jk}}{NL} \qquad e_k^- = \frac{\sum_{j:b_j=-1} a_{jk}}{NL} \tag{5}$$

The following theorem demonstrates the existence of a closed form solution for (4)

Theorem 3.1. *Given the feature independence assumption, the value of* $\mathbf{w}_1 = -\mathbf{H}_0^{-1}\mathbf{g}_0$ *takes the following form*

$$w_{1k} = \frac{4e_k^+}{e_k^+ + e_k^-} \tag{6}$$

Proof. Proof is available in supplementary material. □

The above simplification is possible due to the feature independence assumption and the fact that hessian at $\mathbf{w}_0 = \mathbf{0}$ is independent of the relevance scores (1).

The theorem suggests a simple hard thresholding scheme where the dimensions corresponding to largest values of $v(k) = \frac{e_k^+}{e_k^+ + e_k^-}$ are retained. A useful interpretation of these $v(k)$ values, for the special case of binary features, is as

$$v(k) = \mathbb{P}(b_j = 1 | a_{jk} > 0) \tag{7}$$

In other words, the approximation step retains only those voters (c, d) such that if $x_{ic} > 0, z_{ld} > 0$, then the label l is relevant to point i with high probability.

Let $\mathbf{w}_a, \mathbf{W}_a$ denote, respectively, the parameter vector obtained after thresholding and the reconstructed matrix form of the same. The approximation step can be efficiently implemented as shown in the pseudo-code in supplementary material. The computational complexity of this step is $O(N\hat{C}\hat{D}\log L)$. On Wikipedia-1M dataset with $N \approx 1M, \log L \approx 10, \hat{C} \approx 100, \hat{D} \approx 10$, the total computations are $\approx 10^{10}$ which is just a few minutes on a present-day CPU.

Refinement step: While the $\mathbf{W}_a$ solution provided by the approximation step is itself reasonably accurate from a practical standpoint (see Section 5), it can be further improved by taking a refinement step. This step is necessary to rectify the errors caused due to the quadratic approximation and the assumption of independently generated features which does not strictly hold. This stage fixes the voter sparsity pattern recovered in $\mathbf{W}_a$ and only re-learns their weights. For efficiency, it samples only a small number ($S \approx 100$) of most erroneous labels per training point through the ZestXML's efficient label shortlisting algorithm (discussed in next part) and only optimizes on these difficult point-label pairs by minimizing the original objective (2). Since the parameter matrix is sparse and the training sample is small, this step can be highly efficient with a training time of less than an hour on large datasets and complexity

$O(NS\hat{K})$ where $\hat{K}$ is the average active feature pairs for an point-label pair. Liblinear [13] library was used for this optimization. The resulting refined parameter matrix is denoted as $\mathbf{W}_r$.

3.3 Prediction

During prediction, given a test point, the most relevant labels need to be retrieved for it, *i.e.* labels with the highest relevance score (1). Since both point and label features are sparse vectors, ZestXML adopts an inverted index based search algorithm for prediction. It begins by creating an inverted index where the keys are label features and each label is indexed against its own set of features. When a test point $\mathbf{x}$ arrives, its features are first projected onto the label feature space by applying the learnt matrix $\mathbf{W}$ to get a sparse embedding $\tilde{\mathbf{x}} = \mathbf{W}^\top \mathbf{x}$. Due to the sparsity pattern in $\mathbf{x}$ and $\mathbf{W}$, this takes $O(\hat{C}K)$ operations. Now, for each non-zero dimension of $\tilde{\mathbf{x}}$, its inverted index list is perused and the scores for the indexed labels are aggregated. Finally, all resulting label scores are sorted decreasingly to get top label predictions.

While it is desirable to directly use the refined parameter matrix $\mathbf{W}_r$ for the above projection operation, this has a major drawback. Some of the inverted indices corresponding to popular label features might be extremely long and contain too many irrelevant labels. Therefore, although the point projection itself takes just $O(10^3)$ operations, the overall prediction procedure can be inefficient. To sidestep this problem, ZestXML uses a two-step prediction where $S \approx 100$ potentially relevant labels are first shortlisted by using $\mathbf{W}_a$ and then these labels are re-ranked by using $\mathbf{W}_r$ based scoring.

The above choice hinges on the observation that a provably efficient shortlisting procedure exists for $\mathbf{W}_a$ based scoring. This is based on the interpretation in (7) that $\mathbf{W}_a$ values measure the voter precision in recovering relevant labels. As a result, high (low) scoring inverted index lists contain few (many) indexed labels in them. Therefore, we can safely ignore the low scoring, lengthy lists without significantly changing the estimated label scores.

Theorem 3.2. *Let $\mathbf{x}$ be a test point, σ_x, σ_z be the bounds over $L1$ norms of $\mathbf{x}, \mathbf{z}$ respectively and ϵ be a small error tolerance parameter. Further, let $s^* = \max_l \mathbf{x}^\top \mathbf{W}_a \mathbf{z}_l$ be the score of the top-ranked label by approximate prediction. Then, an efficient algorithm exists which instead uses $\tilde{\mathbf{W}}_a$ obtained by truncating parameters smaller than ϵ and predicts, in time $O(\frac{\hat{C}\hat{D}K \log L}{\epsilon})$, a top-ranked label with score $\tilde{s}$ whose regret bounded by $s^* - \tilde{s} \leq \sigma_x \sigma_z \epsilon$.*

Proof. Proof is available in supplementary material. □

The prediction time scales as $O(\hat{C}\hat{D}K \log L)$ as suggested by the theorem. Since $logL \approx 10, \hat{C} \approx 100, \hat{D} \approx 10, K \approx 10$, prediction requires around 10^5 computations per test instance which is a few milliseconds, after discounting for ϵ, on modern CPUs.

This paper uses ZestXML-XHTP and ZestXML-tuned to refer to predictions of label shortlisting and label re-ranking stages respectively. The pseudo-code for label shortlisting algorithm is available in supplementary material.

4 SPONSORED SEARCH ADVERTISING

This paper applies ZestXML to the task of Sponsored Search Ads in Bing. This section briefly summarizes the application as well as the use of ZestXML in it.

Sponsored Search Advertising: Sponsored Search Advertising (SSA) fulfills users' need for relevant search results and advertisers' motive to direct traffic to their web pages and hence it is considered to be amongst the most important sources of revenue for search engines. The ads are retrieved from a dynamic corpus provided by large number of advertisers and shown alongside the organic search results. This corpus usually consists of advertiser hyperlinks which are annotated with keyword tags, titles, and descriptions. The advertiser provides bids on specified keywords that are valuable to capture traffic. Building efficient retrieval models is thus needed to fetch relevant advertisements/documents for user queries.

Feature-based SSA: Traditionally, ad retrieval in SSA involved directly matching a user query to an advertiser provided bid phrase. Recently, however, a query feature-based matching approach is being increasingly adopted where ads are indexed in an inverted index against the relevant query features as keys. At serving time, a user query is featurized and then searched in the inverted index to retrieve the relevant ad candidates. Inverted index is traditionally created for TF-IDF matching where the ad documents are collected, linguistic pre-processing is performed for featurization and each ad is indexed against its own features. This feature-based matching finds its relevance in especially showing documents/ads for rare or completely unseen queries since the disintegration of queries to features can lead to a good recall. This paper proposes to improve the inverted index creation by directly learning the most informative ad feature to query feature mappings corresponding to the non-zeros in the learnt sparse parameter matrix. Furthermore, during inverted index search, the associated novel prediction algorithm can be applied instead of traditional TF-IDF search resulting in more accurate and efficient retrieval. The inverted index based retrieval has several advantages over the low-dimensional dense embeddings where retrieval involves doing an ANNS (Approximate nearest neighbor search) with an expensive search data structure as discussed in section 2

XC in Ad retrieval: The ZestXML model is a natural fit for feature-based SSA and its generalized zero-shot learning abilities can help procure relevant ads for unseen queries. To deploy this in an online system of ads, the historical click logs are mined to collect ads documents and queries for which they have received a click. A search query is tagged to be relevant to an ad if there have been at least 2 historical clicks. Feature vectors of ads are created by extracting TF-IDF bag-of-words representation from the ad landing pages provided by advertisers. Feature vectors of queries are also created similarly as a TF-IDF bag-of-words vector. The features contain unigrams as well as bigrams occurring at least twice in the data. ZestXML is learned on the training data and an inverted ad-index is built over its learnt instance-label feature pairs. For every ad, the ZestXML's recommended query features are inserted into an inverted ad index and will be retrieved when the corresponding feature is matched to a query being entered in the search engine.

Table 1: Dataset Statistics.

Dataset	Num points		Num features		Num labels	
	Train	Test	Point	Label	Seen	Unseen
EURLex-4.3K	45,000	6,000	100,000	24316	4,108	163
Amazon-1M	914,179	1,465,767	1,000,000	1,476,381	476,381	483,725
Wikipedia-1M	2,271,533	2,705,425	2,000,000	1,438,196	495,107	776,612
Ads-31M	6,989,516	4,512,480	2,813,614	3,254,839	21,950,748	9,525,826

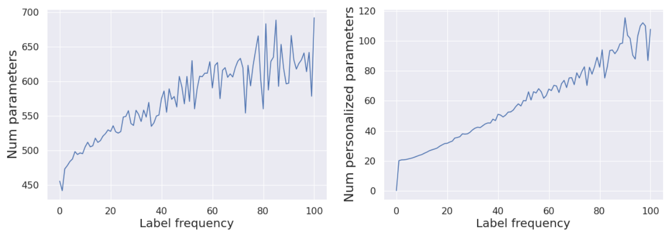

Figure 2: Number of parameters vs frequency of label

ZestXML's scoring function is an effective alternative to naive TF-IDF or BM25 scoring and can help improve the efficacy of retrieval.

5 EXPERIMENTS

5.1 Experiment Settings

Datasets and features: Experiments were carried out on large-scale datasets with up to 7 million training points and 31 million labels comprising both seen and unseen (*i.e.* zero-shot) labels (see Table 1 for dataset statistics). The applications considered include document tagging where a web-page is assigned with descriptive tags (EURLex-4.3K [7],Wikipedia-1M), item-to-item recommendation on Amazon where a product is recommended its frequently co-purchased products (Amazon-1M), and advertising where an ad landing page is assigned with its relevant search queries to be shown against (Ads-31M). The following procedure is used for generating all datasets: The training and test data were generated to belong to two successive non-overlapping time periods to mimic zero-shot learning in real applications. A 5% of test data was kept aside for hyper-parameter tuning. Most baseline approaches use the same bag-of-ngrams ($n \leq 2$) TF-IDF feature representation for both data points and labels. All datasets, apart from the proprietary Ads-31M, will be publically released.

Baseline algorithms: ZestXML was compared against leading algorithms from following 4 classes of methods :

(1) State of the art extreme classifiers such as AttentionXML[66], Astec[11], DiSMEC[2], Parabel[45] and Bonsai[26]
(2) Extreme classifiers which improve performance on few-shot labels such as DECAF[40], XReg[46] and PFastreXML[20]
(3) Dense retrieval methods based on the state of the art natural language modelling architectures such as Sentence BERT bi-encoder[48], Fasttext[24] and WarpLDA (topic model)[10], these algorithms provide strong scalable baseline to compare ZestXML's performance over zero-shot and few-shot labels
(4) Leading zero-shot multi-label learners such as 0-BIGRU-WLAN, 0-CNN-LWAN[50] and CoNSE [43], these baselines don't scale on extreme datasets, hence, ZestXML's comparison against these baselines is reported only for EURLex-4.3K in Table 5.

The implementation of all the aforementioned algorithms were provided by their authors. The hyper-parameters of all baseline algorithms were set as suggested by their authors wherever applicable and by fine grained validation otherwise. The prediction for the Topic Model was performed using a scalable, map-reduce based implementation of Blei inference [5]. Blei inference was used because it gave better results on our datasets. All baselines have been run on only on those datasets on which it trains within a week and predicts within few days. ANNS on dense embeddings has been carried out using HNSW[35] for Sentence BERT and Fasttext.

Evaluation metrics: Prediction accuracies are primarily measured using widely used propensity-scored precision@K(PSP) evaluation metric. Propensity scores for zero-shot labels are extrapolated according to standard propensity-score formula [20].

Table 2 reports PSP@1, 3, 5 metrics for Generalized ZSL task on ZestXML and baselines. The section **??** provides evaluation numbers for all datasets and baseline with respect to standard precision@K(P) as well. For a predicted score vector $\hat{\mathbf{y}} \in R^L$ and ground truth vector $\mathbf{y} \in \{0,1\}^L$, $P@k = \frac{1}{k}\sum_{l \in rank_k(\hat{\mathbf{y}})} y_l$, $PSP@k = \frac{1}{k}\sum_{l \in rank_k(\hat{\mathbf{y}})} \frac{y_l}{p_l}$. Here, p_l is propensity score of the label l proposed in [20].

Hyper parameters: ZestXML uses three tunable hyper parameters : (1) Model sparsity K (2) shortlist size S (3) common regularization parameters for ZestXML-tuned optimization λ. $\lambda = 1$ was found to work consistently well for all datasets. S impact the recall values of the shortlist which dictates the overall quality of label shortlist generated for ZestXML-tuned and therefore should be maximized for good overall performance. As demonstrated in figure 4, with larger S recall continues to improve but by smaller amounts. A shortlist size of $S = 100$ seemed adequate for all the datasets without incurring too much prediction cost. K dictates the number of learnable parameters in the model and can impact the final performance of the model. Figure 4 plots the variation in final PSP@5 with variation in K, we observe that with increasing K model performance improves but with diminishing gains. We choose K between 10 to 40 for all datasets.

Hardware: Training and prediction for all CPU based methods were done on a 110 GB RAM intel Xeon processor machine. All GPU based baselines were run on 4 NVIDIA TESLA P100 GPUs. Figure 3 reports training and test times normalized for single core for CPU based methods and single GPU for GPU based methods.

5.2 Results

Results on EURLex-4.3K: ZestXML is among the best performing methods on EURLex dataset in terms of PSP@5, although on PSP@1 traditional extreme classifiers are better than ZestXML. Note that, on EURLex all of the baselines which rely on label features to make predictions(DECAF, Bert-ANNS, Fasttext-ANNS) perform poorly as compared to traditional extreme classifiers. This is primarily because label features on EURLex are not very informative and many times lead to noise in predictions. The smaller EURLex dataset allows comparison to standard zero-shot multi-label learners(Zero-BIGRU-LWAN, Zero-CNN-LWAN, ConSE) which don't scale to

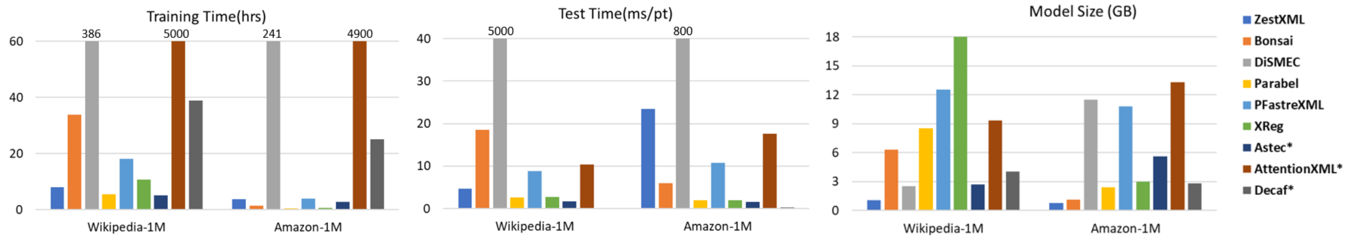

Figure 3: ZestXML learns most compact model on both Wikipedia-1M and Amazon-1M datasets and is comparable to most scalable extreme classification methods w.r.t training/prediction time (* marked methods were run on GPU)

Table 2: ZestXML achieves the highest or close to the highest PSP for Generalized Zero-Shot(G.ZSL) task among other XML and dense ANNS baselines on all datasets

Algorithm	G.ZSL PSP@1	PSP@3	PSP@5	Algorithm	G.ZSL PSP@1	PSP@3	PSP@5	Algorithm	G.ZSL PSP@1	PSP@3	PSP@5
EURLex-4.3K				Amazon-1M				Wikipedia-1M			
ZestXML-tuned	48.01	60.29	66.15	ZestXML-tuned	21.58	**30.62**	**36.79**	ZestXML-tuned	**14.43**	**15.80**	**17.31**
ZestXML-XHTP	36.32	40.05	42.9	ZestXML-XHTP	18.56	26.94	33.39	ZestXML-XHTP	1.5	2.54	3.68
AttentionXML	53.92	**63.59**	**67.85**	AttentionXML	10.18	13.76	15.91	AttentionXML	3.82	4.54	5.20
Astec	50.32	59.14	63.00	Astec	10.32	13.97	16.44	Astec	2.66	2.74	2.99
Decaf	40.68	46.96	49.89	Decaf	11.91	16.26	19.09	Decaf	3.48	3.97	4.48
Parabel	46.82	58.8	64.29	Parabel	9.06	12.38	14.29	Parabel	2.99	3.32	3.65
DiSMEC	47.26	59.82	65.55	DiSMEC	10.46	14.57	17.13	DiSMEC	2.35	2.99	3.48
Bonsai	46.41	58.83	64.44	Bonsai	9.88	13.68	15.93	Bonsai	3.19	3.61	4.05
XReg	**58.06**	62.99	65.97	XReg	9.87	13.25	15.48	XReg	3.48	3.51	3.83
PfastreXML	55.30	58.00	59.91	PfastreXML	7.76	10.39	12.14	PfastreXML	2.97	2.90	3.10
FastText ANNS	17.10	15.74	16.13	FastText ANNS	16.80	20.63	23.64	FastText ANNS	7.16	6.01	6.19
Bert ANNS	4.64	3.66	3.57	Bert ANNS	**22.97**	29.69	34.95	Bert ANNS	10.34	8.17	8.20
Topic Model	6.85	6.14	6.13	Topic Model	1.98	1.96	1.95	Topic Model	1.88	1.93	2.19

Table 3: Comparing the shortlisting performance of ZestXML XHTP with popular IR samplers TF-IDF, BM25

Sampler	Recall@10	Recall@50	Recall@100
ZestXML-XHTP	39.77	54.61	60.05
TF-IDF	31.08	44.70	50.65
BM25	33.62	46.61	52.25

larger datasets. ZestXML achieves the best accuracies on both seen and unseen labels when compared to such methods (see Table 5).

Results on large datasets: On large datasets, Table 2 demonstrates that ZestXML outperforms most of the leading extreme classifiers, dense embedding based ANNS approaches and topic model on the GZXML task. ZestXML consistently outperforms dense ANNS by at most 10% and can be upto 16% better than extreme classifiers at PSP@5. In terms of P@5, ZestXML is the best performing method on Amazon-1M and Ads-31M by a margin of 2.5% and 11% respectively and is second to only AttentionXML on Wikipedia-1M where seen labels play a significant role in predictions. However, AttentionXML is around 50x slower to train and has 10x larger model size than ZestXML beside requiring expensive GPUs for training and prediction. To summarize, ZestXML consistently outperforms most baselines in diverse recommendation scenarios whereas all the existing baselines fail significantly in one or the other scenarios.

Complexity: ZestXML learns a more compact models across all datasets than existing extreme classification methods as demonstrated in Figure 3. Moreover, ZestXML is highly scalable even on a single thread of a standard desktop due to efficient training and its prediction time is consistently about a few milliseconds across datasets. On the largest dataset with 31 million labels (Ads-31M), ZestXML trains in just 30 hours on single core and predicts in ~20 ms latency which is desirable for practical use.

ZestXML is also capable of carefully allotting its voters or model parameters as per the needs of different labels. As seen from Figure 2, more voters are alloted to popular labels which have more data to learn from. At the same time, ZestXML shares many voters for predicting tail labels accurately through data pooling while also reserving voters for personalization in head labels. In contrast, existing XML approaches attempt to personalize models for all labels and hence suffer on tail labels.

Performance on rare labels: ZestXML performs better than all baselines on rare labels as shown in Figure 5, which plots PSP@5 of different methods in zero-shot and few-shot label regime on Amazon-1M dataset. Existing extreme classification baselines make no predictions for zero-shot labels and perform poorly on few-shot

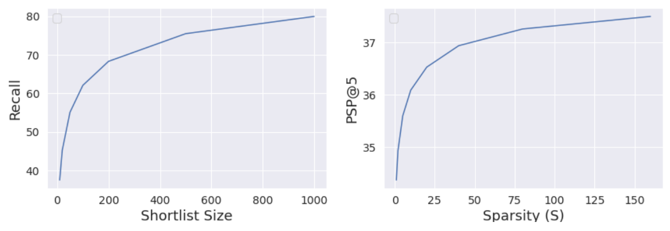

Figure 4: Hyperparameter ablation

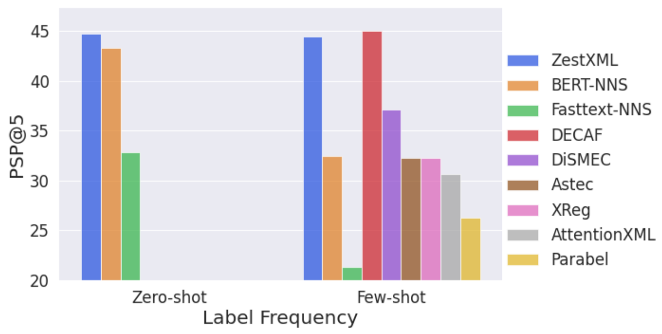

Figure 5: ZestXML consistently outperforms the baselines across both zero shot and few shot labels on Amazon-1M

labels since not much training data is available for such labels, although recently proposed DECAF algorithm (which utilises a combination of per-label classifier and label text embedding) performs on par with ZestXML on few-shot labels. BERT and Fasttext embedding based ANNS approaches perform reasonably well on zero-shot labels but ZestXML significantly outperforms them on few-shot labels.

Sampler choice: Table 3 shows the superior shortlisting performance of ZestXML-XHTP as compared to popular IR based text-matchers like TF-IDF and BM25 on Amazon-1M dataset. ZestXML-XHTP is better by an absolute margin of 10% and 8% than TF-IDF and BM25 respectively on Recall@10, 50, 100. This demonstrates that ZestXML-XHTP brings many useful cross features which TF-IDF and BM25 fail to capture.

Application to computational advertising: To test the efficacy of the ZestXML in uncovering missing recall, we deployed it in a system consisting of ensemble of algorithms ranging from neural information retrieval, graph and IR based techniques. We performed online A/B testing on Bing sponsored search engine in order to evaluate the results. One of the fundamental objective is to maximize the ad clicks generated, hence we evaluate the metric click-yield (CY) for the algorithm, which can be defined as the number of clicks per 1000 queries searched. The deployment of ZestXML led to significant gains and helped in boosting the contribution of CY of IR based system by 17%. The algorithm also helped expanding the coverage to completely unseen queries by a factor of 3.4%.

Qualitative results: ZestXML also provides useful explanations to its label predictions based on the most contributing voters. In product recommendation applications, where explainable predictions are useful, this can be leveraged. The figure in supplementary materialpresents the top item recommendations by ZestXML on Amazon for a few test points and also shows the top voters (instance-label feature pairs) responsible for these predictions. The green ticks indicate that the predictions are relevant to the user as judged by user clicks. Examples of the useful explanations that ZestXML generate include (1) predicting book recommendations by the same author that user is viewing (Jordan Rubin); (2) predicting books on same topic that the user is viewing (carpentry).

Table 4: Comparison of ZestXML with other dense ANNS and XML algorithms on proprietary Bing Ads-31M dataset

Algorithm	G.ZSL PSP@1	G.ZSL PSP@3	G.ZSL PSP@5
	Ads-31M		
ZestXML-tuned	**15.22**	**20.24**	**22.01**
ZestXML-XHTP	9.97	15.97	18.89
Parabel	2.16	3.09	3.53
Xreg	2.89	3.96	4.43
FastText ANNS	4.78	6.94	8.16
Bert ANNS	6.79	9.57	11.15
Topic Model	1.41	2.23	2.87

Table 5: Comparison of ZestXML with zero-shot multi label algorithms on EURLex-4.3K

Algorithm	P@5 Seen+Unseen	P@5 Unseen
	EURLex-4.3K	
ZestXML-tuned	**69.5**	**12.81**
ConSE-FastText	34.71	4.27
0-BIGRU-LWAN	61.90	9.30
0-CNN-LWAN	58.90	6.90

6 CONCLUSION

This paper studied the problem of zero-shot labels in extreme classification and developed a novel ZestXML classifier capable of generalized zero-shot predictions. ZestXML scalably learns a highly sparsified linear transform between the data point's and label's features through an innovative optimization technique. ZestXML can be up to 14% and 10% more accurate at generalized zero-shot learning relative to leading extreme classifiers and dense embedding search approaches. Deploying ZestXML in Sponsored Search Advertising on Bing improved the click yield and unseen query coverage by 17% and 3.4% respectively. ZestXML's source code and benchmark datasets for zero-shot extreme classification will be released for research purposes.

REFERENCES

[1] 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*.
[2] R. Babbar and B. Schölkopf. 2017. DiSMEC: Distributed Sparse Machines for Extreme Multi-label Classification. In *WSDM*.
[3] R. Babbar and B. Schölkopf. 2019. Data scarcity, robustness and extreme multi-label classification. *ML* (2019).
[4] K. Bhatia, H. Jain, P. Kar, M. Varma, and P. Jain. 2015. Sparse Local Embeddings for Extreme Multi-label Classification. In *NIPS*.
[5] D. M. Blei, A. Y. Ng, and M. I. Jordan. 2003. Latent dirichlet allocation. *JMLR* 3, Jan (2003), 993–1022.
[6] Chang W. C., Yu H. F., Zhong K., Yang Y., and Dhillon I. S. 2019. A Modular Deep Learning Approach for Extreme Multi-label Text Classification. *CoRR* (2019).

[7] I. Chalkidis, M. Fergadiotis, P. Malakasiotis, N. Aletras, and I. Androutsopoulos. 2019. Large-Scale Multi-Label Text Classification on EU Legislation. In *ACL*. 6314–6322.

[8] W. C. Chang, H. F. Yu, K. Zhong, Y. Yang, and I.S. Dhillon. 2020. Taming Pretrained Transformers for Extreme Multi-label Text Classification. (August 2020).

[9] J. Chen and Q. Gu. 2017. Fast Newton Hard Thresholding Pursuit for Sparsity Constrained Nonconvex Optimization. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 757–766.

[10] J. Chen, K. Li, J. Zhu, and W. Chen. 2016. WarpLDA: a Cache Efficient O(1) Algorithm for Latent Dirichlet Allocation. In *VLDB*.

[11] K. Dahiya, D. Saini, A. Mittal, K. Dave, H. Jain, S. Agarwal, and M. Varma. 2021. DeepXML: A Deep Extreme Multi-Label Learning Framework Applied to Short Text Documents. *WSDM to appear* (2021).

[12] Mostafa Dehghani, Hamed Zamani, Aliaksei Severyn, Jaap Kamps, and W. Bruce Croft. 2017. Neural Ranking Models with Weak Supervision. In *SIGIR*. 65–74.

[13] R. E. Fan, K. W. Chang, C. J. Hsieh, X. R. Wang, and C. J. Lin. 2008. LIBLINEAR: A library for large linear classification. *JMLR* (2008).

[14] A. Gaure, A. Gupta, V. K. Verma, and P. Rai. 2017. A probabilistic framework for zero-shot multi-label learning. In *UAI*, Vol. 1. 3.

[15] J. Guan, A. Zhao, and Z. Lu. 2018. Extreme Reverse Projection Learning for Zero-Shot Recognition. In *ACCV*. Springer, 125–141.

[16] C. Guo, A. Mousavi, X. Wu, D. N. Holtmann-Rice, S. Kale, S. Reddi, and S. Kumar. 2019. Breaking the Glass Ceiling for Embedding-Based Classifiers for Large Output Spaces. In *NeurIPS*.

[17] B. Hariharan, S. V. N. Vishwanathan, and M. Varma. 2012. Efficient max-margin multi-label classification with applications to zero-shot learning. *Machine learning* 88, 1-2 (2012), 127–155.

[18] D. Huynh and E. Elhamifar. 2020. A shared multi-attention framework for multi-label zero-shot learning. In *CVPR*. 8776–8786.

[19] 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*.

[20] H. Jain, Y. Prabhu, and M. Varma. 2016. Extreme Multi-label Loss Functions for Recommendation, Tagging, Ranking and Other Missing Label Applications. In *KDD*.

[21] A. Jalan and P. Kar. 2019. Accelerating Extreme Classification via Adaptive Feature Agglomeration. *IJCAI* (2019).

[22] 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*.

[23] Y. Jernite, A. Choromanska, and D. Sontag. 2017. Simultaneous Learning of Trees and Representations for Extreme Classification and Density Estimation. In *ICML*.

[24] A. Joulin, E. Grave, P. Bojanowski, and T. Mikolov. 2017. Bag of Tricks for Efficient Text Classification. In *Proceedings of the European Chapter of the Association for Computational Linguistics*.

[25] Singh R. Keshari, R. and M Vatsa. 2020. Generalized Zero-Shot Learning Via Over-Complete Distribution. *arXiv preprint arXiv:2004.00666* (2020).

[26] S. Khandagale, H. Xiao, and R. Babbar. 2019. Bonsai - Diverse and Shallow Trees for Extreme Multi-label Classification. *CoRR* (2019).

[27] V. Kumar Verma, G. Arora, A. Mishra, and P. Rai. 2018. Generalized zero-shot learning via synthesized examples. In *CVPR*. 4281–4289.

[28] A. Kusupati, M. Singh, K. Bhatia, A. Kumar, P. Jain, and M. Varma. 2018. FastGRNN: A Fast, Accurate, Stable and Tiny Kilobyte Sized Gated Recurrent Neural Network.. In *NeurIPS*.

[29] Y. Le Cacheux, H. Le Borgne, and M. Crucianu. 2019. From classical to generalized zero-shot learning: A simple adaptation process. In *International Conference on Multimedia Modeling*. Springer, 465–477.

[30] Lee, C. W., W. Fang, Yeh, C. K., Frank Wang, and Yu-Chiang. 2018. Multi-label zero-shot learning with structured knowledge graphs. In *CVPR*. 1576–1585.

[31] Z. Lin, G. Ding, M. Hu, and J. Wang. 2014. Multi-label Classification via Feature-aware Implicit Label Space Encoding. In *ICML*.

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

[33] S. Liu, M. Long, J. Wang, and M. I. Jordan. 2018. Generalized zero-shot learning with deep calibration network. In *NIPS*. 2005–2015.

[34] W. Lu, J. Jiao, and R. Zhang. 2020. TwinBERT: Distilling Knowledge to Twin-Structured Compressed BERT Models for Large-Scale Retrieval. In *CIKM*. 2645–2652.

[35] Y. A. Malkov and D. A. Yashunin. 2018. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. *IEEE transactions on pattern analysis and machine intelligence* (2018).

[36] S. Mallat and Z. Zhang. 1993. Matching pursuits with time-frequency dictionaries. *IEEE Transactions on Signal Processing* (1993), 3397–3415.

[37] T. K. R. Medini, Q. Huang, Y. Wang, V. Mohan, and A. Shrivastava. 2019. Extreme Classification in Log Memory using Count-Min Sketch: A Case Study of Amazon Search with 50M Products. In *Neurips*.

[38] T. Mensink, E. Gavves, and C. G. Snoek. 2014. Costa: Co-occurrence statistics for zero-shot classification. In *CVPR*. 2441–2448.

[39] P. Mineiro and N. Karampatziakis. 2015. Fast Label Embeddings via Randomized Linear Algebra. In *Joint European conference on machine learning and knowledge discovery in databases*.

[40] A. Mittal, K. Dahiya, D. Saini, S. Agarwal, P. Kar, and M. Varma. 2021. DECAF: Deep Extreme Classification with Label Features. *WSDM to appear* (2021).

[41] J. Nam, E. L. Mencía, H. J. Kim, and J. Fürnkranz. 2015. Predicting unseen labels using label hierarchies in large-scale multi-label learning. In *ECML PKDD*. Springer, 102–118.

[42] A. Niculescu-Mizil and E. Abbasnejad. 2017. Label Filters for Large Scale Multilabel Classification. In *AISTATS*.

[43] M. Norouzi, T. Mikolov, S. Bengio, Y. Singer, J. Shlens, A. Frome, G. S. Corrado, and J Dean. 2013. Zero-shot learning by convex combination of semantic embeddings. *arXiv preprint arXiv:1312.5650* (2013).

[44] 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*.

[45] Y. Prabhu, A. Kag, S. Harsola, R. Agrawal, and M. Varma. 2018. Parabel: Partitioned label trees for extreme classification with application to dynamic search advertising. In *WWW*.

[46] Y. Prabhu, A. Kusupati, N. Gupta, and M. Varma. 2020. Extreme Regression for Dynamic Search Advertising. *WSDM* (2020).

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

[48] N. Reimers and I. Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *EMNLP-IJCNLP* (2019).

[49] N. Reimers and I. Gurevych. 2020. The Curse of Dense Low-Dimensional Information Retrieval for Large Index Sizes. *arXiv preprint arXiv:2012.14210* (2020).

[50] A. Rios and R. Kavuluru. 2018. Few-shot and zero-shot multi-label learning for structured label spaces. In *EMNLP*, Vol. 2018. 3132.

[51] Romera-Paredes, B., and P. Torr. 2015. An embarrassingly simple approach to zero-shot learning. In *ICML*. 2152–2161.

[52] Y. Shen, X. He, J. Gao, L. Deng, and G. Mesnil. 2014. Learning Semantic Representations Using Convolutional Neural Networks for Web Search. In *Proceedings of the International Conference on World Wide Web (WWW '14 Companion)*.

[53] S. Si, H. Zhang, S. S. Keerthi, D. Mahajan, I. S. Dhillon, and C. J. Hsieh. 2017. Gradient Boosted Decision Trees for High Dimensional Sparse Output. In *ICML*. 3182–3190.

[54] J. Tropp and A. Gilbert. 2007. Signal Recovery From Random Measurements Via Orthogonal Matching Pursuit. *IEEE Transactions on Information Theory* 53 (2007), 4655–4666.

[55] T. Wei and Y. F. Li. 2018. Does Tail Label Help for Large-Scale Multi-Label Learning. In *IJCAI*.

[56] J. Weston, S. Bengio, and U. Nicolas. 2011. Wsabie: Scaling up to large vocabulary image annotation. In *IJCAI*.

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

[58] Y. Xiao-Tong and L. Qingshan. 2014. Newton Greedy Pursuit: A Quadratic Approximation Method for Sparsity-Constrained Optimization. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*. 4122–4129.

[59] S. Xie and S. Y. Philip. 2017. Active zero-shot learning: A novel approach to extreme multi-labeled classification. *International Journal of Data Science and Analytics* 3, 3 (2017), 151–160.

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

[61] C. Xu, D. Tao, and C. Xu. 2016. Robust Extreme Multi-label Learning. In *KDD*.

[62] N. Yadav, R. Sen, D. N. Hill, A. Mazumdar, and I. S. Dhillon. 2020. Session-Aware Query Auto-completion using Extreme Multi-label Ranking. *arXiv preprint arXiv:2012.07654* (2020).

[63] 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*.

[64] E.H. I. Yen, X. Huang, K. Zhong, P. Ravikumar, and I. S. Dhillon. 2016. PD-Sparse: A Primal and Dual Sparse Approach to Extreme Multiclass and Multilabel Classification. In *ICML*.

[65] Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L. Hamilton, and Jure Leskovec. 2018. Graph Convolutional Neural Networks for Web-Scale Recommender Systems. In *KDD*. 974–983.

[66] 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*.

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

[68] H. Zamani, B. Mitra, X. Song, N. Craswell, and S. Tiwary. 2018. Neural Ranking Models with Multiple Document Fields. *WSDM* (2018).

[69] Y. Zhang, B. Gong, and M Shah. 2016. Fast zero-shot image tagging. In *CVPR*. 5985–5994.