Experimenting with ensembles of pre-trained language models for classification of custom legal datasets

Tamara Matthews and **David Lillis**
School of Computer Science, University College Dublin, Belfield, Dublin 4, Ireland
tamara.matthews@ucd.ie; david.lillis@ucd.ie

Abstract

Document corpora owned by law and regulatory firms pose significant challenges for text classification; being multi-labelled, highly imbalanced, often having a relatively small number of instances and a large word count per instance. Deep learning ensemble methods can improve generalization and performance for multi-label text classification but using pre-trained language models as base learners leads to high computational costs.

To tackle the imbalance problem and improve generalization we present a fast, pseudo-stratified sub-sampling method that we use to extract diverse data subsets to create base models for deep ensembles based on fine-tuned models from pre-trained transformers with moderate computational cost such as BERT, RoBERTa, XLNet and Albert. A key feature of the sub-sampling method is that it preserves the characteristics of the entire dataset (particularly the labels' frequency distribution) while extracting subsets. This sub-sampling method is also used to extract smaller size custom datasets from the freely available LexGLUE legal text corpora. We discuss approaches used and classification performance results with deep learning ensembles, illustrating the effectiveness of our approach on the above custom datasets.

1 Introduction

The increasing volume of regulations relating to activities in the law and regulation domain require efficient methods for automated multi-label classification (MLC), which can replace expensive and time-consuming data labelling by domain experts. Examples include legal professionals who may need to categorize the types of business activities a contract relates to, or label types of individual clauses (e.g. restrictive covenants, penalty clauses). Similarly, companies involved in financial regulation may need to label regulatory texts with the focus of the regulation (e.g. fraud, accounting obligations or mis-selling).

Since the creation of BERT (Devlin et al., 2018) the "foundation model" for pre-trained transformer-based language models, various new types of transformer models for NLP have been developed: some improve the pre-training method such as RoBERTa (Liu et al., 2019) and DeBERTa (He et al., 2020), the vocabulary size and specificity such as LEGAL-BERT (Chalkidis et al., 2020), or enable input sequence lengths larger than 512 tokens (Tay et al., 2020a) such as Reformer (Kitaev et al., 2020), CogLTX (Ding et al., 2020), Longformer (Beltagy et al., 2020), Big bird (Zaheer et al., 2020).

The major advantage of using pre-trained transformers for MLC is their large, context and semantic aware language models (LM), which can significantly improve classification results even on small datasets (Sun et al., 2019). Fine-tuning pre-trained language models on legal texts can be challenging due to their uncommon vocabulary (containing domain specific, rare, and conceptually complex words) and the large text length in each document, exceeding the maximum sequence input length of pre-trained transformer-based models.

There is a trade-off between the pre-trained model size and the available computational power. The pre-trained model size increases with the vocabulary size and with the number of parameters (the number of bi-LSTM layers). The larger models enable improved performance at the cost of increased computational effort (memory size and processing units) (Tay et al., 2020b).

While pre-trained transformer models can provide efficient solutions for text classification, fine-tuned models often lack flexibility as the fine-tuning restricts generalization to the new (narrower) domain. MLC based on deep ensemble models can be improve generalization but these approaches are constrained by high computational costs.

Here we present MLC results using deep-learning ensembles of fine-tuned models discussing their suitability for limited resources environments and small size datasets (6000 to 7000 instances). Our approach is based on ensembles of fine-tuned models obtained by transfer learning from pre-trained transformers. We use transformers that require moderate fine-tuning costs as BERT, RoBERTa, XLNet (Yang et al., 2019) and ALBERT (Lan et al., 2020). This study includes homogeneous and heterogeneous ensembles of deep base-learners each generated from data re-sampling with replacement, using our custom pseudo-stratified random sampling method.

Our contribution includes: a custom pseudo-stratified sampling method for sub-sampling and train/test splitting of imbalanced multi-label datasets, used for generating diverse datasets; aggregating ensemble models using fine-tuned base-models (transfer learning from pre-trained NLP transformers), and discussing results. The datasets used in our work, sub-sampled from benchmark legal text datasets are available.

2 Multi-label classification

Real-life datasets as multi-domain business documentation or collections of legal and regulatory documents are multi-labelled and highly imbalanced, presenting a complex MLC problem, relating one example to multiple categories.

The imbalanced label distribution increases the complexity of the learning problem as stratified sampling cannot be applied for creating a balanced test/train split that includes all labels. Iterative methods for test-train split have been proposed to ensure a similar label distribution in both test and train sets (Sechidis et al., 2011).

Several approaches are designed to enable classic algorithms (Naive Bayes, SVM, Random Forests, k-means) to perform on multi-label datasets: (a) Algorithm Adaptation (Szyma, 2019) (b) Problem Transformation (Binary Relevance, Label Powerset and Classifier Chains methods) and (c) Ensemble learning, described in recent reviews dedicated to MLC (Bogatinovski et al., 2022; Kowsari et al., 2019). Since neural networks and deep learning methods can generate a multiple prediction output by design, such methods support multi-label classification and can be applied with or without approaches (a)-(c), although these can significantly improve the performance of deep learning methods also.

The multi-label data can produce ensembles of models, using one-vs-all and one-vs-one techniques. Other methods are based on re-sampling and sub-sampling (Bagging and Boosting) or re-arranging the data into convenient domains matching labels' distribution: random k-labelsets (Tsoumakas and Vlahavas, 2007), hierarchical arrangements, pruned sets (Read et al., 2008).

2.1 Performance measures and thresholding

A variety of performance measures have been designed to assess the various goals in multi-label classification: Hamming loss, ranking loss, one-error, average precision, coverage, micro-F1 and macro-F1. Used for imbalanced datasets, F-scores (measures) can be optimized either as an empirical utility maximization EUM (optimal classifier) or as a decision-theoretic approach DTA (predictions by optimal classifiers) (Lewis, 1995), DTA being better for handling rare classes and for domain adaptation tasks.

The F-measure optimization is usually performed in two steps: learn a score function from a ML algorithm (optimized for DTA) and then select a threshold to maximize the empirical F-measure (Ye et al., 2012).

3 Ensemble deep learning

Ensemble learning methods (Madjarov et al., 2012; Read et al., 2008; Tsoumakas and Vlahavas, 2007; Dong and Han, 2004) use multiple base learners to form an ensemble learner (model) to improve generalization and model performance.

The models can be generated using the same ML algorithm or a combination of algorithms. Ensemble methods are commonly used on imbalanced datasets with data over-sampling techniques.

The ensemble prediction is obtained from a suitable aggregation rule: plurality or majority voting, (weighted) predictions' mean, best performance model, or using learning systems to combine predictions (Zhou et al., 2002).

Through the use of multiple models, ensemble deep learning can improve generalization as well as prediction performance (Yang et al., 2021). Using base models generated from fine-tuning pre-trained language models is expected to further improve performance due to the large context and semantically aware base models.

Ensemble methods provide an improved performance based on the diversity of multiple models. Creating a diverse committee of base learners that is still consistent with the training data was demonstrated to be of high importance in generating a good ensemble (Dietterich, 2000) as predictions from each learner are combined into the classification outcome.

Methods to generate diversity such as bagging (Breiman, 1996) and boosting are usually applied. In bagging ensembles, multiple models are created using data subsampling methods (i.e. random drawing with replacement) and a joint prediction is obtained through a voting mechanism. In boosting using AdaBoost (Freund et al., 1996) and its variants, data is sampled according to weights assigned to instances and sampling weights are updated based on classification outcomes to improve the scores. These meta-learner methods can be applied to any base learner, including deep learners.

Using the disagreement of an ensemble member with the ensemble's prediction as a diversity measure, (Melville and Mooney, 2004) show that there is a significant Spearman rank correlation between diversity and error reduction of the ensemble. They conclude that increasing ensemble diversity leads to reducing generalization error of the ensemble.

The success of these meta-learners has led to applications in a variety of fields, including text categorization (Shapire and Singer, 2000; Dong and Han, 2004) also revealing some weaknesses. While bagging can reduce the error due to variance of the base classifier, using stable learners, such as Naive Bayes, will not reduce the error. Also, small datasets can generate a limited amount of diversity.

Boosting can perform poorly with insufficient data (Freund and Schapire, 1999) or noisy labels (incorrect class labels in training) (Dietterich, 2000). Other drawbacks for deep model ensembles are the high costs of computing power, memory and process time when training multiple deep learners (Yang et al., 2021).

Using transfer learning from pre-trained transformer language models and the available cloud computing GPUs, our work investigates the feasibility of deep learning ensemble models for text classification. We use a bagging-type data resampling with homogeneous and heterogeneous ensembles to assess how data diversity and type of pre-trained model improve the ensemble model.

4 Legal Datasets

In recent years, curated collections of legal texts have been made available, along with their dedicated language models as: CUAD (Hendrycks et al., 2021), ECHR (Chalkidis et al., 2019), EURLEX57k (Fergadiotis et al., 2018). Based on these, several benchmark datasets included in LexGLUE (available on the HuggingFace[1] platform) (Chalkidis et al., 2021) enable comparison of various AI approaches using the same datasets and metrics.

The LexGLUE multi-labelled datasets (ECtHR, EUR-LEX) include the ECHR-A and ECHR-B datasets containing case descriptions of the European Court of Human Rights, where labels represent articles of the European Convention on Human Rights that have been violated or allegedly violated. LexGLUE also includes EUR-Lex data , which consists of EU legislation that has been labelled according to 7,000 EuroVoc concepts[2] (with 4 granularity levels). The available EUR-Lex data in LexGLUE contains 65,000 documents annotated with the 100 most frequent concepts from level 2.

LexGLUE datasets are designed for Natural Language Undersanding (NLU) and include a temporal 'concept drift' in the datasets for development and test (i.e. data issued at a later time, within five years of training set documents). Since in our study we do not approach NLU problems, we use only the original training datasets which we split into new, smaller train/test sets that both include data collected within the same time interval, limiting the 'concept drift'.

Here we only use the training data of the multi-labelled datasets from the LexGLUE set (ECHR-A, ECHR-B and EUR-Lex datasets) and sub-sample from each, the datasets A, B and C (respectively) without the 'concept drift'. The datasets A and B were sub-sampled using the custom function described in Section 5.1 while for dataset C we select from the EUR-Lex train data only the instances labelled as 'Regulations' and take 50 of the most frequent labels (out of 2941, see Table 1). The unlabelled instances (where these are included in the original train set) have also been removed.

The complexity characteristics of the new datasets are shown in Table 1 in comparison to the original data, where the labels' Cardinality (*Car*) and Density (*Den*) for a dataset with N the number of instances, Y_i the set of labels for instance i and L the number of instances are defined as:

[1]https://huggingface.co/datasets/lex_glue

[2]https://eurovoc.europa.eu

Dataset	N	L	Car	Den	IR
ECHR-A train	8086	10	1.32	0.39	114.00
ECHR-B train	8866	10	1.48	0.45	67.12
EUR-Lex train (Regulations)	29600	2941	4.94	0.002	3554.00
A	5651	10	1.23	0.37	125.34
B	5925	10	1.35	0.41	89.98
C	7201	50	3.34	0.07	8.77

Table 1: Data complexity for the sub-sampled datasets

Car = $\frac{1}{N}\sum_{i=1}^{N}|Y_i|$; and *Den* = $\frac{1}{N}\sum_{i=1}^{N}\frac{|Y_i|}{L}$; while the maximum imbalance ratio (IR) is the ratio of the most common label against the rarest one. The new datasets have a smaller number of instances (close to that of custom datasets) and an IR that is higher for datasets A and B and much lower for dataset C.

4.1 Dealing with large text lengths

One characteristic of text in the law and regulatory domains is the large text length, often exceeding 5000 words/entry. Using pre-trained transformers with low to moderate computational demands (as BERT, ALBERT, RoBERTa) is a cost-effective approach to perform MLC on legal text, with limitations due to their maximum input sequence length.

General-purpose NLP transformers can process a maximum input length of 512 tokens, much smaller than the text length in legal datasets. This problem is usually solved by applying 'text truncation', 'hierarchical methods' or 'data transformation'. Text truncation uses only the 'head', 'tail' or 'head + tail' parts of the document, considered to contain the key information. In 'hierarchical' methods, the text is split into sequences (of lengths smaller than 510) their pre-trained embedded representations are extracted from the [CLS] token in the last hidden layer and then combined using max-pooling or mean-pooling (attention weighted) to obtain the embeddings representation of the entire text for classification (Sun et al., 2019).

Other hierarchical methods use hierarchical transformers that generate embeddings and encode these again using a shallow transformer (Chalkidis et al., 2021). The 'data transformation' method (applied here) performs text splitting into sections of suitable length using them to create an expanded dataset (preserving the associated labels). For inference, these entries can be re-joined after fine-tuning along with their predictions.

While truncation methods can miss important information, the 'hierarchical' methods (which create embedding representations of the whole input text) and the 'data transformation' method that re-joins text sections and predictions can both improve prediction outcomes.

5 Methods

5.1 Generating diverse datasets

Real-life, custom legal datasets have a small number of instances and are highly imbalanced – requiring efficient data sampling/splitting methods that enable all labels to be represented in both train and test sets. The iterative method for stratified sampling of imbalanced multi-label datasets proposed by Sechidis et al. (2011) requires reaching convergence, which is time-consuming when used for repeated sub-sampling.

Here we have designed a fast, pseudo-stratified sampling function that provides a train/test split configured to follow the label distribution in the original data and ensuring that all labels are represented in both train and test sets.

```
$C$: corpus of labelled text instances
$L$: set of all labels
$M$: desired minimum instances per label (M=10)
ratio_in: proportionality factor (configurable)
$N_l$: number of instances to be selected for label $l$
labelled$(i,l)$: true iff instance $i$ is labelled with label $l$
distinct_labels$(I)$: set of all distinct labels associated
    with instances in set $I$
choose_random$(I,n)$: set of $n$ randomly chosen instances
    from set $I$
```

```
for $l \in L$ do
    $I_l = \{i | i \in C \wedge \text{labelled}(i,l)\}$
    if $|I_l| < M$ then
        $N_l = 1$
    else
        $N_l = |I_l|/\text{ratio\_in}$
    end if
end for
do
    $TEST = \emptyset$
    for $l \in L$ do
        $TEST = TEST \cup \text{choose\_random}(I_l, N_l)$
    end for
    $TRAIN = C \setminus TEST$
while distinct_labels$(TEST) \neq$ distinct_labels$(TRAIN)$
```

Figure 1: Pseudo-code describing the basic algorithm for the pseudo-stratified sampling function

This pseudo-stratified sampling method is designed to randomly extract (without replacement) instances that represent each label in a number proportional to the frequency of each label. We define two integer parameters: M (related to the minimum number of instances per label) and 'ratio_in', which configure the number of entries to be extracted for the test set for each label (basic algorithm described in Figure 1).

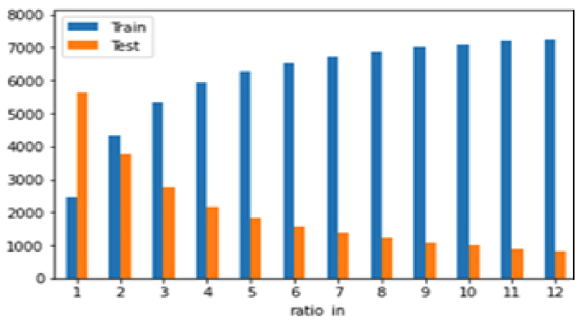

Figure 2: Train and Test size after split using the proposed pseudo-stratified function, for a range of 'ratio_in' values (starting from the ECtHR-A train dataset).

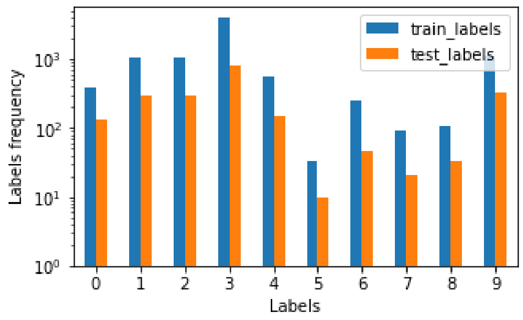

Figure 3: Labels' distribution when splitting the ECtHR-A training dataset into train/test using 'ratio_in'=7

For the training set we keep only the instances that are not included in the test set. The 'ratio_in' values can be chosen from a value of 2 up to a maximum depending on availability of instances for each label. A plot of the train and test sizes for various ratios is shown in Figure 2.

This method provides an efficient splitting ensuring that all labels are represented in both train and test sets. Low values of the 'ratio_in' (2 to 6) give a close representation of the original label distribution, while values above 10 generate only a roughly similar distribution. It can be argued that more data diversity can be obtained using a more distorted distribution. The typical label distribution for the train and test sets is shown in Figure 3.

5.2 Data preparation

The main data is one-hot-encoded and split as train/test datasets, then the train set is further split by re-sampling into train/test sets to create the base-models.

We apply the 'data transformation' method, splitting each text entry into sections of up to 120 words (such that only entire sentences are included in each sequence) while preserving the corresponding labels, creating an expanded dataset. The split sequences are then re-joined along with their predictions at testing or inference time.

The medium and average lengths of the sequences in each dataset are shown in Table 2. Other improvements of the sequence content (i.e. adding to each sequence the last sentence from the previous sequence) are not applied here.

Datasets	**A**		**B**		**C**	
	train	test	train	test	train	test
Median length	108	108	103	105	73	64
Mean Length	104.29	104.42	94.84	96.11	72.51	69.5

Table 2: The medium and mean lengths of sequences in the tran and test sets for the datasets A, B and C after data transformation

The 'data transformation' method is context and semantic aware within each sequence, as each is expected to contain a representative amount of information, that enables classification. Moreover, such phrases generally refer to a specific topic – identified by a certain label group. The 'data transformation' generates much larger train and test datasets on which the model is defined, being a type of boosting technique.

An advantage of text splitting into sequences is that we obtain an increased number of instances, improving the statistical basis of our model (and its bias). This is a type of 'data over-sampling' leading to an ensemble model where predictions are obtained by aggregating over each original instance, which improves prediction variance.

5.3 Generating deep learning ensembles

The proposed method for generating diverse datasets is applied for supervised multi-label classification of legal datasets, creating a train/test split for each. The text for every entry in the train and test datasets is split into sequences of up to 120 words (as described in Section 5.2) forming an expanded dataset where each sequence keeps its original labels, on which each base-model is defined. For testing after fine-tuning, these 'split' entries are re-joined along with their predictions generating an aggregated prediction set.

We create homogeneous and heterogeneous ensembles of deep models, where base-models are fine-tuned from the following pre-trained transformer models: BERT-based-uncased, RoBERTa-base, ALBERT-base-v2 and XLNet-base-cased. The **simpletransformers**[3] library (based on PyTorch) has been used, taking advantage of the unified data, model output formats and default model arguments available (i.e. batch sizes of 16, sequence length of 128, learning rate of 4.e-5).

[3] https://simpletransformers.ai/

To enable generalization, the base learners can be ‘incomplete’ (not fully trained) models and as the fine-tuning reaches convergence fast (within 3 epochs), we have used only 2 epochs for fine-tuning the base-learners when working on datasets sub-sampled from ECtHR-A and ECtHR-B (10 labels) and on 5 epochs for datasets sub-sampled from the EUR-Lex dataset, (from which we selected data including the most frequent 50 labels). Label weights are applied during training of the model as a means of regularization. Label weights are proportional to labels’ frequency in each newly sub-sampled dataset for each base model. The Label Ranking Average Precision[4] (LRAP) metric is used for fine-tuning with the same threshold of 0.5 for all labels.

A set of 10 base-models has been generated for each homogeneous ensemble, each base-model being fine-tuned on a newly resampled dataset. The resampling is performed using the described pseudo-stratified random sampling function, which is used as a type of bagging technique (sampling without replacement to generate the train and test data). As the fine-tuning for the 10 base-models is performed within a loop, to ensure independent results, the model outputs and checkpoints are deleted at the end of each fine-tune and the pre-trained model type is re-initialized before starting the fine-tuning for the next base model.

The base-model prediction arrays are obtained from each ‘raw_output’ (as given by the prediction outputs of the **simpletransformers** library) after applying a *sigmoid* activation. We aggregate the base-models as the mean of their prediction arrays thus generating the ensemble model (which is also a prediction array).

In the present study we use three types of optimization: (i) we fine-tune (optimize) each classifier using LRAP metrics (the average over each ground truth label assigned to each instance, of the ratio of true vs. total labels with lower score); (ii) we use micro-F1 with thresholding to optimize the predictions on the expanded dataset and find the best three models; (iii) to find the optimal prediction threshold on the re-joined dataset, we use sample averaged F1 as the harmonic mean of Precision (P) and Recall (R) averaged over the sample size S where Y_i and $h(x_i)$ are the true and the predicted labels for an example x_i: P = $\frac{1}{S}\sum_{i=1}^{S}\frac{Y_i \cap h(x_i)}{h(x_i)}$;

R = $\frac{1}{S}\sum_{i=1}^{S}\frac{Y_i \cap h(x_i)}{Y_i}$; F1 = $2\frac{PR}{P+R}$

The choice for the optimizations (ii) and (iii) was made based on best results obtained using these types of performance measures for the expanded and re-joined datasets, respectively.

[4] https://scikit-learn.org/stable/modules/generated/sklearn.metrics.label_ranking_average_precision_score.html

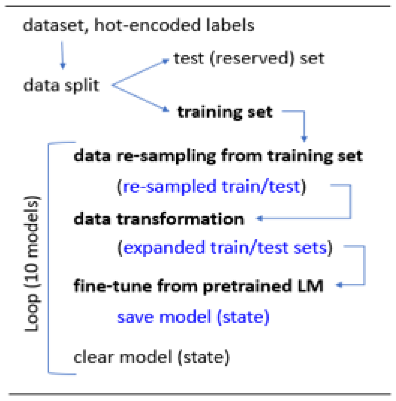

Figure 4: Schematic of the workflow generating the deep ensemble model.

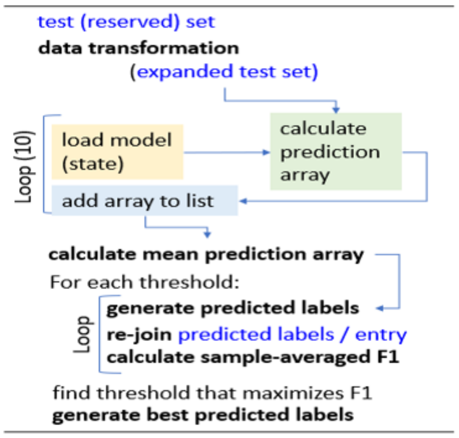
test (reserved) set
data transformation
(expanded test set)
Loop (10)
load model (state)
calculate prediction array
add array to list
calculate mean prediction array
For each threshold:
Loop
generate predicted labels
re-join predicted labels / entry
calculate sample-averaged F1
find threshold that maximizes F1
generate best predicted labels

Figure 5: Schematic of the workflow for test and inference using the deep ensemble model.

5.4 The workflows

The workflow for generating the ensemble model of deep base-learners is shown in Figure 4 while the workflow for testing and inference is shown in Figure 5 . Both show a concise, intuitive overview of the main steps in generating the models and performing inference.

For inference, we load the saved models, generate the prediction array from each base model and their ensemble as the mean prediction array. For a given threshold in the range 0.1 - 0.9 we generate the predicted labels for the expanded dataset, then aggregate (re-join) this back into the original, while keeping all unique labels obtained from each split text entry. The threshold optimization is performed by seeking the threshold that gives the F1 maximum (calculated as sample averaged).

6 Experiments

The ‘main’ datasets (A, B and C) have been subsampled from each original dataset (ECtHR-A, ECtHR-B and EUR-Lex, respectively) using the pseudo-stratified sampling function at ratio_in=7, extracting a ‘main’ dataset (Table 1) and a test dataset (for inference) with a size of about 25-30% of the ‘main’ dataset.

During fine-tuning, the ‘main’ dataset is split into train/test using the pseudo-stratified sampling function at ratio_in =5 to generate diverse models. The train/test datasets are pre-processed (according to the description in Section 5.2) obtaining expanded datasets with a maximum text length of 120 words per entry. Fine-tuning is performed on the expanded datasets optimizing the LRAP metrics using model configuration arguments defaults from the **simpletransformers** library.

Several types of deep ensembles are generated: (a) homogeneous and (b) heterogeneous, where the ensemble is created from base-models fine-tuned on the same pre-trained model, or on different ones, respectively. For each (a), and (b) two types of ensemble predictions are then constructed: the ‘mean model’ by averaging the prediction arrays of the 10 base-learners and the ‘best three’ model by averaging the prediction arrays of the best three models.

The best three models are selected as those reaching the highest micro-F1 values at the threshold that maximizes micro-F1 on the expanded dataset, obtaining an optimal threshold at values of 0.2-0.3 for the A and B datasets and of 0.5 for dataset C.

The ensemble prediction arrays are optimized using thresholding applied on the ‘re-joined’ dataset with merged predictions, choosing as threshold the value that maximizes the sample averaged F1. For the A and B datasets the optimal threshold is 0.8-0.85 while for dataset C the optimal threshold is 0.5. All predictions with probabilities above the set threshold are kept for each instance.

7 Results and Discussion

Homogeneous ensemble models have been created using the four pre-trained models considered. The ensembles are aggregated as the mean prediction over 10 models, the mean prediction over the best three models, and compared to the best model. The best models are those with the highest micro-F1 values after applying thresholding.

	Expanded data			**Re-joined data**		
	Mean	**Best3**	**Best**	**Mean**	**Best3**	**Best**
RoBERTa	0.73	0.73	0.72	**0.81**	0.8	0.78
XLNet	0.73	0.73	0.71	**0.81**	**0.81**	0.78
BERT	0.73	0.66	0.73	**0.81**	**0.81**	0.78
ALBERT	0.72	0.73	0.72	**0.81**	0.78	**0.81**

Table 3: Optimised micro-F1 scores for dataset A (expanded and re-joined data) on homogeneous ensembles.

	Expanded data			**Re-joined data**		
	Mean	**Best3**	**Best**	**Mean**	**Best3**	**Best**
RoBERTa	0.71	0.71	0.69	**0.79**	**0.79**	0.77
XLNet	0.71	0.7	0.69	**0.79**	**0.79**	0.76
BERT	0.71	0.7	0.68	0.78	0.78	0.75
ALBERT	0.7	0.69	0.68	0.78	0.78	0.76

Table 4: Optimised micro-F1 scores for dataset B (expanded re-joined data) on homogeneous ensembles.

	Expanded data			**Re-joined data**		
	Mean	**Best3**	**Best**	**Mean**	**Best3**	**Best**
ROBERTA	0.74	0.74	0.73	0.79	0.78	0.78
XLNET	0.75	0.75	0.74	**0.80**	0.79	0.78
BERT	0.74	0.73	0.72	0.78	0.78	0.77
ALBERT	0.73	0.74	0.73	0.77	**0.80**	0.76

Table 5: Optimised micro-F1 scores for dataset C (expanded and re-joined data) on homogeneous ensembles.

	Expanded dataset			**Re-joined dataset**		
F1 scores	**micro**	**macro**	**weigh.**	**micro**	**macro**	**weigh.**
dataset A	0.74	0.67	0.73	0.82	0.75	0.82
dataset B	0.73	0.65	0.68	0.79	0.67	0.80
dataset C	0.75	0.67	0.74	0.79	0.71	0.81

Table 6: Optimized F1 scores for the heterogeneous ensemble model using the Best 3 models from each model type on datasets A, B and C.

Comparative results for threshold optimized micro-F1 for the ensemble models calculated as the ‘Mean’ (over the 10 prediction arrays), ‘Best 3’ (mean of best three models’ predictions) and ‘best’ (predictions from the best model) are shown in Table 3, Table 4 and Table 5 for datasets A, B and C, respectively. Table 6 shows F1 scores for the heterogeneous ensembles built on the four types of fine-tuned models using the mean prediction from each ‘Best 3’ models).

While scores’ improvements between the homogeneous models and heterogeneous ones are only within 1-2%, there is an important increase of about 5-8% in scores between the expanded model (based on the sequence split dataset) where average optimized micro-F1 is 0.73 and the one for the re-joined data with micro-F1 averages at 0.81 (i.e:

	LinSVC			MultinomialNB		
	micro	macro	weigh.	micro	macro	weigh.
dataset A	0.8	0.69	0.79	0.47	0.09	0.33
dataset B	0.76	0.64	0.75	0.44	0.09	0.29
dataset C	0.85	0.71	0.83	0.58	0.33	0.44

Table 7: Baseline scores for micro, macro and weighted averaged F1 from sklearn LinSVC and MultinomialNB multi-output methods.

values from Table 3).

The expanded dataset created from sequences with their original labels generates another type of ensemble (a type of over-sampling) which is aggregated at prediction time by joining predictions belonging to each original entry. This type of ensemble appears to be more efficient than aggregating (as a mean) prediction arrays.

In Table 7 we show results obtained for the same datasets, using a strong baseline, the **tf-idf** text processing with LinSVC linear support vector classification from **sklearn** multi-label classification. The LinSVC baseline has high scores, especially for dataset C, as the LinSVC algorithm with the one-vs-rest approach performs very well for datasets with a low imbalance ratio (Table 1). Other multi-output classifiers as 'Multinomial Naive Bayes' cannot handle the data complexity and obtains low scores.

8 Conclusions

We performed multi-label classification on imbalanced datasets sub-sampled from legal text datasets provided in LexGLUE, using deep learning ensembles of base-learners built as fine-tuned transfer models on four well-known NLP transformer models (BERT, RoBERTa, ALBERT and XLNet).

We designed a pseudo-stratified sampling method for imbalanced multi-label datasets to re-sample diverse datasets which were used to generate base-models for homogeneous deep-ensembles.

Using the GPUs on Google-Colab the training times per epoch for the base models were in the range of 2 - 15 min/epoch. Training times increase with the number of instances as do the inference times (between 1 min - 3 min per model), leading to acceptable times for generating and using such ensemble models.

The values for micro-F1 scores from homogeneous and heterogenous ensembles on the expanded datasets reach close values that improve very little from the 'best model' score (only by 1% - 2%). This also occurs for the scores on the re-joined datasets. Such small differences in the overall outcomes show the close performance of the fine-tuned models, even when starting from different transformer models. These results were also due to the threshold optimization applied, which levels out the original difference in outcomes.

Nevertheless, there is an important score improvement between the expanded and re-joined datasets, indicating that the ensemble generated by text splitting can substantially improve classification outcomes at aggregation time, as threshold optimized scores for the re-joined dataset are 7%-10% higher than those obtained on the expanded dataset.

Overall, the best micro-F1 scores obtained on the re-joined dataset reach 0.78-0.82 which are satisfactory results for imbalanced multi-label datasets of up to 50 labels. Higher scores can be obtained by fine-tuning with optimized model hyperparameters to obtain improved base-models.

The text length in each entry and the degree of label imbalance play a significant role in the fine-tuning performance. For the deep learning models, dataset C (50 labels) with short mean text length per entry and low imbalance obtained similar scores to those for datasets A and B (10 labels) with larger text length/entry and high imbalance. The LinSVC baseline (using one-vs-rest scheme) obtained lower scores on datasets A and B than on dataset C, due to difficulties of binary-relevance type algorithms on highly imbalanced data.

As we have considered small datasets, these can only generate a limited amount of diversity, leading to less variability in the ensemble models. To improve results using ensembles, improved methods to create diversity in datasets should be tested.

Acknowledgements

This work is supported by the Department of Enterprise, Trade and Employment of the Government of Ireland, and by Enterprise Ireland, through the Disruptive Technologies Innovation Fund (Grant No. DTIF2019-074 "TRANSPIRE").

References

Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The Long-Document Transformer. *arXiv:2004.05150*.

Jasmin Bogatinovski, Ljupčo Todorovski, Sašo Džeroski, and Dragi Kocev. 2022. Comprehensive comparative study of multi-label classification methods. *Expert Systems with Applications*, 203:117215.

Leo Breiman. 1996. Bagging Predictors. *Machine learning*, 24.2:123–140.

Ilias Chalkidis, Ion Androutsopoulos, and Nikolaos Aletras. 2019. Neural Legal Judgment Prediction in English. *arXiv:1906.02059v1*.

Ilias Chalkidis, Manos Fergadiotis, Prodromos Malakasiotis, Nikolaos Aletras, and Ion Androutsopoulos. 2020. Legal-bert: The muppets straight out of law school. *arXiv preprint arXiv:2010.02559*.

Ilias Chalkidis, Abhik Jana, Dirk Hartung, Michael James Bommarito, Ion Androutsopoulos, Daniel Martin Katz, and Nikolaos Aletras. 2021. LexGLUE: A Benchmark Dataset for Legal Language Understanding in English. *SSRN Electronic Journal*.

Jacob Devlin, Ming Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of deep bidirectional transformers for language understanding. *arXhiv: 1810.04805*.

Thomas G Dietterich. 2000. Ensemble Methods in Machine Learning. In *Multiple Classifier Systems. MCS 2000.*, volume 1857, pages 1–15.

Ming Ding, Chang Zhou, Hongxia Yang, and Jie Tang. 2020. CogLTX: Applying BERT to long texts. *Advances in Neural Information Processing Systems*, 2020-December(NeurIPS).

Yan-Shi Dong and Ke-Song Han. 2004. A comparison of several ensemble methods for text categorization. In *Proceedings of the 2004 IEEE International Conference on Services Computing (SCC'04)*, pages 0–3. IEEE.

Manos Fergadiotis, Prodromos Malakasiotis, Ion Androutsopoulos, and Ilias Chalkidis. 2018. Large-Scale Multi-Label Text Classification on EU Legislation. *arXiv:1906.02192v1*.

Yoav Freund and Robert E Schapire. 1999. A Short Introduction to Boosting. *Journal of Japanese Society for Artificial Intelligence*, 14(5):771–780.

Yoav Freund, Robert E Schapire, and Murray Hill. 1996. Experiments with a New Boosting Algorithm. *icml*, 96:148–156.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2020. DeBERTa: Decoding-enhanced BERT with Disentangled Attention. *arXiv:2006.03654*.

Dan Hendrycks, Collin Burns, Anya Chen, and Spencer Ball. 2021. Cuad: An expert-annotated nlp dataset for legal contract review. *arXiv preprint arXiv:2103.06268*.

Nikita Kitaev, Anselm Levskaya, and Łukasz Kaiser. 2020. REFORMER: THE EFFICIENT TRANSFORMER. In *ICLR 2020*, pages 1–12.

Kamran Kowsari, Kiana Jafari Meimandi, Mojtaba Heidarysafa, and Sanjana Mendu. 2019. Text Classification Algorithms : A Survey. *information*, 10(150):1–68.

Zhenzhon Lan, Chen Mingda, Goodman Sebastian, Gimpel Kevin, Piyush Sharma, and Soricut Radu. 2020. ALBERT: A LITE BERT FOR SELF-SUPERVISED LEARNING OF LANGUAGE REPRESENTATIONS. In *ICLR 2020*, pages 1–17.

D Lewis. 1995. Evaluating and optimizing autonomous text classification systems. In *Proceedings of the 18th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 246–254.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Gjorgji Madjarov, Dragi Kocev, and Dejan Gjorgjevikj. 2012. An extensive experimental comparison of methods for multi-label learning. *Pattern Recognition*, 45:3084–3104.

Prem Melville and Raymond J. Mooney. 2004. Diverse Ensembles for Active Learning. In *Proceedings of the 21st International Conference on Machine Learning, (ICML-2004)*, pages 584–591, Banff, Canada.

Jesse Read, Bernhard Pfahringer, Geoff Holmes, and New Zealand. 2008. Multi-label Classification using Ensembles of Pruned Sets. In *Eighth IEEE International Conference on Data Mining*, pages 995–1000.

Konstantinos Sechidis, Grigorios Tsoumakas, and Ioannis Vlahavas. 2011. On the stratification of multi-label data. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 6913 LNAI(PART 3):145–158.

Robert E. Shapire and Yoram Singer. 2000. BoosTexter : A Boosting-based System for Text Categorization. *Machine Learning*, 39:135–168.

Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang. 2019. How to Fine-Tune BERT for Text Classification? *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 11856 LNAI(2):194–206.

Piotr Szyma. 2019. scikit-multilearn : A scikit-based Python environment for performing multi-label classification. *Journal of Machine Learning Research*, 20:1–22.

Yi Tay, Mostafa Dehghani, Samira Abnar, Yikang Shen, Dara Bahri, Philip Pham, Jinfeng Rao, Liu Yang, Sebastian Ruder, and Donald Metzler. 2020a. Long range arena: A benchmark for efficient transformers. *arXiv preprint arXiv:2011.04006*.

Yi Tay, Mostafa Dehghani, Dara Bahri, and Donald Metzler. 2020b. Efficient Transformers: A Survey. *ACM Computing Surveys*, pages 1–28.

Grigorios Tsoumakas and Ioannis Vlahavas. 2007. Random k -Labelsets : An Ensemble Method for Multilabel Classification. In *European conference on machine learning. Springer, Berlin, Heidelberg*, pages 406–417.

Yongquan Yang, Haijun Lv, and Ning Chen. 2021. A Survey on Ensemble Learning under the Era of Deep Learning. *arXiv:2101.08387*.

Zhilin Yang, Zihang Dai, Yiming Yang, and Jaime Carbonell. 2019. XLNet : Generalized Autoregressive Pretraining for Language Understanding. In *33rd Conference on Neural Information Processing Systems (NeurIPS 2019)*, NeurIPS, pages 1–11, Vancouver, Canada.

Nan Ye, Kian Ming A. Chai, Wee Sun Lee, and Hai Leong Chieu. 2012. Optimizing F-Measures: A Tale of Two Approaches. In *Proceedings of the 29th International Confer- ence on Machine Learning, Edinburgh, Scotland, UK*. arXiv:1206.4625.

Manzil Zaheer, Guru Guruganesh, Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, and Amr Ahmed. 2020. Big bird: Transformers for longer sequences. *Advances in Neural Information Processing Systems*, 2020-December(NeurIPS).

Zhi-hua Zhou, Jianxin Wu, and Wei Tang. 2002. Ensembling neural networks: Many could be better than all . *Artificial Intelligence*, 174(18):1570.