Proceedings of the 13th Conference on Language Resources and Evaluation (LREC 2022), pages 4597–4605
Marseille, 20-25 June 2022

Investigating Active Learning Sampling Strategies for Extreme Multi Label Text Classification

Lukas Fromme[*†], **Katsiaryna Mirylenka**[†], **Jonas Kuhn**[*], **Jasmina Bogojeska**[†]
[*]Institute of Natural Language Processing (IMS) - University of Stuttgart, [†]IBM Research - Zurich
lukas.fromme@ims.uni-stuttgart.de, kmi@zurich.ibm.com

Abstract

Large scale, multi-label text datasets with a high number of classes are expensive to annotate, even more so if they belong to specific language domains. In this work, we aim to build classifiers for these datasets using Active Learning in order to reduce the labeling effort. We outline the challenges when dealing with extreme multi-label settings and show the limitations of existing pool-based Active Learning strategies by considering their effectiveness as well as efficiency in terms of computational cost. In addition, we present five multi-label datasets which were compiled from hierarchical classification tasks to serve as benchmarks in the context of extreme multi-label classification for future experiments. Finally, we provide insights into multi-class, multi-label evaluation and present an improved classifier architecture on top of pre-trained transformer language models.

Keywords: Active Learning, Text Classification, Multi-Label

1. Introduction

In many fields, one could regard text classification as a solved problem. Even before pre-trained language models were available, common classification tasks such as Spam/Ham sorting or Positive/Negative detection could be completed by a variety of machine learning architectures with a very high accuracy. Consequently there has been a trend to move toward more complex classification tasks. One such task is commonly known as *Extreme Multi Label Text Classification (XMTC)* - datasets for this task contain large numbers of distinct classes ranging from hundreds to more than a million. We combine XMTC with Active Learning (AL), which is a technique used to reduce annotation effort by strategically selecting samples from an unlabeled set. The combination of XMTC and AL has not been exposed to much systematic research although it is relevant in many practical scenarios. Large, domain-specific datasets are often assigned numerous, fine-grained categories and each text can belong to multiple of them. For example, one can easily imagine a dataset of law texts which is annotated with several hundred different target classes. Such datasets are also very expensive to annotate due to requiring domain experts who will take a lot of time working through all the classes. Therefore, AL is a particularly attractive prospect, albeit not without its own set of challenges. In order to be truly effective, AL uses various strategies in order to select the most profitable samples from the unlabeled set. However, many of these strategies are not effective on all kinds of data or all classification models. In this work we motivate and introduce the task of Active Learning for Extreme Multi-Label Classification. Our goal is to gain insights into the performance of established AL approaches on datasets with more than 100 distinct classes. We are also interested in both the efficiency and effectiveness of these approaches and consider the tradeoff between increases in accuracy and computational cost. Our work contributes the following: **1.** An investigation of AL selection strategies on Extreme Multi Label datasets including the analysis of the computational cost. **2.** Where possible, we provide the datasets directly. In addition we detail the steps necessary to compile the dataset from its source. All of our Extreme Multi-Label datasets contain hierarchical label structures. **3.** A multi-label classification evaluation technique using variable thresholds. **4.** A more accurate classification network architecture that uses a Convolutional Neural Network as the classification head on top of BERT.

2. Related Work

Active Learning has been the subject of extensive research and a number of strategies have been shown to reduce the manual annotation effort in a variety of supervised machine learning tasks. Past experiments have shown that AL is effective on Text Classification with SVMs (Tong and Koller, 2001), (Goudjil et al., 2018) as well as with Deep Learning models performing image classification (Wang and Shang, 2014). In recent years there have been more approaches of applying AL to Deep Models, for example for Named Entity Recognition (Shen et al., 2017) and Text Classification (An et al., 2018). The challenge is that traditional AL approaches often rely on classifier confidence which is typically very high in Deep Neural Networks and as such, may not be a good indication of the actual classifier accuracy (Ren et al., 2021). In our experiments, we focus on *pool-based* AL which means that we select one text from the unlabeled pool and annotate it completely. Furthermore, we deal with the task of Multi-Label text classification, which makes the usage of some AL strategies (for example prediction confidence) more difficult (Wu et al., 2020), (Esuli and Sebastiani, 2009). Apart from the methods presented in Section 5.1, there are noteworthy AL approaches us-

ing Generative Adversarial Networks (Mayer and Timofte, 2020), custom loss functions with label correlation (Ranganathan et al., 2018), uncertainty gradient methods (Ash et al., 2019) or using network layers to separate the label space (Liu et al., 2021).

Extreme Multi-Label Text Classification (XMTC) has come up as a research topic often focusing on industrial tasks such as recommender systems (Agrawal et al., 2013), that can easily have several thousands of distinct classes. XMTC methods often find ways to reduce or condense the label space to both save memory during computation and build representations of the label space (Bi and Kwok, 2013), (Prabhu and Varma, 2014). However, with the rise of contextualized pre-trained language models and large scale computation thanks to powerful GPUs, XMTC has also been successfully used with BERT (Devlin et al., 2018) or specifically adapted architectures (Chang et al., 2020), (Chang et al., 2019).

3. Active Learning for Extreme Multi Label Text Classification

We are given a labeled training set *T*, an unlabeled set *U* and a model *M* which requires supervised training on a downstream task. In AL, we start by training *M* on the initial training set *T*. We then sample *n* new data points from *U* after the training has completed. These data points are then annotated and added to the training set. This cycle continues until a stopping criterion is fulfilled - typically until a certain accuracy threshold is reached, the model converges or we exhaust the annotation budget. The goal is to increase the accuracy of *M* without the need to annotate *U* in its entirety which allows us to achieve higher classification scores with less annotation effort. The general scheme is illustrated in Algorithm 1. In an experimental setting, it is a common practice to split a fully annotated dataset into *T* and *U* in order to simulate the AL setup. The annotation step is then performed by simply using the original labels of *U*.[1] The main challenge in AL is to select texts from U that are valuable to the classifier and thus lead to a higher classification accuracy compared to a random selection of the same number of texts. We refer to this selection mechanism as *sampling strategy* or *selection strategy*.

The goal of text classification is to automatically sort a text into a number of predefined categories called *classes* or *labels*. In Multi-Label classification specifically, each text is assigned to a subset of all available classes instead of just one. Compared to other classification tasks, XMTC deals with datasets that contain many distinct classes from a few hundreds to more than a million. For our purposes we consider a multi-label dataset with at least 100 classes to fall under the XMTC task. The datasets used in our experiments have from 100 to around 700 classes and as such, they are relatively small compared to many of the benchmark datasets in XMTC.[2]. One reason for this is that we want to introduce new datasets that can be used in the context of XMTC. However, we also require our datasets to be easy enough for a classifier so that we can better observe the effects of AL without training for hundreds of iterations. As such, we believe that this is an important step towards a more systematic research approach to AL for XMTC. We are also dealing with a multi-label classification problem which means that each text belongs to any number of classes instead of just one. Though Multi-label classification requires little changes in the architecture of a Deep Classification Model, the evaluation needs a specialized treatment in order to be informative (see Section 5.2).

Algorithm 1 Active Learning (AL)

```
procedure AL(labeled set T, unlabeled set U, model M, stopping criterion c)
    D ← T
    while criterion c not met do
        train M_t on D
        D* ← sampled n data points from U
        fully annotate D*
        D ← D ∪ D*
```

4. Resources

4.1. Datasets

For our experiments, we consider 5 extreme multi-label datasets which we compile using publicly available, fully annotated datasets with a hierarchical label structure. We extract the leaf nodes of the label hierarchy and use only those as classes. Some datasets have additional cleaning steps which are detailed in their respective descriptions.

EurLex. It is a subset of the EurLex57k dataset (Chalkidis et al., 2019). This dataset deals with European Law and is specific to the legal domain. Starting with the original dataset, we first filter out all classes that are not leaves in the topic hierarchy. In addition we also remove classes that appear in less than 50 samples. A small number of texts with no labels are removed as well.

ArXiv. Taken originally from (kaggle, 2020) (`https://www.kaggle.com/Cornell-University/arxiv`), this dataset contains titles and scientific abstracts from arXiv documents. The documents are categorized using the special arXiv hierarchy.

NYT. This is a subset of the New York Times Annotated Corpus (Sandhaus, 2008) consisting of manually annotated news articles that are sorted into a fine-grained hierarchy of topics.

[1] This kind of simulation has been found to ignore issues that come up when actually deploying an AL system with human annotators (Settles, 2011).

[2] An overview of XMTC datasets can be found here: `http://manikvarma.org/downloads/XC/XMLRepository.html#metrics` (Bhatia et al., 2016)

	size			number of	classes per	texts	average class
	train	test	dev	classes	text (avg)	per class (avg)	co-occurence
EurLex	44,689	5,954	5,963	739	4.38	265.32	1.88
arXiv	239,347	29,174	26,309	113	1.68	4385.46	42.85
NYT	22,991	10,941	2,554	303	2.62	315.52	3.03
RCV1	20,816	781,265	2,333	100	1.53	12321.26	116.72
Yelp	100,911	48,272	11,287	580	2.3	401.68	2.33
AGNews	112,400	7,600	7,600	4	0.75	28115.0	0
Toxic	102,124	25,532	31,915	6	0.22	3702.83	2065.47

Table 1: **Sizes and class statistics for all datasets. Classes per text and texts per class are averaged over all texts of the respective dataset. Average class occurrence is calculated for each class individually and then averaged.**

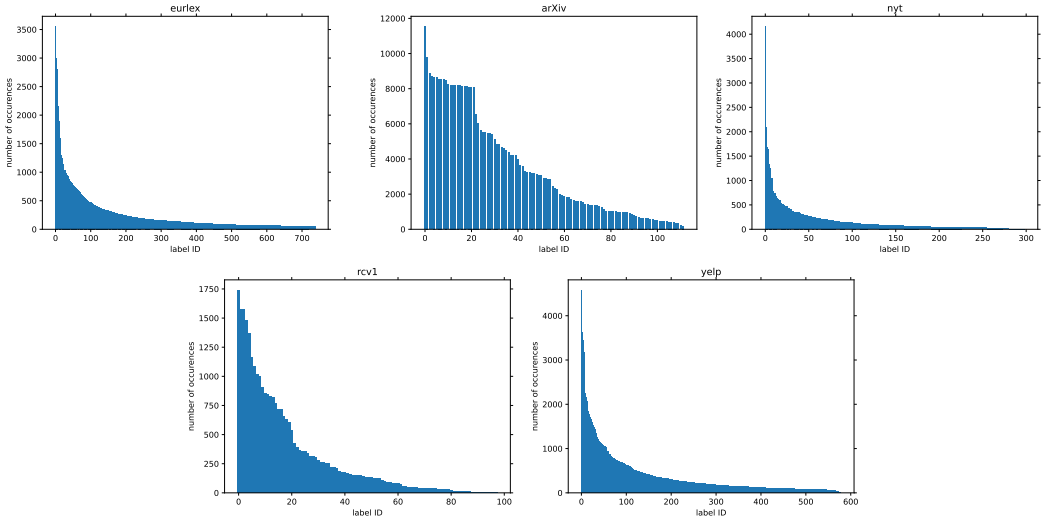

Figure 1: **Class distribution of extreme multi-label datasets. Class frequencies on all datasets are highly unbalanced, containing many rare classes and few very frequent classes.**

RCV1. The Reuters Corpus Volume 1 dataset (Lewis et al., 2004) consists of manually categorized newswire articles.

Yelp. This datasets consists of reviews taken from Yelps database (`https://www.yelp.com/dataset`) annotated with fine grained topics concerning the subject of the review.

In addition, we also run experiments for two datasets with fewer classes - one single-label and another multi-label dataset. These datasets are used without any additional modifications and do not contain a hierarchical label structure.

AGNews. This is a collection of News Articles grouped into 4 broad categories: World, Sports, Business and Tech. Taken directly from kaggle: `https://www.kaggle.com/amananandrai/ag-news-classification-dataset`. This is a multi-class, single label dataset.

Toxic. It is a mutli-Label dataset, which is a collection of comments from online conversations annotated with 6 different aspects of negative behaviour. Available on kaggle: `https://www.kaggle.com/c/jigsaw-toxic-comment-classification-challenge/overview`

The descriptive statistics in Table 1 indicate the distinct characteristics of the datasets included in our collection. We expect the individual sampling strategies to respond differently to the properties of the datasets. ArXiv and RCV1 have the fewest number of classes, least classes per text and most occurrences of each class (texts per class) compared to the other 3 Extreme Multi-Label datasets. We also calculate average class co-occurrence by counting how often each class appears together with another class and then average over all classes. Intuitively, this gives us an idea of how correlated the classes are in a dataset. We see that arXiv and RCV1 have a relatively high co-occurrence compared to the other 3 Extreme Multi-Label datasets. For the two datasets AGNews and Toxic, we see that both datasets can have zero labels as average classes per text is below 1. Toxic has a high class co-occurence while the classes on AGNews are mutually exclusive, as it is a single-label dataset. AGNews and Toxic are not consider XMTC datasets. We include them to show AL performance on more standard NLP datasets. As

can be seen in Figure 1, all five Extreme Multi-Label datasets are unbalanced.

5. Experiment

5.1. AL Sampling Strategies

We implement three state-of-the-art sampling strategies for multi-label AL:

ALPS (Active Learning by Processing Surprisal) (Yuan et al., 2020) calculates *surprisal embeddings* with the Masked-Language-Modeling objective of the underlying BERT transformer network. In essence, the less certain the language model is about the words in the text, the higher the loss in the surprisal embeddings. In order to improve generalisation, the surprisal embeddings are then clustered and samples are selected closest to the the cluster centroids. *ALPS* is a similar approach to *BADGE* (Ash et al., 2019), which embeds unlabeled texts using the classifier loss instead of the language model objective. The disadvantage is, that *BADGE* relies on the classifier accuracy directly which might be extremely low especially in early AL stages. The authors show that *ALPS* is the more effective approach on a variety of datasets which is why we do not use *BADGE*.

DAL The Discriminative Active Learning (Gissin and Shalev-Shwartz, 2019) approach operates on the idea to bring the distributions of labeled and unlabeled set closer together. They train a classifier that learns to distinguish between texts that come from the labeled set and those that come from the unlabeled set. The more certain this discriminator is that the text came from the unlabeled distribution, the higher it is ranked for the selection.

CVIRS (Reyes et al., 2018) combine category vector inconsistency (CVI), i.e. how different the classifier predictions on the unlabeled set are compared to the label set with a ranking of uncertainty scores (RS) taken directly from the classifier predictions. This is the only strategy that uses the classifier probabilities for sample selection.

In addition, we also test a simple sampling strategy of our own that relies on the tokenization of pre-trained language model and is easy to compute.

Subword approach processes all words in a text with the sentence-piece tokenizer of the transformer language model used for classification. Words unknown to the language model are split into subword-units. We assume that a higher ratio of subword-units in a text means that it contains more rare words and thus is more interesting to the classifier. We rank the texts by the average amount of subword-units per word.

All these strategies are compared against random selection to test their effectiveness.

5.2. Variable Threshold and Evaluation

In our experiments, we focus on multi-class multi-label datasets. This means, that there are more than two different classes and each data point can belong to any number of those classes. Given a data point x and a set of classes $C = (c_1, ..., c_n)$, a neural network M returns its predictions as log-probabilities for each class: $M(x) = (p_1, p_2, ..., p_n)$. In a single-label setting we get the prediction y^* by simply taking the class with highest p_i. In a multi-label setting however, it is not immediately clear which of the probabilities signify a positive prediction. Therefore, we define a threshold τ. All p_i that are greater then τ are considered to indicate positive predictions. Formally, the prediction y^* is determined as such: $y^* = \{c_i | p_i > \tau\}$. Rather than fixing τ to a single number, we design it to be variable by selecting it from the development set. When calculating the classification accuracy on the development set, we test a wide range of manually defined values for τ and select the one that gives the best $F1$-score (In our experiments, we optimize τ with regards to Macro-$F1$. See also Section 6). The τ that was used to evaluate the best trained model is then also used to calculate $F1$ on the test set. When no development set is available, τ can also be estimated on the training set. In that case, τ is set so that y^* matches the average number of classes per text of the training set. In practice, τ can also be adjusted post-hoc once enough data has been labeled. The evaluation is done using the $F1$, calculated as follows:

$$\text{F1}(y, y^*) = 2 * \frac{precision * recall}{precision + recall} \tag{1}$$

$$\text{micro-precision}(y, y^*) = \frac{tp}{tp + fp} \tag{2}$$

$$\text{micro-recall}(y, y^*) = \frac{tp}{tp + fn} \tag{3}$$

$$\text{macro-precision}(y, y^*) = \frac{1}{|C|} \sum_{c \epsilon C} \frac{tp_c}{tp_c + fp_c} \tag{4}$$

$$\text{macro-precision}(y, y^*) = \frac{1}{|C|} \sum_{c \epsilon C} \frac{tp_c}{tp_c + fn_c} \tag{5}$$

where y are the annotated classes, and y^* are the predictions from the classifier. *tp*, *fp* and *fn* stand for the error types true positive, false positve and false negative between y and y^*. Due to the multi-class setting, we consider *Macro* and *Micro* precision and recall. For the macro scores, we calculate precision and recall per class and average over all classes as illustrated in equation (4) and (5). For the micro scores we sum up *tp*, *fp* and *fn* over the whole dataset, ignoring the class boundaries. This is shown in Equation (2) and (3). Macro-$F1$ and Micro-$F1$ are then calculated by using the respective *Micro*/*Macro* precision and recall. A more detailed discussion on the difference between the two is found in Section 6.2.

	single output layer		CNN output	
	micro-F_1	macro-F_1	micro-F_1	macro-F_1
EurLex	0.62	0.47	**0.78***	**0.69***
arXiv	0.52	0.42	**0.64***	**0.58***
NYT	0.58	0.32	**0.65***	**0.51***
RCV1	0.78	0.54	**0.81***	**0.64***
Yelp	0.38	0.26	**0.58***	**0.52***
AGNews	0.91	0.91	**0.94***	**0.94***
Toxic	0.75	0.62	**0.79***	**0.68***

Table 2: **Macro and Micro F_1 classification scores on the full datasets (without using Active Learning). Experiments were run with a BERT transformer using a single output layer in the left two columns and a CNN (Convolutional Neural Network) instead in the right two columns. Best results marked in bold. *significance tested with unpaired t-test $p < 0.05$.**

5.3. Experimental Setup

We use a pre-trained BERT transformer[3] for the classification task, truncating the input to the maximum size of 512 tokens. Usually, classification models with BERT are built by adding a single output layer on top of the transformer network. In our experiments instead of the feed-forward layer we add a simple convolutional neural network (CNN) which was implemented in pytorch. We find that the CNN greatly increases the classification performance on the full datasets (see Table 2). We train for a maximum of 15 epochs with early stopping and evaluate the classification $F1$ on a held-out development set after each epoch using a variable threshold as explained in Section 5.2. The criterion for early stopping is also the $F1$ on the development set. After the final epoch the best model (according to dev $F1$) is used to calculate the $F1$ on the test set which we report in Section 5.4. We train with a batch size of 16 on a NVIDIA RTX A6000 using Adam optimizer with a starting learning rate of $5e^{-5}$. For the AL, we set an annotation budget of 2000 texts and add 100 texts in each iteration. Each experiment is run three times with different random seeds in order to increase the robustness of the results.

5.4. Experimental Results

We give a short summary of our results illustrated in Figure 2, which shows the micro $F1$ and Figure 3, which shows the Macro $F1$ on all datasets. Looking at Figure 2, we see that random selection is on par with or better than all strategies on the arXiv, RCV1, AGNews and Toxic datasets. For the AGNews dataset, micro $F1$ is very high at at 0.8 to 0.9 even at 100 texts leaving little room for improvements as the upper bound (Table 2) is 0.94 micro $F1$. *DAL* and *subword* perform slightly worse, while the other approaches are on par with random selection. Similarly, no strategy can significantly improve upon random selection on the Toxic dataset. On the arXiv dataset, random selection consistently outperforms all selection strategies by a margin of up to 0.15 except for *ALPS* which is only slightly worse by a margin of around 0.01. The RCV1 dataset benefits most from the *subword* strategy for up to 500 texts, though the improvements are non significant. Starting at 500 texts which point random selection is on par with the both *subword* and *ALPS*. Here *DAL* performs worse by margin of around 0.2 micro $F1$ compared to the random selection.

On the EurLex dataset, *ALPS* and *subword* score the best micro $F1$ with an improvement of up to 0.15[4] compared to random selection. The *CVIRS* and *DAL* strategies perform similar or worse than random selection by a margin of up to 0.1 micro $F1$. *CVIRS* in particular starts learning much later than the other approaches. The rise in micro $F1$ can be observed starting from 1000 texts while this happens at around 400 to 500 texts for the other strategies.

ALPS is also the most powerful strategy on the NYT dataset, outperforming all other approaches significantly with an increase of around 0.1 compared to the second best approach *CVIRS* and around 0.25 compared to random selection[4]. *Subword* and *DAL* perform overall perform worse or similarly to the random selection.

On the Yelp dataset both *DAL* and *ALPS* achieve improvements over random selection though with a margin of around 0.1 micro $F1$ though the improvements are not significant. *CVIRS* and *subword* achieve very similar scores to random selection.

Compared to the micro $F1$ results, the results on the 5 extreme multi-label datasets as well as the Toxic dataset for the macro $F1$ scores in Figure 3 are lower overall, which is expected since the macro $F1$ score is generally harder to increase. Otherwise, the results on most datasets behave very similarly to the micro $F1$ scores though the margins of improvement do not generally exceed 0.1 of macro $F1$. On the eurlex dataset, *ALPS* and *subword* improve upon random selection by a small but significant[4] margin of 0.05. One big difference to Figure 2 can be observed for the Toxic dataset, which benefits significantly from the CVIRS strategy yielding an improvement of around 0.15 compared to all other strategies and random selection. We offer further analysis and interpretation of the results in Section 5.5 and 6.

5.5. Additional Analysis on EurLex Dataset

We are interested in the high performance of *ALPS* and *subword* on the domain-specific EurLex dataset. We first investigate, if the AL approaches sample similar unlabeled texts to random selection. We find that there is a small overlap of around 70 samples on of selected texts between each AL strategy and random selection on the EurLex dataset while there is little to no over-

[3]Using the huggingface library with the 'bert-base-uncased' model available from `https://huggingface.co`

[4] significance tested with unpaired t-test $p < 0.05$

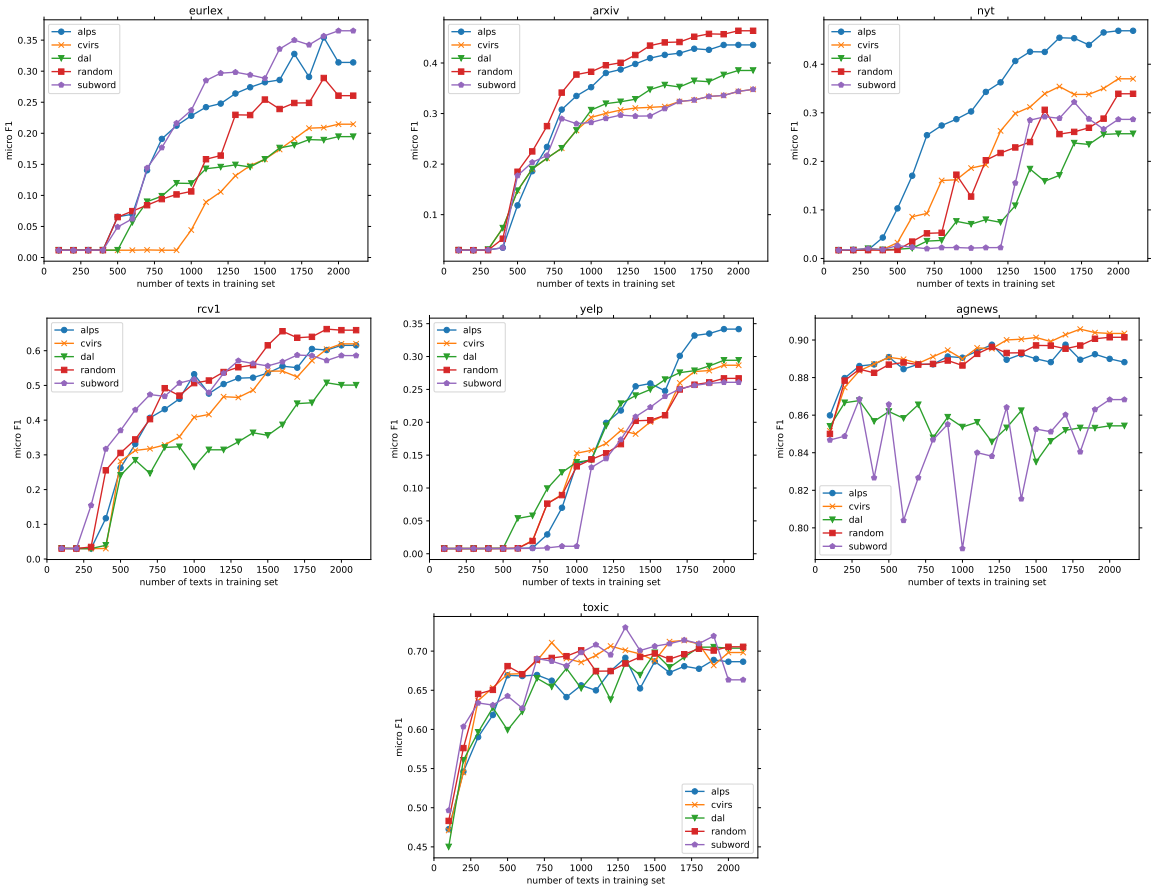

Figure 2: **Micro $F1$ results across all datasets. The x-axis describes the number of texts used to train the classifier while the y-axis shows the resulting micro $F1$.**

lap between the high scoring strategies compared to the lower scoring strategies. *ALPS* and *subword* do show an overlap of 122 texts out of 2000 which happens most likely because both methods leverage the information from the pre-trained language model. We assume given the class distributions (Figure 1), that improving Micro and Macro $F1$ is dependent on learning the large amounts of rare classes. We check how many classes the classifier sees in each AL iteration and find, that *DAL* generally covers around 70% of all classes while all other methods, including random selection cover from 80% up to 94% of all classes. This might be a reason why *DAL* achieves the overall worst Macro F1 on EurLex.

5.6. Efficiency Analysis

For each sampling strategy, we analyse the runtime to process 100 texts averaged over all datasets. It needs to be noted that whenever a strategy requires a pass over the unlabeled set, we do so in batches with a batch size of 16 which is the maximum size the memory of the GPU can handle in our setup (see Section 5.3). When using a smaller batch size we find that one can generally assume that the computation will take proportionally longer, i.e. with a batch size of 1, the computation will take around 16 times as long. All results can be found in Table 3

strategy	computation time for 100 batches (in seconds)
subword	3.80 (computed once)
ALPS	50
DAL	56
CVIRS	123

Table 3: **Runtime Analysis of AL sampling strategies. Computation time is measured for 100 batches of size 16 (1600 texts), averaged over all 5 extreme multi-label datasets and reported in seconds. Random sampling has negligible computational cost.**

Subword requires only a single tokenization pass over the unlabeled dataset and can be finished in a couple of minutes on most datasets. The main advantage is that *subword* calculates its scores only once instead of every AL iteration making it by far the cheapest approach.
ALPS requires masked language modeling on the unlabeled set as well as a clustering and sorting step of the resulting embeddings. In our implementation, completing 100 batches takes 50 seconds, which means that on most of the datasets a single pass of *ALPS* can be completed in around half an hour to an hour.

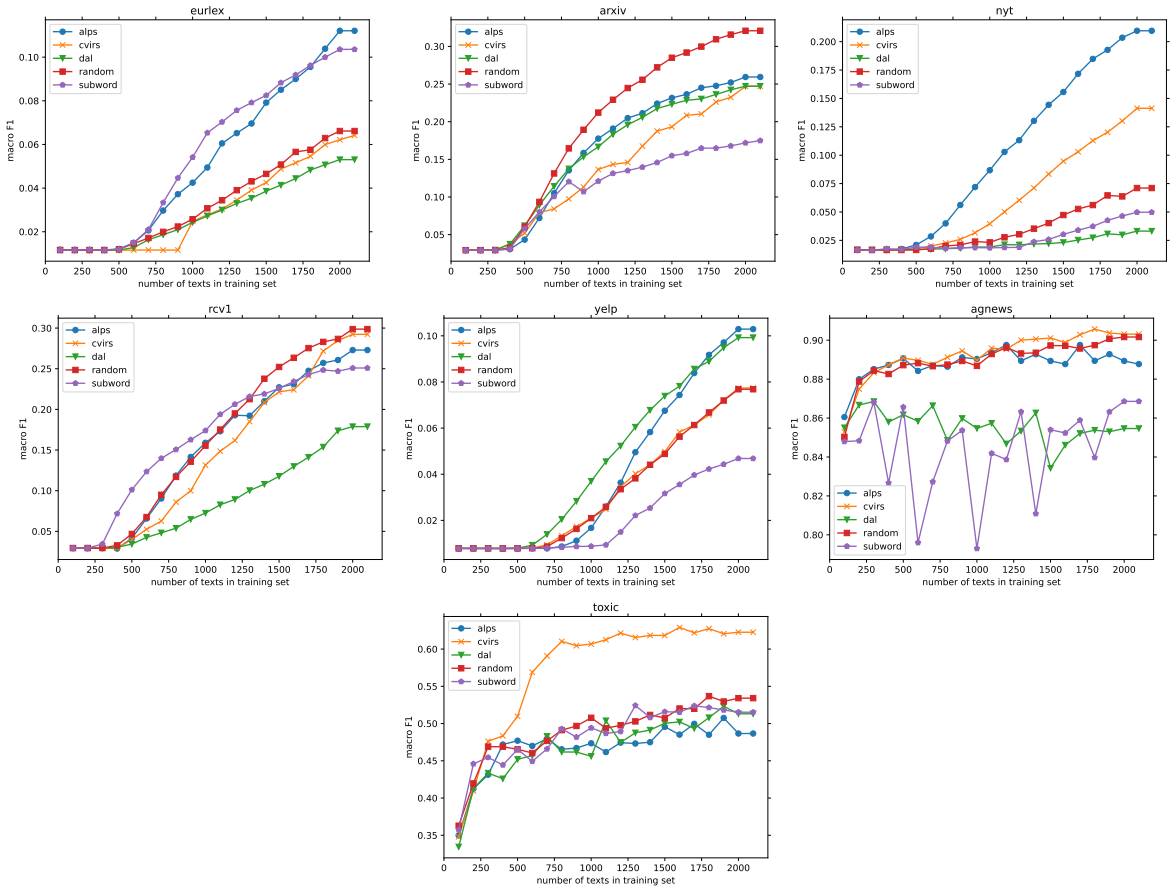

Figure 3: **Macro F1 results across all datasets. The x-axis describes the number of texts used to train the classifier while the y-axis shows the resulting Macro F1.**

DAL requires training of a separate model which we implement as a simple feed forward network and train for 2 epochs as suggested by the authors (Gissin and Shalev-Shwartz, 2019). All in all, *DAL* is slightly slower than *ALPS* but a single pass over the unlabeled data can be computed at similar speeds.

Finally, *CVIRS* is the most expensive approach. For the computation, *CVIRS* requires an inference pass over the unlabeled data, which is expensive with a transformer network and then calculates multiple similarity measures that each require more passes over the unlabeled data. Depending on the dataset, a single calculation of *CVIRS* can take several hours to complete in our implementation.

6. Discussion

6.1. Interpretation of Results

Concerning the overall results in Section 5.4 we find, that AL improves the classifier on datasets which have low label co-occurence: EurLex, NYT and Yelp. This might be an indication that the selection strategies struggle to find correlated labels. While the approaches detailed in Section 5.1 all use different information from the classifier, they all try to find rare or uncertain data points. It might well be, that when labels are highly correlated, the rare classes are assigned to data points which are not very different from the rest and as such, are harder to identify with the proposed methods. We also find that on AGNews and Toxic, CVIRS performs well especially compared to the other dataset. A possible reason for this is, that the classifier F1 is higher on these two datasets for small training sets (especially the Macro F1) and as such, the selection strategy can properly make use of the classifier confidence. Another interesting observation is, that the *subword* approach performs well on the EurLex dataset but is among the worst AL strategies for most of the other datasets. A possible explanation for this is, that on EurLex, rare words also indicate rare classes as they describe detailed legal concepts. In addition, since EurLex has a low class co-occurrence, we know that the labels are fairly singular and spaced out across the dataset. The only other dataset which is similar in both the number of classes and average class co-occurence is Yelp. However, the Yelp dataset contains mostly simple, everyday language that a pre-trained language model will be very familiar with. While we find, that the average number of subword-units is similar across datasets, we expect the split words in a common domain dataset such as Yelp to be less indicative of the class.

6.2. Micro and Macro F1

As already mentioned in Section 5.2, evaluating multi-class, multi-label classification requires some finesse. In the experiments, we choose to evaluate both micro $F1$ and macro $F1$ as the information they provide about the performance of the classifier is actually quite different. There is an argument that micro $F1$ is the more representative metric, as the distributions of classes in the datasets are imbalanced. Even if we assume that our classifier has few errors overall, it can still score a low Macro F1 if the errors are concentrated in many, low frequency classes. As such, macro $F1$ is much harder to increase for an imbalanced set of classes. This is even more true in our AL setting where we might not even find all the classes present in the test set in our small training set. Consequently, micro $F1$ offers a way to ignore the class boundaries and count error types over the whole dataset instead which can be considered a more accurate representation of the classification capabilities of our system. However, especially when we are dealing with imbalanced datasets in practice, the rare classes are often the ones that are particularly interesting. While a classifier with high micro $F1$ might make fewer errors on the whole dataset, it is possible that it really only learns to predict the few majority classes. This effect is again magnified when dealing with AL. One could argue that one of our goals in AL is to cover all the classes of the dataset quickly. Micro $F1$ however will continue to increase even if only the predictions on a few classes or even a single frequent class continue to improve. In essence, while it is true that Micro $F1$ is a better description of how many mistakes our classifier makes overall we should keep in mind that Macro $F1$ reflects how many of the classes our system is able to discriminate. We conclude that while Micro F1 is important to judge the overall accuracy of the classifier, we should pay equal, if not more attention to Macro $F1$ since we want to observe how many classes our classifier learns to cover over the course of AL.

6.3. Tradeoff between effectiveness and efficiency

We see that *subword* is the cheapest method and that it is very effective on the EurLex dataset and yields slight improvements on the RCV1 dataset. While the performance depends heavily on the dataset, given the low cost of the approach we conclude that the approach is worth trying, especially when dealing with domain-specific data that contains many rare words. In general though, the effectiveness across datasets seems too low to be practical. *ALPS* is expensive to compute compared to random selection but it achieves consistent improvements on 3 out of the 5 extreme multi-label datasets which makes it the most effective strategy across the datasets. However, the results for the remaining datasets are often worse than random selection. Overall, *ALPS* appears to be the most appropriate approach in our work but the performance is sub par on the Arxiv and RCV1 datasets. Interestingly, these two datasets have a relatively low number of classes and high class co-occurrence(see Table 1) which might explain the good performance of random selection, especially for Macro F1 since any random text is more likely to include several classes.
DAL appears to be the weakest strategy overall. It is the best approach for the Yelp dataset in terms of Macro F1 from 500 to 1500 texts but even there the differences compared to *ALPS* are marginal. On all other datasets, *DAL* is among the worst performing approaches. Given that it is also fairly expensive to compute, we do not recommend its usage for Extreme Multi-Label AL. Finally, *CVIRS* is only effective on the NYT and Toxic datasets. For both datasets it significantly improves macro $F1$ in particular. However, at least with a transformer-based classifier it is very expensive to compute. *CVIRS* needs to be calculated from scratch each AL iteration and each computation can take several hours in our implementation depending on the amount of unlabeled data. For some datasets this increased the total computation time of the experiment to more than a week, which is extremely impractical especially in an applied AL setting where an annotator would need to check in regularly to label the selected texts.

7. Conclusion

Overall, we conclude that none of the selection strategies investigated in our experiments manage to consistently improve the AL scheme across all the datasets. In addition, we show that a much less computation-intensive approach based on the pre-trained language model can be just as effective on certain types of datasets. Consequently, we demonstrate the need for more effective AL selection strategies that also keep the computational requirements in mind. Our work offers an empirical basis for future experimentd by making the data available or possible to compile and giving a detailed description of computing F1 with a variable threshold. A next step to continue our initial investigation of this task is to extend our hypotheses to established XMTC benchmarks with thousands of labels. Undoubtedly this will generate the need for larger initial training sets in order to even have a chance of covering a significant part of the label space. In addition, we expect that specialised XMTC models (see Section 2) will make a difference. We encourage experimenting with more AL selection strategies that are based on the pre-trained language model and do not require repeated inference passes over the entire unlabeled dataset. Finally, we are looking into ways of combining multiple different sampling strategies in a single combines approach. The combined approach that leverages the strengths of multiple selection strategies has the chance of being effective on a wide range of datasets.

8. Bibliographical References

Agrawal, R., Gupta, A., Prabhu, Y., and Varma, M. (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*, pages 13–24.

An, B., Wu, W., and Han, H. (2018). Deep active learning for text classification. In *Proceedings of the 2nd International Conference on Vision, Image and Signal Processing*, pages 1–6.

Ash, J. T., Zhang, C., Krishnamurthy, A., Langford, J., and Agarwal, A. (2019). Deep batch active learning by diverse, uncertain gradient lower bounds. *arXiv preprint arXiv:1906.03671*.

Bi, W. and Kwok, J. (2013). Efficient multi-label classification with many labels. In *International conference on machine learning*, pages 405–413. PMLR.

Chang, W.-C., Yu, H.-F., Zhong, K., Yang, Y., and Dhillon, I. (2019). X-bert: extreme multi-label text classification with using bidirectional encoder representations from transformers. *arXiv preprint arXiv:1905.02331*.

Chang, W.-C., Yu, H.-F., Zhong, K., Yang, Y., and Dhillon, I. S. (2020). Taming pretrained transformers for extreme multi-label text classification. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 3163–3171.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Esuli, A. and Sebastiani, F. (2009). Active learning strategies for multi-label text classification. In *European Conference on Information Retrieval*, pages 102–113. Springer.

Gissin, D. and Shalev-Shwartz, S. (2019). Discriminative active learning. *arXiv preprint arXiv:1907.06347*.

Goudjil, M., Koudil, M., Bedda, M., and Ghoggali, N. (2018). A novel active learning method using svm for text classification. *International Journal of Automation and Computing*, 15(3):290–298.

Liu, Q., Zhu, Y., Liu, Z., Zhang, Y., and Wu, S. (2021). Deep active learning for text classification with diverse interpretations. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, pages 3263–3267.

Mayer, C. and Timofte, R. (2020). Adversarial sampling for active learning. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 3071–3079.

Prabhu, Y. and Varma, M. (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.

Ranganathan, H., Venkateswara, H., Chakraborty, S., and Panchanathan, S. (2018). Multi-label deep active learning with label correlation. In *2018 25th IEEE International Conference on Image Processing (ICIP)*, pages 3418–3422.

Ren, P., Xiao, Y., Chang, X., Huang, P.-Y., Li, Z., Gupta, B. B., Chen, X., and Wang, X. (2021). A survey of deep active learning. *ACM Computing Surveys (CSUR)*, 54(9):1–40.

Reyes, O., Morell, C., and Ventura, S. (2018). Effective active learning strategy for multi-label learning. *Neurocomputing*, 273:494–508.

Settles, B. (2011). From theories to queries: Active learning in practice. In *Active Learning and Experimental Design workshop In conjunction with AISTATS 2010*, pages 1–18. JMLR Workshop and Conference Proceedings.

Shen, Y., Yun, H., Lipton, Z. C., Kronrod, Y., and Anandkumar, A. (2017). Deep active learning for named entity recognition. *arXiv preprint arXiv:1707.05928*.

Tong, S. and Koller, D. (2001). Support vector machine active learning with applications to text classification. *Journal of machine learning research*, 2(Nov):45–66.

Wang, D. and Shang, Y. (2014). A new active labeling method for deep learning. In *2014 International Joint Conference on Neural Networks (IJCNN)*, pages 112–119.

Wu, J., Sheng, V. S., Zhang, J., Li, H., Dadakova, T., Swisher, C. L., Cui, Z., and Zhao, P. (2020). Multi-label active learning algorithms for image classification: Overview and future promise. *ACM Computing Surveys (CSUR)*, 53(2):1–35.

Yuan, M., Lin, H., and Boyd-Graber, J. L. (2020). Cold-start active learning through self-supervised language modeling. *CoRR*, abs/2010.09535.

9. Language Resource References

Bhatia, K., Dahiya, K., Jain, H., Kar, P., Mittal, A., Prabhu, Y., and Varma, M. (2016). The extreme classification repository: Multi-label datasets and code.

Chalkidis, I., Fergadiotis, M., Malakasiotis, P., and Androutsopoulos, I. (2019). Large-scale multi-label text classification on eu legislation. *arXiv preprint arXiv:1906.02192*.

kaggle. (2020). *arXiv dataset*. Public Domain.

Lewis, David D and Yang, Yiming and Russell-Rose, Tony and Li, Fan. (2004). *Rcv1: A new benchmark collection for text categorization research*. Goldsmiths, University of London.

Evan Sandhaus. (2008). *The New York Times Annotated Corpus*.