Enzo POGGIO Projet 2016 MELS

1 Baseline

Ce que j'ai fait :

Démarche adoptée :

Afin de répondre au problème de manière exhaustive, je décide d'adopter la démarche scientifique. Ainsi, après une observation minutieuse des données, je formule d'abord un ensemble d'hypothèses que j'ai testé par rapport à une affectation aléatoire de la valeur des tweets (Moyenne pos & neg de 33%). J'écarte ensuite les traits trop peu distinctifs (par exemple, distinguer les tweets contenant des smileys). Ayant obtenu de bons résultats avec l'addition de la valeur sentimentale de chaque mot des tweets (avec Sentiword 40%), je tente de corroborer cette hypothèse en utilisant une autre liste de mots positifs et négatifs. Je me suis basé sur les travaux de la NRC-SentimentAnalysis, qui a fini première dans la "Building the State-of-the-Art in Sentiment Analysis of Tweets". J'utilise alors maintenant la pmilexicon.

Gestion et prétraitement des données de base :

Systèmes modulaire & Dictionnaire de données :

Je choisis de lire une fois le fichier testé et de le mettre en mémoire dans un dictionnaire de données. Cette variable sera modifiée, voir rééditée dans la majorité des modules de ce programme. Le système modulaire est plus simple pour essayer une hypothèse portant sur un trait. Je conditionne l'information ainsi :

```
d[i]=(firstNum , secondNum, value, tweet, posScore, negScore)
```

Où i est un int (différent pour chaque tweet) ;
firstNum, secondNum sont les deux premiers numéros (désignant le numéro utilisateur et le numéro du tweet) ;
value est la valeur du tweet ["positive", "negative", "neutral"] ;
tweet est un tableau contenant toutes les chaînes de caractère du tweet ;
posScore, negScore sont des variables de types float qui permettront de définir la valeur du tweet lors de l'affectation.

Description des modules :

readFile(file)
Permet de lire un fichier de tweet ;
Crée une variable dictionnaire contenant l'enregistrement de chaque tweet (firstNum, secondNum, value, tweet, posScore, negScore).

writeData(data)
Permet d'écrire les données de façon à ce que le script scoredev.py puisse les utiliser

sentiWordNet(swntxt) & pmilexicon(pmitxt):
Créent et retournent des dictionnaires à partir des ressources entrées.
Ils sont ainsi constitués (voir le code pour plus de détails)
swn[wordkey]=[posScore,negScore, occWord]
pmi[key]=(score, occPos, occNeg)

termeAnalyzerSWN(swn, data) & termeAnalyzerPMI(pmi, data):
Utilisent les dictionnaires créés pour pondérer les tweets ;
Changent les valeurs de posScore, negScore et retournent le dictionnaire changé.

affectationPNN(data) & affectationPNNCoef(data,coef):
Affectent une valeur ["positive", "negative", "neutral"] au tweet selon posScore, negScore et coef[] (calculé dans coefficateur (data, version))
Retournent le dictionnaire évalué

randomizer(data)
Permet d'attribuer une valeur aléatoire à chaque tweet et retourne un dictionnaire évalué aléatoirement.

understandData(data)
Affiche des informations qui peuvent aider à trouver un critère d'évaluation à partir des données générées

coefficateur(data, version)
Crée des coefficients à partir de données empiriques développées avec des tests.

```
        Procédure d'évaluation :
                Explication de mes critères d'affectation

Dans affectationPNNCoef(data,coef) nous avons la clé de notre affectation !
Procédure :
Tout d'abord on va initialiser la valeur à neutre

                value="neutral"

Ensuite on s'occupe des tweets ambigus qui pourraient être positifs ou négatifs. Ils vont d'abord dans
un ensemble très tolérant et sont ensuite sélectionnés selon leur plus haut score entre negScore et
posScore:

                if (negScore>negAmb and posScore>posAmb ):
                        if negScore>posScore:
                                value="negative"
                        elif negScore<posScore:
                                value="positive"

Et pour finir nous traitons les tweets non-ambigus. Si leur valeur dépasse les seuils suivants, ils
sont déclarés comme positifs ou négatifs :

                elif negScore>negNonAmb:
                        value="negative"
                elif posScore>posNonAmb:
                        value="positive"

La valeur de sélection a été trouvée en étudiant les données grâce à la fonction understandData(data)
et coefficateur(data, version) (data = données de développement) :

Moyenne Positive = 0.6791802897448531    Moyenne Négative = 0.39006549775717836
Minimum Positif = -1.6969848655423443    Minimum Négatif = -3.126057203751694
Maximum Positif = 12.126889641038899     Maximum Négatif = 9.599009709434053

Ces coefficients ont été calculés grâce aux valeurs de affectation(data), qui elles, proviennent de
understandData(data). Elles ont ensuite été ajustées empiriquement jusqu'à avoir un score
satisfaisant.

                        coef = moyenneAjustée / moyenne

coefposAmb =0.868694232         coefposNonAmb = 0.927588756
coefnegAmb =0.743464884         coefnegNonAmb = 0.820375044
coefmoyenneAmb = 0.806079558    coefmoyenneNonAmb = 0.8739819

Ici j'utilise les moyennes positives et négatives comme critère de sélection. Pour l'améliorer de
quelques pourcents j'ai un peu diminué les moyennes afin de traiter beaucoup plus de tweets ambigus et
un peu plus de tweets non-ambigus !

        Comment ça marche :
                Comment lancer ma baseline ?

Pour lancer ma baseline il vous suffit d'écrire dans le bash du dossier contenant ma baseline, le
dossier pmilexicon (, le fichier sentiWordNet.txt) et le fichier à traiter :
python3 baseline.py -*le nom du fichier à traiter*- > -*le nom du fichier que vous voulez créer*-
Remplacer les -*balises*- par les noms de fichiers appropriés.
```

Résultats & Conclusion

Avec les données de développement j'arrive à :

Confusion table:

gs \ pred	positive	negative	neutral
positive	259	102	91
negative	46	179	36
neutral	206	223	154

Scores:

class	precision	recall	fscore
positive	(259/511) 0.5068	(259/452) 0.5730	0.5379
negative	(179/504) 0.3552	(179/261) 0.6858	0.4680
neutral	(154/281) 0.5480	(154/583) 0.2642	0.3565
average(pos and neg)			0.5029

Et avec les données de test j'arrive à :

Scoring T13:
- positive: P=48.62, R=51.50, F1=50.02
- negative: P=74.04, R=26.89, F1=39.45
- neutral: P=27.89, R=58.22, F1=37.71
- OVERALL SCORE : 44.73

Scoring T14:
- positive: P=58.04, R=59.92, F1=58.97
- negative: P=63.33, R=21.99, F1=32.65
- neutral: P=25.22, R=46.13, F1=32.61
- OVERALL SCORE : 45.81

Scoring TS1:
- positive: P=42.42, R=33.33, F1=37.33
- negative: P=35.00, R=50.00, F1=41.18
- neutral: P=23.08, R=18.75, F1=20.69
- OVERALL SCORE : 39.25

On remarque une baisse significative entre les résultats d'entraînement et les résultats du test. Une adaptation plus générique aux données est certainement nécessaire. De plus, on s'aperçoit que les tweets neutres sont largement sous-représentés avec mon système : une augmentation de leur rappel serait bénéfique. Il me faudrait de meilleurs facteurs de discernement pour ne pas surestimer les tweets neutres.

Sources :

pmiLexicon tiré de cette page :
http://saifmohammad.com/WebPages/Abstracts/NRC-SentimentAnalysis.htm