AVIGNON UNIVERSITÉ

M2 IA – 2022/23

UE Techniques d'apprentissage automatique

ECUE Traitement Automatique des Langues

Mickael Rouvier

TP

1 Contexte et objectif

L'analyse des sentiments est un domaine de recherche extrêmement actif en Traitement automatique du langage naturel (TALN). En effet, ces dernières années ont vu se multiplier les sources de données textuelles porteuses d'opinion disponibles sur le Web (Twitter, Facebook, Instagram, WhatsApp...). Devant cette abondance de données et de sources, l'automatisation de la synthèse des multiples avis devient cruciale pour obtenir efficacement une vue d'ensemble des opinions sur un sujet donné.

L'objectif de ce TP est de classifier les tweets selon l'opinion/sentiment/émotion exprimé par son auteur, en l'occurrence ici 3 classes : positif, négatif ou neutre.

Vous devez créer un système de détection automatique de sentiment en appliquant la modélisation par sac-de-mots et un classificateur de type SVM (Séparateur à Vaste Marge) à un corpus de tweets.

2 Mise en œuvre

Données. Pour réaliser ce TP vous avez à votre disposition un corpus de tweets en anglais (disponible sur la plateforme e-upav, cf. *Données TP*). Ce corpus, qui provient de la conférence SemEval 2013, est séparé en trois parties :

- Un jeu d'**apprentissage** : `twitter-2013train-A.txt`.
- Un jeu de **développement** : `twitter-2013dev-A.txt`.
- Un jeu de **test** : `twitter-2013test-A.txt`.

Les trois fichiers sont structurés de la même façon, en trois colonnes séparées par des **tabulations** : numéro du tweet dans le jeu, sentiment associé au tweet et contenu textuel du tweet.

> **Attention :** le classificateur doit être entraîné sur le corpus d'apprentissage et évalué sur le corpus de développement et de test. Les données de développement et de test ne doivent surtout pas être intégrées dans les données d'apprentissage.

Étapes. Les différentes étapes à réaliser sont les suivantes :

1. **Chargez** les fichiers correspondant au jeu d'apprentissage, de développement et de test.
2. **Extrayez** le lexique du corpus d'apprentissage, c'est-à-dire la liste de tous les mots apparaissant au moins une fois dans un tweet.
3. **Assignez** un numéro unique à chaque mot du lexique. **Important :** commencez la numérotation à 1.
4. **Décomptez**, pour chaque message du corpus entier, le nombre d'occurrences de chaque mot qu'il contient.
5. **Convertissez** séparément les trois jeux de données (apprentissage, développement et test) de façon à obtenir trois fichiers au format supporté par le SVM (cf. Section 3).
6. **Appliquez** le classificateur aux fichiers obtenus (cf. Section 3.

3 Classificateur

Comme indiqué en introduction, nous allons utiliser un SVM pour effectuer la classification. Le plus simple est d'utiliser l'implémentation *LibLinear*[1].

Format d'entrée. Les fichiers donnés en entrée du SVM doivent être au format suivant :

```
<label> <index1>:<value1> <index2>:<value2>...
...
...
...
```

Chaque ligne représente un tweet et se finit par un `\n`. Les différents éléments sont séparés par des espaces. Pour la classification, `<label>` est un entier indiquant le label de la classe (par exemple : 0 pour objectif, 1 pour positif, 2 pour négatif et 3 pour mixte).

Chaque paire `<index>:<value>` permet d'attribuer une valeur à une entrée du classificateur. Dans le cas des sacs-de-mots, `<index>` est le numéro unique du mot (un entier) dans le lexique du corpus entier, et `<value>` est le nombre de fois que ce mot apparaît dans le tweet considéré (un autre entier).

Quelques remarques concernant le format du fichier :

- Il faut que les `<index>:<value>` soient triés par ordre **croissant** selon l'index.
- La **numérotation** des **mots** dans le lexique (i.e. `<index>`) doit commencer par 1.

Par exemple, si le tweet est positif et contient les mots de numéros 4, 89, 145, avec des fréquences respectives de 5, 1, et 2, alors on aura la ligne suivante :

```
1 4:5 89:1 145:2
```

Utilisation. Téléchargez le logiciel LibLinear[1] et exécutez-le en ligne de commande. Alternativement, certains langages proposent une intégration sous forme de bibliothèque.

Pour essayer les paramètres en validation croisée (ici : 5-fold) sur les données `train.svm` :

```
liblinear-1.21/train -c 4 -e 0.1 -v 5 train.svm
```

Pour entrainer sur les données `train.svm` un modèle qui sera stocké dans le fichier `tweets.model` :

```
liblinear-1.21/train -c 4 -e 0.1 train.svm tweets.model
```

Pour tester ce `tweets.model` modèle sur les données `test.svm` et produire la sortie `out.txt` :

```
liblinear-1.21/predict test.svm tweets.model out.txt
```

4 Améliorations

Quelques pistes possibles pour améliorer votre système :

- Normaliser les données :
  - Casse : supprimer la distinction entre minuscules et majuscules.
  - Stemming (racinisation) : ne garder que la racine des mots. Des outils disponibles en ligne permettent d'effectuer cette opération[2]. Encore mieux : faire une lemmatisation.
- Supprimer les mots-outils : prépositions, conjonctions, pronoms, etc. Des listes de mots-outils sont disponibles en ligne[3].

1. https://www.csie.ntu.edu.tw/~cjlin/liblinear/
2. http://snowball.tartarus.org/algorithms/french/stemmer.html
3. https://github.com/stopwords-iso/stopwords-fr/tree/master/raw