Natural Language Processing

Toxic Comment Classification

Nicolas BOIZARD

Contents

Introduction :

Dans le cadre du cours de Natural Language Processing 2022 à l'ISEN Lille, un exercice de classification de commentaires toxiques a été retenu comme évaluation finale. Ce rapport a pour objectif d'expliquer pas à pas la méthode utilisée pour construire un modèle de Machine Learning répondant à cette problématique. Le modèle devra prendre en entrée une chaine de caractères et classifier cette dernière en six classes : Toxic, Severe toxic, Obscene, Thread, Insult, Identity Hate.

Matériel :

Pour la réalisation de ce projet un jeu de donnée de 159571 commentaires classifié en six classes (cf. introduction) nous à été proposé. Ce jeu de commentaires est tiré d'un concours mis en place par la plateforme Kaggle.

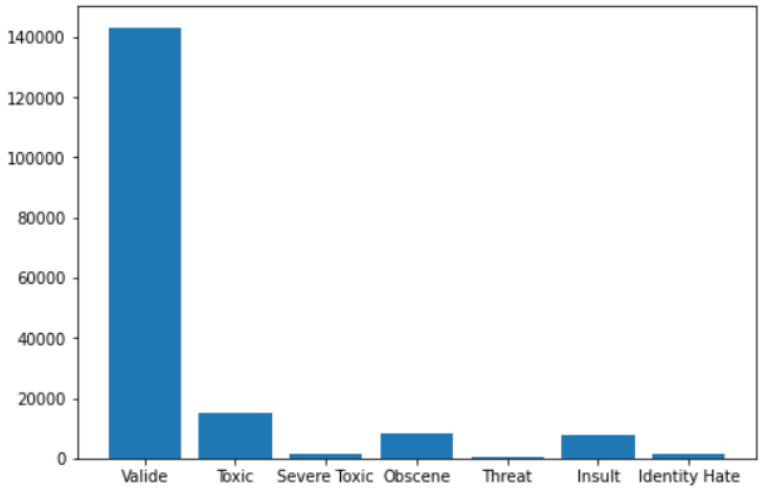
A l'aide du graphique ci-dessous ainsi que des extraits de lignes du dataset plusieurs observations sont relevées :

1er : le jeu de donnée est inéquitablement distribué

2ème : Un commentaire peut appartenir à plusieurs classes simultanément.

3ème : Un commentaire peut n'appartenir à aucune classe et sera désigné comme valide pour la suite du projet.

	id	comment_text	toxic	severe_toxic	obscene	threat	insult	identity_hate
6	0002bcb3da6cb337	cocksucker piss around work	1	1	1	0	1	0
12	0005c987bdfc9d4b	hey... it.. @ | talk . it... exclusive group w...	1	0	0	0	0	0
16	0007e25b2121310b	bye! look, come think comming back! tosser.	1	0	0	0	0	0
42	001810bf8c45bf5f	gay antisemmitian? archangel white tiger meow!...	1	0	1	0	1	1
43	00190820581d90ce	fuck filthy mother ass, dry!	1	0	1	0	1	0



Comme énoncé dans l'introduction, des algorithmes de Machine Learning et Deep Learning seront favorisés comme approche pour résoudre la classification. Effectivement, la quantité importante de données disponibles permet la mise en place de ce type de solution.

Solutions :

Machine Learning :

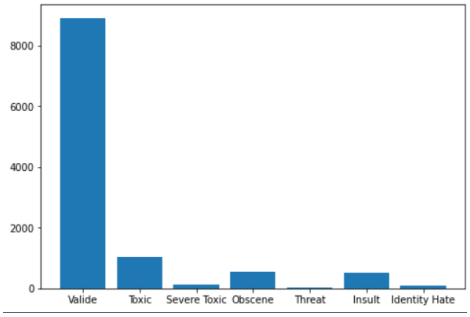
Afin de se donner un premier aperçu des résultats envisageables, l'utilisation d'un algorithme de Machine Learning simple à comprendre et rapide à entrainer est favorisé. Aujourd'hui, l'un des algorithmes de Machine Learning le plus puissant pour la classification multi classes n'est autre que le Random Forest. Nous décidons donc de partir sur ce dernier.

Notre première étape sera de nettoyer les commentaires. Pour ce faire nous séparons les contractions, supprimons les caractères spéciaux, les nombres, ainsi que les URL et les mentions. Un mot ne pouvant être passé directement à notre Random Forest nous devons tout d'abord convertir nos commentaires bruts en un TF-IDF. Le TF-IDF signifiant respectivement Terme Frequency et Inverse Document Frequency nous permet de mesurer la pertinence d'un mot ou d'un terme clé dans un commentaire en se basant sur sa rareté dans l'ensemble de notre data set. Plus le score TF-IDF d'un mot est élevé et plus le mot sera intéressant lexicalement à prendre en compte. Notre corpus étant particulièrement volumineux nous déciderons de ne prendre seulement que 10.000 commentaires afin de ne pas excéder 16gb de ram (plus il y a de commentaires et plus il y a de vocabulaires à prendre en compte). Pour finir, nous découpons notre data set en deux afin d'obtenir un jeu d'entrainement (80%) et un jeu de validation pour tester notre modèle (20%).

Notre data set correctement configuré, nous pouvons donc entrainer notre Random Forest avec les hypers paramètres suivants : n_estimators = 15 (nombre d'arbres de décisions du Random Forest).

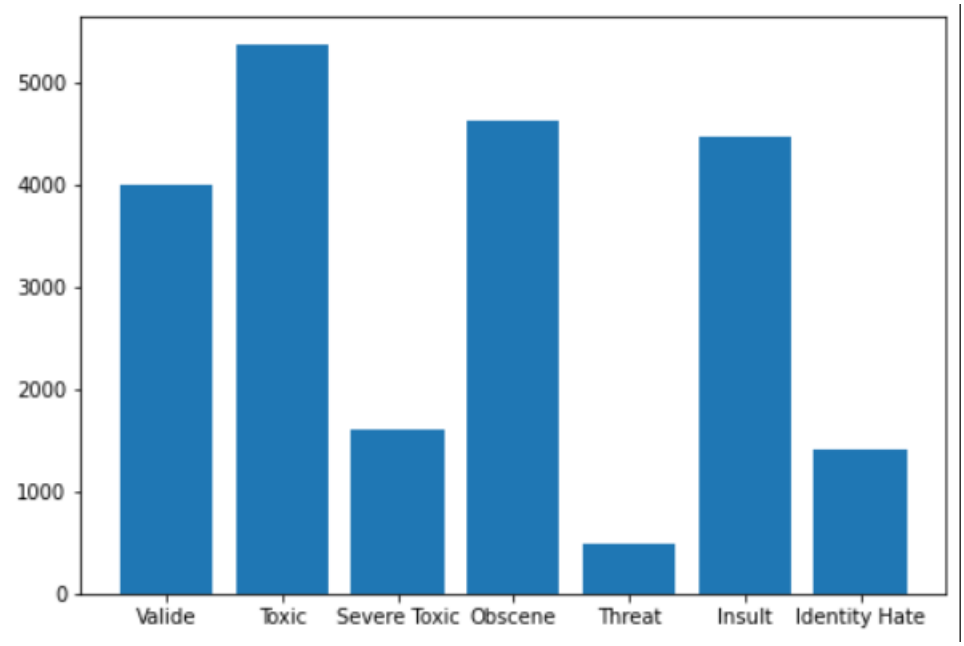
Notre premier modèle délivre une accuracy de 0.91 ce qui semble à première vue un bon résultat. Cependant, en regardant la précision ou le rappel sur les données sous représentés nous nous rendons compte que les résultats ne sont pas bons.

```
For label : toxic we have 242 success (20 true positive and 222 true negative) and 8 errors (0 false positive and 8 false negative)
For label : severe_toxic we have 247 success (2 true positive and 245 true negative) and 3 errors (0 false positive and 3 false negative)
For label : obscene we have 245 success (11 true positive and 234 true negative) and 5 errors (1 false positive and 4 false negative)
For label : threat we have 248 success (0 true positive and 248 true negative) and 2 errors (0 false positive and 2 false negative)
For label : insult we have 242 success (9 true positive and 233 true negative) and 8 errors (0 false positive and 8 false negative)
For label : identity_hate we have 245 success (0 true positive and 245 true negative) and 5 errors (0 false positive and 5 false negative)
```



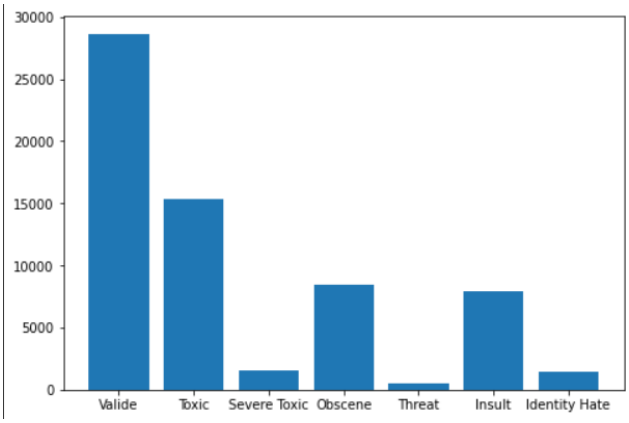
Nous décidons donc de refaire le même modèle mais cette fois ci en sélectionnant le maximum de commentaires sous-représentés et en limitant le nombre de commentaires valides. Cette fois nous obtenons une accuracy inférieure de 0.58 ce qui est plus représentatif des résultats obtenus. Nous notons également le rappel qui sera utilisé pour comparer ultérieurement le Random Forest à un modèle de Deep Learning.

```
For label : toxic we have 219 success (118 true positive and 101 true negative) and 26 errors (9 false positive and 17 false negative) / Precision: 0.9291338582677166, Recall 0.8740740740740741.
For label : severe_toxic we have 218 success (10 true positive and 208 true negative) and 27 errors (4 false positive and 23 false negative) / Precision: 0.7142857142857143, Recall 0.30303030303030304.
For label : obscene we have 211 success (89 true positive and 122 true negative) and 34 errors (11 false positive and 23 false negative) / Precision: 0.89, Recall 0.7946428571428571.
For label : threat we have 236 success (2 true positive and 234 true negative) and 9 errors (0 false positive and 9 false negative) / Precision: 1.0, Recall 0.18181818181818182.
For label : insult we have 204 success (85 true positive and 119 true negative) and 41 errors (20 false positive and 21 false negative) / Precision: 0.8095238095238095, Recall 0.8018867924528302.
For label : identity_hate we have 223 success (9 true positive and 214 true negative) and 22 errors (1 false positive and 21 false negative) / Precision: 0.9, Recall 0.3.
```



Deep Learning :

Une fois les premiers résultats obtenus nous décidons de tenter de les améliorer en utilisant un modèle de Deep Learning. Nous commençons par créer un sous-échantillon du jeu de données afin de limiter le temps de traitement et de rebalancer les données.



Ensuite, nous supprimons les stop word ainsi que les caractères spéciaux et les chiffres afin de réduire le nombre de vocabulaire et ainsi également diminuer le temps de calculs pour l'entrainement.

Afin de rendre les commentaires compatibles avec un algorithme de Machine Learning nous transformons tous les mots en chiffres en les tokenisants. Nous décidons ensuite de créer 6 réseaux de neurones identiques mais séparés afin de les entrainer à classifier chaque commentaire sur une seule classe. Chaque réseau disposera d'une couche d'embedding de 100 neurones. Interviendra ensuite une couche bidirectionnel GRU ayant pour nombre de neurone la taille moyenne d'un commentaire qui permettra au neurone de prendre en compte la « temporalité » de la phrase. Nous finirons enfin par deux couches Dense respectivement de 6 puis 1 neurone avec deux dropout de 50% afin de forcer chaque neurone et éviter ainsi un surapprentissage notamment avec les classes les moins représentées. Une fois les 6 réseaux de neurones entrainés nous regroupons leurs 6 prédictions pour n'en former plus qu'une. Le seuil de projection de la probabilité est adapté en fonction de chaque réseau pour améliorer les résultats.

```
tmp = []
tmp.append(1 if tox[index] > 0.6 else 0)
tmp.append(1 if srv_tox[index] > 0.95 else 0
tmp.append(1 if obsc[index] > 0.7 else 0)
tmp.append(1 if threat[index] > 0.92 else 0)
tmp.append(1 if insult[index] > 0.8 else 0)
tmp.append(1 if hate[index] > 0.9 else 0)
Y_predict.append(tmp)
```

Comme le montre les résultats ci-dessous obtenus avec un jeu de test nous remarquons que les réseaux de neurones obtiennent de bien meilleurs résultats que notre modèle basé sur un Random Forest. En effet, les réseaux prédisent mieux (notamment en regardant le recall) ou aussi bien que le Random Forest alors que l'ensemble de test est beaucoup plus grand et donc le risque d'erreurs aussi.

```
For label : toxic we have 7949 success (2423 true positive and 5526 true negative) and 1030 errors (367 false positive and 663 false negative) / Precision: 0.8684587813620072, Recall 0.7851587815942969.
For label : severe_toxic we have 8498 success (225 true positive and 8273 true negative) and 481 errors (373 false positive and 108 false negative) / Precision: 0.3762541806020067, Recall 0.6756756756756757.
For label : obscene we have 8295 success (1488 true positive and 6807 true negative) and 684 errors (398 false positive and 286 false negative) / Precision: 0.7889713679745494, Recall 0.8387824126268321.
For label : threat we have 8812 success (42 true positive and 8770 true negative) and 167 errors (128 false positive and 39 false negative) / Precision: 0.24705882352941178, Recall 0.5185185185185185.
For label : insult we have 7992 success (1174 true positive and 6818 true negative) and 987 errors (557 false positive and 430 false negative) / Precision: 0.6782206816868862, Recall 0.7319201995012469.
For label : identity_hate we have 8524 success (177 true positive and 8347 true negative) and 455 errors (340 false positive and 115 false negative) / Precision: 0.34235976789168276, Recall 0.6061643835616438.
```

Améliorations :

Pour des raisons de temps de calcul, beaucoup de compromis ont été fait sur la précision des résultats. Il serait donc intéressant de relancer l'apprentissage des 6 réseaux de neurones en augmentant le nombre d'époques (jusqu'à présent entraînements avec seulement 5 époques). Nous pourrions aussi mettre en place des callbacks pour faire varier le learning rate pendant l'apprentissage. Un autre point d'apprentissage concerne aussi le jeu de données, il faudrait utiliser le jeu de données entier afin de généraliser au maximum nos modèles.

Lien GitHub du projet (Ml et DeepL fichier séparés) : [NLP to detect toxic comments. (github.com)]