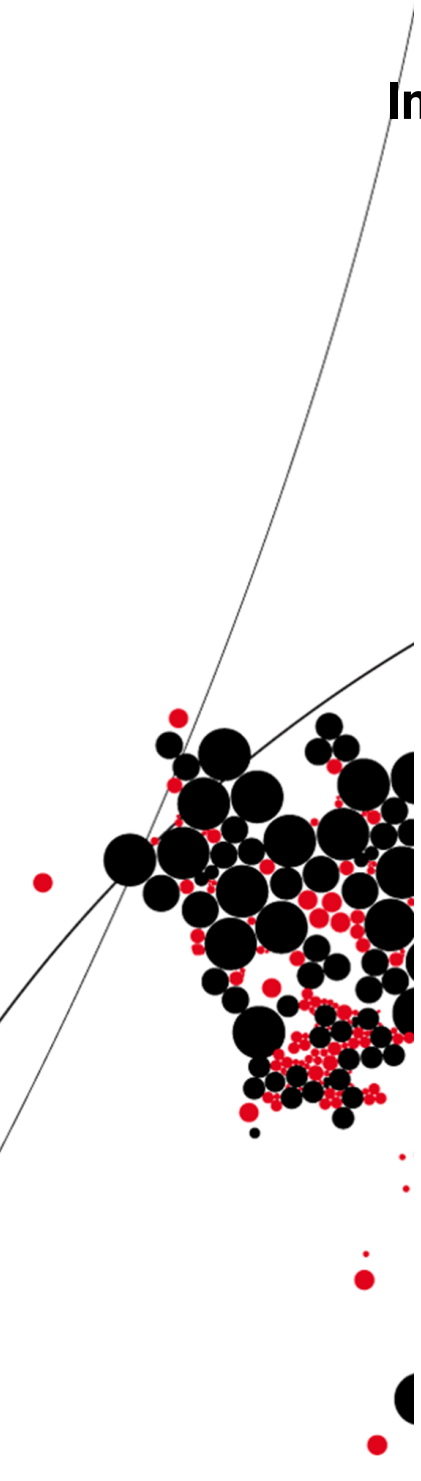
UNIVERSITY OF TWENTE.

Faculty of Electrical Engineering, Mathematics & Computer Science

Improving Extreme Multi-Label Text Classification with Sentence Level Prediction

Ashish Singhal

M.Sc. Final Thesis
August 2022



Supervisors:
dr. Christin Seifert
dr.ing. Gwenn Englebienne
dr. Shenghui Wang

Telecommunication Engineering Group
Faculty of Electrical Engineering,
Mathematics and Computer Science
University of Twente
P.O. Box 217
7500 AE Enschede
The Netherlands

Acknowledgements

This thesis marks the end of my master's journey where I feel I have outgrown myself in various areas and have become a better individual professionally and personally. I would like to express my gratitude to all of them for playing a part in my journey and helping me in improving myself.

I would like to express my gratitude to my supervisors who helped me in growing academically and personally during the thesis. I would like to start by thanking Dr.Shenghui Wang and Dr.Gwenn Englebienne for giving me the opportunity to work on this research problem. I liked the discussions in weekly meetings, which helped me increase my critical thinking about the problems. Their insights really helped me in my thesis. I appreciate their patience and support in the face of mistakes, as well as their enthusiasm in the face of successes, whether excellent or poor. I would also like to thank Dr.Christin Seifert for giving valuable feedback in crucial times that helped in shaping the thesis report.

Next, I would like to thank my parents and family without whom I would never be where I am. They have supported my decisions throughout my life and have been an inspiration to me. Their impeccable guidance at various important life junctions have put me in the right direction.

Summary

The Extreme Multi-Label Text Classification (XMTC) problem aims to assign a small number of relevant labels to document text from a large label space. XMTC label spaces follow a power law distribution, that results in data sparsity for tail labels and aggressive prediction of head labels. Existing methods for tackling XMTC problems have utilized the whole document text to predict relevant labels. This project attempts to identify and use meaningful sentences of document text to predict relevant labels. Relevant labels are predicted for the sentences and they are empirically concatenated to form relevant labels set for the document. This method is based on the idea that not all text of a document is informative of the relevant labels. Whenever whole document text is used, informative text is often get polluted with noisy text which hampers the performance. Instead, predicting relevant labels for the sentences can facilitate augmented focus on the informative text, and more relevant and tail labels can be predicted. This project also explores the idea of using focal loss in XMTC problems with label propensities to overcome the influence of power law distribution and treat every label equally.

Contents

List of acronyms

XMC	Extreme Multi-Label Classification
XMTC	Extreme Multi-Label Text Classification
BCE	Binary Cross Entropy Loss
SLEEC	Sparse Local Embeddings for Extreme multi-label Classification
DXML	Deep Extreme Multi-Label Learning
LPSR	Label Partitioning by Sub-linear Ranking
nDCG	Normalized Discounted Cumulative Gain
CNN	Convolutional Neural Network
BoW	Bag of Words
XML-CNN	Extreme Multi-Label Learning - Convolutional Neural Network
GNN	Graph Neural Networks
BERT	Bidirectional Encoder Representations from Transformers
HIBERT	Hierarchical Bidirectional Encoder Representations from Transformers
LSTM	Long Short term Memory
BiLSTM	Bidirectional Long Short term Memory
LDA	Latent Dirichlet Allocation
LSA	Latent Semantic Analysis
PLSA	Probabilistic Latent Semantic Analysis
DISMEC	Distributed Sparse Machines for Extreme multi-Label Classification
SMOTE	Synthetic Minority Oversampling Techniques

Chapter 1

Introduction

1.1 Problem Statement and Challenges

Extreme Multi-Label Classification (XMC) is a machine learning problem of assigning an item a relevant subset of labels from an extremely large set of labels [1]. Number of labels are often of the order of 10^4 and sometimes even in 10^6 order which makes predicting relevant labels quite challenging. The labels in XMC problems follow a power-law distribution which poses its foremost challenge in predicting rare labels. Often due to power law distribution of labels, a machine learning model becomes biased towards frequent labels while ignoring more informative rare labels. A lot of research has been taken to overcome this challenge in XMC as it has highly impactful real-world applications in online ad-recommendations [20] [11], product classification in ecommerce [57] [63] [53] and large scale classification of images and text for their hashtags recommendation for social media [65]. XMC also introduces, including scalability issues due to enormous label spaces, data sparsity issues due to inadequate training samples for seldomly appearing labels, severe class imbalance among labels, etc. XMC is a supervised machine learning task. Supervised classification problems involves identifying the class of an item, for example, identifying whether a given image is a bicycle or a car. These problems involve datasets consisting of labelled instances of the items being classified (like a series of labelled pictures of bicycle and car) which is then used to train the classifier to learn from these labelled examples and correctly identify the classes for unseen examples. Based on the number of classes to which an observation might belong and the total number of potential classes, classification is further divided into three types: binary, multi class and multi-label. Multi-class classification and multi-label classification are not the same as XMC. In multi-class classification, a single label is predicted from a series of mutually exclusive labels i.e., one and only one label should be associated with each item.The goal of multi-label classification is to predict all relevant labels from a small number of labels that are not mutually exclusive. XMC is most comparable with multi-label classification. With the proviso that the label spaces in XMC-style problems can reach vast proportions, XMC is most comparable to multi-label classification. XMC is a one-of-a-kind challenge that is best described as a multi-class, multi-label problem with a label space ranging from thousands to millions of labels per data point. When the application domain is text, it is called as Extreme Multi-Label Text Classification (XMTC). For instance, assigning relevant wiki tags to Wikipedia articles, chosen from a label space of over a million unique labels making the process of selecting a limited relevant collection of labels significantly more challenging.

1.2 Goal

There has been a lot of research in the XMTC domain, with many methodologies such as Label Embedding Methods [16] [14] [48], Tree-based methods [20] [24], Deep Learning methods, etc [67] [33]. These methods took documents as inputs and attempts to assign relevant subset of labels. These methods find the relationships between the document text and labels. This experiment is focused on learning relationships between the sentences of the documents and the labels and find relevant labels for the documents. Documents contain unimportant content or noise which is considered in previous methods while model training and labels prediction. In practice, humans annotate documents with the relevant labels based on either certain keywords in the documents or semantic meaning of the whole document. It implies humans tends to ignore noisy content from the documents for label assignment since noise is unrelated with any labels. Hence, with the presence of noise in the documents it becomes vital to not predict labels based on whole document content. But previous studies done in XMTC have always tried to learn a correlation between the documents text and the labels. Additionally, humans focusing on keywords also leads to only certain labels getting assigned. Human annotators often focuses only on certain keywords which are seen more frequent than others. Similarly, annotators tends to remember frequent occurring labels for annotation due to which only these few labels get assigned for most of the time [24]. Labels getting assigned are often general labels (e.g., disease) which are not quite informative at all about the documents when document is about a specific disease like Angelman Syndrome. The Angelman Syndrome label though a informative label does not get assigned as often as general labels due to being more specific and harder to remember. This leads to a power law distribution in the labels' assignment frequency where few labels are assigned with most of the documents and most of the labels are assigned rarely. There are around 40% of labels that does not get even assigned to documents [51].

Recent works in XMTC have included attention mechanisms [48] [47] [40] focusing on finding the relationship between keywords and labels. These keywords are extracted from the document and attention scores are computed among all combinations of keywords and labels. These method intuitively consider the whole document text for keyword extraction. These approaches ignore the semantic meaning of the sentences keywords belongs to. Keywords' meaning changes based on the sentences it is used in. Hence, rather than considering keywords individually the whole sentence should be considered to compute attention scores between labels and sentences. Consequently, attention scores need to be computed with the semantic representation of the sentences and relationships between the labels and sentences could be found. Additionally, attention based methods convert documents into fixed size semantic embeddings. Such fixed embeddings for a large document lead to information loss as a huge lengthy text is confined to a fixed size vector. Instead, if semantic embeddings of individual sentences are computed and utilized it would provide the model with more granular information present in the sentences with no information loss as in document embeddings case.

All these reasons motivated the prediction of labels against the sentences of the documents rather than the document itself. Intuitively, considering sentences separately for label predictions tends to give more rare and correct labels as each sentence could have a different granular semantic information relating to different set of labels when compared with another sentence. The model is trained with the sentences rather than the documents. In the end, labels predicted against the sentences of a document are combined in an empirical fashion to form a label set corresponding to the document. The challenge in this approach is to obtain ground truth at sentence levels for training. Typically, the

ground truth is readily available against the documents as human annotators assign labels to the documents. The sentences cannot be assigned labels from document's ground truth randomly, but it needs a relevant semantic method to assign labels to sentences.

The seminal paper of using deep learning in XMTC [33] has proposed to use the binary cross entropy loss function for the training. In the same lines, later works based on this seminal paper have also used binary cross entropy loss function. However, the BCE loss function is more suited for plain multi-label problems, where label assignment is balanced. As stated earlier, Extreme Multi-Label Text classification has a huge label space following a power law distribution. Due to this high imbalance in label assignment BCE tends to get biased towards frequent labels [33]. Data imbalance could be avoided by balancing the dataset so that each label has approximately equal number of data points for training. Balancing dataset often needs either the generation of synthetic data points for less frequent labels or removing data points for frequent labels. Both of these lead to data quality issues. Generating synthetic information for the huge label space introduces noise in the dataset hampering dataset quality. Whereas removing data points leads to loss of information. The better approach to tackle such huge data imbalance is to weigh down frequent labels and boost rare labels data points while training the model. Focal Loss [56] is the loss function which executes this approach. It is introduced for object recognition in images with huge dense scene. Focal loss tends to focus on required objects more than the unwanted negative objects in the background of images. A similar intuition can be extended in the text domain of XMC as well. Focal loss allows to provide weights to the labels while computing losses. Rare labels could be provided with higher weights than frequent labels making losses large for rare labels and leading the model model to focus more on rare labels. Propensity values, introduced in [24], of the labels can be considered as the weights of the labels in Focal Loss. Propensity value is a measure of how rare a label is. The reciprocal of propensity values can serve suitable weights for each label in focal loss. Quite interestingly, focal loss has never been used in XMTC settings.

With this background, the motivating research questions for this project are defined as:

- **RQ 1:** Does focal loss with propensity as weights improves predictive performance in XMTC setting?
- **RQ 2:** Is predicting labels at sentence level gives better performance than predicting labels at document level?
  - **RQ 2.1:** How can one generate the dataset where relevant labels present per document are actually assigned to different sentences in the document?
  - **RQ2.2:** How can a sentence level dataset where document relevant labels are assigned to its sentence segments be used to train the model?
  - **RQ2.3:** How can labels be predicted for the documents from the labels predicted for the multiple sentences of the document?

Chapter 2

Background and Related Work

2.1 Related Work

This section gives a brief overview of the work done in Extreme Multi-Label Text Classification (XMTC). Since the advent of Extreme Multi-Label Classification, several methods and algorithms have been devised. These methods can be broadly grouped into following types of approaches: (1)One-vs-All, (2) Embedding Based Methods, (3) Tree Based Methods, (4) Deep Learning Methods, (5) Graph Neural Network Methods. These approaches are described below:

2.1.1 One-vs-All Methods

A very frequent method in XMTC is to split the multi-label classification problem into several binary classification problems such as Dismec [3], PD-Sparse [1], PPD-Sparse [4], XML-CNN [33]. These approaches have high accuracies and less model sizes. PD-Sparse [1] minimizes the complexity via training a classifier for each label. This work assumes that there exists a few accurate labels for each instance and feature space is rich enough to distinguish between labels clearly.PD-Sparse employs margin-maximizing loss function in conjunction with an L1 penalty to create an exceedingly sparse solution in extreme classification. An extension to PD-Sparse is the PPD-Sparse [4] which parallelise the PD-Sparse algorithm by utilising large-scale distributed computing. This allows PPD-Sparse to be 100x faster at training than DISMEC [3]. DISMEC learns a classifier for each label based on distributed computing. It uses a double layer of parallelization to sufficiently exploit computing resource (400 cores), implementing a significant speed-up of training and prediction. Pruning spurious weight coefficients (close to zero), DISMEC makes the model thousands of times smaller, resulting in reducing the required computational resource to a much smaller size than those by other state-of-the-art methods.

Despite PPD-Sparse being fast, it is unable to scale to large problems due to significant overhead of generating the shortlist at each item. The prediction times of DISMEC, PD-Sparse, PPD-Sparse are also too high to meet the latency and throughput requirements of real-world applications. Approaches such as PLTs, negative samplings and learned label hierarchies have been proposed to speed up training and predictions. However, they rely on sub-linear search structures such as nearest-neighbor structures or label-trees that are well suited for fixed or pre-trained features such as bag-of-words or FastText [18] but not support jointly learning deep representations since it is expensive to repeatedly update these search structures as deep-learned representations keep getting updated across learning epochs. Thus these approaches are unable to utilize deep-learned features which leads to inaccurate solutions. This project avoids these issues by using deep learning.

2.1.2 Embedding Based Methods

The enormous number of labels is a significant challenge in XMTC. The goal of embedding-based approaches is to reduce the large label space to a low-dimensional space while retaining the label correlations. More specifically, given n training instances $(x_i, y_i), i = (1, ..., n)$ where $x_i \in \mathbb{R}^d$ is a $d-$dimensional feature vector and $y_i \in \{0, 1\}^L$ is a $L-$dimensional label vector z_i by $f_C(y)$ where f_C is called as compression function. After training regression model f_R for predicting embedding vector z_i, decompression function f_D is called on predicted vector z_i to predict label vector $\hat{y}_i$. The disadvantage in this approach is that feature space X and label space Y are projected into a lower dimensional space Z for efficiency. Due this, some information would be lost leading to only limited success in the solution.

Major difference between algorithms in this approach is design of compression function f_C and decompression function f_D. SLEEC [16] the most typical approach computes embedding vectors z_i by maintaining the pairwise distance between label vectors y_i and y_j by capturing non-linear label correlations i.e $d(z_i, z_j) \approx d(y_i, y_j)$ if i is in k-nearest neighbour of j. Since KNN has high computational capacity, so for a test instance, SLEEC uses clusters into which this instance has fallen for prediction. AnnexML [14] is an extension of SLEEC and uses KNN Graph of label vectors in the embedding space improving upon SLEEC accuracy and efficiency.

Another state-of-the-art algorithm is DXML [22] that uses deep neural network to simulate label embedding in a non-linear fashion. As these approaches compresses whole feature vector into a low dimensional vector there is information loss in compression and decompression. This project attempts to overcome this by compressing and decompressing sentence's feature vectors and not document's feature vectors. Sentences being a smaller portion of document, their compressed and decompressed representations doesn't include huge information loss.

2.1.3 Tree-based Methods

Due to the success of tree-based algorithms in binary classification problems, tree-based techniques are also utilized in multi-label classification. These approaches employ an ensemble of decision trees. Tree based approaches in extreme classification have the advantages of less training and prediction time but have high model sizes and poor prediction scores. With constructing a hierarchical structure over a benchmark classifier, the label partitioning by sub linear ranking (LPSR) [15] focusses on minimizing prediction time. However, LPSR has expensive cost since it requires to learn hierarchy additionally. One of the most prominent algorithm in this category is FastXML [20]. It learns a hyperplane and optimizes nDCG-based ranking loss function at each node. PFastreXML [24] is FastXML extension that employs same architecture as FastXML but differs in the loss function it utilizes. It uses propensity scored nDCG and propensity scored precision@k loss functions that attempts to resolve the issue of missing labels in ground truth facilitating system to predict tail labels more accurately. However, these algorithms can be expensive in terms of training time or model size.

Parabel [11] is a newer tree based approach which instead of focusing on training instances partitioning focuses on partitioning of labels. To achieve state-of-the-art prediction performance while keep training time and computational power relatively low, Parabel implements the one-vs-all approach in conjunction with tree-based partitioning.

2.1.4 Deep Learning Methods

In spite of achieving great success in binary and multi-class classification [26] [19], deep learning has not been well explored in XMC setting. FastText [30] recreates document representations by averaging the word embeddings in the documents, accompanied by a softmax transformation. As demonstrated in sentiment analysis and multi-class classification [30], it is a simple yet effective and accurate multi-class text classifier. However, FastText may not be directly applicable for more complicated problems like XMTC [31].

BoW-CNN [27] applies CNN to high dimensional text data that learn powerful word embeddings of small text regions. The aggregated embedding of all regions are passed to one or multiple convolutional layers, a pooling layer and the output layer at the end. XML-CNN [33] inspired from [26], accomplishes computational efficiency by training a deep neural network with a hidden bottleneck layer much smaller than output layer.CNN-Kim [26] similar to BoW-CNN, creates feature maps by superimposing convolutional layers over concatenated word embeddings of a document, which are then max pooled and utilized by a fully connected layer and softmax output layer. XML-CNN adapted this architecture to make it more suitable for XMTC. However, XML-CNN has a few disadvantages [31]. To begin, it is trained with the binary cross entropy loss. This loss is often sensitive to label noise, which is common in extreme multi-label data. Because the label vocabulary is so large, it is very common for human annotators to overlook relevant tags. In such cases, whenever the classifier's predictions are disagreed with the annotated labels, the cross entropy loss function can assign the classifier penalty during the training phase. Secondly, in XML-CNN labels are trained as separate binary classification tasks. Due to this label's prediction scores are not comparable to with each other. This is troublesome because many applications require you to rank all labels based on their relevance, rather than making an independent binary decision on each label. In C2AE [19], the algorithm is trained with ranking loss. It does not scale well to extreme data as the ranking loss used required to compare all positive and negative label pairs. Furthermore, C2AE only accepts the bag-of-words representation as input, making it more difficult to learn powerful representations from extreme multi-label datasets.Apart from these methods, latest methods in embedding based approach such as DXML [22], AnnexML [14] uses deep learning to generate the embeddings whose shortcomings have been stated earlier.

2.1.5 Graph Neural Network Based Methods

Extreme classification may be seen from a different angle as a bipartite graph link prediction challenge given by $G = (DUL, E)$ where D and L represents documents and labels and E represents edges between document $d \in D$ and label $l \in L$. This interpretation allows for the inference of complex correlation structures. Assume that documents d_1 and d_2 have the same label l_1. If another label l_2 is significant to d_2, it may be assumed that l_2 is also relevant to d_1. In extreme classification contexts, where missing labels abound and training documents are seldom tagged comprehensively with all labels pertinent to them, such transitive inferences can be quite useful. Current extreme classifiers, on the other hand, struggle to represent such implicit transitive links unless the pair (d1, l2) appears explicitly in the training set. Recent advancements in Graph Neural Networks (GNNs) [19, 51, 55] allow using node neighborhoods to collaboratively learn more discriminative features. However, existing works mostly use document-document graphs and not joint document-label graphs at extreme scales.

2.1.6 Attention Mechanisms

Focusing on the most relevant parts of the input to formulate decisions by using attention mechanisms has become a standard choice in lot of tasks [41], [50]. AttentionXML [40] proposed the idea of per-label attention in extreme classification, where the network attends to the most important bits of the document embedding for each label. AttentionXML uses BiLSTM to effectively represent documents for XMTC and self-attention mechanism to learn relevant parts of text for each label. AttentionXML did get promising results but it ignored the label relationships which has been demonstrated to be critical in embedding-based and tree-based multi-label learning methods. Additionally, the method's complexity grows with the number of labels, making it unsuitable for large-scale applications [45]. HBLA [47] constructs the label graph using a novel strategy to consider the label relationships. It also proposes a new attention approach - adjustive attention to establish the semantic relationships between words and labels to retrieve label-specific word representations. The hybrid representation of context-aware feature and label specific word feature is fed to the document encoder for classification. LAHA [48], a state-of-the-art in this category, uses two attention modules. First attention module implements self-attention on the document representation which is generated by the BiLSTM. Second attention module is interaction attention between the label embeddings and the document representation from BiLSTM. Label embeddings are produced by the node2vec algorithm from the label co-exist graph.

2.1.7 Focal Loss

Focal Loss introduced in Lin et al. [56] has been used to avoid the model from being biased toward frequent labels in previous work related to XMTC. Focal Loss has two hyper parameters α and γ which together helps model focus on rare classes more than the frequent labels.α is a weight parameter corresponding to class labels whereas γ is a *focusing* parameter that led model focus more on rare labels. Lin et al. in [56] found $\alpha = 0.25$ and $\gamma = 2$ works best. Jiang and He [76] tries to improve focal loss by introducing the idea of using Gaussian weights. Here a set of infrequent labels was given a constant weight α_1 and frequent labels were given weight α_2 for focal loss. Han et al. [78] used Focal Loss in extreme multi-label text classification setting and applied it on AttentionXML [40] and RoBERTa with mix-up [69] strategy. Interestingly $\alpha = 1$ was set for all the labels and γ parameter was hyper-tuned. In the cases of extreme data imbalance such as in extreme classification where there is a huge label set and each label has a different extreme frequency from other, it is recommended to not give a constant weight to all labels or a set of labels. PFastreXML [24] talked about propensity-scored loss functions, which are an augmentation of current loss functions multiplied by inverse propensity weights. Because the propensity weight was difficult to get in most datasets, it was proposed that the propensity defined by the sigmoidal function be used instead. The propensity weights are related to the labels and frequent labels tend to have higher propensity weights than rare labels. This work use focal loss with α parameters set to the inverse of these propensity weights corresponding to each label.

2.2 Background Work

This project deals with the extreme class imbalance with Focal Loss function and proposes new strategy of training and predicting at the sentence level of the document. With this in mind, related work section is divided in two parts: one about class imbalance and another about different training strategies that covers aspect or sentence level, meta level and topic modelling training.

2.2.1 Methods for class imbalance

Existing deep learning methods in text classification exhibit good classification performance on balanced dataset. The stability and generality of these models, however, might be harmed by data imbalance. Data imbalance is data distribution imbalance that reflects different numbers of samples among classes which generates a biased model based on generic loss functions. In extreme multi-label classification, data imbalance is very extreme where a large number of labels have around 2-3% of data samples assigned and very few labels have 98% of the data samples assigned. The learnt model does well in categories where there are enough examples, but not so well in classes where there are not enough. There are various ways suggested in the literature to overcome the problem of data distribution imbalance.

Data distribution shift

These methods alter the data distribution such that all classes have almost equal number of data samples and there is data balance. Sampling methods, such as oversampling, under-sampling, and hybrid sampling are some of the prominent methods in this category. Synthetic minority oversampling techniques (SMOTE) [71] is one of the most popular over-sampling methods that balances data distribution by generating synthetic samples of minor categories. The work in [64] employed SMOTE based class-specific extreme learning machine to increase the classifier's attention to samples of minor categories. Experimental results demonstrated that the algorithm had a high efficacy on real benchmark datasets. Under-sampling, as contrast to over-sampling, randomly samples a subset of categories with a high number of samples. To mitigate the harmful effects of class imbalance, [73] used random under-sampling. The classification performance of this strategy is better than that of a classifier that does not use any data sampling, according to computational results. Due to the success of both these methods, hybrid sampling methods are developed. [59] proposed ant colony optimization re-sampling strategies to address the problem of class imbalance. To find the optimal subset from the balanced dataset created by over-sampling, this model used the colony optimization technique. When compared to traditional sample approaches, this results in a substantial improvement. However, sampling approaches have disadvantages. To begin with, the oversampling method puts more noise into the dataset, lowering model performance. Oversampling approaches in natural language processing are challenging to use because the meaning of natural language can vary substantially with minor input transformations. Second, under-sampling causes data loss, and the model may overlook important information during training.

Reformulating loss functions and propensity

Increasing the model's loss for minority labels miss-classification is another way to resolve class imbalance. The goal of this strategy is to modify the model such that it pays greater attention to minority samples. The most common way is to use the cost sensitive loss function to replace the original classification loss function. Authors in [74] created a cost-sensitive stacking learning model based on inverse mapping that merged cost-sensitive and ensemble approaches. With unbalanced datasets, the efficacy and efficiency of this strategy were evaluated using both linear and ensemble forest classifiers. In 2016, [70] presented the online hard case mining (OHEM) method in 2016. The OHEM algorithm provides training dataset with the greatest loss and filters out problematic samples that have a higher effect on classification and detection for retraining by utilising its loss function. Although the OHEM method gives miss-classified samples more weight, it ignores easy-to-categorize cases. [56] suggested a novel Focal Loss to overcome this problem, which expands OHEM by reweighing miss-

classification, simple samples, and difficult samples, and makes the model pay more attention to the problematic samples by lowering the weight of easy-to-classify data in training.

2.2.2 Different Training Strategies

Often in text classification, vanilla strategy to train a model is to consider the whole document as input and train against its ground truth. There are various use cases where certain different information of a document are being employed for training such as sentences of a document, document's meta data and topic modelling. This section lays out the related work for these strategies.

Sentence based training

Hierarchical Attention Network [18] is the first to propose a sentence-level model for document classification. It adopted Bi-GRU with attention in the architecture. Other work researching on the effectiveness of sentence-level model based on BERT includes HTML [21] and HIBERT [25]. HTML [21] presented a sentence-level BERT model coupled with audio characteristics to estimate the price of a financial asset over time, whereas HIBERT [25] is intended for document summarisation. Similar to HTML [21], Att-BLSTM [42] proposed a neural network architecture for relation classification. It utilized neural attention mechanism with Bi-LSTM on sentences of documents to capture most useful semantic information for the classification. Authors in [37] proposed bidirectional GRUs which integrates a novel attention pooling mechanism with max-pooling operation to force the model to pay attention to the keywords in a sentence and maintain the most meaningful information of the text automatically. Along the same lines, a lot of work has been done in sentiment analysis domain where instead of whole document sentences from the document are used to classify. [44] proposed an attention based LSTM method with target embedding, which was proven to be an effective way to enforce the neural model to attend to the related part of a sentence. The attention mechanism is used to enforce the model to attend to the important part of a sentence, in response to a specific aspect. [58] extended the attention modeling by differentiating the attention obtained from the left context and the right context of a given target/aspect. These studies have taken different sentences of a document individually as input and concatenated the computation to form a final document representation in the model which is then passed to the output layer and model is trained. In this project, the proposed architecture takes sentences as inputs separately and are trained at the sentence level itself.

Topic Modelling

Topic models have been studied since researchers developed Latent Semantic Indexing [72] in 1990. Topic modelling is the method of determining the underlying semantic structure of a text using a hierarchical Bayesian analysis on a group of documents [79]. In simpler words, documents are represented with a mixture of topics, topics are represented with a probability distribution over words and the documents are represented by a probability distribution of topics. Basic, inter-document correlated, intra-document correlated, temporal, and supervised probabilistic directed topic models are the five primary types of probabilistic topic modeling methodologies [75]. Though several topic modeling approaches exist in the literature, the key works in the field are latent semantic analysis, probabilistic latent semantic analysis, and latent Dirichlet allocation [80]. Latent semantic analysis is a natural language processing approach that uses statistical and mathematical computations to extract and represent the contextual use meaning of words in a vast collection of text corpora [81]. Singular value decomposition is used in latent semantic analysis (LSA) to lower the dimensionality

of the TF-IDF scheme. LSA may capture synonyms of words, however determining the number of themes in LSA is challenging [62]. Probabilistic latent semantic analysis (PLSA) is a probabilistic topic model in which each word in a document is represented as a sample from a mixture of models [62].There is no probabilistic model at the document level in PLSA. LDA is a generative probabilistic approach for modeling collections of discrete data, such as a text corpus [68]. LDA can provide a full generative model and can handle long-length documents [62]. Authors in [82] used LDA for lawsuits classification. It was a multi-label classification task. Similarly, another method in [77] used LDA to model topics and then use the derived topic vectors for fake news classification. SLLDA [81] labeled LDA to address extreme multi-label teaxt classification. SLLDA efficiently scale up to problems with hundreds of thousands of labels. But LDA suffers from ”order effects” which is different topics are generated if training data is shuffled. This leads to inaccurate results and overall performance of the model takes a hit [83].

Chapter 3

Datasets and Preprocessing

A high-quality evaluation of a machine learning model entails determining if the model performs well across all relevant domains and sub populations. In this project, two datasets were utilized to assess the robustness of the model and methodology. We use EURLEX57K dataset [51], and Medline dataset [49]. EURLEX and Medline are prominent benchmark datasets in Extreme Multi-label Text Classification (XMTC) and various works have evaluated their experiments against these datasets. We provide the details on these two datasets in following sections.

3.1 EURLEX57K Dataset

EURLEX57K dataset is an improved version of the dataset released by Mencia and Furnkranzand [52]. This dataset contains 57,000 legislative documents in English with an average length of 727 words. One peculiar challenge XML tackles is trying to learn patterns for labels which have very few training instances in the dataset. Hence, learning document representations for these labels is challenging. In XMTC task, only a few labels have suitable number of documents they being assigned to. In EURLEX57K dataset there are approximately 7,000 labels, out of which only 4,271 labels are assigned to the documents. Out of the total assigned labels, only 2,049 have been assigned to more than 10 documents.

EURLEX57K labels exists in a tree structure with the directed edges among the label nodes. In case a label is a parent in the tree it has one or more children who further down have one or more children. The edges between the two label nodes in the tree represents the two labels are related. Between any two linked label nodes considered, parent label is a *general* label whereas child label node gives specific information within the parent label. A label tree has *0* to *n* levels. Suppose, a random label at level *7* is annotated to a document then its parent nodes from level *6* to *0* becomes relevant labels as well. This randomly selected label's siblings and all its parent sibling labels would not be relevant labels. EURLEX57K labels does not belong to large tree. There exist multiple trees of different sizes in the label space and with no connecting edge among them. Each tree consist of labels of one kind where its label nodes are related with each other based on a certain topic or a category.

3.1.1 Data Preprocessing - EURLEX57K

Originally, EURLEX57K dataset has 7201 labels of which only 4,271 labels have been assigned to the documents. These assigned 4,271 labels belong to different independent label trees. Each label

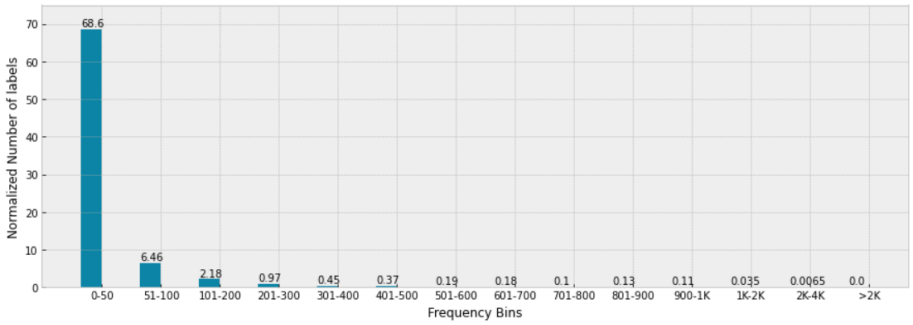


Figure 3.1: EURLEX57K Dataset Labels Distribution: The horizontal axis represents labels' frequencies in window/bin format whereas vertical axis represents how many labels have their frequencies falling in particular frequency bins. The number of labels in a bin have been normalized by the size of the bin.

of a tree is related to other labels based on a certain topic or category. Due to this, if one label is assigned to the documents, its parents upto level 0 or root node becomes potential relevant labels. Whereas, if none of the labels of a tree are assigned to the documents, none of its labels could potentially be relevant labels. These labels could not be predicted by the model as none of the labels of the tree are present in the ground truth. Hence, all labels of the tree whose atleast one label exist in the training dataset are kept. After preprocessing of the dataset, all 7,201 labels are not kept. The assigned 4,271 labels are part of different independent tree hierarchy structures. Each independent tree hierarchy structures' labels are related to each other based on a certain topic or category. Due to this, if one label is assigned to the documents, its children till last *n* level becomes potential relevant labels. Whereas, if none of the labels from a tree hierarchy structure These labels could not be predicted by the model as none of the related labels are assigned to the document. Hence, all the labels from different tree hierarchy structure are kept if atleast one label from particular tree hierarchy structure is assigned to the document. In the end, 4,531 labels are kept in the dataset out of which 4,271 labels are assigned to the documents.

EURLEX57K dataset consists of 45000 legislative documents in training dataset, and 6000 documents in validation and test dataset each. The final 4,271 assigned labels follows the power law distribution where the majority of the labels have very few instances in training dataset (5-10 instances) making them difficult to learn and predict in evaluation. In numbers, out of utilized 4,271 labels; 3,430 labels have 1-50 data points in the whole dataset which is 80% of the total labels. There are only 51 labels with more than and equal to 1000 data points whereas 541 labels have 51-200 data points (fig.3.1) . Due to these characteristics, machine learning models tend to become biased towards frequent labels and prediction of rare labels becomes a challenge. In the case of applying plain supervised classification models, the models will learn only about frequent labels whereas other suggested methods [22] tends to be unscalable for huge label spaces.

3.1.2 Segmentation - EURLEX57K

The documents in EURLEX57K dataset are litigation based documents where the text is not present in a sructured format of plain paragraphs with formal boundaries of punctuations. Rather, documents are organized in an unstructured format where headings, sub headings and bullet points are present. These documents are highly noisy (e.g. grammatical and spelling mistakes) as well, as they

are manually typed [55]. Along with these characteristics, EURLEX57K legal documents consist of litigation-based domain specific lexicons. At times, a legal domain specific lexicon is followed by its sub domain lexicons in an unstructured format. Due to presence of such unstructured document in the corpus, it is highly unlikely to be able to segment the documents just based on delimiters (i.e. full-stops or question marks, etc) as it raises the possibility of segmenting the documents into incomplete coherent sentences. A sentence segment should be coherent in its own, explaining certain topic or an idea.

In plain textual documents, punctuation denoting the boundaries between the different sentences always results in coherent sentences and these boundaries often coincide with a change in topic. A similar idea of segmenting legal documents cannot be utilized in the case of legal documents due to nuances described above. Therefore we need a tool specializing in legal documents segmenting by handling its unstructured format. Blackstone [60] is a python library which is used in this project to segment the EURLEX57K dataset's documents into different sentence segments.

Blackstone is a Spacy based library for processing long-form, unstructured legal text. It is maintained by the Incorporated Council of Law Reporting for England and Wales' research lab and written by Daniel Hoadley. Blackstone is an open-source project specifically trained for use on long-form text containing common law entities and concepts. It has been trained on data since 1860s making it a comprehensive model which has seen different forms of legal structures. Specifically, the model has been trained on English case laws and it has the peculiarities of the legal system of England and Wales. With this in mind, the model is generalized well to understand other geographical specific legal text. Since, EURLEX57K consists of legal documents from European Union, Blackstone library can be used to execute the segmentation of the legal documents.

With the help of Blackstone library, EURLEX57K documents are segmented. There are 6,04,898 sentence segments generated from the training dataset with 79,711 and 78,264 segments in test and validation dataset respectively.

3.2 Medline Dataset

The Medline dataset [49] is a publicly accessible dataset of medical scientific papers that includes abstracts, citations, authors, labels, and other meta data. It consists of about 27 million records. This dataset is maintained by National Center for Biotechnology Information (NCBI) at the US National Library of Medicine (NLM) through PubMed, a free online resource that hosts citations and abstracts in the field of medicine. The majority of medline articles are biomedical in nature, while the rest are related to the life sciences. Medical Subject Headings (MeSH), a regulated vocabulary vetted and updated annually by the National Library of Medicine, are used to annotate each record. The primary goal of MeSH terms is to create an organised collection of labels for indexing and cataloguing the huge number of medical publications in the PubMed database so that they may be easily found. The MeSH Tree View Browser provides access to the MeSH words, which are organised in a tree hierarchical format [54]. There are 16 branches in MeSH tree, where the MeSH headings at the top of these branches are generalize headings and down the tree MeSH headings becomes more specific. Down the branches in the tree, many MeSH headings are repeated in different (sub)branches. MeSH is a restricted vocabulary of labels that serve as keywords in the PubMed system to aid in document retrieval. MeSH provides indexers with a definite framework to refer to as a regulated vocabulary. If no particular words are provided in the MeSH terms vocabulary for an article, an indexer will assign

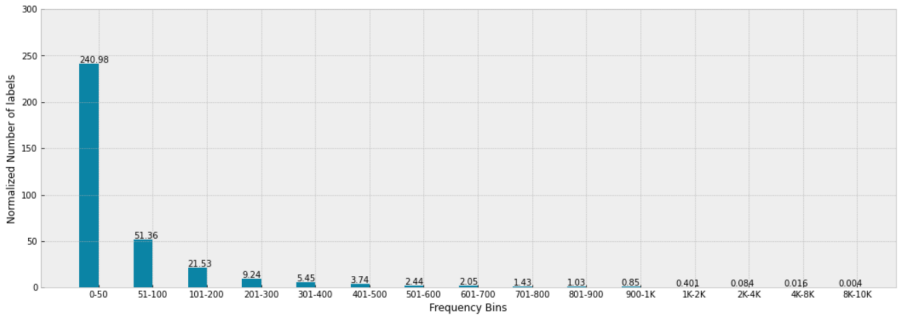


Figure 3.2: Medline Dataset Labels Distribution: The horizontal axis represents labels' frequencies in window/bin format whereas vertical axis represents how many labels have their frequencies falling in particular frequency bins. The number of labels in a bin have been normalized by the size of the bin.

the closest generic term possible. Due to this, like other extreme classification domains, medline labels follow a power law distribution or tail distribution. Similar to EURLEX57K dataset, the vast majority of the labels are extremely sparse i.e. have fewer instances or data points.

3.2.1 Data Preprocessing - Medline Dataset

Medline dataset in use consists of 15,741,875 documents and 60,682 MeSH headings or labels. As MeSH headings exists in tree hierarchical structure, MeSH headings of only top 8 out of 16 branches are kept in the final dataset. The total number of labels from 8 branches are 54,579. There are 27,854 labels in top 8 branches which are repeating several times. Effectively, there are 26,725 unique labels in the final processed dataset. There are 40,3620 documents in the training dataset and 55,000 documents in validation and test dataset each. Out of the total considered 26,725 labels, 20,055 labels are being assigned to the documents. 16,770 labels from total assigned labels have less than 200 documents in 40,3620 documents. This sparsity in labels distribution makes this dataset suitable for extreme classification tasks. The label distribution can be seen in detail in Fig.3.2

Dataset	Train Set	Labels	Avg. points per label	Avg. labels per point	Validation	Test
EURLEX	57,000	4,531	55	5	5,500	5,500
Medline	513,620	26,725	6,888	12	55,000	55,000

Figure 3.3: Document Level dataset statistics for EURLEX57K and Medline dataset

Dataset	Labels	Train Sentences	Noise Sentences	Validation Sentences	Test Sentences
EURLEX	4,532	340,330	15,505	78,264	79,711
Medline	26,726	946,382	65,245	113,743	128,539

Figure 3.4: Sentence Level dataset statistics for EURLEX57K and Medline dataset

3.2.2 Segmentation - Medline Dataset

Unlike EURLEX57K documents where it contains sub-headings, bullets, etc., Medline dataset's documents does not have such unstructured format of text. But interestingly, the dataset documents are cleaned from every kind of punctuation marks, and hence there are no formal boundary among the document's text. The text is present in the form of a plain sequence of words with no full stops or any other punctuation marks which could denote the boundary between adjacent sentences. Evidently, it is quite difficult to segment such text into different sentence segments randomly such that sentences are coherent in their meaning and also follows the sentence structure.

There are popular libraries such as NLTK, Spacy, Stanford Core NLP which are equipped with functionality to identify the boundaries among the sequence of words to generate the sentences. But these libraries use statistical modelling or depends on heavily language patterns to perform sentence boundary detection. These libraries performs miserably on text with bad punctuation and wrong capitalisation. Therefore, DeepSegment, a python library, is used to segment the text and generates the coherent sentence segments that follows the sentence structure as well. DeepSegment uses BiLSTM and Conditional Random Fields (CRF) for automatic sentence boundary detection between the sequence of words. It significantly has outperformed NLTK, Spacy standard libraries.

With the help of DeepSegment, Medline dataset's documents have been segmented. There are 9,46,382 number of segments generated from the training dataset. 1,28,539 segments and 1,13,743 segments are generated from test and validation dataset respectively.

Chapter 4

Methodology

This section gives detail knowledge about the methodology and tries answer research question stated in section 1.2. This project deals with finding out whether sentence level prediction results in better results than document level prediction in extreme multi-label classification area. This is answered by building two different models where one model is trained at the sentence level and another at document level and comparing their scores. This project introduces **L**abel **A**ware **F**eatures with **F**ocal Loss (**LAFF**) model trained for sentence and document level. The fig 4.1 lay out the methodology framework for sentence and document level. The datasets used in this project consists of documents and

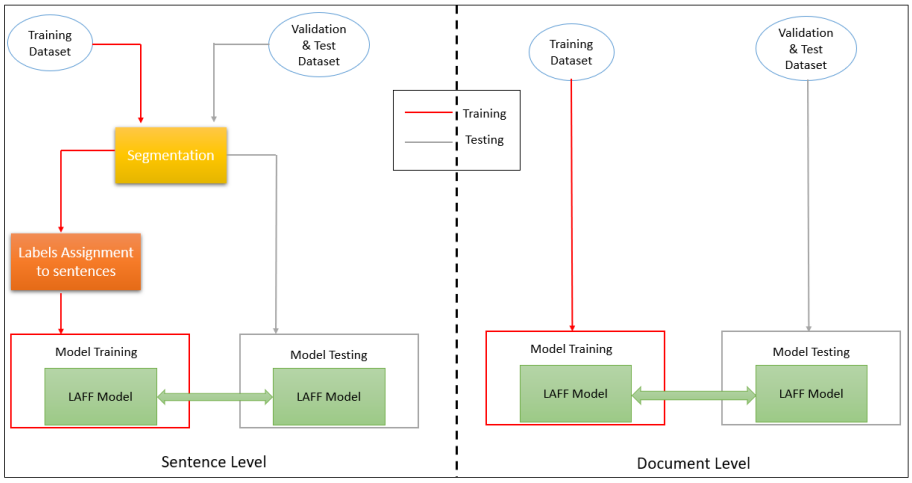


Figure 4.1: Methodology Framework: Left-side represents methodology for sentence level. Training dataset that contains documents are segmented into sentence segments and gets labels assigned. This sentence level training dataset is used to train LAFF model. Validation and test dataset is segmented and with the help of trained LAFF model relevant labels are predicted. Right-side represents methodology for document level. There is no segmentation involved. Similar to sentence-level LAFF model is trained and evaluated but at the document level.

their corresponding relevant labels. These datasets could be readily used for document level model training. However, for sentence level model training, a sentence level dataset is need to derived from the existing document level dataset such that the labels are associated with the sentence segments of the document. Section 4.1 deals with the methodology of generating such sentence level training dataset where labels are associated with sentence segments of documents from already available datasets.

In document level trained model, inference is done for the document of the test dataset directly and evaluation is carried out. In sentence level trained model, inference is done by predicting labels for the sentence segments of the test documents. To compare both document and sentence level trained model, predictions at sentence level are need to be aggregated to form final prediction set for the document. This final set per document can be used for model performance evaluation and fair comparison can be done with the document level trained model's performance. Section 4.3 discusses three strategies of aggregating sentence level prediction for generating labels prediction set at document level.

Section 4.2 discusses the algorithm and model architecture of LAFF that would be used for document and sentence level training and testing. The last research question deals with using Focal Loss function for extreme multi-label text classification which is answered in section 4.2.3. This section derives the focal loss from cross entropy loss and explains how it could be used with label propensities to solve the rare labels prediction challenge.

4.1 Label Generation at Sentence Segments Level

This section introduces methodology to generate labels at sentence segments' level from any given dataset that contains labels corresponding to the documents. Labels generated for the sentence segments would later be used to train the model and predict labels at the sentence segments level. The core idea is to bring labels and features (i.e. documents or sentence segments) in the same embedding space. Once, the training set features and the labels are in the same embedding space, unseen features can be assigned with the labels which are closer to them. This same idea is used in Deep Extreme Multi-Label Learning(DXML) [22]. DXML implementation is used here to bring the documents and labels in the same space. This is followed by predicting labels for the sentence segments by using the same trained model. Consecutively, the methodology of generating labels at sentence segments level is comprised of mainly two components: (i) *bringing labels and documents in the same space* (ii) assigning labels to sentence segments by predicting labels for them.

4.1.1 Bringing Documents and Labels in same space: DXML's approach

This approach aims at learning non-linear mapping from documents X to the labels' (Y) embedding space which could be understood as bringing X and Y in the same embedding space. The visualization of the framework can be seen in Fig. 4.2. The model is trained with the documents and their corresponding labels to learn a non-linear mapping from the documents to the label space. The labels are used in the form of label graph in training which utilizes the labels structure. A label graph is constructed where two labels have an edge if they co-occur with each other in any documents' annotation. With this label structure, each vertex of the graph is represented by a high-dimensional vector with the help of Graph Convolutions Networks(GCN) [61]. The documents are also represented by a high-dimensional vector with the help of BERT and neural network. The documents are fed to the BERT whose output is fed to the neural network and we get a high-dimensional vector representation of document. This vector representation of documents are used along with averaged corresponding label embeddings to compute embedding loss. The aim of model training is to reduce this embedding loss consequently mapping the documents and labels arrive in the same embedding space.

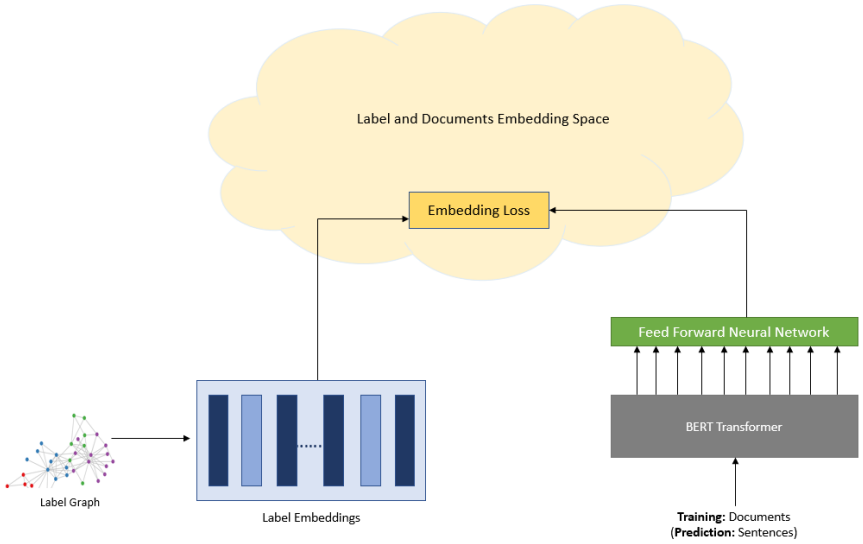


Figure 4.2: Framework for Label Generation at Sentence Segment Levels. From the label co-occurrence graph label embeddings are generated with Graph Convolution Networks. While training documents are fed to the BERT and its output is passed to the neural network. Embedding loss between label embeddings and output of neural network is computed and minimized to bring documents and labels in same embedding space. To assign labels to sentences, sentences are fed to BERT followed by neural network and whichever label embeddings are closest to neural network output that corresponding label is assigned to sentence.

Let $D = (X_1, Y_1), ..., (X_N, Y_N)$ be training dataset containing N documents that belongs to K labels. X_i represents a documents and $Y_i \subseteq \{0,1\}^K$ its corresponding ground truth label vector where $y_{ij} = 1$ if j-th label is assigned to the i-th document. The label graph is constructed with labels as node, and edge is constructed between two nodes if labels share at least one document in the training dataset. The label graph is defined as $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathbf{v_i} \in \mathcal{V}$ refers to labels $\mathbf{k_i} \in \mathbf{K}$, edges $(\mathbf{v_i}, \mathbf{v_j}) \in \mathcal{E}$, an adjacency matrix $\mathbf{A} \in \mathbb{R}^{K \times K}$ and a diagonal degree matrix $\mathbf{D_{ii}} = \sum_j A_{ij}$. The adjacency matrix elements values belongs to (0,1) where value 0 represents no edges presents and 1 represents presence of edges between two nodes. The label graph contains undirected edges i.e. $A_{ij} = A_{ji} = 1$. Graph Convolution Networks (GCN) [61] is employed which propagates messages through the graph and learns the contextualized label embeddings.To update the current node, GCN, combines the degree values of all adjacent nodes. Only the first-order neighbourhood information is recorded by each convolution later in GCN. By stacking many convolution layers, multi-order neighbourhood information can be gathered. GCN is used with multi-convolution layers to learn a high-dimensional latent space representation of labels. Finally, each label is represented by a r-dimensional dense vector, i.e. $I_k \in \mathbb{R}^r$ for the k-th label (k=1,2,...K). The whole set of labels in embedding form can be described as $L = (I_1, ..., I_k) \in \mathbb{R}^{r \times K}$. All label embeddings can be represented by matrix **V** with K rows and r columns.

The documents are passed to the BERT followed by the Feedforward Neural Network to learn a non-linear high-dimensional representation of it. Document's representation in label's embeddings space is represented by non-linear function $f_X = F(X, W) \in \mathbb{R}^r$ where W denotes BERT's as well as neural network's weight parameters. Each document f_x would have corresponding processed label

embeddings represented by f_y where $f_y = \frac{1}{nnz(\mathbf{y})}\mathbf{V}*\mathbf{y}$. $nnz(\mathbf{y})$ denotes number of non-zero elements in $y \in Y$. Set of f_y is represented as $f_Y \in \mathbb{R}^r$. Therefore, the distance between each document embedding represented by f_x; and corresponding processed label embeddings f_y is represented by:

$$d(f_x, f_y) = \sum_{m}^{r} h((f_x)_m, (f_y)_m) \tag{4.1}$$

where

$$h(a,b) = \begin{cases} 0.5(a-b)^2 & if|a-b| \leq 1, \\ |a-b| - 0.5 & otherwise \end{cases} \tag{4.2}$$

The objective function that will be minimized can be formulated as:

$$\mathcal{L}(W) = \sum_{j=1}^{N} d(f_{x_j}, f_{y_j}) \tag{4.3}$$

With minimizing the loss function equation 4.3, f_x and f_y would be close to each other as measured by the embedding loss. After getting the final f_X once training is complete, it is partitioned into several different clusters with k-means algorithm. These clusters are used to predict the labels for documents while doing evaluation. Similarly, it could be used to predict labels for sentence segments as well. The training algorithm is shown in Algorithm 1

Algorithm 1: Training Algorithm to bring documents and labels closer in same embedding space

1 **Input:** $D = (x_{(1)}, y_{(n)}), ..., (s_{(n)}, y_n)$ where $1 \leq i \leq N$ (total no. of documents); No. of clusters: C; Embedding dimensionality: *I*;

2 Build the label graph with the labels where edges exist between two labels if they share a document in dataset;

3 Use GCN [61] to generate label embeddings matrix **V**;

4 Project original label matrix *Y* to

$$f_Y = \left\{ f_y | f_y = \frac{1}{nnz(y)} V * y, y \in Y \right.$$

where $nnz(y)$ denotes the number of non-zero elements of the original binary vector y;

5 Train BERT and neural network to obtain mapping from feature vector X to feature vector f_X. Update **W** for T epochs to

$$min\mathcal{L}(W) = \sum_{j=1}^{N} d(f_{x_j}, f_{I_k})$$

6 Partition f_X into $Z^1, ..., Z^C$ by k-means

7 **Output:** $\{Z^1, ..., Z^C\}$

4.1.2 Generating Labels for Sentence Segments

After training the model with the documents and their corresponding labels, the same model is used to assign labels to document's sentence segments. With the model training, the documents vector representation has come in the same embedding space as labels' vector representation. This could also be understood as each documents' vector representation is closest to its potentially relevant labels vector representations in the same embedding space. The sentence segments are derived

from the segmentation of original documents. Hence, even sentence segments vector representation would be in same embedding space as of labels and they will be closer to their potentially relevant labels.

After training the model with documents and corresponding annotated labels, the non-linear mapping of documents f_X is clustered into C partitions represented by $Z^1, ..., Z^C$ with k-means algorithm. For predicting labels for sentence segments, first its vector representation is computed from the trained model denoted by f_x. This f_x gets assigned with the index of the closest cluster by

$$i^* = min_{i \in \{1,...,C\}} ||f_x - z_c^i||^2 \tag{4.4}$$

where z_c^i is the center of the i-th cluster. With the closest cluster found, with the help of $k-$NN search, similar training samples are found in Z^{i^*}. The union of k-nearest neighbour's labels is set as the provisional set of labels for the sentence segments.The intersection between the provisional set of predicted labels and sentence segments original documents label set is kept as the final label set for the sentence segment. If no intersection is found then the sentence segment is ignored and not kept in the dataset. This way, only document relevant and important sentence segments are kept in the final dataset.

Algorithm 2: Algorithm to predict labels for sentence segments

1 **Input**: Sentence Segment **x**, clusters $\{Z^1, ..., Z^C\}$, No. of k-NN: k, No. of desired labels
2 $\tilde{x} \leftarrow f_x = F(x, W)$
3 Z^{i^*}: partition closest to $\tilde{x}$
4 $K_{\tilde{x}} \leftarrow k$ nearest neighbours of $\tilde{x}$ in Z^{i^*}
5 $P_x \leftarrow$ All labels for points in $K_{\tilde{x}}$
6 $provisional_x \leftarrow Top_p(P_x)$
7 $y_{pred} \leftarrow$ intersection of $provisional_x$ and sentence segment **x** original documents annotated labels

Training Dataset Generation

One of the research question this project tries to answer is how a newly derived dataset where labels from the document level are assigned to the sentence segments can be used to train the model. This question is important since the sentence level training dataset generated with above methodology contain only document relevant important sentence segments. The methodology removes the noisy sentence segments whose provisional label set does not have any common labels with the document's ground truth label set. Whereas, test and validation dataset would contain all sentence segments including the noisy ones. Considering noisy and unimportant sentence segments would for the evaluation would unnecessarily result into worse results.

Since, in test or validation dataset, there is no way to know and provide only document relevant sentences to the model for prediction, model should have some capability to detect these unimportant and noisy sentence segments. Hence, to augment the model's capability in this direction a small portion of noisy sentences are added to the sentence level training dataset. These added sentence segments are assigned "NOISE" label which model will learn and predict while validation or testing. The dataset generated above and augmented with noisy sentence segments would give final sentence level training dataset.

4.2 Final Label Predictive Model

This section introduces **L**abel **A**ware **F**eature with **F**ocal Loss (LAFF) algorithm with its model architecture to predict relevant labels for sentence segments and as well as documents. With using the same algorithm, it helps in concluding whether predicting labels for sentence segments is better and results in better results than predicting labels for documents. The model has been trained and evaluated once with the original dataset of documents and other time with sentence segment dataset obtained from previous section and their results are compared to arrive at a conclusion. When labels are predicted for sentence segments level, final predicted label set is created for document level by clubbing all the labels and ordering them based on frequency. Fig. 4.3 introduces the model architecture which attempts to answer whether doing prediction at sentence segment level is better than at document. It does so by trying to leverage two different *attention* mechanisms which integrates different label representations with different document representation. Fig. 4.3 shows model architecture consists of 4 main components:

- The document embedding module generates semantically meaningful embeddings of input texts which could be sentence segments or original documents.
- The Label graph embedding module generates label embeddings capturing semantic correlations between labels from the label graph input.
- The Interaction Attention computes attention score embedding vector of the input text with respect to each label.
- The Aggregation layer combines attention score embedding vector from interaction attention module with input text embeddings from the document embedding module which is used predicting relevant labels.

Let $D = \{(X_1, Y_1), ..., (X_N, Y_N)\}$ be a training dataset. X_i where $1 <= i <= N$ represents both documents and sentence segments interchangeably. Each X_i has its corresponding label vector Y_i that contains upto K labels i.e. $Y_i \subseteq 0, 1^K$. N is the total number of data points in the training dataset which could be documents or sentence segments. The training starts with documents or sentence segments passing to BERT to fetch the input corresponding contextualized embeddings. BERT generates r dimensional embeddings. For a training dataset with N data points, BERT's output, H, could be represented as:

$$H = (h_1, ..., h_N) \in \mathbb{R}^{N \times r} \tag{4.5}$$

Label graph has been used to focus on the label correlations. Similar to how label graph is used in label generation for sentence segments in section 4.1.1 and in DXML [22], label graph is used here as well. The label graph is constructed with labels as the nodes and two labels have an edge between them if they co-exist in any document's annotation. Similar to the label embeddings generation described in section 4.1.1, GCN [61] is employed here as well to generate the label embeddings. Each label node is represented by an s-dimensional dense vector i.e. $I_k \in \mathbb{R}^s$ for the k-th label $(k = 1, 2, .., K)$. Whole set of labels can be described as $L = (I_1, ..., I_k) \in \mathbb{R}^{s \times K}$. BERT generates r-dimensional embeddings for documents, segments and GCN generates s-dimensional embeddings for labels.

4.2.1 Interaction Attention

Interaction attention seeks to compute attention scores regard to each label in order to establish the semantic relationship between labels and features. All text features be it in sentence form or

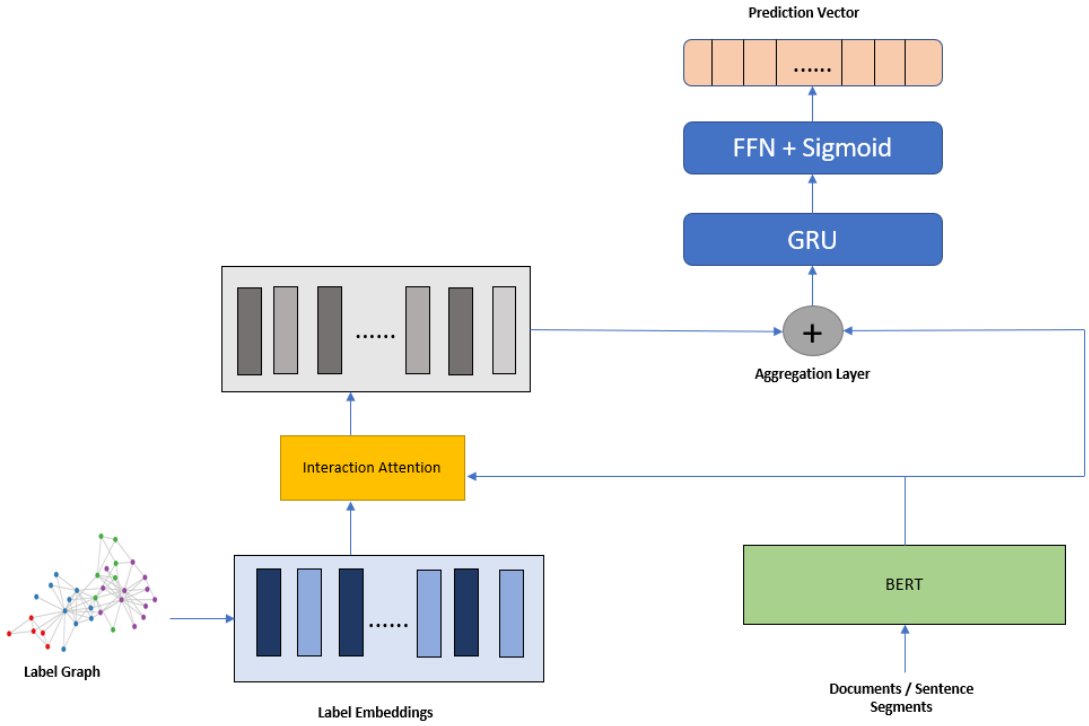


Figure 4.3: LAFF Model Architecture. Interaction attention weights are computed between label embeddings and features' embeddings i.e. documents or sentence embeddings. Label embeddings are generated from label co-occurrence graph with graph convolutional networks and features' embeddings are generated from BERT. The BERT embeddings are aggregated with the interaction attention weights and passed to the Gated Recurrent Unit (GRU) and neural network to label prediction vector.

paragraph form and labels are represented in the r-dimensional latent space using the sentence embedding and label graph embedding techniques as $H = (h_1, ..., h_n)$ and L respectively. Documents or sentence segment embeddings and label embeddings are in different latent spaces. To align them in single latent space, matrix $W_q \in \mathbb{R}^{s \times r}$ is trained via $Q = LW_q$. Similar to [48], $Q \in \mathbb{R}^{k \times r}$ is taken as queries for each label and use H to construct key-value pairs for each sentence. Then, interactive score $M^{(I)} \in \mathbb{R}^{n \times k}$

$$M^{(I)} = HQ^T \tag{4.6}$$

To make sure the attentions scores fall into the range of [0,1], $M^{(I)}$ is normalized to obtain the interaction score $A^{(I)}$ of label and sentences computed as:

$$A^{(I)} = (A^{(I)}_{tj})_{t=(1,...,n),j=(1,...,k)} \tag{4.7}$$

$$A^{(I)}_{tj} = e^{M^{(I)}_{tj}} / \sum_{i=1}^{n} e^{M^{(I)}}_{tj} \tag{4.8}$$

4.2.2 Aggregation Layer

From the above steps we obtain $A^{(I)} \in \mathbb{R}^{n \times k}$ and $H \in \mathbb{R}^{n \times r}$. The former is about the importance of each label with respect to each document or sentence segments, while the latter focuses on the

semantic meaning of the sentences. Aggregation layer is formed to aggregate the information from $A^{(I)}$ and H. For simplicity, embeddings from $A^{(I)}$ and H are concatenated as shown in 4.9, where $\hat{H} \in \mathbb{R}^{n \times (r+k)}$ is the final hybrid sentence/document embedding.

$$\hat{H} = A^{(I)} \oplus H \tag{4.9}$$

$\hat{H}$ is provided as the input to GRU to generate the final encoded representation of input sentence segments or documents as provided. GRU learns the concise representation of text input with labels' attention scores and text embedding providing more information to the network for classification.At time t, the hidden state corresponding to each sentence can be formulated as:

$$\begin{aligned} \overrightarrow{h_i} &= \overrightarrow{GRU}(\overrightarrow{h}_{t-1}, \hat{h_t}), t \in [1, r+k], i \in [1, n] \\ \overleftarrow{h_i} &= \overleftarrow{GRU}(\overleftarrow{h}_{t-1}, \hat{h_t}) \\ h_i &= \overrightarrow{h_t} \oplus \overleftarrow{h_t} \end{aligned} \tag{4.10}$$

Final hidden state h_{r+k} is representing the input for classification. h_{r+k} is inputted to feed classifier to predict the confidence score of each label for input. The classifier consists of feed forward neural network with *sigmoid* activation function:

$$\hat{y} = sigmoid(W h_{r+k}^T) \tag{4.11}$$

here W is the trainable parameter for FNN. $\hat{y}$ is the predicted label set for sentence segments and documents. From $\hat{y}$ top p labels are taken based on their probability scores. When, prediction is done for documents, $\hat{y}$ is directly compared with the ground truth label vector y. Whereas, while predicting labels for sentence segments, the label predictions for different segments belonging to one document are combined together to form a final resultant label prediction set that corresponds original document. For a document X_i consisting of t sentence segments s, model predicts labels $\hat{y_{it}}$ for each sentence segment $s_t, (t = 1, ..., n)$. For $t = (1, ..., n)$ all top m labels from $\hat{y_{it}}$ are combined to obtain $\hat{y_i}$. $\hat{y_i}$ is ordered in descending order of each label's frequency.

4.2.3 Loss Function

A lot of works in extreme multi label classification have used loss functions that treats every label equally such as Binary Cross Entropy (BCE). In extreme multi label classification, dataset is highly imbalanced with few labels having more frequency than other labels. These infrequent labels with very few training instances are tail labels and their frequency distribution follows power law distribution. With cross-entropy losses or with any other common classification losses, classifier is able to learn frequent labels much easily than infrequent labels as these losses penalizes classifier equally for misclassification of all labels. Classifiers can achieve higher score metrics by predicting head (frequent) labels well and eliminating tail labels but this behaviour is detrimental in real world applications. Tail labels contain descriptive information about the documents not found in head or torso labels and exhibits label space's diversity. Hence it is quite important to get rid of popularity bias in the model's prediction results and influences model to predict tail labels as well.

One way to make classifier learn infrequent labels well is by penalizing classifier more for misclassification of infrequent labels than of frequent labels. This could be achieve by giving more weightage to rare or tail labels than frequent labels while computing loss. Focal loss and propensity loss functions attempts to down weight frequent labels while giving more weights to tail labels. Focal loss

achieves this with the help of focussing parameter, γ, that increases the penalty for missclassification. It understands tail labels will be misclassified more number of times than frequent labels where γ parameter would increase the loss penalty for such labels. Propensity loss functions down weights frequent labels by dividing individual labels losses by a coefficient which is directly related to their frequency, p_k.

Focal loss was introduced in [56] and is an improved version of Cross-Entropy Loss (CE). In multi-label classification, the classification task is reduced to a series of binary classification tasks for each label. Given K labels, the network outputs one logit for each label, z_k, which is independently activated by the sigmoid function $\sigma(z_k)$. Let's denote y_k as the ground truth for label k. The total classification loss, L_{tot}, is obtained by aggregating a binary loss from K labels:

$$L_{tot} = \sum_{k=1}^{k=K} L(\sigma(z_k), y_k) \tag{4.12}$$

where L is binary cross entropy loss per label k, given by:

$$L = -y_k L_+ - (1 - y_k) L_- \tag{4.13}$$

L_+ and L_- are positive and negative loss parts i.e. $L_+ = log(\sigma(z_k))$, $L_- = log(1-\sigma(z_k))$. From [56], focal loss is obtained by setting L_+ and L_- as:

$$\begin{aligned} L_+ &= \alpha(1-p)^\gamma log(p) \\ L_- &= \alpha p^\gamma log(1-p) \end{aligned} \tag{4.14}$$

where $p = \sigma(z_k)$, α is the weight of the label and γ is the focussing parameter. By setting $\gamma = 0$, binary cross entropy is yield. As $p \to 1$, the factor $(1-p)^\gamma$ tends to zero and the loss for well classified examples is down-weighted. α parameter could take a set of values corresponding to set of labels. To facilitate model to focus on hard-to-classify labels, α could get assign a greater value and frequent well classified examples could have smaller α values

In addition to having tail labels in XMC datasets, it has been demonstrated that while learning to assign tail labels, one must additionally account for missing labels in the training data due to huge label space. A human annotator cannot explore every potential label when selecting which labels to assign to a given data point.Rather, human annotators on examining the record assigns a set of relevant labels that spring to mind. It is difficult to manually verify for the existence or absence of each label for each document in a dataset where the labels for each example are picked from a label space with millions of elements, thus some instances will have missing labels. Worse, the likelihood of a label being absent is greater for tail labels than for head labels. This results into incomplete ground truth label vectors for each record.Due to this, propensity model is introduced in [1] which advocates using propensity model of each label in loss function and evaluation ranking metrics. The propensity score for label *k*, p_k is modelled as a sigmoidal function of the frequency of label *k* as:

$$p_k = P(y = 1 | y^* = 1) = \frac{1}{1 + Ce^{-Alog(d_k + B)}} \tag{4.15}$$

where d_k is the number of documents that are indexed with label *k* in the training set. Parameters *A* and *B* depends on the meta data of the specific dataset in use and $C = (log(N) - 1)(B + 1)^A$ where *N* is size of training set. [24] proposed to use $A = 0.55$ and $B = 1.5$ in case the meta data is not available for the dataset and same has been used in this project as well for both the datasets. $\mathbf{y}^*$ is the complete but unobtainable ground truth label vector without any missing labels. Propensity

model for a label can be pulled into original loss functions by dividing the ground truth values of label for each document by p_k [1], i.e. $y_k = \frac{y_k}{p_k}$. Propensity coefficient can be included in the focal loss as:

$$L_{tot} = \sum_{k=1}^{k=K} L(\sigma(z_k), \frac{y_k}{p_k}) \tag{4.16}$$

where L can be expanded according to equations 4.13 and 4.14. This experiment tries to find whether loss functions treating all labels equally perform better or worse than loss functions that treats rare frequent with more importance than frequent labels.

4.3 Sentence Level Prediction to Document Level Prediction

This project answers whether predicting labels at the sentence level gives better performance than predicting labels at the document level. On one side of this comparison, there is a model trained and infer on the documents itself. On the other side, a second model is trained on the sentence segments of the documents and infers on the same. This model's predictions on the sentence segments is needed to be converted for the documents to compare the performances of both the models at the same document level. This also facilitates the comparison with the previous XMTC research works since their scores are also reported at the document level too. Real world XMTC related applications also have labels assigned at the document level. All these reasons compel to lay out strategies to derive documents level predictions from the sentence segments predictions. There are 3 strategies identified to generate document level label predictions from the corresponding sentences' predictions. In each sentence prediction vector, top nine labels are kept as relevant labels for the sentence. The number nine is selected since the evaluation at the document level is done until the rank 7 in the prediction vector.

- The first strategy is **voting strategy** where all the nine labels of the sentences' prediction are considered to form the label prediction vector for the document. The labels from all the sentences predictions of a document are aggregated and sorted in reverse order based on their counts in the aggregated set. This strategy works on the idea that if a certain label has occurred multiple times in the sentences' prediction vector belonging to a document, then that label related relevant information is present multiple times across the document. Hence, the label with highest count has its relevant information present in the document with highest amount and hence it is most relevant label of the document. This strategy is called voting strategy since all the labels of sentences' prediction vector votes in the form of their count to form the prediction vector for the parent document.

- The second strategy is **probability strategy** which is similar to voting strategy. Instead of labels' count like in voting strategy, labels' probabilities are considered to order the labels for document level prediction vector. All the labels of the sentences' prediction vector are aggregated and their probabilities are taken to sort them in descending order. In the cases where labels have occurred more than once in the aggregated set, their probabilities average is considered. This method consider label probability instead of its count to avoid labels with high count but low probability from coming in top ranks. Intuitively, if labels have lower probability then it is not much relevant to the document hence it should not be in top ranks. On the contrary if the label has occurred few times or even once across all the sentences' prediction vector of a document but has high probability it implies that label is relevant to the sentence in whose prediction vector it has occur and hence relevant to the document too.

- Third and last strategy is **first label strategy** and it considers only one top-most label from each sentence prediction vector to form the document level prediction vector. These top-most labels are aggregated and are ordered based on their probabilities. If a label has occurred at the top of sentences prediction vector belonging to a document multiple times then only its highest probability is considered. It is the simplest of three strategies by considering only the most relevant labels of each sentence to form document level label prediction set. This strategy works on the idea that if a label has highest probability for a sentence and since sentence is part of the document, that label is also relevant to the document with high confidence due to its high probability score.

4.4 Evaluation Metrics

Multi class and multi label classification tasks use *accuracy* as the evaluation metric often. However, since the dataset tends to be highly imbalanced in extreme multi label classification problems, model would easily classify *majority classes*. As a result, model can attain better accuracy score overall by classifying just majority classes correctly while ignoring minority classes. Due to this *accuracy* does not provide entire story related to the model and its performance. This precludes the use of *accuracy* metric.

Each record in XMTC problem has small set of relevant labels that need to be predicted though the label space is huge. As a small set of labels is required to be predicted compared to the huge label space, predicted labels are sorted in decreasing order of their probability of being a relevant label to the document and a ranked list of predictions is generated. In case of ranked lists of predictions, ranking metrics such as Precision@n is a good choice and as been extensively used in extreme multi-label problems. Precision@n indicates how many labels out of top n labels are correct and relevant labels. The most often n values used for ranked metrics are $\{1,3,5,7\}$. For a dataset, let K be total number of labels, $\mathbf{y} \in \{0,1\}^K$ be *annotated* ground truth label vector for a test document and $\hat{\mathbf{y}} \in R^K$ be the predict score vector. Then Precision@n (P@n) is given by:

$$P@n(\mathbf{y}, \hat{\mathbf{y}}) = \frac{1}{n} \sum_{k \in r_n(\hat{\mathbf{y}})} \mathbf{y}_k \tag{4.17}$$

where $r_n(\hat{\mathbf{y}})$ is the set of rank indices of the annotated relevant labels among the top-*n* portion of the predicted ranked list for a document. , and $||y||_o$ counts the number of labels in the annotated ground truth label vector $\mathbf{y}$.

As described earlier how XML datasets has incomplete ground truth label vectors due to huge label space faced by human annotators. Due to this, propensity model is introduced in [24] and encourages to use it in evaluation metrics as well. Eqn 4.15 represents the propensity score p_l for each label l. This propensity score is used with Propensity score for each label *l* i.e. p_l is used with ranking metrics as Precision@n and called as Propensity Score Precision (PSP@n). PSP@n is formulated as:

$$PSP@n(\mathbf{y}, \hat{\mathbf{y}}) = \frac{1}{n} \sum_{k \in r_n(\hat{\mathbf{y}})} \frac{\mathbf{y}_k}{p_k} \tag{4.18}$$

For M test data points, propensity score precision could be defined as:

$$\zeta@n(\hat{\mathbf{y}}) = \frac{-1}{M} \sum_{i=1}^{M} PSP@n(\mathbf{y}, \hat{\mathbf{y}}) \tag{4.19}$$

Note that ζ could be greater than 1 due to the propensity scores. Therefore, results reported under experiments section are $100 * \zeta@n(\hat{\mathbf{y}})/\zeta@n(\mathbf{y})$ for $PSP@n$.

In extreme multi-label problems, one of the challenge is to avoid model from predicting frequent labels often and expect it to predict relevant but rare or unseen labels. This characteristic of model predicting variety of labels can be evaluated with the second evaluation metric coverage@n. It denotes the percentage of unique labels predicted across the test dataset at top n=1,3,5,7. Coverage@n $(C@n)$ is formulated as:

$$C@n = \frac{\sum_{t=1}^{T} U(\hat{\mathbf{y}}_t, n)}{\sum_{t=1}^{T} U(\mathbf{y}_t, n)} \tag{4.20}$$

where $\hat{\mathbf{y}}$ is predicted label set, $\mathbf{y}$ is ground truth label set and $U()$ returns number of unique labels from $\hat{\mathbf{y}}$ and $\mathbf{y}$ top $n = 1, 3, 5, 7$.

Chapter 5

Experiments and Results

This chapter lays out the experiments done with their experimental-settings. This project proposes **L**abel **A**ware **F**eatures with **F**ocal Loss (LAFF) model which is evaluated with PFastreXML [24]. Experiment 5.1 evaluates LAFF model. The first research question is about focal loss and whether using focal loss gives better results in extreme multi-label setting. This experiment is discussed in section 5.2 where focal loss is compared with binary cross entropy loss function. The second research question is whether sentence level prediction of labels gives better accurate results than document level prediction. This is answered by obtaining the results for sentence level prediction and comparing it with document level prediction. Sentence level prediction experiment is explained in section 5.3. The second research question have sub questions which are explained further in the same section.

5.1 Document Level Prediction: LAFF Model Evaluation

This experiment is concern with the evaluation of LAFF model performance with respect to PFastreXML [24] to validate how good or bad LAFF model is. LAFF model is trained with EURLEX4K dataset, an older version of EURLEX57K dataset and corresponding results are compared with EURLEX4K results from PFastreXML. In this project EURLEX57K dataset has been used in spite of EURLEX4K being used profoundly in previous works is because EURLEX4K dataset contains pre-processed documents without any sentence delimiter. EURLEX4K dataset contains stemmed words without any punctuation and it becomes difficult to generate the coherent sentence segments from it. Hence, this dataset is only used to analyze the LAFF model performance with the previous work of PFastreXML.

Documents of EURLEX4K dataset are fed to the LAFF as the input. There are 3,956 labels in the dataset and label embeddings are generated of dimension 100. There are 15,449 data points in training dataset and 3,865 data points in validation and test dataset. The model is trained with focal loss function and with a learning rate of $1e^{-05}$. Momentum and weight decay is set to 0.9 and 0.1 respectively. Focal loss's hyper-parameter is set to $\gamma = 4$ and α to each label's propensity score.

The results of this experiment is shown in table in fig 5.1. It can be seen that PFastreXML algorithm performs better than LAFF at all ranks. One of the reason for bad performance of LAFF is due to already pre-processed documents in EURLEX4k dataset. EURLEX4K dataset contain documents which already stemmed and cleaned from stop words. LAFF utilizes BERT which requires its input to be present in unprocessed form to learn and generate a better contextualize representation. PFastreXML is a tree classifier that has each classifier trained at each non-leaf node to focus on only

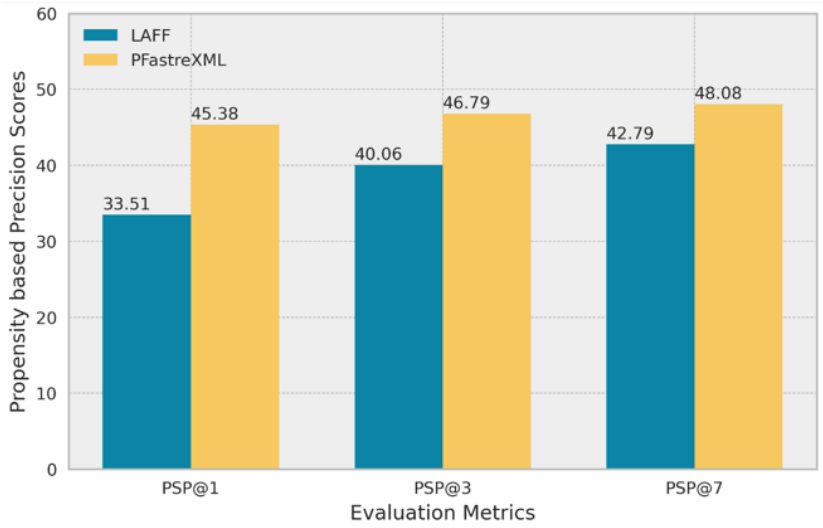


Figure 5.1: LAFF v/s PFastreXML scores

few labels. In a way, there are number of classifiers in PFastreXML, and each classifier deals with only few labels. Additionally, PFastreXML being an ensemble tree classifier takes vocabulary token for training and does not need a contextual representation of document to train with. Due to all these reasons, LAFF model perform worse than PFastreXML. The difference between scores at $PSP@1$ is high and it keeps decreasing at $PSP@3$ and $PSP@5$. This implies that LAFF model's performance gets better with respect to PFastreXML scores and it has generated more rare labels than at $PSP@1$.

5.2 Document Level Prediction: Focal Loss v/s BCE

This experiment answers question whether focal loss utilisation gives better predictive performance in XMTC setting. To answer this question a baseline is created where LAFF model is trained at the document level with binary cross entropy (BCE) loss function. This baseline is created for both EURLEX57K and Medline datasets. Documents are directly served to the LAFF model as input and baseline scores are generated at the document level. On the other side of the comparison, another LAFF model is trained at document level with focal loss function.

Both EURLEX57K and Medline datasets are cleaned as mentioned in their respective section in 3. EURLEX57K dataset is cleaned and labels that are not present in either train, validation or test datasets are removed. From the initial number of 7201 labels, 4531 labels are kept. Since BERT can capture the semantic of input that is not processed and cleaned from punctuation and stopwords better, document text does not undergo any kind of pre-processing. Document text is provided to the BERT tokenizer that convert it into numerical tokens which are then fed to the model. The label embeddings are generated from the label graphs constructed from 4,531 labels. Each node in the graph is a label and edge exist between them if they have been assigned together to a document in training dataset. Graph Convolutional Networks are employed on the label graph to generate each label node embedding of dimension-size 100. Document text and label embeddings are fed to the model for training and prediction of labels. Two LAFF models; one with BCE and another with focal loss are trained for 105 and 110 epochs respectively. Learning rate is set to $1e^{-05}$, momentum to

0.9 and weight decay to 0.1. Focal loss' γ hyper-parameter is set to 3 and α is set to each label's propensity score.

Similarly, for Medline dataset, document text is not pre-processed. From 60,682 labels, 26,725 labels from eight branches are kept in the dataset.As number of labels is huge, label embeddings created are of size 300 [22]. Similar to EURLEX57K experiment, two LAFF models are trained for with BCE and focal loss. The model was trained for 30 epochs for 403,620 documents. The other hyper-parameters are same as EURLEX57K dataset experiment.

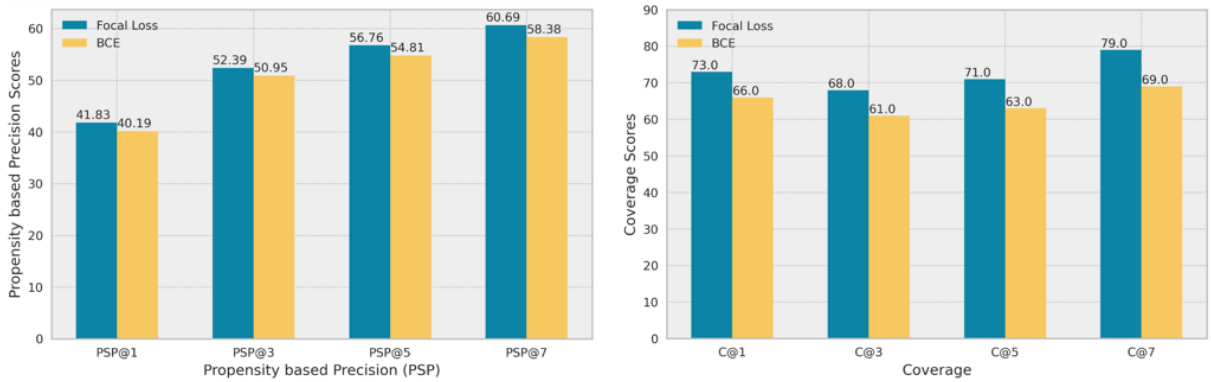


Figure 5.2: Focal Loss v/s BCE scores for EURLEX57K dataset.

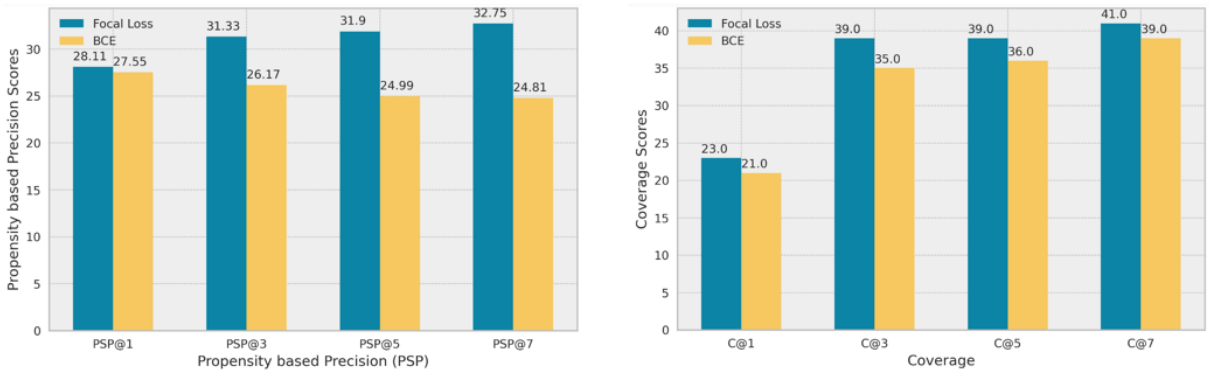


Figure 5.3: Focal Loss v/s BCE scores for Medline dataset.

The results for both EURLEX57K and Medline dataset are shown in fig 5.2 and 5.3 respectively. The BCE baseline scores are lesser than the focal loss results at the document level prediction. The difference between $PSP@1$ score is not big between the two results but the following PSP scores at various rank n have higher difference between them. This implies that with focal loss model is able to predict relevant and rarer labels. The propensities used for each label in focal loss helped the model to focus more on less frequent labels due to high loss for infrequent labels. This allowed the model to get train on highly imbalanced dataset well and treat dataset as balanced. Coverage metric shows percentage of unique labels predicted. One of the challenges in Extreme Multi-label Text classification (XMTC) setting is model predicts few labels frequently due to labels' power-law distribution. With higher $coverage$ scores for both EURLEX57K and Medline dataset, it can be implied that due to focal loss model is predicting more unique labels than BCE loss trained model.

5.3 Sentence Level Prediction

Another question this project answers is whether predicting labels at the sentence level gives better performance than predicting labels for the document level. To answer this question we need to train LAFF model with the sentence segments and infer at the sentence level from the model. Later, these sentence level prediction vectors is needed to be converted to the document level prediction vector for the parent document and model's performance evaluation is done. These results are compared with document level trained model which is already explained in section 5.2. From section 5.2, focal loss related results are considered for comparing with the sentence level predictions results.

Before training the model with the sentence segments of the document, sentence level training dataset is required which has ground truth corresponding to the sentence segments. Section 4.1 discusses about the methodology that generates labels at the sentence segments. Following experiment discusses about this methodology experiment along with the evaluation for the generated dataset.

5.3.1 Sentence Level Label Generation with Evaluation

As stated earlier, for training LAFF model with sentence segments, sentence's level ground truth is required which is not available. All the supervised learning datasets consists of labels assigned to the documents. Methodology explained in section 4.1 lays out the process of generating sentence level dataset from any supervised dataset. It is inspired from DXML [22] and assigns labels to the sentence segments after bringing the labels and documents in the same embeddings space. Sentence level ground truth is generated for both EURLEX57K and Medline datasets.

A label co-occurrence graph is built for both the datasets from which label embeddings are generated by Graph Convolutional Networks. As both dataset have different number of labels, different size label embeddings are generated [22]. Since EURLEX57K has lesser number of labels i.e., 4,513, label embeddings of dimension 100 is generated. Whereas Medline dataset has 26,725 labels for which label embeddings of size 300 are created. EURLEX57K dataset has 4,513 labels and Medline dataset has 26,725 labels. Documents are fed to BERT to generate its contextualize embeddings. After having label and documents embedding representation, a weight matrix learned that maps documents to labels. This weight matrix represented by a feed forward neural network takes documents' contextualize embeddings from BERT and generates the label mapped representation. This mapped version of document and document's corresponding labels are brought together in same embedding space by minimizing their distance with the help of embedding loss function with equation 4.1. BERT and neural network are trained with the help of embedding loss function. For both the datasets, learning rate is set to $1e^{-05}$, momentum to 0.9 and weight decay to 0.5. After model training, EURLEX57K and Medline document's embedded representation are divided into 50 and 100 clusters respectively with $k - means$ algorithm. Number of clusters formed are in proportion with the number of documents. Since, EURLEX57K has 57,000 train documents 50 clusters are constructed whereas Medline has 513,620 documents hence 100 clusters are formed.

Once the documents and labels are brought in same embedding space, labels can be predicted for the sentence segments which would be considered as their ground truth labels for further LAFF model training. Train documents of EURLEX57K and Medline datasets are segmented into sentence segments with Blackstone and Deepsegment libraries as explained in section 3.1.2, 3.2.2 respectively. These sentence segments of the documents are passed to the BERT and neural network to

obtain their embedded representation and closest labels are assigned based on the algorithm 2 For each sentence segments 7 closest clusters are found and labels associated with these clusters are assigned to the sentence segments provisionally. Out of these provisionally assigned labels to the sentence segments, labels which are already present in the parent documents ground truth are selected as ground truth labels for the sentence segments. In this way, ground truth labels are assigned to the sentence segments of the document. If none of the provisionally assigned labels are present in the document ground truth, no label gets assigned and sentence segment is removed from the final dataset.

Evaluation

The quality of model trained and prediction can be gauged with how much percentage of document labels are actually assigned to the sentence segments. The labels which are assigned to sentence segments are common between provisional assigned label set and document's ground truth. Hence, if a model is well trained, provisional label set of sentence segment would ideally contain labels from document's ground truth. Hence, if a huge amount of document labels are getting assigned to the sentence segments it concludes model has been trained well. Hence to evaluate the generated dataset, an average percentage of document ground truth labels utilized is computed.

$$C = \sum_{i}^{N} \frac{P(D_i)}{N} \tag{5.1}$$

where $P(D_i)$ represents percentage of document D_i ground truth labels are assigned to the sentence segments. N represents total number of documents. EURLEX57K dataset has 80% of its document's ground truth labels being assigned and 73% for Medline documents. This represents model's generated sentence segments embedded representation lies closer to the document's embedded representation if they have same context otherwise in spite of same context less number of labels would be assigned from document's ground truth.

5.3.2 Training Dataset Generation and Model Training

The dataset generated from previous section where labels were assigned to the sentence segments as their ground truth labels can be considered as the training dataset for sentence level model training. In the generated dataset, only sentence segments whose atleast one provisionally assigned label present in the document ground truth are kept. If a label is present in both document ground truth and provisional label set it intuitively implies that the concerned sentence segment has information which is semantically similar to the document information and hence it is document relevant. In a document there would be sentence segments which are not much informative about the document's context and would have different information altogether not relevant with the document. Hence, in the above dataset all the important and document relevant sentence segments are kept and all unimportant, *noisy* sentence segments have been removed. Unfortunately, this filtering out of noisy, irrelevant sentence segments are not present in the test and validation dataset. All the sentence segments of test and validation dataset would be considered since information about *noisy* sentence segments would not be available before hand.

Model would infer relevant labels for the sentence segments of the test and validation dataset. Since these datasets contain document irrelevant sentence segments, their relevant labels would be unrelated to the document. Considering these sentence segments labels in performance evaluation would worsen the performance of the model. To have a fair evaluation of the model, model should be

be able to detect irrelevant and *noisy* sentence segments to avoid them being considered from the evaluation. To make model able to detect *noisy* sentence segments, *noisy* sentence segments are added in the existing training dataset. A smaller portion of already ignored *noisy* sentence segments are added in the training dataset after assigning a different "NOISE" label. While training LAFF model would be able to learn about *noisy* sentence segments and would detect them in test and validation dataset. In the EURLEX57K dataset, 15,505 *noisy* sentence segments are added which is equal to the highest frequency a label has in sentence level training dataset. Similarly, in Medline dataset 65,245 *noisy* sentence segments are added. Following sections 5.3.2 and 5.3.2 evaluate dataset with "NOISE" label and the three strategies for generating document level prediction from sentence level prediction. Before going through these evaluations, let's see how test and validation dataset for sentence level are generated since evaluation is done on these datasets and how inference is done.

Sentence Level Validation and Test Dataset with Inference

Validation and test dataset for sentence level prediction are generated by segmenting the documents into sentence segments. For EURLEX57K dataset, Blackstone library has been used. DeepSegment library has been used for Medline dataset as explained in section 3. Labels are not generated for sentence segments since evaluation would be done at the document level.

After training the model with sentence level training dataset, sentence segments of documents from test dataset are given as the input to the model and it predicts relevant label corresponding to the sentence segment. Since, evaluation is done at the document level, the prediction at sentence segments level is used to derive document level prediction by three different strategies explained in section 4.3. First strategy is voting strategy where all top predicted labels of sentences are considered for document level prediction set. In voting strategy, document level prediction set is ordered based on the labels counts across all the sentences prediction set of the document. Second strategy is probability strategy where document level labels are ordered based on the labels' probabilities. In case there are multiple occurrences of labels, their probabilities' average is computed. In last strategy only top-most label from each sentence prediction set is considered for the document level prediction set and are ordered based on their probabilities.

NOISE Label Evaluation

This section evaluates whether having *noisy* sentence segments associated with a "NOISE" label augments the performance of the model or not. This evaluation is done on both EURLEX57K and Medline dataset. Two sets of training dataset is created for both the datasets: one with *noisy* sentence segments with "NOISE" label and another without the *noisy* sentence segments. With "NOISE" label, a new label co-occurrence graph is constructed and new set of label embeddings are generated with Graph Convolutional Networks (GCN). EURLEX57K has label embeddings of dimensions 100 and Medline has label embeddings of dimension 300. With two set of training datasets of each dataset, two LAFF models are trained with learning rate of $1e^{-05}$, weight decay of 0.1 and momentum of 0.9. Focal loss hyper-parameter γ is set to 3 and α are label propensity scores. In EURLEX57K dataset, training dataset with "NOISE" label has model trained for 30 epochs and without "NOISE" label model is trained for 33 epochs. In Medline dataset, model is trained for 20 epochs with "NOISE" label training dataset and for 25 epochs without "NOISE" label.

The evaluations for both the EURLEX57K and Medline dataset are shown in figure 5.6 and 5.7 respectively. Since, model inference is done at the sentence level, document level predictions are derived using the three strategies explained in section 4.3.

Sentence to Document Prediction Strategies	Training Method	PSP@1	PSP@3	PSP@5	PSP@7	C@1	C@3	C@5	C@7
Voting Strategy	"NOISE" Label	**57.77**	**49.81**	**47.37**	**50.87**	**0.77**	0.63	**0.65**	**0.67**
	Without "NOISE" Label	53.9	47.89	46.39	50.40	0.70	0.66	0.63	0.64
Probability Strategy	"NOISE" Label	**59.10**	**48.71**	42.57	45.02	**0.78**	0.62	0.58	0.60
	Without "NOISE" Label	59.0	47.44	**43.94**	**45.55**	0.76	**0.65**	**0.62**	**0.63**
First Label Strategy	"NOISE" Label	63.29	**48.09**	**39.07**	**37.05**	**0.805**	0.62	0.55	0.53
	Without "NOISE" Label	**63.89**	47.56	38.7	36.5	0.79	**0.66**	**0.57**	**0.56**

Figure 5.4: EURLEX57K: "NOISE" label v/s Without "NOISE" label

Sentence to Document Prediction Strategies	Training Method	PSP@1	PSP@3	PSP@5	PSP@7	C@1	C@3	C@5	C@7
Voting Strategy	"NOISE" Label	**39.27**	**30.62**	**29.38**	**27.74**	**0.46**	**0.38**	**0.394**	**0.376**
	Without "NOISE" Label	37.02	28.36	28.03	27.46	0.45	0.374	0.369	0.37
Probability Strategy	"NOISE" Label	**40.5**	**31.23**	**27.73**	**28.03**	**0.469**	**0.39**	0.37	**0.371**
	Without "NOISE" Label	39.5	29.14	27.41	27.96	0.46	0.378	**0.372**	0.37
First Label Strategy	"NOISE" Label	**43.7**	**32.05**	**30.08**	**27.49**	**0.487**	**0.403**	**0.389**	**0.368**
	Without "NOISE" Label	41.87	29.04	28.8	26.67	0.472	0.38	0.362	0.356

Figure 5.5: Medline: "NOISE" label v/s Without "NOISE" label

In EURLEX57K dataset results, "NOISE" label dataset has performed well across all the ranks in voting strategy. Voting strategy takes into account of label counts for generating the document level prediction vector. This implies that "NOISE" label dataset has resulted into more correct number of labels in the document predictions. In the probability strategy, the first two ranks of $PSP@n$ metric has better results for "NOISE" label dataset. Scores are lower for $PSP@5$ whereas there is not much difference in scores at $PSP@7$. With scores of $PSP@1,3$ and $C@1$, "NOISE" label dataset has performed well. In probability strategy, due to "NOISE" label dataset, correct labels are present at the higher ranks. In first label strategy, for dataset without "NOISE" label dataset is greater for $PSP@1$. but $C@1$ is higher for "NOISE" label dataset. Overall, "NOISE" dataset has better scores than without "NOISE" dataset more number of times. The ranks at which without "NOISE" label dataset has better scores are at lower ranks i.e 5 and above which are not as significant as higher ranks.

In Medline dataset, "NOISE" label dataset has outperformed without "NOISE" label dataset in all the three strategies. It might be due to huge amount of dataset in Medline dataset abling model to generalize more and better. With the "NOISE" label dataset, LAFF model is able to detect irrelevant sentence segment and avoid it from considering them in final evaluation. This helps in focusing only on important and relevant parts of the document for label prediction. Hence, for sentence level model training, training dataset should have *noisy* sentence segments associated with "NOISE" labels. This answer the question of how sentence level training dataset can be used for model training.

Sentence to Document Prediction Strategies Evaluation

In the section 4.3, three strategies are introduced which can be used to generate document level prediction vector from the sentence level prediction vector. Voting strategy consider the label count for document level label prediction vector generation. Similarly, in probability strategy, labels' probabilities are considered. In first label strategy, the label from each sentence segment of the document are considered. In the previous section, it is seen that "NOISE" label dataset has better performance than without "NOISE" label dataset. Hence for the evaluation, results of "NOISE" label dataset are considered for strategies evaluation.

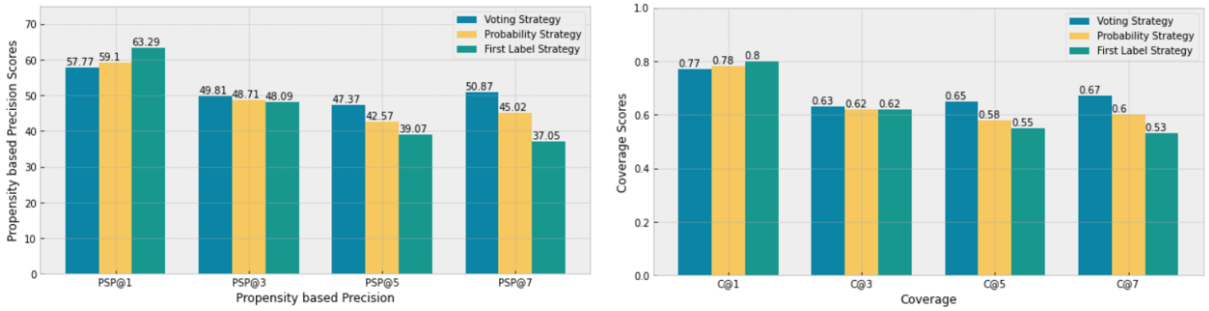


Figure 5.6: EURLEX57K: Voting v/s Probability v/s First Label Strategy

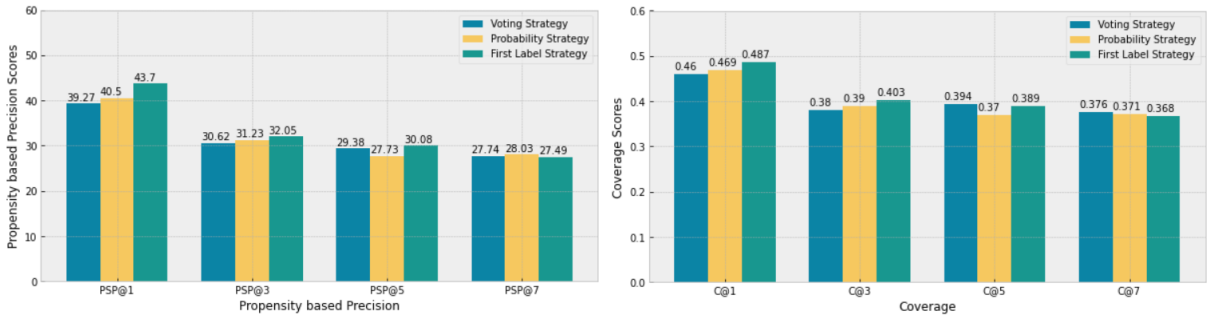


Figure 5.7: Medline: Voting v/s Probability v/s First Label Strategy

In EURLEX 57K dataset, barring $PSP@1$ and $C@1$, voting strategy has performed better than probability and first label strategy. First label strategy has performed best for rank one but its scores drastically decreases for later ranks. Voting strategy seems to be more stable than first label strategy. It's scores from rank 3 to 7 lies in a narrow interval. Same trend is also seen in probability strategy and is more stable than first label but lesser than voting strategy for ranks 3 to 7. Probability strategy surpasses voting strategy at rank 1 but fall behind for other ranks. Voting and probability strategy are similar in the sense that all the labels of sentence segments are considered for final document prediction vector. In Medline dataset, first label strategy show same trend as in EURLEX57K dataset. It's scores drastically decreases after rank 1. But, first label strategy has higher scores than other two strategies at all ranks and makes it best performing strategy. This is due to availability of huge data for model training. This leads to model predicting more sentence relevant and document relevant labels at the first rank for the sentence level. Coverage scores also shows that top-most labels are more unique in numbers than other two.

With the above results it can be implied that there is not just one strategy that is best but all three are best based on different ranks and different datasets. In EURLEX57K dataset , voting and prob-

ability strategies does not have drastic changes in scores as first label strategy has but it surpasses the other two rank 1. At other ranks, voting strategy performs better. In Medline dataset, first label strategy surpasses other two strategies at all ranks.

Until now in sentence level prediction experiment, training dataset has been generated which has *noisy* sentence segments associated with ”NOISE” labels. On evaluation in section 5.3.2 it is found that training dataset with *noisy* sentence segments gives better performance. Later, three strategies of sentence level prediction to document level prediction is evaluated in section 5.3.2 which found all three strategies are better in different situations. The following section evaluates sentence level prediction with document level prediction.

Document Level v/s Sentence Level Prediction Evaluation

The main question this project answer is whether sentence level label prediction gives better predictive performance than the document level label prediction. To answer this, one model is trained with documents and another with the sentence segments. Both these models uses focal loss for training. Similar to experiment in section 5.2 LAFF model is trained with documents. Since,sentence level training dataset that has *noisy* sentences gives better results as per section 5.3.2, this dataset is used for sentence level model training. The model is trained similar to how sentence level model is trained in section 5.3.2. Figure 5.8 and 5.9 gives results for EURLEX57K and Medline datasets for sentence level prediction compared with document level prediction.

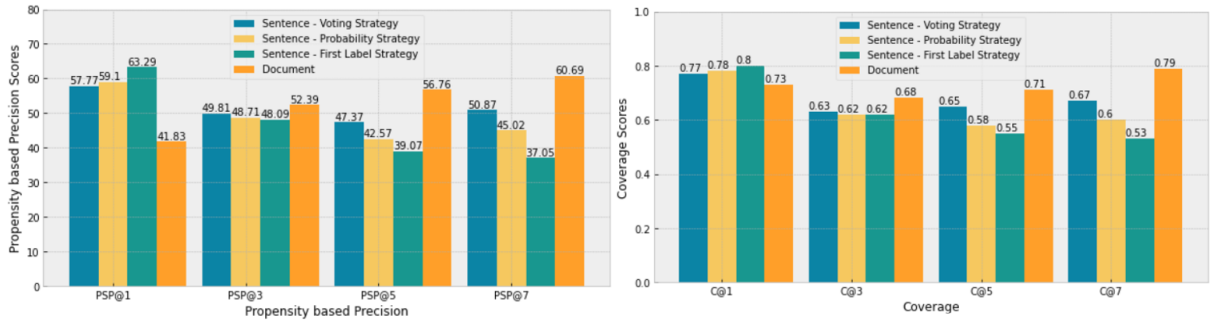


Figure 5.8: EURLEX57K: Sentence Level Prediction v/s Document Level Prediction Strategy

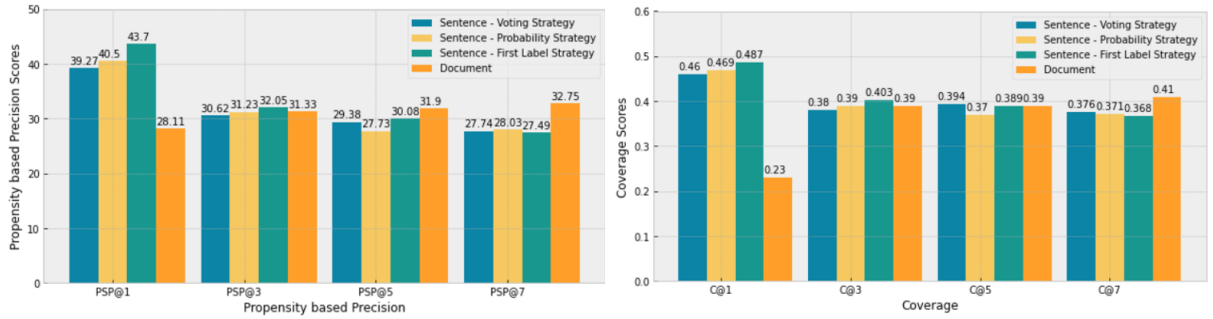


Figure 5.9: Medline: Sentence Level Prediction v/s Document Level Prediction Strategy

From the above results, it is clearly seen that sentence level prediction performs better than document level prediction for rank 1. All the three sentence prediction to document prediction have

outperformed document level prediction for rank 1. For later ranks, document level prediction perform better. In Medline dataset, for rank 3, sentence level is slightly better than document level but the difference is not too high. Overall, it can be said that sentence level prediction is better for rank 1 but for later ranks document level prediction is better. Worse scores for sentence level prediction for rank 3 on wards can be due how ground truth of sentence segments are generated and how sentence segments are aggregated to form final document level prediction. Since, sentence level ground truth is a derived dataset there could be some noise in the dataset. A document intuitively have more sentence segments than the labels assigned to them. These sentence segments though are informative about the document it is not necessary that these are associated with the labels assigned to the document. As stated in [24], each document in XMTC have missing labels, sentence segments would have relevant labels which are missing from the document's ground truth. Additionally, aggregation of sentence labels for formulation of document prediction needs a more efficient method than employed here. In the current aggregation strategies there might be document relevant labels which would be getting ignored due to low count or low probability.

The better performance of sentence level trained model than document-level trained model can also be observed with an actual respective prediction example. The image below 5.10 shows a test document from the EURLEX57K dataset. The test document text has been highlighted with the possible important phrases or keywords which an annotator would refer to assign relevant labels. The highlighted phrases majorly talk about specific different third countries and items being imported from them. Overall, the context of this document is about which items to import. The actual assigned

'15.10.2011 EN Official Journal of the European Union L 270/48COMMISSION IMPLEMENTING DECISION of 14 October 2011amending and correcting the Annex to Commission Decision 2011/163/EU on the approval of plans submitted by third countries in accordance with Article 29 of Council Directive 96/23/EC(notified under document C(2011) 7167)(Text with EEA relevance)(2011/690/EU) THE EUROPEAN COMMISSION, Having regard to the Treaty on the Functioning of the European Union, Having regard to Council Directive 96/23/EC of 29 April 1996 on measures to monitor certain substances and residues thereof in live animals and animal products and repealing Directives 85/358/EEC and 86/469/EEC and Decisions 89/187/EEC and 91/664/EEC\xa0(1), and in particular the fourth subparagraph of Article 29(1) and Article 29(2) thereof. Whereas:(1) Directive 96/23/EC lays down measures to monitor the substances and groups of residues listed in Annex I thereto. Pursuant to Directive 96/23/EC, the inclusion and retention on the lists of third countries from which Member States are authorised to import animals and animal products covered by that Directive are subject to the submission by the third countries concerned of a plan setting out the guarantees which they offer as regards the monitoring of the groups of residues and substances listed in that Annex. Those plans are to be updated at the request of the Commission, particularly when certain checks render it necessary.(2) Commission Decision 2011/163/EU\xa0(2) approves the plans provided for in Article 29 of Directive 96/23/EC ('the plans') submitted by certain third countries listed in the Annex thereto for the animals and animal products indicated in that list. Decision 2011/163/EU repealed and replaced Commission Decision 2004/432/EC of 29 April 2004 on the approval of residue monitoring plans submitted by third countries in accordance with Council Directive 96/23/EC\xa0(3).(3) In the light of the recent plans submitted by certain third countries and additional information obtained by the Commission, it is necessary to update the list of third countries from which Member States are authorised to import certain animals and animal products, as provided for in Directive 96/23/EC and currently listed in the Annex to Decision 2011/163/EU ('the list').(4) Belize is currently included in the list for aquaculture and honey. However, Belize has not provided a plan as required by Article 29 of Directive 96/23/EC. Therefore, Belize should be removed from the list.(5) Ghana has submitted a plan for honey to the Commission. That plan provides sufficient guarantees and should be approved. Therefore, an entry for Ghana for honey should be included in the list.(6) India has now carried out corrective measures to address the shortcomings in its residue plan for honey. That third country has submitted an improved residue plan for honey and a Commission inspection confirmed an acceptable implementation of the plan. Therefore, the entry for India in the list should include honey.(7) Madagascar has submitted a plan for honey to the Commission. That plan provides sufficient guarantees and should be approved. Therefore, honey should be included in the entry for Madagascar in the list.(8) Mauritius is currently included in the list for poultry but with a reference to footnote 2 in the Annex to Decision 2011/163/EU. That footnote restricts such imports to those from third countries using only raw material either from Member States or from other third countries approved for imports of such raw material to the Union, in accordance with Article 2 of that Decision. However, Mauritius has not provided the required guarantees for the plan for poultry. Therefore, the entry for that third country in the list should no longer include poultry.(9) Turkey has submitted a plan for eggs to the Commission. That plan provides sufficient guarantees and should be approved. Therefore, eggs should be included in the entry for Turkey in the list.(10) The entry for Singapore in the list includes aquaculture but with a reference to footnote 2 in the Annex to Decision 2011/163/EU. However, in the Annex to Decision 2004/432/EC, as amended by Commission Decision 2010/327/EU\xa0(4), there is no reference to footnote 2 as Singapore submitted an approved plan for aquaculture. The Commission has not been advised of any change since the approval of that plan. Therefore, the entry for that third country in the list should be corrected by deleting the reference to that footnote for imports of aquaculture. For reasons of legal certainty, the entry for Singapore should apply retroactively from 15 March 2011, the date of application of Decision 2011/163/EU when the error in the entry regarding Singapore occurred. The competent authorities of the Member States have been informed accordingly and no disruption to imports has been reported to the Commission.(11) The Annex to Decision 2011/163/EU should therefore be amended accordingly.(12) The measures provided for in this Decision are in accordance with the opinion of the Standing Committee on the Food Chain and Animal Health, The Annex to Decision 2011/163/EU is replaced by the text in the Annex to this Decision. This Decision shall apply from 1 November 2011. However, the amendment concerning the entry for Singapore shall apply from 15 March 2011.This Decision is addressed to the Member States.'

Figure 5.10: A sample EURLEX57K document

labels and its predicted labels are shown in the table 5.11. The actual assigned labels have labels for countries mentioned in the document text along with different poultry items related labels which are supposed to be imported. The *document-level* trained model predicted more general labels which are related to the context of the document text. It predicted labels such as *import (EU), animal product, live animal, third country*, and *veterinary inspection*. The other predicted labels are vaguely

related to the document's context and their correctness is debatable. Whereas the labels predicted with *sentence-level* trained model are general as well as specific labels. It predicted *import (EU), veterinary inspection, third country* like general labels along with *India, Singapore* like specific labels. It was also able to predict *aquaculture* label which is not present in the actual assigned labels but has its mention in the document as one of import items. While predicting correct and rare labels, *sentence-level* model also predicted some loosely document related labels such as *exchange of information, intra-EU trade, administrative cooperation*. But overall, *sentence-level* model is able to predict rare and specific labels which *document-level* model did not able to.

Actual Labels	Document-Level Prediction	Sentence (Voting Strategy) Prediction	Sentence (Probability Strategy) Prediction	Sentence (First Label Strategy) Prediction
food contamination, Singapore, egg, import license, fishery product, live poultry, India, Mauritius, veterinary inspection, Belize, Madagascar, surveillance concerning imports, Turkey, Ghana, import (EU), honey, food safety	import (EU), animal product, live animal, third country, health control, international civil servant, rural migration, veterinary inspection, customs inspection	veterinary inspection, plant health control, foodstuffs legislation, Singapore, poultry, veterinary legislation, food inspection, intra-EU trade, third country,	veterinary inspection, health control, import (EU), third country, aquaculture, exchange of information, hormone, administrative cooperation, tariff quota	veterinary inspection, third country, health control, India, Singapore, aquaculture, administrative cooperation, exchange of information, intra-EU trade

Figure 5.11: Labels sets for the test document

This experiment clearly found *sentence-level* trained model performing better than *document-level* trained model. The scores for *sentence-level* trained model are not directly compared with the existing state-of-the-art results because this project uses EURLEX57K dataset whereas state-of-the-art used EURLEX4K dataset; an earlier version of EURLEX57K. EURLEX4K dataset documents does not have sentences with any context but a sequence of stemmed and processed words due to which sentences could not be generated. Similarly, in Medline dataset, previous state-of-the-art models used a subset of Medline dataset and there is no standard subset available for evaluation. Due to all these reasons, *sentence-level* trained model's results for both the dataset could not be compared with state-of-the-art results. But, the proposed LAFF model which has been used in *sentence-level* and *document-level* trained model has been evaluated withe one of the state-of-the-art model, PFastreXML, at the document level with the help of EURLEX4K dataset.

Chapter 6

Conclusions and Future Work

6.1 Contributions and Limitations

This project aimed to answer whether sentence level prediction is better than document level prediction in extreme multi-label setting where model tends to get biased toward majority class and in the process answered several other related questions. To execute the project, it require a sentence level training dataset. This project inspired from DXML [22] proposed the methodology to generate ground truth for sentences of a document from document's ground truth labels. This sentence level dataset is generated by bringing text feature space and label space in one domain. Generated dataset is evaluated by means of percentage of labels in document ground truth assigned. It introduced the idea of using a dummy class label depicted by label "NOISE" to ignore the noisy and unwanted sentence segments from the sentence level training dataset. This makes sure model gets train from quality sentence segments and also able to detect unwanted sentence segments in the test and validation dataset. It is found that, using a dummy class label increased the model's performance as all unwanted sentence segments are not considered for evaluation. The labels predicted for sentence segments are aggregated and sorted in reverse order based on 3 different strategies viz. voting, probability and first label strategy. First label strategy where only top-most label of a sentence segment is considered for document level prediction has best score at rank 1 but it drastically decreases for rank 3 on wards. Voting strategy is much a stable method and whose scores lie in a narrow interval but its score at rank 1 is lesser than first label strategy. Overall, none of the three strategies are clear winner and each of them gives good performance based on different situations. Sentence level prediction is then compared with document level prediction. It is found that sentence level prediction gives better results at rank 1 and predicts more rarer labels than document level prediction. However, for rank 3 on wards, document level classification performs better than sentence level. This is might be due to unknown issues in sentence level training dataset as it is a derived one. Similarly, there are issues in generating document level prediction vector from sentence level prediction vector. This project also answer whether focal loss gives better performance in extreme-multi label classification setting and found it does. Focal loss related results are compared with often used binary cross entropy (BCE) loss trained model, and clearly focal loss trained model predicted more infrequent labels than BCE trained model. Focal loss utilized each label propensity score that helped model to treat infrequent labels with more weightage than frequent labels.

6.2 Future Work

For future research, it would be interesting to explore different loss functions in extreme multi-label learning setting. Other than Focal Loss there are class-balanced focal loss and distribution-balanced loss functions. Class-balanced focal loss is an improved version of focal loss which reduces redundant information of *frequent* labels. Whereas, distribution-balanced loss function is another loss function which down-weighs the easy-to-classified labels by utilizing integrated re-balanced weighting and negative tolerant regularization (NTR).The another area which could be explored in future is how can sentence level training dataset methodology be improved. We know that often documents contains sentences which are semantically similar. In this project, these semantically similar sentence segments are kept as individual sentences. It would be interesting to see on concatenation, does model performs better than the current performance. It's also to be seen how semantic similarity between these sentence segments could be computed as it involves selecting an optimal threshold value to consider similar. The basis of attempting to predict at sentence level is to focus on granular information present in the document which is often lost in fixed size document embeddings. Currently in this project, sentence segments have been used. But even sentence segments contains extra noisy information and it would be really interesting to extract only few words or phrases from the documents to be used for model training and prediction. So, another future work would be to extract these important granular informative phrases from the document rather than sentences and associate them with labels. Currently, "NOISE" label have been used to ignore noisy sentences from the test evaluation. Semantically different noisy sentences that are unrelated with each other have been assigned same "NOISE" label in training dataset. It would be interesting to explore assigning different kinds of noise labels to noisy sentences which are clubbed based on their noise semantic similarity. In this way model would be able to learn different types of noise and could detect in test dataset. Another limitation that could be further researched is how to effectively aggregate sentence level labels to form document level predicted labels. In current methodology, every sentence is treated equally while forming document level label predictions but some sentences are more important than others with respect to the document's context. Hence, labels should be considered according to the importance of the sentence to the document.

In other directions, recently, large language models (LLM) have been able to solve then challenging tasks such as Neural Machine Translation (NMT), text generation effectively. It would be interesting to train an LLM model from scratch that not only understands documents but also labels. Existing LLM models learn about documents easily as documents are long. Labels are often one or maximum two words and hence LLM models does not get long input to understand the label's context. Hence, making an LLM model understand labels would be a challenge.

Bibliography

[1] Ian En-Hsu Yen, Xiangru Huang, Pradeep Ravikumar, Kai Zhong, and Inderjit Dhillon. 2016. Pdsparse: A primal and dual sparse approach to extreme multi class and multi label classification. In International Conference on Machine Learning, pages 3069–3077

[2] Grigorios Tsoumakas and Ioannis Katakis. 2007.Multi-label classification: An overview. International Journal of Data Warehousing and Mining (IJDWM), 3(3):1–13.

[3] Rohit Babbar and Bernhard Scholkopf. 2017. Dismec: Distributed sparse machines for extreme multi-label classification. In Proceedings of the Tenth ACM International Conference on Web Search and Data Mining, pages 721–729. ACM.

[4] Ian EH Yen, Xiangru Huang, Wei Dai, Pradeep Ravikumar, Inderjit Dhillon, and Eric Xing. 2017. Ppdsparse: A parallel primal-dual sparse method for extreme classification. In Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pages 545–553. ACM.

[5] D. Hsu, S. Kakade, J. Langford, and T. Zhang. Multi-label prediction via compressed sensing. In Advances in neural information processing systems, 2009.

[6] M. M. Cisse, N. Usunier, T. Artieres, and P. Gallinari. Robust bloom filters for large multilabel classification tasks.In Advances in Neural Information Processing Systems, pages1851–1859, 2013

[7] W. Bi and J. Kwok. Efficient multi-label classification with many labels.In Proceedings of The 30th InternationalConference on Machine Learning, pages 405–413, 2013.

[8] Y. Zhang and J. Schneider. Multi-label output codes using canonical correlation analysis. pages 873–882, 2011

[9] Hsiang-Fu Yu, Prateek Jain, Purushottam Kar, and Inderjit Dhillon. 2014. Large-scale multi-label learning with missing labels. In International conference on machine learning, pages 593–601.

[10] Y.-N. Chen and H.-T. Lin. Feature-aware label space dimension reduction for multi-label classification. In Advances in Neural Information Processing Systems, pages 1529–1537,2012

[11] Yashoteja Prabhu, Anil Kag, Shrutendra Harsola, Rahul Agrawal, and Manik Varma. 2018. Parabel: Partitioned label trees for extreme classification with application to dynamic search advertising. In Proceedings of the 2018 World Wide Web Conference on World Wide Web. International World Wide Web Conferences Steering Committee, 993–1002.

[12] F. Tai and H.-T. Lin. Multi-label classification with principal label space transformation.Neural Computation, pages 2508–2542, 2012

[13] Krishnakumar Balasubramanian and Guy Lebanon. 2012. The landmark selection method for multiple output prediction. arXiv preprint arXiv:1206.6479

[14] Yukihiro Tagami. 2017. Annexml: Approximate nearest neighbor search for extreme multi-label classification. In Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data

[15] Jason Weston, Ameesh Makadia, and Hector Yee. 2013. Label partitioning for sublinear ranking. In International Conference on Machine Learning, pages 181–189

[16] K. Bhatia, H. Jain, P. Kar, M. Varma, and P. Jain. Sparse local embeddings for extreme multi-label classification. In Advances in Neural Information Processing Systems, pages730–738, 2015

[17] C. Xu, D. Tao, and C. Xu.Robust extreme multi-labellearning. InProceedings of the ACM SIGKDD internationalconference on Knowledge discovery and data mining, 2016.

[18] Yang, Z., Yang, D., Dyer, C., He, X., Smola, A., Hovy, E.: Hierarchical attention networks for document classification. In: Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (2016)

[19] Chih-Kuan Yeh, Wei-Chieh Wu, Wei-Jen Ko, and YuChiang Frank Wang. 2017. Learning deep latent space for multi-label classification. In AAAI, pages 2838–2844.

[20] Yashoteja Prabhu and Manik Varma. 2014. Fastxml: A fast, accurate and stable tree-classifier for extreme multi-label learning. In Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining, pages 263–272. ACM.

[21] Yang, L., Ng, T.L.J., Smyth, B., Dong, R.: HTML: hierarchical transformer-based multi-task learning for volatility prediction. In: The Web Conference 2020 (2020)

[22] Zhang W, Yan J, Wang X, Zha H. Deep extreme multi-label learning. In: Proc. ofACM ICMR, 2018: 100-107

[23] Rahul Agrawal, Archit Gupta, Yashoteja Prabhu, and Manik Varma. 2013. Multi-label learning with millions of labels: Recommending advertiser bid phrases for web pages. In Proceedings of the 22nd 2829 international conference on World Wide Web, pages 13–24. ACM.

[24] Himanshu Jain, Yashoteja Prabhu, and Manik Varma. 2016. Extreme multi-label loss functions for recommendation, tagging, ranking & other missing label applications. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM, 935–944.

[25] Zhang, X., Wei, F., Zhou, M.: HIBERT: document level pre-training of hierarchical bidirectional transformers for document summarization. arXiv preprint arXiv:1905.06566 (2019)

[26] Yoon Kim. 2014. Convolutional neural networks for sentence classification. arXiv preprint arXiv:1408.5882.

[27] Rie Johnson and Tong Zhang. 2014. Effective use of word order for text categorization with convolutional neural networks. arXiv preprint arXiv:1412.1058.

[28] M. Gori, G. Monfardini, and F. Scarselli. 2005. A new model for learning in graph domains. In Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005., Vol. 2. IEEE, 729–734.

[29] A. Sperduti and A. Starita. 1997. Supervised neural networks for the classification of structures.IEEE Transactions on Neural Networks8, 3 (1997), 714–735.

[30] Edouard Grave, Tomas Mikolov, Armand Joulin, and Piotr Bojanowski. 2017. Bag of tricks for efficient text classification. In Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics, EACL, pages 3–7.

[31] Bingyu Wang, Li Chen, Wei Sun, Kechen Qin, Kefeng Li and Hui Zhou. 2019. Ranking-Based Autoencoder for Extreme Multi-label Classification. In proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies

[32] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini. 2008. The graph neural network model.IEEE Transactions on Neural Networks20, 1, 61–80.

[33] Jingzhou Liu, Wei-Cheng Chang, Yuexin Wu, and Yiming Yang. 2017. Deep learning for extreme multi-label text classification. In Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 115–124. ACM.

[34] J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun. 2013. Spectral networks and locally connected networks on graphs.arXiv preprint arXiv:1312.6203(2013).

[35] M. Defferrard, X. Bresson, and P. Vandergheynst. 2016. Convolutional neural networks on graphs with fast localized spectral filtering. In Advances in neural information processing systems. 3844–3852.

[36] T. Mikolov, K. Chen, G. Corrado, and J. Dean, “Efficient estimation of word representations in vector space,” 2013, arXiv:1301.3781. [Online]. Available: http://arxiv.org/abs/1301.3781

[37] D. Zhang, M. Hong, L. Zou, F. Han, F. He, Z. Tu, Y. Ren, Attention pooling-based bidirectional gated recurrent units model for sentimental classification, Int. J. Comput. Int. Sys. 12 (2019), 723–732.

[38] T. N. Kipf and M. Welling. 2016. Semi-supervised classification with graphconvolutional networks.arXiv preprint arXiv:1609.02907(2016).

[39] W. Hamilton, Z. Ying, and J. Leskovec. 2017. Inductive representation learning on large graphs. In Advances in neural information processing systems. 1024–1034.

[40] R. You, S. Dai, Z. Zhang, H. Mamitsuka, and S. Zhu. 2019. AttentionXML: Extreme Multi-Label Text Classification with Multi-Label Attention Based Recurrent Neural Networks. (2019).

[41] J. Gehring, M. Auli, D. Grangier, and Y. N. Dauphin. 2016. A convolutional encoder model for neural machine translation. arXiv preprint arXiv:1611.02344 (2016)

[42] Z. Peng, W. Shi, J. Tian, Z. Qi, B. Li, H. Hao, B. Xu, Attention based bidirectional long short-term memory networks for relation classification, in Meeting of the Association for Computational Linguistics, Berlin, Germany, 2016.

[43] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio. 2018.Graph Attention Networks. arXiv:1710.10903 [stat.ML]

[44] M. Yang, W. Tu, J. Wang, F. Xu, X. Chen, Attention based LSTM for target dependent sentiment classification, in: Thirty-First AAAI Conference on Artificial Intelligence, 2017, pp. 5013–5014

[45] Deepak Saini, Arnav Kumar Jain, Kushal Dave, Jian Jiao, Amit Singh, Ruofei Zhang and Manik Varma. 2021. GalaXC: Graph Neural Networks with Labelwise Attention for Extreme Classification. Proceedings of the Web Conference 2021

[46] K. Xu, W. Hu, J. Leskovec, and S. Jegelka. 2018. How powerful are graph neuralnetworks?arXiv preprint arXiv:1810.00826(2018).

[47] L. Cai, Y. Song, T. Liu and K. Zhang, "A Hybrid BERT Model That Incorporates Label Semantics via Adjustive Attention for Multi-Label Text Classification," in IEEE Access, vol. 8, pp. 152183-152192, 2020, doi: 10.1109/ACCESS.2020.3017382.

[48] Huang, X., Chen, B., Xiao, L. et al. Label-Aware Document Representation via Hybrid Attention for Extreme Multi-Label Text Classification. Neural Process Lett (2021). https://doi.org/10.1007/s11063-021-10444-7

[49] https://www.nlm.nih.gov/databases/download/pubmed_medline.html

[50] D. Bahdanau, K. Cho, and Y. Bengio. 2014. Neural machine translation by jointly learning to align and translate. arXiv preprint arXiv:1409.0473 (2014)

[51] Chalkidis, Ilias and Fergadiotis, Emmanouil and Malakasiotis, Prodromos and Androutsopoulos, Ion. 2019. Large-Scale Multi-Label Text Classification on EU Legislation. Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics. Download Link: https://huggingface.co/datasets/eurlex

[52] Eneldo Loza Mencia and Johannes Furnkranzand. 2007. An Evaluation of Efficient Multilabel Classification Algorithms for Large-Scale Problems in the Legal Domain. In Proceedings of the LWA 2007, pages 126–132, Halle, Germany.

[53]] Samy Bengio, Jason Weston, and David Grangier. 2010. Label embedding trees for large multi-class tasks. In Advances in Neural Information Processing Systems. 163–171.

[54] U.S National Library of Medicine. [n. d.]. MeSH Browser. https://meshb.nlm.nih.gov/treeView. ([n. d.]). (Accessed on 03/18/2019).

[55] Vijit Malik,Rishabh Sanjay, Shouvik Kumar Guha, Shubham Kumar Nigam, Angshuman Hazarika, Arnab Bhattacharya, Ashutosh ModiSemantic. Segmentation of Legal Documents via Rhetorical Roles. https://arxiv.org/abs/2112.01836

[56] T. Lin, P. Goyal, R. Girshick, K. He and P. Dollár, "Focal Loss for Dense Object Detection," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 42, no. 2, pp. 318-327, 1 Feb. 2020, doi: 10.1109/TPAMI.2018.2858826.

[57] Dan Shen, Jean David Ruvini, Manas Somaiya, and Neel Sundaresan. 2011. Item categorization in the e-commerce domain. In Proceedings of the 20th ACM international conference on Information and knowledge management. ACM, 1921–1924

[58] J. Liu, Y. Zhang, Attention modeling for targeted sentiment, in: Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers, 2017, pp. 572–577

[59] M. Li, A. Xiong, L. Wang, S. Deng, J. Ye, Aco resampling: Enhancing the performance of oversampling methods for class imbalance classification, Knowledge-Based Systems (2020) 105818.

[60] BlackStone Link: https://github.com/ICLRandD/Blackstone

[61] T. N. Kipf and M. Welling. 2016. Semi-supervised classification with graphconvolutional networks.arXiv preprint arXiv:1609.02907(2016).

[62] Lee, S., Baker, J., Song, J., Wetherbe, J.C.: An empirical comparison of four text mining methods. In: Proceedings of the 43rd Hawaii International Conference on System Sciences, pp. 1-10. IEEE, New York (2010)

[63] Rahul Agrawal, Archit Gupta, Yashoteja Prabhu, and Manik Varma. 2013. Multi-label learning with millions of labels: Recommending advertiser bid phrases for web pages. In Proceedings of the 22nd international conference on World Wide Web. ACM, 13–24.

[64] B. S. Raghuwanshi, S. Shukla, Smote based class-specific extreme learning machine for imbalanced learning, Knowledge-Based Systems 187 (2020) 104814.

[65] Emily Denton, Jason Weston, Manohar Paluri, Lubomir Bourdev, and Rob Fergus. 2015. User conditional hashtag prediction for images. In Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining. ACM, 1731–1740.

[66] Jianxiang Dong. Focal Loss Improves the Model Performance on Multi-Label Image Classifications with Imbalanced Data. ICNSER2020: Proceedings of the 2nd International Conference on Industrial Control Network And System Engineering Research

[67] Zhouhan Lin, Minwei Feng, Cicero Nogueira dos Santos, Mo Yu, Bing Xiang, Bowen Zhou, and Yoshua Bengio. 2017. A structured self-attentive sentence embedding. arXiv preprint arXiv:1703.03130 (2017).

[68] Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent dirichlet allocation. Journal of Machine Learning Research 3, 993-1022 (2003)

[69] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, “mixup: Beyond empirical risk minimization,” in ICLR, 2018.

[70] Shrivastava, A., Gupta, A., Girshick, R.: Training region-based object detectors with online hard example mining. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 761–769. IEEE, NJ (2016)

[71] N. V. Chawla, K. W. Bowyer, L. O. Hall, W. P. Kegelmeyer, Smote: synthetic minority oversampling technique, Journal of artificial intelligence research 16 (2002) 321–357.

[72] Deerwester S, Dumais ST, FurnasGW, Landauer TK, Harshman R. Indexing by latent semantic analysis. Journal of The American Society for Information Science. 1990;41(6):391–407.

[73] B. Liu, G. Tsoumakas, Dealing with class imbalance in classifier chains via random undersampling, Knowledge-Based Systems 192 (2020) 105292.

[74] C. Cao, Z. Wang, Imcstacking: Cost-sensitive stacking learning with feature inverse mapping for imbalanced problems, Knowledge-Based Systems 150 (2018) 27–37.

[75] Daud, A., Li, J., Zhou, L., Muhammad, F.: Knowledge discovery through directed probabilistic topic models: a survey. Frontiers of Computer Science in China 4(2), 280-301 (2010)

[76] Jiang D, He J (2021) Text semantic classification of long discourses based on neural networks with improved focal loss. Comput Intell Neurosci 2021

[77] Şule Öztürk Birim, Ipek Kazancoglu, Sachin Kumar Mangla, Aysun Kahraman, Satish Kumar, Yigit Kazancoglu: Detecting fake reviews through topic modelling, doi: https://doi.org/10.1016/j.jbusres.2022.05.081

[78] S. Han, C. Lim, B. Cha and J. Lee, "An Empirical Study for Class Imbalance in Extreme Multi-label Text Classification," 2021 IEEE International Conference on Big Data and Smart Computing (BigComp), 2021, pp. 338-341, doi: 10.1109/BigComp51126.2021.00073

[79] Blei, D.M., Lafferty, J.D.: Topic models. In: Srivastava, A., Sahami, M.(eds.) Text Mining: Classification, Clustering, and Applications. pp. 71-93. Chapman Hall/CRC, London (2009)

[80] Alghamdi, R., Alfalqi, K.: A survey of topic modeling in text mining. International Journal of Advanced Computer Science and Applications 6(1), 147-153 (2015)

[81] Landauer, T.K., Laham, D., Rehder, B., Schreiner, M.E.: How well can passage meaning be derived without using word order? A comparison of latent semantic analysis and humans. In: Proceedings of the 19th Annual Meeting of the Cognitive Science Society, pp. 412-417. Erlbaum, New Jersey (1997).

[82] Pedro Henrique Luz De Araujo, Teófilo De Campos: Topic Modelling Brazilian Supreme Court Lawsuits. Volume 334: Legal Knowledge and Information Systems, pp. 113-122, doi: 10.3233/FAIA200855

[83] Agrawal, Amritanshu Fu, Wei Menzies, Tim. (2018). What is Wrong with Topic Modeling? (and How to Fix it Using Search-based SE). Information and Software Technology. 98. 10.1016/j.infsof.2018.02.005.