Adapting Transformers for Multi-Label Text Classification

Haytame Fallah, Patrice Bellot, Emmanuel Bruno, Elisabeth Murisasco

Adapting Transformers for Multi-Label Text Classification

Haytame Fallah[1,3,*], Patrice Bellot[1], Emmanuel Bruno[2] and Elisabeth Murisasco[2]

[1] *Aix-Marseille Univ, University of Toulon, CNRS, LIS, Marseille, France*

[2] *University of Toulon, Aix-Marseille Univ, CNRS, LIS, Toulon, France*

[3] *Hyperbios, Toulon, France*

Abstract

Pre-trained language models have proven to be effective in multi-class text classification. Our goal is to study and improve this approach for multi-label text classification, a task that has been surprisingly little explored in the last few years despite its many real world applications. In this paper, our originality is to propose architectures for the classification layers that are used on top of transformers to improve their performance for multi-label classification. Our contribution involves the evaluation of thresholding methods on several transformers, either by computing an individual threshold for each label (***IT***) or a global one (***GCT***). We also propose two approaches for multi-label text classification. The first consists in adding a parameter for learning the number of labels present for a given example (***NHA***). The second approach consists in adding a layer to the classification layers in order to learn the features for selecting the relevant labels while avoiding the use of thresholds (***TL***). We evaluate these approaches on two English corpora of newspaper articles and scientific papers and then on a new multi-label dataset of French scientific article abstracts publicly available. The evaluations show that the performance of our proposals exceeds that of state-of-the-art multi-label text classification methods for the evaluated datasets, and are transposable to any multi-label classification problem.

Keywords

Multi-label classification, Transformers, BERT, French Transformers,

1. Introduction

Multi-label classification is a generalization of the multi-class classification problem, where each instance is associated with exactly one label. In multi-label text classification, the goal is to associate one or more labels to the input text sample. It is an important natural language processing task that has many applications in other NLP tasks such as question answering or entity recognition, but also real-world applications such as information retrieval (e.g. metadata enrichment and analysis in digital libraries) or content recommendation, and so on.

Multi-label text classification is a challenging task due to the fact that several factors must be taken into account such as the dependencies that can exist between the labels, the complexity of

CIRCLE (Joint Conference of the Information Retrieval Communities in Europe) 2022, July 04–07, 2022, Samatan, FRANCE

[*] Corresponding author.

haytame.fallah@lis-lab.fr (H. Fallah); patrice.bellot@univ-amu.fr (P. Bellot); emmanuel.bruno@univ-tln.fr (E. Bruno); elisabeth.murisasco@univ-tln.fr (E. Murisasco)

0000-0001-8698-5055 (P. Bellot)

CEUR Workshop Proceedings (CEUR-WS.org)

extracting semantic features from the noisy input text that can contain redundant information, and mapping those features to multiple targets, while also finding the discriminative information that allows the identification of each label of the document.

Several methods have been proposed to tackle the multi-label classification problem, whether it's traditional methods such as Binary Relevance [1], or deep learning-based approaches such as CNN, RNN (and a combination of both approaches [2]) or the attention mechanism. These methods manage to capture the semantic features of the document but fail to consider the dependencies that can exist between labels. Hierarchical models [3] and graph neural networks [4] as well as other architectures [5] have been introduced to better capture those dependencies. But with the emergence of attention-based transformers [6] and their ability to better extract the semantic representations of text documents, and adaptation of these models for multi-label text classification, that has the potential of achieving overall better results, is yet to be explored.

Few multi-label text datasets are popular among the papers treating the multi-label problem. AAPD [5] and Reuters [7] seem to be the most used datasets in the literature. This is even more true for the french language, only a few studies have involved french datasets in multi-label text classification [8, 9]. We, therefore, introduce in this paper a new french multi-label dataset.

In this article, our main contributions are :

- The creation of a French corpus of multi-label text classification, MFHAD (for **M**ultilabel **F**rench **H**AL **A**bstracts **D**ataset), containing the abstracts of scientific articles obtained from the open archive HAL (https://hal.archives-ouvertes.fr) that contains more than one million papers,
- The adaptation of available transformer models for multi-label classification,
- The study of threshold selection methods for more efficient exploitation of the results of the transformer models, in particular, the choice of a global threshold , or a threshold specific to each label for the individual optimization of the labels,
- The proposal of two alternative approaches to thresholding for the selection of relevant labels. The first one consists in introducing a parameter at the last layer of the transformer which will be trained for the calculation of the number of labels present in an example, the value of this parameter will be used to select the labels having the strongest activations; The second one consists in adding a final layer to the model, which will have the same number of parameters as the second last layer (equal to the number of labels), in order to obtain more discriminating activation values for the given example, i.e. high activation if labels are present, low activation in the opposite case.

The article is organized as follows: the section 2 presents the approaches that address the multi-label classification problem, the sections 3 and 4 describe the proposed approaches and the section 5 is dedicated to experiments.

2. Related Work

In multi-class classification, each example (instance) X in the dataset is associated with a single label. Multi-label classification is furthermore about being able to associate each entry with multiple $\mathbf{Y}$ labels, rather than just one.

2.1. Multi-label Classification Strategies

Multi-label classification methods can be classified into three categories: problem transformation, adaptation, and ensemble methods.

2.1.1. Problem Transformation (PT)

Problem transformation consists in 'transforming' the dataset to change the problem into a single-label multi-class classification. One such method is to consider all possible unique combinations of labels, *label powerset* [10], and train a multi-class classifier $M : X \to P(Y)$, where $P(Y)$ is the powerset of $\mathbf{Y}$, the set of unique and distinct subsets of labels. In addition to the high number of possible labels that can reach $2^{|Y|}$, the challenge lies in finding enough examples for each combination of labels. For a large $|\mathbf{Y}|$, the training and inference time of the models is high. It is important to note, that by transforming the problem into a multi-class classification, the dependencies that may exist between the different labels are no longer considered [11].

2.1.2. Ensemble Methods

A set of multi-class classifiers can be combined to create a multi-label classifier. For a given instance, each classifier will predict a single label and all outputs of these classifiers are then combined via an ensemble method. One of these methods consists in considering a label as present if a percentage of classifiers having predicted this label is reached, also called the discriminative threshold. The $RAKEL$ algorithm [12] is another variation of this method. Classifiers trained on random subsets of the *labels powersets* are used for the creation of a multi-label classifier, the predictions of these classifiers go through a voting system for the final prediction. The use of multiple classifiers imposes strong constraints in terms of memory use, as well as the need to optimize a number of models that increase linearly with the number of labels in the dataset.

2.1.3. Problem Adaptation (PA)

Problem adaptation methods do not require a transformation of the dataset but an adaptation of classification algorithms, such as ML-kNN [13] which extends the kNN algorithm for multi-label data, or BP-MLL [14] an adaptation of the backpropagation algorithm for neural networks.

The adaptation of deep learning algorithms for multi-labels remains in general an avenue with few contributions. An adaptation of these approaches could contribute to a significant increase in performance. The use of a single model without the need for prior data transformation is an efficient method to try to address the multi-label problem.

2.2. Thresholding Methods

Thresholding methods directly impact the choice of a label for the multi-label problem. The threshold can be adjusted in several ways, either to optimize all the labels (a global threshold), or to optimize each label individually (number of thresholds equal to the number of labels). Let

m be the number of examples in the test dataset (or validation) and n_y the number of labels. The four most commonly used strategies for choosing the threshold(s) are:

- **SCut**: Labels are optimized individually, thresholds are chosen based on the validation set, measured either by maximizing a score or minimizing a cost function and without guaranteeing a global optimum. This method can also be used to obtain a global threshold;

- **RCut** (Rank Cut) : Labels are ordered according to their score, the t first labels are chosen as relevant labels. The t parameter is either predefined or set from the validation dataset [15];

- **PCut** (Proportion Cut): for each label y_i, the instances of the test data set are ordered according to the score obtained for this label. The first k_i instances are chosen for the label y_i where $k_i = P(y_i) \times x \times n_y$ is the number of instances assigned to this label. $P(y_i)$ is the probability an instance belonging to the label y_i (computed from the training set), and x the average number of instances to be assigned for any label previously set. If $x = n$ all instances are taken, for $x = 0$ no instance is considered to be part of the evaluated label [16, 17];

- **MCut** (Maximum Cut): the labels are ordered from the scores obtained for an instance of the dataset, the threshold is equal to the average of the two contiguous labels for which the score difference is the most important [18].

Variations of these methods aiming to overcome the constraints they may impose have been proposed [15]. The classification score-based methods are the best among the thresholding approaches [19]. In this paper, the proposed approaches are similar in nature to the Scut and RCut methods, but with different implementations.

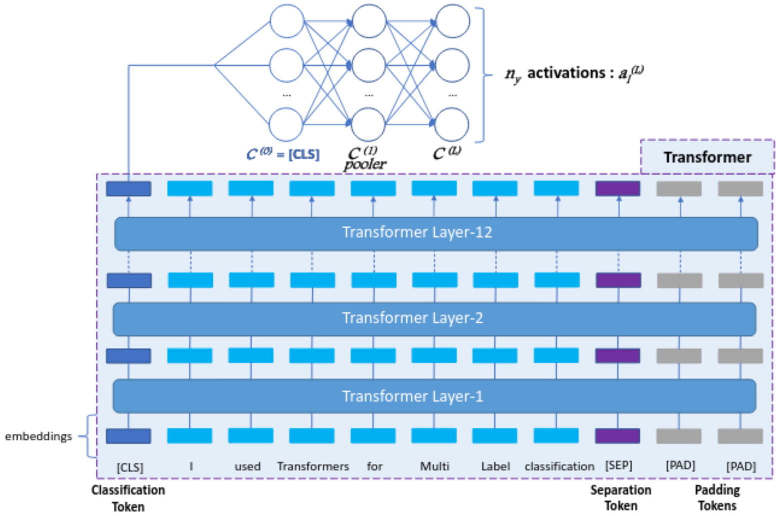


Figure 1: BERT architecture with a dense classification layer, on top of the transformer layers, connected to the [CLS token].

2.3. Deep learning based approaches

[20] uses a neural network for multi-label classification in spectroscopic multi-composition

analysis, a network composed of a classifier to which a parameter is added for learning an activation threshold. This parameter will be optimized according to the threshold computed by applying the model on the training set, the target value of the threshold will thus be different for each iteration of the training phase, rendering the training process much more difficult.

On the other hand, MAGNET [4], a graph network implementing the attention mechanism to capture the dependency structure between labels that uses BERT's embeddings, explicitly tries to tackle the Multi-label text classification problem and manages to have good performances in F1 for the AAPD and Reuters datasets (cf. section 5.2). DocBERT [21], which is now the state of the art reference, adds a linear network to the head of BERT, but without subsequent processing of the model outputs that can enhance the performance of the transformer (e.g. thresholding techniques to better choose the correct present labels).

Transformers have been used for the 'extreme' multi-label classification of text where very large corpora of texts with a number of labels that can reach tens of thousands are processed. Such architectures are not well suited for short texts.

Attempts to use neural networks for text classification do not focus on multi-label classification. Those that do deal with this problem generally do not give importance to how the output layer activations are exploited (e.g. use of thresholds).

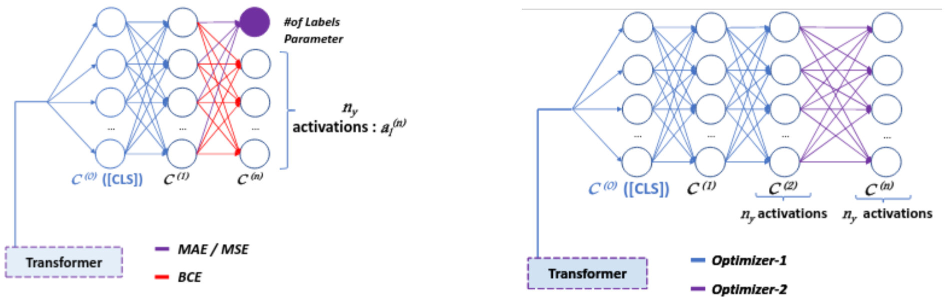
3. BERT Adaptation

BERT introduces bi-directionality in the prediction of masked tokens, where both the left and right semantic contexts of the word to be predicted are considered. In addition to masked language modeling, BERT is trained for next-sentence prediction, a task where the model receives a pair of sentences and tries to predict whether the second sentence follows the first. BERT introduces a special classification token **[CLS]** (also having an identifier and an embedding vector) containing a *hidden state* of the sentence, updated in each layer of the model.

A feed-forward neural network (FFNN) of L dense layers (usually L=2) is added on top of the last transformer layers of the model. This is done to fine-tune the pre-trained transformer for the desired NLP task (text classification in our case). The token **[CLS]** is the input of this FFNN, and n_y outputs corresponding to the labels of the dataset (similar approach to [21]. Figure 1 shows the architecture of the model.

For multi-label classification, the values of the $A^{[L]}$ activations of the $C^{[L]}$ output layer can be used to determine the presence of a label. Each activation can be a value between 0 and 1 representing a probability of the presence of the corresponding label. A threshold is then used for the label selection process, the trivial value of 0.5 is usually used for this purpose. The *Sigmoid* σ function is the activation function that is suited for this case, alongside the Binary Cross Entropy as a loss function.

This approach cannot be considered completely as a problem transformation, it does not require a transformation of the dataset or the creation of multiple binary classifiers. It also does not a complete adaptation of neural networks for multi-label classification, the results must be processed later on for the final classifications.



(a) Added number of labels parameter architecture (**NHA** approach)

(b) Added Thresholding Layer architecture (**TL** approach)

Figure 2: Alternative approaches' architectures

4. Thresholding methods

Thresholding approaches can be applied to the transformer if we consider each activation in the final layer to be a representation of a binary classifier of the label it represents. If the activation is high, the label is considered to be a valid label, and vice-versa.

We propose two alternative methods to thresholding in an attempt for more efficient use of the $A^{[L]}$ activation values of the output layer. These approaches aim to avoid thresholding by learning text-specific features for a better label selection.

4.1. Global Classification Threshold (GCT)

The classification threshold s can be chosen to maximize the classification scores. During the training of the model and after each iteration, the value of s is varied from 0 to 1, with a step defined beforehand (10^{-2}). The micro-F1 score is then computed for each threshold s. We finally obtain the global optimal threshold gct which provides the best performance on the training dataset. This threshold is then used for the validation and test datasets. The set of labels L present for an example x can be expressed as:

$$Y_{x \in X} = \cup_{y \in Y} \{y_i\} : \sigma(a_{y_i}^{[L]})) \geq gct \tag{1}$$

$a_{y_i}^{[L]}$ being the activation corresponding to label y_i among the activations $A^{[L]}$. A variant of the SCut method (with a global threshold) consists in fine-tuning the optimal threshold from the validation dataset and then applying it to the test dataset. Only the first approach has been studied in this article.

4.2. Individual Thresholds (IT)

The use of a shared threshold s for all labels assumes that the features associated with the activations $A^{[L]}$ are the same for each label, which is not the case, the activation intensity of a neuron in the last layer $a_{y_i}^{[L]}$, when the label y is present, varies from label to label, an effect that is accentuated if the dataset is unbalanced.

We, therefore, propose to evaluate the SCut method (with individual thresholds) by setting a threshold it_y for each label y in the dataset, assigned to the $a_{y_i}^{[L]}$ activations of $C^{[L]}$. The values of it_y are the values that maximize the classification scores for a label y. These thresholds are computed as for gct during the training phase of the model, and on the training dataset, by varying each threshold to maximize the F1 score for each label.

$$Y_{x \in X} = \cup_{y \in Y} \{y_i\} : \sigma(a_{y_i}^{[L]}) \geq it_y \tag{2}$$

4.3. N Highest Activations (NHA)

Using a threshold to determine the presence of a label leads in several cases to an under-classification (respectively over-classification) of an instance when the predicted number of labels is lower (respectively higher) than the actual number of labels. The first proposed alternative consists in introducing a neuron at the last layer which will be used only for computing the number of labels y present for an instance. Let $A'^{[L]}$ be the list of the N highest activations, and N being the actual number of labels present for a given instance:

$$Y_{x \in X} = \cup_{y \in Y} \{y_i\} : a_{y_i}^{[L])} \in A'^{[L])} \tag{3}$$

The number of labels present for an instance is the target value, the mean absolute error **MAE** is used as a loss function for this regression problem. We use a single optimizer in the backpropagation, therefore, the error of the regression must be scaled to match the classification error, done by reducing the regression error by a factor of 5. The value of this neuron will be used to recover the N highest activations that will be considered as predicted labels, similarly to the RCut method. Figure 2a shows the model architecture for this approach.

The objective of this architecture is to extract from the [CLS] token, criteria or features about the number of distinct topics present in the text. This token, the final output of the transformer layers, contains an information-rich representation of the input text.

4.4. Threshold Layer (TL)

The second proposed approach is based on the addition of a dense layer, after the output of the classifier, for a total of $L = 3$ layers, having n_y neurons to match the last classification layer of the model. This additional layer aims to make the values of the final activations as close as possible to 1 in the case of the presence of the label, or to 0 in the opposite case. Therefore, the trivial threshold of 0.5 can be used for the classification.

The addition of this layer could result in an increase in the recall of the classifier as many activations of the non-present labels will no longer exceed the classification threshold since they will be as close as possible to 0. A gain in precision can also be expected as the activations of the present labels will be emphasized more than the activations of the irrelevant labels.

For this approach, we use two optimizers, the first one which includes and optimizes all the layers of the transformers as well as the first two layers of the classifier, and another one dedicated to the optimization of the last layer that we added. The error function used is the same for both parts of the classifier, the BCE in this case. The final architecture of the classifier is presented in the figure 2b

Table 1
Dataset used, W is the average number of words per abstract.

	#Train	#Valid	#Test	labels	W
MFHAD	11035	2366	2370	200	140.75
AAPD	53840	1000	1000	54	163,16
Reuter-21578	5827	1943	3019	90	127,76

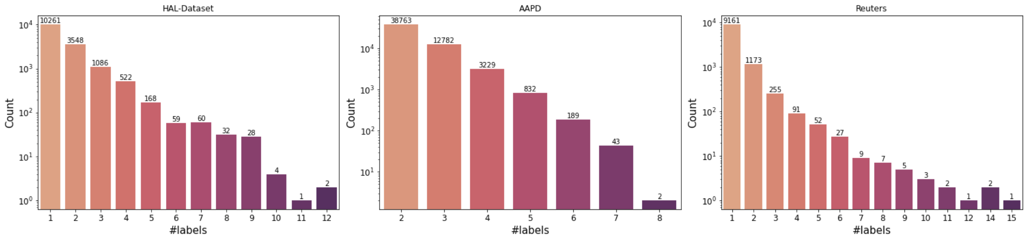


Figure 3: Instance count based on the number of labels for all datasets.

5. Experiments and Results

We present in this section the results of the evaluation of non-deep learning baselines as well as all the previously mentioned methods on three multi-label text datasets, including the French dataset "MFHAD" that we have designed and made available. We compare the different methods, i.e. the thresholding methods as well as the proposed alternatives coupled with BERT transformer and its variants, to baseline approaches, all put in perspective with optimal target results (oracle approaches).

5.1. Evaluated Transformers

For the evaluation of the proposed methods, we use HugginFace's [22] implementation of the *uncased-base* version of **BERT**, with 12 transformer layers and an embedding vector of size 768 dimensions, as well as variants of *BERT* (*uncased-base* versions):

- **RoBERTa** [23] a variant with an optimized training process, where the *Next Sentence Prediction* part of BERT is removed. The dataset used is ten times larger than the one used for BERT. This has led to a performance gain over the original version on the automatic language processing tasks in the GLUE benchmark;

- **DistilBERT** [24], an efficient variant of BERT that uses knowledge distillation to shrink the size of BERT by 40% while keeping 97% of its performance. DistilBERT is based on the fact that after training a very large model, the output distribution can be approximated by a much smaller neural network, using the Kulback-Leiber divergence [25] as the optimization function ;

- **DeBERTa** [26], the most recent variant of BERT where words are represented by two vectors that encode their content and their relative position in the sentence. It is also characterized by the optimization of the decoding of the prediction of hidden tokens, which contributes to a significant gain in efficiency in the pre-training phase of the model, but also in the performances concerning the various natural language processing tasks;

- **CamemBERT** [27] is based on RoBERTa, trained on the French part of OSCAR [28];

- **FlauBERT** [29] trained on various French sub-corpora of different writing styles, from formal writings (e.g. Wikipedia and books), to writings extracted from the internet (e.g. Common Crawl). For this variant, we use the *cased-base* version.

The comparison of different transformers is important to evaluate the performance of our approaches as well as their re-usability and applicability on different transformer architectures.

5.2. Datasets

Multi-label text classification is not present in NLP leaderboards, the GLUE benchmark for example, nor in recent CLEF or Semeval conferences. There are few multi-label corpora frequently used in the literature that can be used to evaluate and compare models, and this is even more true for the French language. To remedy this, we have built our own corpus comprised of abstracts of scientific articles written in French from HAL, a corpus that we make available to the public. In this section, we provide details about the different datasets used [1] for the evaluation of the different models :

- **MFHAD**[2]: Our dataset is comprised of abstracts of french scientific papers collected from 'HAL', an open academic research archive, distributed over three major scientific domains, i.e. computer science, physics, and mathematics. Articles published between 1980 and 2021 that have a french abstract, which represents 15 771 documents distributed over 200 different labels;

- **Reuter-21578** [3] is a collection of articles from the Reuters newswire from the year 1987. It is a dataset that has been often used to evaluate models for multi-label text classification. An article can belong to one or more of the 90 domains of the dataset;

- **AAPD** (or ArXiv Academic Paper Dataset) is, not unlike the "MFHAD", a collection of the "Abstract" of several scientific publications. An article can have one or more classifications among 54 labels. We use the same training, validation, and test distribution as [5].

Table 1) and figure 3 present the different characteristics of these datasets in more detail.

5.3. Evaluation Method

We will compare the two proposed approaches as well as the coupling of thresholding methods to transformers with methods that are more explicit on the label selection criteria, but also with other adaptations of deep learning models to multi-label text classification. We will also compare all these methods to several upper bounds that represent the optimal results that can be achieved.

[1] All datasets can be downloaded here: https://zenodo.org/record/6344750#.Yio1YH_MK-r

[2] The extraction of these abstracts was made using the tool made available by HAL: https://api.archives-ouvertes.fr/docs/search

[3] https://archive.ics.uci.edu/ml/datasets/reuters-21578+text+categorization+collection

Table 2

Scores for test dataset from Reuters and AAPD, best scores are in bold blue, the blue up arrow ↑ indicates when an evaluated approach achieves a gain in performance for the associated model. *Orig.* refers to the original result taken from the corresponding paper.

Modèles	Reuters				AAPD			
	Pr.	R	F1	Acc	Pr.	R	F1	Acc
Baselines								
Decision Tree	78,36	75,05	76,67	74,23	49,67	46,8	48,19	26,6
Bagging	88,12	79,65	83,67	73,34	77,27	48,16	59,34	26,9
Random Forest	**97,18**	57,13	71,96	64,06	**94,2**	25,49	40,12	20
GradientBoost	88,06	80,56	84,14	74,23	79,73	46,8	58,98	27,1
SVM	94,19	79,62	86,29	80,64	80,85	59,98	68,86	36,2
CNN [30] *Orig.*	-	-	86,3	-	-	-	66,4	-
CNN-RNN [2] *Orig.*	-	-	85,5	-	-	-	66,9	-
SGM [5] *Orig.*	-	-	-	-	-	-	71,0	-
MAGNET [4] *Orig.*	-	-	89,9	-	-	-	69,6	-
DocBERT $_{base}$ [21] *Orig.*	-	-	89,0	-	-	-	73,4	-
DocBERT $_{large}$ [21] *Orig.*	-	-	90,7	-	-	-	**75,2**	-
BERT $_{base}$	91.36	90.46	90.91	86.18	76.33	71.95	74.07	41.5
DistilBERT $_{base}$	91.29	90.41	90.84	86.52	80.84	66.41	72.92	41.0
RoBERTa $_{base}$	91.23	87.60	89.38	85.29	74.22	71.50	72.83	40.1
DeBERTa $_{base}$	91.63	90.41	**91.02**	**86.78**	75.99	71.00	73.41	41.0
Thresholding without any specific architecture (GCT and IT)								
BERT $_{base}$+ GCT	90.0	91,.4 ↑	90.86	86.45 ↑	75.56	**72.65** ↑	74.08 ↑	41.7 ↑
BERT $_{base}$+ IT	88.89	**92.30** ↑	90.56	85.72	75.51	72.61 ↑	74.04	41.3
DistilBERT $_{base}$+ GCT	90.83	90.78 ↑	90.80	86.29	75.73	72.08 ↑	73.86 ↑	39.8
DistilBERT $_{base}$+ IT	88.40	91.66 ↑	90.0	85.29	79.06	68,31 ↑	73.3 ↑	41.0
RoBERTa $_{base}$+ GCT	90.73	89.90 ↑	90.31 ↑	86.28 ↑	74.49 ↑	72,49 ↑	73.47 ↑	40.3 ↑
RoBERTa $_{base}$+ IT	89.77	90.49 ↑	90.13 ↑	85.92 ↑	75.35 ↑	71,09	73.15 ↑	40.2 ↑
DeBERTa $_{base}$+ GCT	91.63	90.41	**91.02**	**86.78**	74.46	73.56 ↑	74.01 ↑	39.4
DeBERTa $_{base}$+ IT	90.67	90.92 ↑	90.79	86.61	75.87	72.08 ↑	73.92 ↑	40.7
Transformer Architecture Adaptation (NHA et TL)								
BERT $_{base}$+ NHA	92.33 ↑	85.87	88.98	85.92	73.48	66.38	69.75	40.3
BERT $_{base}$+ TL	90.60	90.41	90.50	86.12	73.48	72.20 ↑	72.83	39.5
DistilBERT $_{base}$+ NHA	92.08 ↑	86.03	88.95	86.15	73.93	66.29	69.90	41.8 ↑
DistilBERT $_{base}$+ TL	90.0	89.66	89.83	85.89	73.63	72.32 ↑	72.97 ↑	40.2
RoBERTa $_{base}$+ NHA	89.43	83.20	86.20	83.40	75.04 ↑	66.58	70.56	42.0 ↑
RoBERTa $_{base}$+ TL	91.17	89.05 ↑	90.09 ↑	86.02 ↑	76.48 ↑	71.58 ↑	73.94 ↑	**42.5** ↑
DeBERTa $_{base}$+ NHA	92.22 ↑	86.41	89.21	86.35	75.32	65.67	70.16	41.9 ↑
DeBERTa $_{base}$+ TL	90.97	90.43 ↑	90.70	86.12	75.97	71.70 ↑	73.78 ↑	41.8 ↑

5.3.1. Baselines

We will first do a comparison with non-neuronal approaches, with interpretable machine learning classification criteria, using TF-IDF extracted features (with no maximum number of features for each dataset), as inputs:

- **Decision** trees using the *"Gini"* criterion, without a maximum tree depth, and 2 as the minimum samples for splitting nodes;

- **Random** Forest with 100 estimators (number of trees) and the same parameters used in the previous method for each tree;
- **Bagging** using decision trees as main estimator (10 estimators);
- **GradientBoosting** with logistic regression as error function, a learning rate of 0.1 and 100 estimators;
- **Support Vector machine** using RBF as the kernel and a regularization parameter of 1.0.

We also include deep learning approaches as a comparison for the evaluation of our approaches:

- **CNN** [30] and **CNN-RNN** [2] which use convolutional neural networks to extract text-specific features;
- **SGM** [5] which applies a sequence generation model with a new decoder structure for multi-label classification;
- **MAGNET** [4] a graph network implementing the attention mechanism to capture dependencies between labels;
- **DocBERT** [21]: a fine-tuning of the *base* and *large* versions of BERT for document classification.

To evaluate the performance gain of our approaches, we compare them to an unchanged version of each transformer model, where no architecture modification is applied, and the classification threshold is the trivial value of 0.5.

5.4. Oracle Approaches

The target theoretical optimums (oracle approaches) are represented by the following two approaches:

- The oracle approach for the N highest activation, where we consider the number of labels present for an instance as a given and then take the N highest activations as the present labels;
- The oracle approach for both SCut thresholding methods, where the calculation of the global threshold and the individual thresholds is done from the test dataset. Considered as the optimal results towards which these methods should come as close as possible.

5.5. Results

In this section, we present the performances of all the approaches mentioned in the previous sections, tested on the different transformers presented in section 3. We propose the following notations for these approaches:

- ***"GCT"***: The Global Optimal Threshold method (see section 4.1);
- ***"IT"***: The Individual Threshold method (see section 4.2);
- ***"NHA"***: The N largest activations approach (see section 4.3);
- ***"TL"***: Denotes the "thresholding layer" approach (see section 4.4). As for the oracle approaches, they will be designated by adding $oracle$ after the corresponding approaches.

Table 3
Scores of the oracle approaches for the test set of the Reuters and AAPD datasets (the best scores are in bold blue).

Models	Reuters				AAPD			
	Pr.	R	F1	Acc	Pr.	R	F1	Acc
BERT$_{base}$+GCT_{oracle}	91.46	90.70	91.08	87.01	80.41	68.85	74.18	43.1
BERT$_{base}$+IT_{oracle}	93.61	91.24	92.41	87.94	**83.0**	70.59	**76.29**	44.9
DistilBERT$_{base}$+GCT_{oracle}	91.61	90.17	90.88	86.52	74.50	73.27	73.88	40.0
DistilBERT$_{base}$+IT_{oracle}	92.48	92.04	92.26	87.45	82.59	70.34	75.97	44.0
RoBERTa$_{base}$+GCT_{oracle}	91.21	89.56	90.38	86.32	74.12	71.83	72.96	39.9
RoBERTa$_{base}$+IT_{oracle}	93.18	90.57	91.86	87.78	81.95	70.14	75.58	44.2
DeBERTa$_{base}$+GCT_{oracle}	92.74	90.73	91.72	87.41	75.21	71.95	73.55	40.4
DeBERTa$_{base}$+IT_{oracle}	**93.74**	91.56	**92.64**	88.27	82.82	70.26	76.02	44.0
BERT$_{base}$+NHA_{oracle}	92.55	**92.55**	92.55	**92.45**	74.06	74.06	74.06	51.8
DistilBERT$_{base}$+NHA_{oracle}	91.99	91.99	91.99	91.95	75.09	**75.09**	75.09	**54.7**
RoBERTa$_{base}$ +NHA_{oracle}	91.69	91.69	91.69	91.45	73.36	73.36	73.36	51.5
DeBERTa$_{base}$ +NHA_{oracle}	92.31	92.31	92.31	92.41	73.48	73.48	73.48	52.7

The maximum length of the sequences used is 512 tokens for all datasets. Tables 2 and 4 show the micro-F1 (with precision and recall) and Accuracy scores for the English and French corpus test datasets respectively. The tables 3 and 5 present the optimum scores for the different approaches. The results of the thresholding methods as well as the proposed architectures were obtained from the *base* versions of the transformers used.

Transformers outperform all other methods, whether it is the classical methods, or other deep learning methods. Our baseline versions of BERT and its variants achieves a better performance than the *base* version of *DocBERT* (the current state of the art in multi-label text classification for the AAPD and Reuters corpora), and for Reuters, better than its *large* version. This is due to the longer training in the fine-tuning process, 150 epochs for the Reuters dataset vs 30 in the case of *DocBert*, and 40 epochs for the AAPD dataset vs 20. For transformers, a longer training process generally yields better performance with a low risk of over-fitting.

We also note that SVMs and decision trees obtain the highest micro-precision scores, at the expense of recall rate, thus lowering the micro F1 score. But precision is not a reliable factor for performance measurement in multi-label classification. SVM can be considered as the best performing non-neural approach due to its high accuracy score, but it falls short of the other methods tested.

The ***GCT*** thresholding techniques seems to be the best among the evaluated approaches, it manages to get a better micro-F1 score than its thresholding counterpart, namely the ***IT*** approach, with a score of **91.02** vs **90.70** for the Reuters dataset, and **74.08** vs **74.04** for AAPD (see table 2). For the Reuters Dataset, the thresholding techniques and the proposed alternatives do not manage to achieve a gain in micro-F1 scores compared to the baseline versions, except for the *RoBERTa* model. This does not apply to the AAPD dataset, gains in performance can be perceived in using thresholding techniques or the ***TL*** approach. Especially for the *RoBERTa* model where the gain is the highest. The *large* version of *DocBERT* remains the best performing

Table 4
Scores for the test dataset from MFHAD.

Models	MFHAD			
	Pr.	R	F1	Acc
Baselines				
Decision Tree	45.29	42.03	43.60	36.84
Bagging	74.99	38.86	51.19	35.49
Random Forest	92.80	32.85	48.52	34.43
GradientBoost	58.10	40.53	47.75	30.42
SVM	84.79	46.08	59.71	43.33
CamemBERT	71.27	58.85	64.47	49.87
FlauBERT	70.10	62.21	65.92	52.44
Thresholding without any specific architecture (GCT and IT)				
CamemBERT+GCT	70.14	59.31 ↑	64.27	49.74
CamemBERT+IT	70.41	59.71 ↑	64.62 ↑	50.00 ↑
FlauBERT+GCT	71.30 ↑	61.40	66.14 ↑	52.74 ↑
FlauBERT+IT	65.04	64.52 ↑	64.78	50.71
Transformer Architecture Adaptation (NHA et TL)				
CamemBERT$+NHA$	60.33	53.60	56.76	48.43
CamemBERT$+TL$	71.45 ↑	58.18	64.14	50.97 ↑
FlauBERT$+NHA$	61.64	54.81	58.02	49.16
FlauBERT$+TL$	70.37 ↑	60.95	65.32	51.05

Table 5
Scores of the oracle approaches for MFHAD.

Models	MFHAD			
	Pr.	R	F1	Acc
CamemBERT+GCT_{oracle}	71.37	59.68	65.01	51.09
CamemBERT+IT_{oracle}	**81.05**	56.70	66.73	52.24
FlauBERT+GCT_{oracle}	75.44	59.12	66.29	52.87
FlauBERT+IT_{oracle}	80.54	59.45	**68.40**	54.55
CamemBERT+NHA_{oracle}	66.85	66.85	66.85	62.83
FlauBERT+NHA_{oracle}	67.49	**67.49**	67.49	**63.37**

model for AAPD. The complex nature of the scientific vocabulary of this corpus underlines the possible benefits of increasing the size of the transforming models (adding several layers and increasing the embedding dimension).

The difference in performance gains between the datasets can be due to their complexity level. The proportion of shared tokens/words between the dictionary of the transformers and the Reuters corpus is much higher than AAPD due to its scientifical nature, which makes the training process much more difficult on the latter, therefore rendering the thresholding and alternative approaches more useful to gain performance on the multi-label classification task. Those techniques won't be as useful if the learning process yields a good overall semantic understanding of the dataset, in that case, a threshold of 0.5 is generally more than enough.

The *RoBERTa* model is the lowest performing variant of *BERT*. But the performance gain on *RoBERTa* is more perceivable for all the evaluated datasets. This can be explained by the fact that for this variant of *BERT*, the Next-Sentence Prediction objective is removed in the pre-training process. The goal of NSP is to learn long dependencies that can exist across sentences, whereas Masked Language Modeling is more focused on understanding relationships on the word level. For multi-label text classification, distinguishing the difference between domains across multiple sentences is vital, and NSP is certainly a good contributing factor for this purpose.

The same conclusions can be drawn from the performance data on the MFHAD dataset, where the baseline version of *FlauBERT* outperforms its *CamemBERT* counterpart (**65.92** vs **64.47** in the F1-score). This comes as no surprise knowing that the latter is a variant based on *RoBERTa* that also lacks the NSP objective, but also the fact that *FlauBERT* is pre-trained, in addition to text crawled from the internet (Common-Crawl), on wikipedia article and books, thus getting more vocabulary coverage of MFHAD than *CamemBERT* that is pre-trained on a corpus derived from Common-Crawl. The evaluated approaches fail in obtaining a gain in performance for *FlauBERT* that achieved good performance with its baseline version but allow *CamemBERT* to get closer to *FlauBERT*'s performance.

The ***TL*** architecture performs better than the *base* version of the *DocBERT* and *MAGNET* models, and matches the *large* version of the former with a score of **90.70** for the *DeBERTa* model on the Reuters dataset. The ***NHA*** approach does not succeed in reaching its theoretical optimal, even though it manages to obtain a score exceeding the *base* version of *DocBERT* for the same dataset (see Transformer Architecture Adaptation part of table 2). These two architectures require less training time than the two thresholding methods ($1.5\times$ faster), but have a higher inference cost, especially for ***TL*** where it's almost twice that of the other approaches, due to the presence of the extra layer that this architecture requires.

As shown in tables 2 and 4, an increase in the recall is observed for almost every model and every dataset for the two thresholding methods. This is due to the fact that modifying the classification thresholds (especially by lowering them) can lead to more labels being predicted as present, but can also lead to a decrease in precision. The same can be said for the ***TL*** alternative method where adding an additional layer contributed in some cases to an increase in the final activations' values, thus using a threshold of 0.5 might include some labels that couldn't otherwise be predicted. The opposite effect is observed for the ***NHA*** method, where the model predicts the most frequent number of labels present in the unbalanced dataset which is always the smallest. This leads to an increase in precision (fewer labels are being predicted thus fewer false positives) but this comes at the cost of the recall.

The ***IT***$_{oracle}$ approach is the best performing among the evaluated oracle approaches. Computing an individual threshold for each label from the test set leads to a global optimal score for all labels, surpassing the optimum for the ***GCT*** method. But this is not the case for its experimental equivalent. ***GCT*** remains the method which comes closest to its theoretical optimum (cf. tables 3 and 2). This can be explained by the fact that for the ***IT*** method several thresholds must be computed for each label, which does not guarantee a global optimal on all the labels for the test set. This effect is amplified if the number of labels is large.

For the theoretical optimal of the method ***NHA***, providing the actual number of relevant labels allows new labels to be considered as present, which increases true positives and decreases false negatives. But this could lead to an increase in false positives as these new labels are in some cases non-valid predictions. The instances for which this approach reduces the number of what would have otherwise been predicted by the model (using 0.5 as a threshold) are rare. Thus the micro F1 score in some cases may be lower compared to the theoretical optimal of the thresholding approaches. On the other hand, a considerable increase of the accuracy is observed, it is in fact the oracle approach that achieves the highest accuracy scores as shown in tables 3 and 5. The actual implementation of this approach did not achieve the same performance as its theoretical optimum. The hidden state of the sentence contained in the [CLS] token may not be sufficient to estimate the number of existing labels, a task that is made harder by the unbalanced nature of the datasets (regarding the number of labels per instance). Exploiting the attention scores obtained in the different layers of the model could be a more efficient method to accomplish this task.

DeBERTa is the best performing variant of *BERT* among all the evaluated models, the changes made by this variant regarding the addition of the relative position of the word to the sentence as an input, seem to improve the performance of *BERT*. This performance improvement comes at the expense of the training speed of the model (**18%** slower training speed than the other variants on average, but with similar inference speeds). As for *DistilBERT*, it achieves results at the same level as its original version, despite its reduced size. Knowledge distillation seems to be an efficient method to overcome the major drawback of transformers and neural networks: the necessity of a long training time.

6. Conclusion and Future Work

As far as the application domain is concerned, multi-label classification is a relevant task. However, multi-label text classification is not a common task and, regrettably, it is not included in the most prominent benchmarks such as GLUE.

Transformer-based language models outperform other deep neural architectures and provide a strong adaptable foundation for a multitude of NLP tasks and text classification, which is the direction we followed in this study.

First, we have tested and shown in this paper that thresholding approaches can achieve a performance gain if the learning process does not yield good results for datasets of a complex nature. Computing a global threshold achieves higher results than computing an individual threshold for each class. The optimization done on each class does not guarantee a general optimal for all labels. We then proposed modifications to the transformer architecture. This is done by adding an additional layer, with a number of activations equal to the number of classes, at the bottom of the model to avoid optimizing thresholds. An approach that, on average, is as effective as thresholding approaches. The analysis of the optimal to be reached showed that the use of the number of classes for the selection of labels significantly increases the performance. However, the difference between the optimal and the actual results of our experiments shows that our proposal needs to be improved in the case of unbalanced data sets. The language of the text corpora, English for AAPD and Reuters, or French for the MFHAD corpus we have built, does not seem to be a factor that impacts the performance of the proposed approaches. These approaches can be used for any multilabel classification problem. Each *BERT* variant seeks to improve the constraining aspects of the original model. *DistilBERT*, despite its reduced size, obtains results as high as the original version. *DeBERTa* is the best performing variant, together with *CamemBERT* which outperforms *FlauBERT*.

We have shown the potential of approaches for adapting the transformer model for multi-label text classification. All these approaches are not limited to text classification but can be used for any other multi-label classification task. Our future work will be focused on exploring other methods of adapting neural networks for multi-label text classification. The thresholds can be learned parameters of the model during the training phase. Finding a more effective method for computing the number of classes present for the ***NHA*** method, and finding a better way to represent the dependencies between the labels for the ***TL*** approach, are other areas we will explore in the future.

References

[1] M.-L. Zhang, Y.-K. Li, X.-Y. Liu, X. Geng, Binary relevance for multi-label learning: an overview, Frontiers of Computer Science 12 (2018).

[2] G. Chen, D. Ye, Z. Xing, J. Chen, E. Cambria, Ensemble application of convolutional and recurrent neural networks for multi-label text categorization, in: 2017 International Joint Conference on Neural Networks (IJCNN), IEEE, Anchorage, AK, USA, 2017, pp. 2377–2383.

[3] Z. Yang, D. Yang, C. Dyer, X. He, A. Smola, E. Hovy, Hierarchical Attention Networks for Document Classification, in: Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Association for Computational Linguistics, San Diego, California, 2016, pp. 1480–1489.

[4] A. Pal, M. Selvakumar, M. Sankarasubbu, Multi-Label Text Classification using Attention-based Graph Neural Network, in: ICAART, 2020.

[5] P. Yang, X. Sun, W. Li, S. Ma, W. Wu, H. Wang, SGM: Sequence Generation Model for Multi-label Classification, in: Proceedings of the 27th International Conference on Computational Linguistics, Association for Computational Linguistics, Santa Fe, New Mexico, USA, 2018, pp. 3915–3926.

[6] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, I. Polosukhin, Attention is All you Need, in: Advances in Neural Information Processing Systems, volume 30, Curran Associates, Inc., 2017.

[7] D. D. Lewis, Y. Yang, T. G. Rose, F. Li, RCV1: A New Benchmark Collection for Text Categorization Research, Journal of Machine Learning Research 5 (2004) 361–397.

[8] A. Bailly, C. Blanc, T. Guillotin, Classification multi-label de cas cliniques avec CamemBERT (Multi-label classification of clinical cases with CamemBERT), in: Actes de la 28e Conférence sur le Traitement Automatique des Langues Naturelles. Atelier DÉfi Fouille de Textes (DEFT), ATALA, Lille, France, 2021.

[9] A. Imane, B. A. Mohamed, Multi-label Categorization of French Death Certificates using NLP and Machine Learning, in: Proceedings of the 2nd international Conference on Big Data, Cloud and Applications, BDCA'17, Association for Computing Machinery, New York, NY, USA, 2017, pp. 1–4.

[10] G. Tsoumakas, I. Katakis, I. Vlahavas, Mining Multi-label Data, in: O. Maimon, L. Rokach (Eds.), Data Mining and Knowledge Discovery Handbook, Springer US, Boston, MA, 2010, pp. 667–685.

[11] O. Luaces, J. Díez, J. Barranquero, J. J. del Coz, A. Bahamonde, Binary relevance efficacy for multilabel classification, Progress in Artificial Intelligence 1 (2012) 303–313.

[12] G. Tsoumakas, I. Vlahavas, Random k -Labelsets: An Ensemble Method for Multilabel Classification, volume 4701, 2007, pp. 406–417.

[13] M.-L. Zhang, Z.-H. Zhou, ML-KNN: A lazy learning approach to multi-label learning, Pattern Recognit. (2007).

[14] Min-Ling Zhang, Zhi-Hua Zhou, Multilabel Neural Networks with Applications to Functional Genomics and Text Categorization, IEEE Transactions on Knowledge and Data Engineering 18 (2006) 1338–1351.

[15] Y. Yang, A study of thresholding strategies for text categorization, in: Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval, SIGIR '01, Association for Computing Machinery, New York, NY, USA, 2001, pp. 137–145.

[16] Y. Yang, An evaluation of statistical approach to text categorization, Technical Report, 1997.

[17] D. Lewis, C. Info, L. Studies, M. Ringuette, A Comparison of Two Learning Algorithms for Text Categorization, Third Annual Symposium on Document Analysis and Information Retrieval (1996).

[18] C. Largeron, C. Moulin, M. Géry, MCut: A Thresholding Strategy for Multi-label Classification, volume 7619, 2012.

[19] R. Al-Otaibi, P. A. Flach, M. Kull, Multi-label Classification: A Comparative Study on Threshold Selection Methods (2014).

[20] L. Gan, B. Yuen, T. Lu, Multi-label Classification with Optimal Thresholding for Multi-composition Spectroscopic Analysis, Mach. Learn. Knowl. Extr. (2019).

[21] A. Adhikari, A. Ram, R. Tang, J. Lin, DocBERT: BERT for Document Classification, arXiv:1904.08398 [cs] (2019).

[22] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, J. Davison, S. Shleifer, P. von Platen, C. Ma, Y. Jernite, J. Plu, C. Xu, T. L. Scao, S. Gugger, M. Drame, Q. Lhoest, A. M. Rush, HuggingFace's Transformers: State-of-the-art Natural Language Processing, Technical Report arXiv:1910.03771, arXiv, 2020.

[23] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, V. Stoyanov, RoBERTa: A Robustly Optimized BERT Pretraining Approach, arXiv:1907.11692 [cs] (2019).

[24] V. Sanh, L. Debut, J. Chaumond, T. Wolf, DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter, arXiv:1910.01108 [cs] (2020).

[25] S. Kullback, R. A. Leibler, On Information and Sufficiency, The Annals of Mathematical Statistics 22 (1951) 79–86.

[26] P. He, X. Liu, J. Gao, W. Chen, DeBERTa: Decoding-enhanced BERT with Disentangled Attention, arXiv:2006.03654 [cs] (2021). URL: http://arxiv.org/abs/2006.03654.

[27] L. Martin, B. Muller, P. J. O. Suárez, Y. Dupont, L. Romary, E. V. d. l. Clergerie, D. Seddah, B. Sagot, CamemBERT: a Tasty French Language Model, 2020.

[28] P. J. O. Suárez, B. Sagot, L. Romary, Asynchronous Pipeline for Processing Huge Corpora on Medium to Low Resource Infrastructures, Leibniz-Institut für Deutsche Sprache, 2019.

[29] H. Le, L. Vial, J. Frej, V. Segonne, M. Coavoux, B. Lecouteux, A. Allauzen, B. Crabbe, L. Besacier, D. Schwab, FlauBERT: Unsupervised Language Model Pre-training for French, in: LREC, 2020.

[30] Y. Kim, Convolutional Neural Networks for Sentence Classification, in: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Association for Computational Linguistics, Doha, Qatar, 2014, pp. 1746–1751.