Impact de l'apprentissage multi-labels actif appliqué aux transformers

Maxime Arens[1, 2] Charles Teissèdre[2] Lucile Callebert [2] Jose G. Moreno[1]
Mohand Boughanem[1]
(1) Université de Toulouse, IRIT UMR 5505 CNRS, 31400 Toulouse, France
(2) Synapse Développement, 7 Boulevard de la Gare, 31500 Toulouse, France
maxime.arens@irit.fr

RÉSUMÉ

L'Apprentissage Actif (AA) est largement utilisé en apprentissage automatique afin de réduire l'effort d'annotation. Bien que la plupart des travaux d'AA soient antérieurs aux *transformers*, le succès récent de ces architectures a conduit la communauté à revisiter l'AA dans le contexte des modèles de langues pré-entraînés. De plus, le mécanisme de *fine-tuning*, où seules quelques données annotées sont utilisées pour entraîner le modèle sur une nouvelle tâche, est parfaitement en accord avec l'objectif de l'AA. Nous proposons d'étudier l'impact de l'AA dans le contexte des *transformers* pour la tâche de classification multi-labels. Or la plupart des stratégies AA, lorsqu'elles sont appliquées à ces modèles, conduisent à des temps de calcul excessifs, ce qui empêche leurs utilisations au cours d'une interaction homme-machine en temps réel. Afin de pallier ce problème, nous utilisons des stratégies d'AA basées sur l'incertitude. L'article compare six stratégies d'AA basées sur l'incertitude dans le contexte des *transformers* et montre que si deux stratégies améliorent invariablement les performances, les autres ne surpassent pas l'échantillonnage aléatoire. L'étude montre également que les stratégies performantes ont tendance à sélectionner des ensembles d'instances plus diversifiées pour l'annotation.

ABSTRACT

Impact of multi-label active learning applied to transformers

Active Learning (AL) is widely used in machine learning to reduce the human annotation effort. While most AL predates transformers, the recent success of such architectures has led the research community to revisit AL in the context of transformers. Moreover the fine-tuning mechanism, where only few annotated data are used to train the transformer on a downstream task, is fully aligned with the goal of AL. We propose to study the impact of AL in the context of transformers for the multi-label classification task. However, most AL strategies, when applied to transformers, lead to excessive computation times, which prevents their use in real time human-machine interaction. To cope with these efficiency issues, we adopt uncertainty-based AL strategies. The paper compares six uncertainty-based AL strategies on transformers and shows that while two strategies consistently improve performances, the others do not outperform random sampling. The study also shows that these well-performing strategies tend to select batches of more diverse instances for annotation.

MOTS-CLÉS : Apprentissage Actif, classification multi-labels, transformers.

KEYWORDS: Active Learning, multi-label classification, transformers..

1 Introduction

L'acquisition de données annotées est un point central en apprentissage automatique et profond (Fredriksson *et al.*, 2020). En effet, la performance des modèles entraînés est souvent relative à la quantité et la qualité des données annotées à disposition. Le processus d'annotation, particulièrement dans les domaines techniques, est une étape onéreuse qui requiert la participation d'humains voire d'experts (Wu *et al.*, 2021). C'est d'autant plus vrai dans la classification multi-labels (Zhang, 2022) où chaque instance peut avoir plusieurs labels, l'annotation devient alors un processus fastidieux pour l'annotateur, surtout quand l'espace des labels est grand. Le principal objectif de l'Apprentissage Actif (AA) est de réduire le coût d'annotation de données (Wang *et al.*, 2021) en choisissant et en limitant les données à annoter. En effet, au lieu d'annoter de façon aléatoire les données, les stratégies d'AA permettent de sélectionner en priorité les meilleurs ensembles de données à annoter dans le but de maximiser le gain d'information du modèle lors de sa prochaine étape d'entraînement. Les données ainsi sélectionnées sont annotées par un oracle (un humain). Ces deux étapes sont répétées jusqu'à ce qu'un critère d'arrêt soit atteint. Plus la stratégie d'AA est efficace, plus l'interaction avec l'oracle est valorisée.

Bien que les récents développements de nouvelles architectures d'apprentissage profond (Vaswani *et al.*, 2017) aient conduit à quelques travaux étudiant l'utilisation de l'AA sur des *transformers* (Ein-Dor *et al.*, 2020; Lu & MacNamee, 2020; Schröder *et al.*, 2022), la plupart des stratégies d'AA ont été et sont conçues pour des architectures classiques d'apprentissage automatique telles que les machines à vecteurs de support (SVM) ou des réseaux de neurones à convolution (CNN) (Kumar & Gupta, 2020; Schröder & Niekler, 2020; Reyes *et al.*, 2018; Nakano *et al.*, 2020; Gui *et al.*, 2021; Chen *et al.*, 2022). Cela est principalement dû au fait que l'utilisation de l'AA avec des *transformers* entraîne une augmentation du temps de calcul de chaque étape d'entraînement, ce qui rend l'utilisation de ces stratégies peu viables (Schröder *et al.*, 2022). En effet, les stratégies d'AA doivent être efficientes en matière de temps de calculs afin de permettre une interaction humain-machine (Wang *et al.*, 2021). Appliquer l'AA à des *transformers* permet notamment de réduire le coût d'annotation durant le *fine-tuning* (spécialisation d'un modèle sur une nouvelle tâche) en sélectionnant le meilleur ensemble de données à annoter (Ein-Dor *et al.*, 2020; Lu & MacNamee, 2020).

L'objectif de cet article est d'étudier l'impact des stratégies d'AA sur les *transformers*. Nous nous focalisons sur les stratégies basées sur l'incertitude qui ont prouvé leur efficacité sur des architectures de modèles antérieures. Dans le but de généraliser nos résultats, notre étude est réalisée sur quatre jeux de données (Demszky *et al.*, 2020; Chalkidis *et al.*, 2022; Lippi *et al.*, 2018; Chalkidis *et al.*, 2021), utilise une métrique commune dans la classification multi-labels (Tsoumakas *et al.*, 2010) et explore deux modèles : distilBERT et distilRoBERTa (Devlin *et al.*, 2019; Sanh *et al.*, 2019; Liu *et al.*, 2019). La contribution de ce papier est triple :

1. Nous implémentons six stratégies d'AA basées sur l'incertitude sur deux *transformers*.
2. Nos résultats indiquent que *Confidence Minimum No weighting (CMN)* et *Max Margin Uncertainty (MMU)* sont les meilleures stratégies parmi les six étudiées, améliorant les performances des deux modèles sur l'ensemble des jeux de données.
3. Nos résultats montrent que les stratégies performantes ont tendance à sélectionner des ensembles de données plus diverses que les stratégies en sous-performance.

2 Méthodologie

2.1 Contexte

L'enjeu pour entraîner un modèle avec de l'AA consiste en l'élaboration d'une stratégie pour sélectionner une instance plutôt qu'une autre (Settles, 2009). Une fois la stratégie appliquée et l'instance sélectionnée, la plupart des travaux sur l'AA dans la classification multi-labels font ensuite une requête à un oracle afin d'obtenir tous les labels associés à cette instance (Reyes *et al.*, 2018). De récentes études sur l'application de stratégies d'AA à des *transformers*, sur la tâche de classification binaire (Ein-Dor *et al.*, 2020; Lu & MacNamee, 2020) ont montré que l'AA réduisait les biais lors des premières étapes de l'entraînement. Nuançant ce constat, d'autres résultats préliminaires (D'Arcy & Downey, 2022) suggèrent que les stratégies prédatant les *transformers* s'appliquent difficilement à eux, ajoutant de l'instabilité à l'entraînement.

Les stratégies d'AA basées sur les ensembles (Krogh & Vedelsby, 1994; Shi *et al.*, 2011) et les gradients (Ein-Dor *et al.*, 2020; Lu & MacNamee, 2020) s'adaptent mal au grand nombre de paramètres des *transformers* (Schröder *et al.*, 2022). Comme suggéré par (Lu & MacNamee, 2020), pour l'AA dans le contexte des transformers, nous nous concentrons sur l'étude des stratégies basées sur l'incertitude (Lewis & Gale, 1994; Cohn *et al.*, 1996; Li *et al.*, 2004; Esuli & Sebastiani, 2009; Li & Guo, 2013; Reyes *et al.*, 2018).

L'étude la plus complète sur l'utilisation des stratégies d'AA basées sur l'incertitude dans le contexte des *transformers* (Schröder *et al.*, 2022) montre que les stratégies efficaces sur les architectures antérieures (SVM ou CNN) ne sont pas toujours intéressantes pour les *transformers*. Cette étude ne porte pas sur la tâche de classification multi-labels bien qu'elle soit une tâche où l'apport de l'AA est capital (Liu *et al.*, 2021). Comme montré dans (Wertz *et al.*, 2022), certaines stratégies d'AA multi-labels (Reyes *et al.*, 2018; Gissin & Shalev-Shwartz, 2019; Yuan *et al.*, 2020), dans le contexte des *transformers*, ne semblent pas apporter d'améliorations et performent même moins bien qu'un échantillonage aléatoire des données d'entraînement. A notre connaissance, il n'y a pas encore d'explications pour ces résultats et notre article tente de combler ce manquement autour de six stratégies différentes basées sur l'incertitude.

2.2 Stratégies d'Apprentissage Actif multi-labels

La tâche de classification multi-labels consiste à assigner les labels appropriés à des instances textuelles. Contrairement à la classification multi-classes, plusieurs labels peuvent être associés à une même instance. Pour chacune de nos expériences, notre espace de label est prédéfini et n'évolue pas au fur et à mesure. L'objectif de l'AA est de sélectionner les meilleures instances possibles à annoter pour l'entraînement. Cette sélection peut se faire selon différentes *stratégies*. Nos stratégies sont basées sur l'estimation de l'*incertitude* du modèle sur chaque instance, c'est-à-dire la confiance du modèle dans la prédiction des labels associés à cette instance. Ces stratégies reposent sur l'hypothèse qu'en s'entraînant sur des exemples difficiles (où le modèle hésite), le modèle va gagner en performance.

À chaque itération d'entraînement, les stratégies sélectionnent les instances à annoter parmi toutes les données non-annotées du jeu de données (Lewis & Gale, 1994). Pour chaque instance non-annotée, nous calculons un score qui indique l'incertitude du modèle sur ses prédictions associées. Nous appliquons six stratégies d'AA multi-labels aux *transformers* :

Soient nos instances textuelles $x_1, ..., x_n$ et notre espace des labels $l = l^1, ..., l^q$. Pour une instance donnée x_i nous représentons sa distribution probabiliste de labels par $y_i = [y_i^1, ..., y_i^q], y_i^j \in [0, 1]$ où plus y_i^j est proche de 1, plus le modèle est confiant que x_i est labelisé comme j et où plus y_i^j est proche de 0, plus le modèle est confiant que x_i n'est pas labelisé comme j.

Comme suggéré dans (Schröder *et al.*, 2022), nous concentrons notre étude sur des stratégies d'AA basées sur l'incertitude et plus précisement sur six stratégies d'AA multi-labels :

Max Loss (ML) sélectionne les instances pour lesquelles la fonction de perte est la plus élevée (Li *et al.*, 2004) :

$$\underset{x_i}{\operatorname{argmax}} \Big[\sum_{j=1}^{q} \max\{1 - m_j * f_j(x_i), 0\} \Big] \tag{1}$$

où $m_j = 1$ si $j = u$, $m_j = -1$ sinon, u correspondant au label l^u associé avec la plus grande probabilité à une instance donnée et où $f_j(x_i)$ est défini par :

$$f_j(x_i) = 2 * y_i^j - 1 \tag{2}$$

Mean Max Loss (MML) sélectionne les instances pour lesquelles la fonction de perte moyenne est la plus élevée (Li *et al.*, 2004) :

$$\underset{x_i}{\operatorname{argmax}} \frac{1}{q} \Big[\sum_{k=1}^{q} \sum_{j=1}^{q} max\{1 - o_{kj} * f_j(x_i), 0\} \Big] \tag{3}$$

où $o_{kj} = 1$ si $j = k$, $o_{kj} = -1$ sinon et $f_j(x_i)$ est défini dans (2).

Minimum Confidence No weighting (CMN) sélectionne les instances pour lesquelles la confiance du modèle est la plus basse (Esuli & Sebastiani, 2009) :

$$\underset{x_i}{\operatorname{argmin}} \Big(\min_{j=1}^{q} f_j(x_i) \Big) \tag{4}$$

avec $f_j(x_i)$ défini dans (2).

Max Margin Uncertainty sampling (MMU) sélectionne les instances qui maximisent la marge de séparation entre les groupes prédits de labels positifs et négatifs (Li & Guo, 2013) :

$$\underset{x_i}{\operatorname{argmax}} \frac{1}{\min pos(x_i) - \max neg(x_i)} \tag{5}$$

où $pos(x_i) = [pos_1(x_i), ..., pos_q(x_i)]$ et $neg(x_i) = [neg_1(x_i), ..., neg_q(x_i)]$, avec :

$$pos_j(x_i) = \begin{cases} f_j(x_i) & \text{si } f_j(x_i) > 0 \\ +\infty & \text{sinon} \end{cases} \quad \text{et} \tag{6}$$

$$neg_j(x_i) = \begin{cases} f_j(x_i) & \text{si } f_j(x_i) < 0 \\ -\infty & \text{sinon} \end{cases} \tag{7}$$

avec $f_j(x_i)$ défini dans (2).

Label Cardinality Inconsistency (LCI) sélectionne les instances qui maximisent la distance entre le nombre de labels positifs prédits et la cardinalité des labels du jeu de données annotées (Li & Guo, 2013) :

$$\underset{x_i}{\text{argmax}} \sqrt{\left(\left(\sum_{j=1}^{q} y_i^j\right) - L\right)^2} \tag{8}$$

avec L le nombre moyen de labels (cardinalité) sur les instances déjà annotées.

Category Vector Inconsistency and Ranking of Scores (CVIRS) sélectionne les intances suivant deux mesures. La première est basée sur une agrégation de rang des marges de différence des prédictions du classifieur. La seconde est basée sur l'incohérence des ensembles de labels prédits par rapport à l'espace des labels de l'ensemble des instances annotées. Cette stratégie, plus complexe que les autres, est détaillée dans (Reyes *et al.*, 2018).

Ces stratégies sont dites *myopic*, dans le sens où elles évaluent l'incertitude instance par instance. Bien qu'il existe des stratégies prenant en compte la composition des lots (Gui *et al.*, 2021), les travaux appliquant des stratégies d'AA aux *transformers* utilisent principalement des stratégies *myopic*. Comme dans (Reyes *et al.*, 2018), nous adaptons ces stratégies à l'apprentissage par lots de données simplement : au lieu de sélectionner uniquement l'instance pour laquelle le modèle est le plus incertain, nous sélectionnons les instances les plus incertaines pour remplir notre lot d'entraînement.

3 Expérimentation et résultats

Dans nos expérimentations, l'AA multi-labels suit le processus suivant : tout d'abord, nos modèles sont initialisés en les entraînant avec 25 instances sélectionnées aléatoirement. Ensuite, pour chaque stratégie, nous effectuons 50 itérations d'AA où 25 instances sont sélectionnées à chaque itération pour être ensuite annotées par un oracle, ce qui permet de collecter un total de 1250 instances annotées par stratégie. Après chaque itération, nous entraînons à nouveau le modèle avec le nouveau lot d'instances annotées.

Nous comparons les six stratégies d'AA multi-labels (Li *et al.*, 2004; Esuli & Sebastiani, 2009; Li & Guo, 2013; Reyes *et al.*, 2018) sur deux *transformers*, distilBERT et distilRoBERTa (Devlin *et al.*, 2019; Sanh *et al.*, 2019; Liu *et al.*, 2019). Nos expériences sont menées sur quatre jeux de données multi-labels et reportons nos résultats via une métrique largement utilisée afin de généraliser nos résultats. Ces expérimentations ont été réalisées grâce à la librairie *small-text* [1] (Schröder *et al.*, 2023). Les résultats sont une moyenne calculée sur cinq exécutions de chaque expérience, l'écart-type est indiqué sur les graphes.

1. https://small-text.readthedocs.io/en/latest/

TABLEAU 1 – Caractéristiques des jeux de données

Nom	Labels	Entraînement	Test	Cardinalité	Densité
Jigsaw_toxic	6	159,571	63,978	0.222	0.037
Go_emotions	27	43,410	5,427	0.848	0.031
EUR_Lex	100	55,000	5,000	4.526	0.036
UNFAIR-ToS	8	5,532	1,607	0.124	0.016

3.1 Jeux de données

Pour les modèles antérieurs aux *transformers*, l'ensemble de jeux de données référence dans la classification multi-labels était Mulan (Tsoumakas *et al.*, 2011). Les stratégies d'AA que nous étudions ont souvent été évaluées sur cet ensemble. Cependant, comme les instances sont données sous forme de caractéristiques numériques et que le texte original brut n'est pas fourni, ce jeu de données n'est pas approprié pour l'apprentissage des *transformers*.

Le Tableau 1 montre les caractéristiques des quatre jeux de données utilisés. La cardinalité est le nombre moyen de labels par instance. La densité est la cardinalité divisée par le nombre total d'instances.

Nous avons choisi ces jeux de données afin de réaliser nos expériences sur des textes présentant une variation du niveau de langue employé (de l'insulte au texte légal, en passant par du commentaire de réseau social) ainsi qu'une variation du nombre de labels associés à chaque instance (de 6 à 100).

Jigsaw toxic comment classification (Jigsaw_Toxic) est le jeu de données associé à une compétition Kaggle [2] dont le but était de détecter et de classifier six différents types de toxicité que l'on peut trouver en ligne. Les instances sont tirées de commentaires sur des pages wikipédia. Les différents labels, se référant à différents types de toxicité, sont souvent corrélés (par exemple, toutes les instances de "toxicité sévère" sont aussi labellisées "toxicité"). Près de 90% des instances du jeu de données ne présentent aucune forme de toxicité et ne sont donc associées à aucun label.

Go_Emotions est un jeu de données composé de commentaires Reddit [3] labellisés sur 27 catégories d'émotions comme "colère" ou "curieux" (Demszky *et al.*, 2020). Un peu plus de 30% des instances sont "neutres", c'est-à-dire non associées avec un label.

EUR_Lex57K (EUR_Lex) est un jeu de données composé de textes légaux (Chalkidis *et al.*, 2021) issus du site du même nom [4].

UNFAIR - Terms of Services (UNFAIR-ToS) est un jeu de données composé de textes annotés avec huit types de termes contractuels injustes (Lippi *et al.*, 2018), c'est-à-dire, des termes qui violent potentiellement les droits des consommateurs selon la loi de consommation européene.

Dans nos travaux, nous avons utilisé les versions de EUR_Lex et UNFAIR-ToS fournit dans *Legal General Language Understanding Evaluation* (LexGLUE) (Chalkidis *et al.*, 2022).

Dans nos expérimentations, 10% du jeu d'entraînement est utilisé en tant que jeu de validation et les performances reportées sont obtenues à partir du jeu de test.

2. https://www.kaggle.com/c/jigsaw-toxic-comment-classification-challenge
3. https://www.reddit.com/
4. https://eur-lex.europa.eu

3.2 Configuration expérimentale

Oracle : La simulation d'un oracle humain, annotant les instances non-annotées sélectionnées par les différentes stratégies, est réalisée par l'utilisation des jeux de données multi-labels annotés. A chaque itération d'entraînement, la stratégie d'AA sélectionne les meilleurs instances à partir du jeu d'entraînement sans avoir accès aux labels correspondants. Ces instances et leurs labels associés composent le prochain lot d'entraînement.

Modèles : Comme dans (Schröder *et al.*, 2022), les deux *transformers* utilisés dans cette études sont basés sur BERT (Devlin *et al.*, 2019) et RoBERTa(Liu *et al.*, 2019). Étant donné que (Tsvigun *et al.*, 2022) montrent que dans les processus d'AA les versions distillées de ces modèles obtiennent des performances similaires à celles obtenues par les modèles originaux, tout en étant moins gourmands en ressources informatiques, nous utilisons également les versions distillées de ces modèles (Sanh *et al.*, 2019). Au-dessus des deux modèles, nous ajoutons une couche de neurones dense ainsi qu'une couche sigmoïde afin de réaliser la classification multi-labels.

Détails d'implémentation : DistilBERT est composé de 6 couches, des unités cachées de taille 768 et de 66G paramètres. DistilRoBERTa est structuré d'une façon comparable, à l'exception de son nombre de paramètres, qui est de 82G. La taille maximum des tokens d'entrées pour les deux modèles est fixée à 128, le nombre d'époques à 15 et la taille des lots d'entraînement à 25. Pour optimiser les paramètres des modèles, nous avons choisi AdamW avec un taux d'apprentissage de 2e-5. Les expériences ont été réalisées sur une Nvidia GTX1080Ti (32 GB). Les mêmes hyper-paramètres ont été utilisés par les deux modèles sur les quatre jeux de données. Nous suivons (Howard & Ruder, 2018) pour notre processus de *fine-tuning*.

Valeurs de références : **Random (RD)** est une référence commune dans l'AA, où les instances à annoter sont échantillonnées de manière aléatoire à partir du jeu de données non-annotées. Afin d'évaluer les performances finales atteintes par les différentes stratégies, nous utilisons comme référence le modèle entraîné sur l'ensemble du jeu de données en supervision totale. Nous nommons cette valeur de référence **Full-Supervision (FULL)**.

Métrique : Pour mesurer la performance, nous utilisons une métrique communément utilisée dans la classification multi-labels (Tsoumakas *et al.*, 2010). Nous utilisons les notations de la section 2.2, en ajoutant : pour un label donné l^j nous notons les vrais positifs (vp^j), les faux positifs (fp^j), les faux négatifs (fn^j) et définissons la F1-mesure comme :

$$F1(vp^j, fp^j, fn^j) = \frac{vp^j}{vp^j + \frac{1}{2}(fp^j + fn^j)} \tag{9}$$

Nous utilisons une F1-mesure avec une micro-moyenne, c'est-à-dire que nous faisons la somme de tous les vrais positifs, faux positifs et faux négatifs pour tous les labels puis nous calculons la F1-mesure (plus la valeur est haute plus la performance est bonne) :

$$M_{iF1} = F_1(\sum_{j=1}^{q} vp^j, \sum_{j=1}^{q} fp^j, \sum_{j=1}^{q} fn^j) \tag{10}$$

3.3 Analyse des données sélectionnées

Pour mieux comprendre comment certains biais de sélection peuvent avoir un impact sur les performances d'une stratégie d'AA, nous avons examiné plusieurs caractéristiques des instances : la taille et la cardinalité, ainsi que certaines caractéristiques des lots : la présence d'aberrations et la similarité des instances. Lorsque l'on adapte des stratégies d'AA *myopic* à de l'apprentissage par lot, il y a un risque d'avoir au sein d'un même lot des instances vecteurs d'informations redondantes (Gui *et al.*, 2021). Cette redondance d'information au sein des lots étant proche conceptuellement de la similarité des instances au sein d'un lot, nous nous concentrons sur cette caractéristique.

Afin de calculer la similarité entre les instances sélectionnées, nous calculons Sim_batch, selon la méthode détaillée dans l'Algorithme 1 : pour commencer nous calculons les plongements lexicaux de chaque instance dans le lot grâce à SentenceTransformer (Reimers & Gurevych, 2019) [5]. Ensuite, nous faisons la moyenne pour chaque lot de la similarité cosinus par paire entre les plongements lexicaux présents dans le lot. Enfin, nous calculons Sim_batch, la moyenne du score de similarité sur tous les lots.

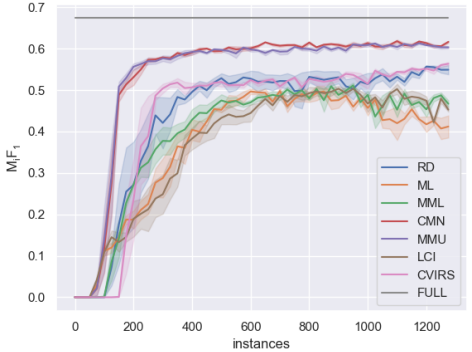
Algorithme 1 Calcul du score Sim_batch

Soit : jeu_annote la liste des lots de données annotées accumulées au fur et à mesure de l'exécution des stratégies d'apprentissage actif
$moyenne\_lot \leftarrow [\,]$
pour chaque lot dans jeu_annote **:**
$pl \leftarrow [\,]$
pour chaque $instance$ dans lot **:**
$pl.ajouter(SentenceTransformer(instance))$
fin pour chaque
$moyenne\_lot.ajouter(\frac{\sum_{x \in paires} similarite\_cosinus(x)}{\frac{|pl|*(|pl|-1)}{2}})$, avec $paires$ la liste des paires uniques d'éléments $\in pl$
fin pour chaque
$Sim\_batch = \frac{\sum_{x \in moyenne\_lot} x}{|moyenne\_lot|}$
retourner Sim_batch

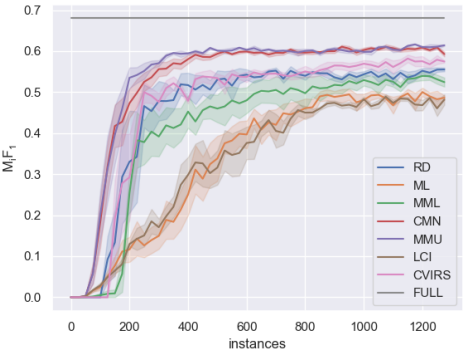
3.4 Résultats

La Figure 1 montre les courbes d'apprentissage en M_{iF1} de nos deux *transformers* selon chaque stratégie d'AA sur les différents jeux de données. Ces résultats indiquent que les stratégies d'AA peuvent améliorer de façon significative les performances atteintes, s'approchant d'une supervision complète à un coût moindre d'annotations. En effet, pour atteindre les performances de l'échantillonnage aléatoire après sélection de 1250 instances, MMU nécessite seulement 175 instances pour distilBERT (225 pour distilRoBERTa) et CMN 175 instances pour distilBERT (275 pour disilRoBERTa). L'écart-type des performances au cours des différentes exécutions est faible pour chaque stratégie d'AA sur les différents jeux de données. En effet, la "graine aléatoire" ne détermine que la composition du lot de

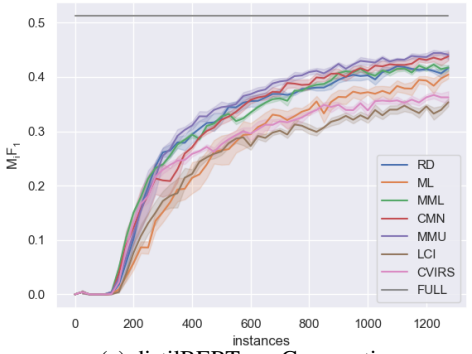
5. nous avons utilisé 'sentence-transformers/nli-distilroberta-base-v2' sur distilRoBERTa et 'sentence-transformers/nli-distilbert-base' sur distilBert



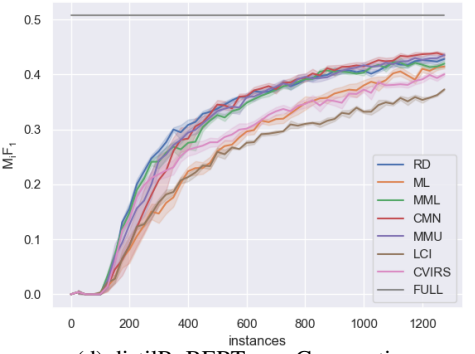
(a) distilBERT sur Jigsaw_toxic



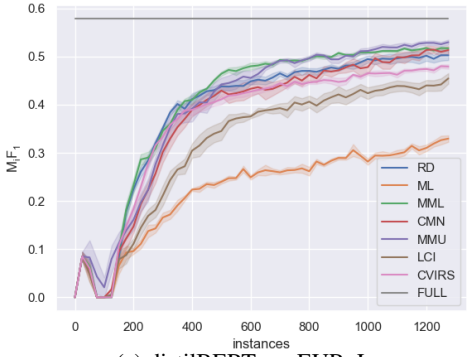
(b) distilRoBERTa sur Jigsaw_toxic



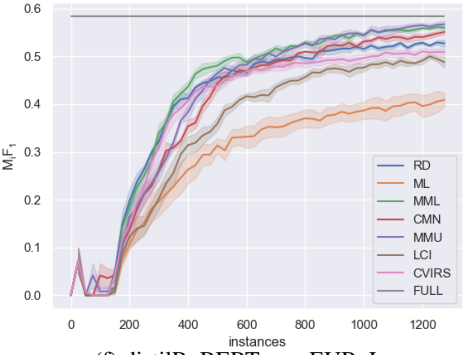
(c) distilBERT sur Go_emotions



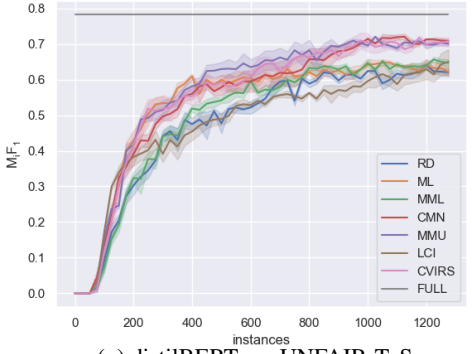
(d) distilRoBERTa sur Go_emotions



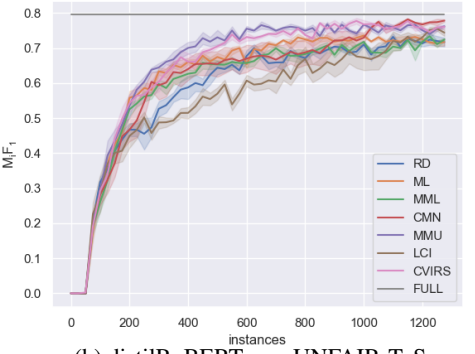
(e) distilBERT sur EUR_Lex



(f) distilRoBERTa sur EUR_Lex



(g) distilBERT sur UNFAIR-ToS



(h) distilRoBERTa sur UNFAIR-ToS

FIGURE 1 – Performances M_{iF1} suivant les différentes stratégies d'AA pour chaque *transformers*

TABLEAU 2 – Pourcentage des jeux de données annotés, associés aux pourcentages de performances atteintes par les stratégies par rapport à une supervision complète.

Jeux de données	Modèles	% jeu de données	% performances M_{iF1}		
			RD	CMN	MMU
Jigsaw	distilBert	0.78	81.45	91.54	89.47
	distilRoBERTa	0.78	81.62	87.21	90.29
goemotions	distilBert	2.88	81.25	85.55	86.13
	distilRoBERTa	2.88	82.84	85.80	85.80
eurlex	distilBert	2.27	86.28	88.00	90.9
	distilRoBERTa	2.27	91.52	95.16	98.1
unfairtos	distilBert	22.60	79.28	90.54	89.51
	distilRoBERTa	22.60	90.19	97.86	95.85

TABLEAU 3 – "M_{iF1}/Sim_batch", obtenu suivant chaque stratégies d'AA. Les résultats en rouge indiquent un score M_{iF1} inférieur à l'échantillonnage aléatoire, les résultats en bleu indiquent un score M_{iF1} supérieur à l'échantillonnage aléatoire.

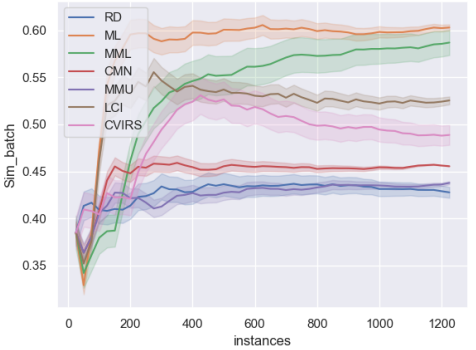
Jeux de données	Modèles	RD	ML	MML	CMN	MMU	LCI	CVIRS	FULL
Jigsaw_toxic	distilBERT	0.549/0.426	0.412/0.602	0.467/0.588	0.617/0.455	0.603/0.439	0.453/0.526	0.564/0.487	0.674/-
	distilRoBERTa	0.555/0.430	0.486/0.590	0.524/0.592	0.593/0.461	0.614/0.448	0.482 /0.578	0.575/0.498	0.680/-
Go_emotions	distilBERT	0.416/0.336	0.404/0.456	0.417/0.341	0.438/0.383	0.441/0.377	0.353/0.474	0.362/0.426	0.512/-
	distilRoBERTa	0.420/0.339	0.414/0.453	0.428/0.341	0.435/0.369	0.435/0.368	0.373/0.463	0.400/0.419	0.507/-
EUR_Lex	distilBERT	0.503/0.741	0.329/0.788	0.517/0.740	0.513/0.736	0.530/0.738	0.454/0.752	0.479/0.758	0.583/-
	distilRoBERTa	0.529/0.736	0.409/0.781	0.560/0.739	0.550/0.746	0.567/0.742	0.488/0.765	0.509/0.756	0.578/-
UNFAIR-ToS	distilBERT	0.620/0.454	0.622/0.490	0.647/0.491	0.708/0.522	0.700/0.504	*0.650*/0.497	0.708/0.523	0.782/-
	distilRoBERTa	0.717/0.455	0.718/0.512	0.724/0.492	0.778/0.553	0.762/0.535	0.744/0.539	0.760/0.511	0.795/-

données d'initialisation dans toutes les stratégies, à l'exception de RD dans laquelle elle joue un rôle plus important.

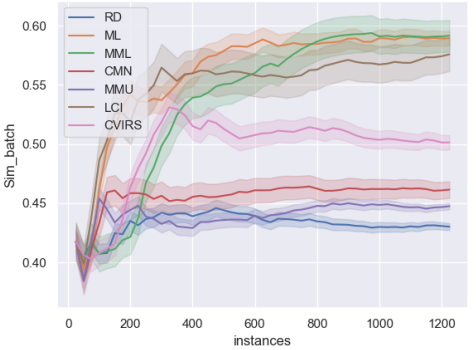
Dans le Tableau 2 nous indiquons le pourcentage du jeu de données que représentent les 1250 instances annotées et le pourcentage des performances de FULL qu'atteignent CMN et MMU, les deux stratégies d'AA les plus performantes, comparativement à l'échantillonnage aléatoire RD. Par exemple pour Jigsaw_Toxic, nous voyons qu'avec moins de 1% du jeu de données (0,78%), nous atteignons plus de 90% des performances de la supervision complète pour CMN (91,54% pour distilBERT et 90,29% pour distilRoBERTa), contre seulement 81% de ces performances avec RD (81,45% pour distilBERT et 81,62% pour distilRoBERTa). De plus, nous voyons sur la Figure 1a que CMN et MMU atteignent ces performances autour de seulement 400 données sélectionnées. Ces résultats mettent en évidence les gains potentiels liés à l'application de stratégies d'AA performantes sur les *transformers*.

Le Tableau 3 compare les performances obtenues après entraînement sur 1250 instances annotées pour chaque combinaison de modèle, jeu de données et stratégies. Afin de mieux illustrer la relation entre les performances du modèle et la similarité des instances sélectionnées, nous avons ajouté le score Sim_batch à côté de la métrique de performance (M_{iF1}/Sim_batch).

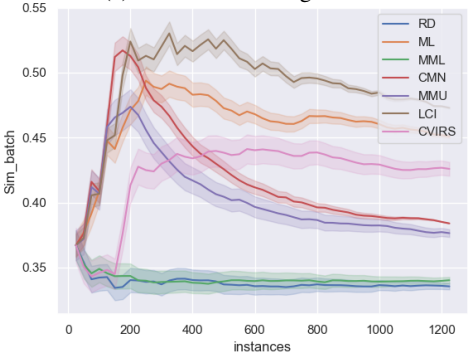
L'un des résultats que nous pouvons tirer du Tableau 3 est qu'à la fin de l'entraînement, la performance (en terme de M_{iF1}) des *transformers* dépend de la stratégie d'AA. Cela suggère que l'ordre des instances sur lequel les *transformers* s'entraînent a bien une importance. Pour les deux *transformers*, CMN and MMU surpassent constamment l'échantillonnage aléatoire et toutes les autres stratégies



(a) distilBERT sur Jigsaw_toxic

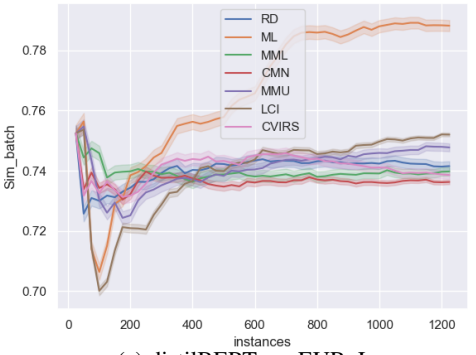


(b) distilRoBERTa sur Jigsaw_toxic

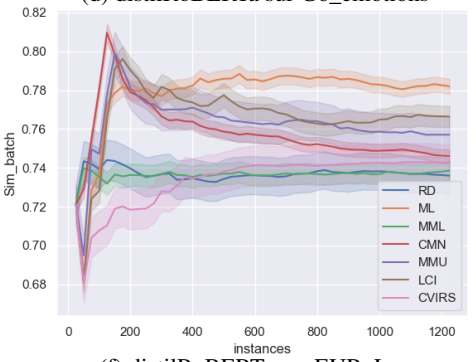


(c) distilBERT sur Go_emotions

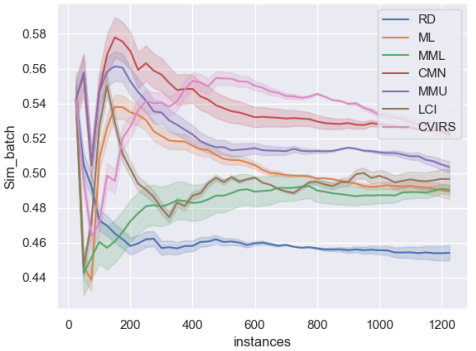
(d) distilRoBERTa sur Go_emotions



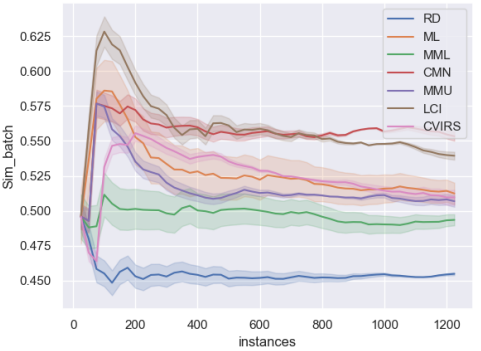
(e) distilBERT sur EUR_Lex



(f) distilRoBERTa sur EUR_Lex



(g) distilBERT sur UNFAIR-ToS



(h) distilRoBERTa sur UNFAIR-ToS

FIGURE 2 – *Sim_batch* suivant les différentes stratégies d'AA pour chaque *transformers*

TABLEAU 4 – Corrélation de Pearson entre M_{iF1} et *Sim_batch*

Jeux de données	Modèles	Pearson	p-value
Jigsaw_toxic	distilBERT	-0.877	0.0096
	distilRoBERTa	-0.843	0.0170
Go_emotions	distilBERT	-0.781	0.0381
	distilRoBERTa	-0.769	0.0431
EUR_Lex	distilBERT	-0.970	0.0003
	distilRoBERTa	-0.930	0.0024
UNFAIR-ToS	distilBERT	0.863	0.0123
	distilRoBERTa	0.776	0.0419

AA, se rapprochant le plus de FULL sur tous les jeux de données.

Une seconde observation que nous pouvons faire, est qu'à chaque fois qu'une stratégie d'AA performe moins bien que l'échantillonnage aléatoire, le score Sim_batch est significativement plus élevé que pour les stratégies performantes ou que l'échantillonnage aléatoire. Dans le Tableau 4, nous remarquons une corrélation linéaire (corrélation de Pearson) entre M_{iF1} et Sim_batch. En effet, il y a une corrélation statistiquement significative (p-value<0.05) entre les performances des stratégies d'AA et le score *Sim_batch*. Plus les instances sélectionnées sont similaires, plus la performance obtenue est basse, à l'exception du jeu de donnée UNFAIR-ToS, pour lequel la corrélation est inversée.

Afin d'expliquer la corrélation obtenue sur les trois premiers jeux de données, en sélectionnant des instances similaires, certaines stratégies entraînent le modèle sur des informations redondantes. La Figure 2, montre l'évolution du score *Sim_batch* au fil de l'apprentissage. Nous remarquons une grande augmentation de score Sim_batch en début d'expérience pour de nombreuses stratégies. Lorsque nous regardons en détail les lots associés à ces fortes augmentations, nous constatons la présence de nombreuses paires de données similaires et même pratiquement identiques. Lorsque ces paires de données très similaires intègrent les lots d'entraînement, cela entraîne un effet domino avec une proportion de paires similaires au sein des lots de plus en plus importante. En poussant plus loin cette analyse, nous avons remarqué ce qui différencie les stratégies performantes des autres : lorsque cet effet domino s'enclenche, CMN et MMU parviennent à en sortir plus rapidement que les autres stratégies. Nous remarquons en effet, que les courbes de Sim_batch pour CMN et MMU ont l'allure d'un pic tandis que les courbes des autres stratégies ont plutôt l'allure d'un pallier.

Pour expliquer la corrélation inversée sur UNFAIR-ToS, un jeu de données composé de phrases annotées de huit différents types de termes contractuels injustes, nous pensons que la similarité des instances joue un rôle clé pour comparer des contrats proches et identifier les parties injustes. Cela peut aussi expliquer pourquoi UNFAIR-ToS est le seul jeu de données pour lequel toutes les stratégies d'AA performent mieux que l'échantillonnage aléatoire.

4 Conclusion

Cet article évalue l'impact sur les *transformers* de six stratégies d'apprentissage actifs basées sur l'incertitude. Nous avons montré que deux de ces stratégies, CMN et MMU, fournissent un gain de performance constant et substantiel par rapport à l'échantillonnage aléatoire, soulignant l'utilité de l'AA appliquée aux *transformers*. Ces deux stratégies seraient notamment de bonnes références pour des prochains travaux sur l'AA multi-labels dans le contexte des *transformers*. Les quatre autres

stratégies fournissent des résultats équivalents ou inférieurs à l'échantillonnage aléatoire. Nous avons investigué les raisons possibles de ces différences de performances. Nous avons d'abord trouvé que CMN et MMU sélectionnent en moyenne des lots d'instances avec une diversité textuelle plus grande que ceux sélectionnés par les stratégies en sous-performance. En poussant notre analyse, nous avons ensuite mis en évidence qu'en début d'expériences, certaines stratégies d'AA sélectionnent des lots de données avec de plus en plus de paires d'instances similaires. CMN et MMU se différenciant par le fait qu'elles arrivent mieux à re-sélectionner des lots avec de moins en moins de paires d'instances similaires dans la suite des expériences. Sur la base de ces résultats, nous nous concentrerons à l'avenir sur des stratégies qui prennent en compte la composition du lot pendant les processus d'AA afin de limiter la présence d'informations redondantes. Par exemple, en maximisant la diversité des instances au sein des lots sélectionnés, en sélectionnant des instances représentatives de l'ensemble du jeu données ou en empêchant la présence de paires d'instances similaires au sein d'un même lot. De plus, nous tenterons de confirmer notre intuition, que les *transformers* sont particulièrement sensibles à la présence d'information redondante dans les lots d'entraînements comparativement à des architectures de modèles antérieurs.

Références

CHALKIDIS I., FERGADIOTIS M. & ANDROUTSOPOULOS I. (2021). MultiEURLEX - A multi-lingual and multi-label legal document classification dataset for zero-shot cross-lingual transfer. *CoRR*, **abs/2109.00904**.

CHALKIDIS I., JANA A., HARTUNG D., BOMMARITO M., ANDROUTSOPOULOS I., KATZ D. M. & ALETRAS N. (2022). LexGLUE : A benchmark dataset for legal language understanding in English. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*, Dubln, Ireland.

CHEN S., WANG R., LU J. & WANG X. (2022). Stable matching-based two-way selection in multi-label active learning with imbalanced data. *Information Sciences*, **610**, 281–299. DOI : https ://doi.org/10.1016/j.ins.2022.07.182.

COHN D. A., GHAHRAMANI Z. & JORDAN M. I. (1996). Active learning with statistical models. *J. Artif. Int. Res.*, **4**(1), 129–145.

D'ARCY M. & DOWNEY D. (2022). Limitations of active learning with deep transformer language models. URL : https://openreview.net/forum?id=Q8OjAGkxwP5.

DEMSZKY D., MOVSHOVITZ-ATTIAS D., KO J., COWEN A., NEMADE G. & RAVI S. (2020). GoEmotions : A Dataset of Fine-Grained Emotions. In *58th Annual Meeting of the Association for Computational Linguistics (ACL)*.

DEVLIN J., CHANG M.-W., LEE K. & TOUTANOVA K. (2019). BERT : Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics : Human Language Technologies, Volume 1 (Long and Short Papers)*, p. 4171–4186, Minneapolis, Minnesota : Association for Computational Linguistics. DOI : 10.18653/v1/N19-1423.

EIN-DOR L., HALFON A., GERA A., SHNARCH E., DANKIN L., CHOSHEN L., DANILEVSKY M., AHARONOV R., KATZ Y. & SLONIM N. (2020). Active Learning for BERT : An Empirical Study. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, p. 7949–7962, Online : Association for Computational Linguistics. DOI : 10.18653/v1/2020.emnlp-main.638.

ESULI A. & SEBASTIANI F. (2009). Active learning strategies for multi-label text classification. In M. BOUGHANEM, C. BERRUT, J. MOTHE & C. SOULE-DUPUY, Éds., *Advances in Information Retrieval*, p. 102–113, Berlin, Heidelberg : Springer Berlin Heidelberg.

FREDRIKSSON T., MATTOS D. I., BOSCH J. & OLSSON H. H. (2020). Data labeling : An empirical investigation into industrial challenges and mitigation strategies. In M. MORISIO, M. TORCHIANO & A. JEDLITSCHKA, Éds., *Product-Focused Software Process Improvement*, p. 202–216, Cham : Springer International Publishing.

GISSIN D. & SHALEV-SHWARTZ S. (2019). Discriminative active learning. *CoRR*, **abs/1907.06347**.

GUI X., LU X. & YU G. (2021). Cost-effective batch-mode multi-label active learning. *Neurocomputing*, **463**, 355–367. DOI : https ://doi.org/10.1016/j.neucom.2021.08.063.

HOWARD J. & RUDER S. (2018). Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1 : Long Papers)*, p. 328–339, Melbourne, Australia : Association for Computational Linguistics. DOI : 10.18653/v1/P18-1031.

KROGH A. & VEDELSBY J. (1994). Neural network ensembles, cross validation, and active learning. In G. TESAURO, D. TOURETZKY & T. LEEN, Éds., *Advances in Neural Information Processing Systems*, volume 7 : MIT Press.

KUMAR P. & GUPTA A. (2020). Active learning query strategies for classification, regression, and clustering : A survey. *Journal of Computer Science and Technology*, **35**(4), 913–945. DOI : 10.1007/s11390-020-9487-4.

LEWIS D. D. & GALE W. A. (1994). A sequential algorithm for training text classifiers. In B. W. CROFT & C. J. VAN RIJSBERGEN, Éds., *SIGIR '94*, p. 3–12, London : Springer London.

LI X. & GUO Y. (2013). Active learning with multi-label svm classification. In *Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence*, IJCAI '13, p. 1479–1485 : AAAI Press.

LI X., WANG L. & SUNG E. (2004). Multilabel svm active learning for image classification. In *2004 International Conference on Image Processing,2004. ICIP '04.*, volume 4, p. 2207–2210 Vol. 4. DOI : 10.1109/ICIP.2004.1421535.

LIPPI M., PALKA P., CONTISSA G., LAGIOIA F., MICKLITZ H., SARTOR G. & TORRONI P. (2018). CLAUDETTE : an automated detector of potentially unfair clauses in online terms of service. *CoRR*, **abs/1805.01217**.

LIU W., WANG H., SHEN X. & TSANG I. (2021). The emerging trends of multi-label learning. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (01), 1–1. DOI : 10.1109/TPAMI.2021.3119334.

LIU Y., OTT M., GOYAL N., DU J., JOSHI M., CHEN D., LEVY O., LEWIS M., ZETTLEMOYER L. & STOYANOV V. (2019). RoBERTa : A robustly optimized bert pretraining approach. DOI : 10.48550/ARXIV.1907.11692.

LU J. & MACNAMEE B. (2020). Investigating the effectiveness of representations based on pretrained transformer-based language models in active learning for labelling text datasets. *CoRR*, **abs/2004.13138**.

NAKANO F. K., CERRI R. & VENS C. (2020). Active learning for hierarchical multi-label classification. *Data Mining and Knowledge Discovery*, **34**(5), 1496–1530. DOI : 10.1007/s10618-020-00704-w.

REIMERS N. & GUREVYCH I. (2019). Sentence-BERT : Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing* : Association for Computational Linguistics.

REYES O., MORELL C. & VENTURA S. (2018). Effective active learning strategy for multi-label learning. *Neurocomputing*, **273**, 494–508. DOI : https ://doi.org/10.1016/j.neucom.2017.08.001.

SANH V., DEBUT L., CHAUMOND J. & WOLF T. (2019). DistilBERT, a distilled version of bert : smaller, faster, cheaper and lighter. *ArXiv*, **abs/1910.01108**.

SCHRÖDER C., MÜLLER L., NIEKLER A. & POTTHAST M. (2023). Small-text : Active learning for text classification in python. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics : System Demonstrations*, p. 84–95, Dubrovnik, Croatia : Association for Computational Linguistics.

SCHRÖDER C. & NIEKLER A. (2020). A survey of active learning for text classification using deep neural networks. *CoRR*, **abs/2008.07267**.

SCHRÖDER C., NIEKLER A. & POTTHAST M. (2022). Revisiting uncertainty-based query strategies for active learning with transformers. In *Findings of the Association for Computational Linguistics : ACL 2022*, p. 2194–2203, Dublin, Ireland : Association for Computational Linguistics. DOI : 10.18653/v1/2022.findings-acl.172.

SETTLES B. (2009). Active learning literature survey. *Technical Report TR-1648. University of Wisconsin-Madison. Department of Computer Sciences*.

SHI C., KONG X., YU P. S. & WANG B. (2011). Multi-label ensemble learning. In D. GUNOPULOS, T. HOFMANN, D. MALERBA & M. VAZIRGIANNIS, Éds., *Machine Learning and Knowledge Discovery in Databases*, p. 223–239, Berlin, Heidelberg : Springer Berlin Heidelberg.

TSOUMAKAS G., KATAKIS I. & VLAHAVAS I. (2010). *Mining Multi-label Data*, In O. MAIMON & L. ROKACH, Éds., *Data Mining and Knowledge Discovery Handbook*, p. 667–685. Springer US : Boston, MA. DOI : 10.1007/978-0-387-09823-4_34.

TSOUMAKAS G., SPYROMITROS-XIOUFIS E., VILCEK J. & VLAHAVAS I. (2011). Mulan : A java library for multi-label learning. *Journal of Machine Learning Research*, **12**, 2411–2414.

TSVIGUN A., SHELMANOV A., KUZMIN G., SANOCHKIN L., LARIONOV D., GUSEV G., AVETISIAN M. & ZHUKOV L. (2022). Towards computationally feasible deep active learning. In *Findings of the Association for Computational Linguistics : NAACL 2022*, p. 1198–1218, Seattle, United States : Association for Computational Linguistics. DOI : 10.18653/v1/2022.findings-naacl.90.

VASWANI A., SHAZEER N., PARMAR N., USZKOREIT J., JONES L., GOMEZ A. N., KAISER L. U. & POLOSUKHIN I. (2017). Attention is all you need. In I. GUYON, U. V. LUXBURG, S. BENGIO, H. WALLACH, R. FERGUS, S. VISHWANATHAN & R. GARNETT, Éds., *Advances in Neural Information Processing Systems*, volume 30 : Curran Associates, Inc.

WANG Z. J., CHOI D., XU S. & YANG D. (2021). Putting humans in the natural language processing loop : A survey. In *Proceedings of the First Workshop on Bridging Human–Computer Interaction and Natural Language Processing*, p. 47–52, Online : Association for Computational Linguistics.

WERTZ L., MIRYLENKA K., KUHN J. & BOGOJESKA J. (2022). Investigating active learning sampling strategies for extreme multi label text classification. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, p. 4597–4605, Marseille, France : European Language Resources Association.

WU X., XIAO L., SUN Y., ZHANG J., MA T. & HE L. (2021). A survey of human-in-the-loop for machine learning. *ArXiv*, **abs/2108.00941**.

YUAN M., LIN H.-T. & BOYD-GRABER J. (2020). Cold-start active learning through self-supervised language modeling. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, p. 7935–7948, Online : Association for Computational Linguistics. DOI : 10.18653/v1/2020.emnlp-main.637.

ZHANG J. (2022). Knowledge learning with crowdsourcing : A brief review and systematic perspective. *IEEE/CAA Journal of Automatica Sinica*, **9**(5), 749–762. DOI : 10.1109/jas.2022.105434.