

## Rapport du projet de traduction automatique de la langue

### Sommaire

Rapport du projet de traduction automatique de la langue.....	1
Sommaire .....	1
Objectif du projet .....	2
Evaluation des plateformes d'analyse linguistique .....	2
Analyse morpho-syntaxique.....	2
Reconnaissance d'entités nommées .....	3
Contribution des membres du groupe .....	5
Sullivan Honnet.....	5
Jules Vittone-Burnel.....	5

## Objectif du projet

L'objectif affiché de ce projet était d'évaluer les résultats des analyses linguistiques réalisées par divers outils et d'en déduire lesquels étaient les plus efficaces dans quelle situation et éventuellement de proposer des pistes d'améliorations intéressantes pour ces outils.

L'objectif plus profond, abritait par ce projet, était de nous pousser à nous intéresser un domaine que nous rencontrions pour la première fois, celui du traitement automatique des langues et, par extension, la linguistique de manière générale.

## Evaluation des plateformes d'analyse linguistique

### **Analyse morpho-syntaxique**

```
root@MSI:/mnt/d/Etude/Etude/Et5/T  
t.txt.pos.stanford.univ  
Word precision: 0.712876762835  
Word recall: 0.712876762835  
Tag precision: 0.712876762835  
Tag recall: 0.712876762835  
Word F-measure: 0.712876762835  
Tag F-measure: 0.712876762835
```

*Figure 1 : Résultat de l'analyse morpho-syntaxique de stanford*

La méthode utilisée par stanford pour ses analyses morpho-syntaxiques obtient des résultats plutôt décevant avec seulement 71% de résultat conforme aux attentes. Ces mauvaises performances peuvent peut-être s'expliquer en partie par le choix de la conversion du fichier de référence dont les lignes comportant plusieurs mots furent coupées. Les résultats sont donc purement indicatifs. Ce souci n'explique pas cette différence marquée.

L'analyse est faite en se basant sur une méthode descendante dont la méthode consiste à s'intéresser d'abord aux éléments centraux de la phrase : groupe verbal, groupe sujet, groupe nominaux... Puis ensuite déconstruire ces éléments et en déduire à partir de leur rôle dans la phrase, leur place grammaticale.

Je pense que cette analyse est une des sources du problème, en effet, en isolant le mot au sein d'un petit bloc qui ne communique pas avec l'extérieur, on limite de fait les possibilités pour l'interprétation des mots par la machine.

```
root@MSI:/mnt/d/Etude/Etude/Et5/1  
.nltk.univ  
Word precision: 0.7719797887  
Word recall: 0.7719797887  
Tag precision: 0.7719797887  
Tag recall: 0.7719797887  
Word F-measure: 0.7719797887  
Tag F-measure: 0.7719797887
```

Figure 2 : Résultat de l'analyse morpho-syntaxique de nltk

Cette méthode avec 23% d'erreur est la meilleure des deux auxquelles nous nous sommes intéressés par manque de temps pour la troisième.

### Reconnaissance d'entités nommées

```
root@MSI:/mnt/d/Etude/Etude/Et5/TA  
ltk.bis.conll.final  
Word precision: 0.880535103184  
Word recall: 0.880535103184  
Tag precision: 0.880535103184  
Tag recall: 0.880535103184  
Word F-measure: 0.880535103184  
Tag F-measure: 0.880535103184
```

Figure 3 : Résultat de la reconnaissance d'entités nommées de nltk

La méthode du module de NLTK pour la reconnaissance d'entités nommées est de créer des arbres avec les noms et les adjectifs qui y sont liés, cette méthode permet de récupérer facilement les différentes entités nommées mais pour placer les tags conll sur les différents mots il faut faire une analyse des différents arbres et ensuite taggé en conséquence. De plus le module de reconnaissance utilise de nombreux autres modules ce qui augmente les incertitudes et donc les erreurs d'approximations ce qui un certain niveau d'imprécision (12%) lors de l'évaluation.

```
root@MSI:/mnt/d/Etude/Etude/Et5/T
rd.conll.final
Word precision: 0.899890753799
Word recall: 0.899890753799
Tag precision: 0.899890753799
Tag recall: 0.899890753799
Word F-measure: 0.899890753799
Tag F-measure: 0.899890753799
```

*Figure 4 : Résultat de la reconnaissance d'entités nommées de stanford*

La plateforme de stanford pour la reconnaissance des entités nommées fait une reconnaissance en faisant une comparaison avec un dictionnaire de mots qui lui sert à faire la distinction entre les différentes entités du texte, l'avantage de cette méthode c'est que l'étude se fait rapidement mais l'analyse ne prend pas en compte des mots qui lieraient différentes entités nommées comme par exemple dans la référence :

« The B-ORG United I-ORG B-LOC States I-ORG I-LOC 's I-ORG largest I-ORG car I-ORG manufacturer I-ORG General I-ORG Motors I-ORG ».

Qui devient avec stanford :

« The O United B-LOC States I-LOC 's O Largest O car O manufacturer O General B-ORG Motors I-ORG ».

On remarque bien que le mot « The » est bien reconnu comme faisant partie de l'entité nommée dans la référence alors que stanford non, ces différences expliquent pourquoi il y a une baisse de la précision.

```
root@MSI:/mnt/d/Etude/Etude/Et5/TAL/Pro  
lima.conll  
Word precision: 0.912222974935  
Word recall: 0.912222974935  
Tag precision: 0.912222974935  
Tag recall: 0.912222974935  
Word F-measure: 0.912222974935  
Tag F-measure: 0.912222974935
```

*Figure 5 : Résultat de la reconnaissance d'entités nommées de lima*

## Contribution des membres du groupe

### **Sullivan Honnet**

Il eut la charge de l'utilisation des outils nltk et stanford, il s'est chargé de générer les fichiers à partir de ces outils.

De plus il s'est occupé du traitement des fichier python pour la reconnaissance des entités nommées pour les trois outils : lima, nltk et stanford.

Le but des différents fichiers est de récupérer les résultats des outils pour pouvoir par la suite faire la comparaison avec les fichiers de référence ici le fichier `ne_reference.txt.conll`. Le premier problème étant les sauts de ligne au milieu des phrases qui était traiter différemment selon l'outil par exemple stanford considère le « . » comme caractère pour faire un saut de ligne et concaténait alors les morceaux de phrase ce qui fait que le fichier qui en résultait ne correspondait donc que pour les premiers mots de la première ligne. On a donc eu le parti pris de concaténer les phrases et de sauter des lignes qu'après un « . ».

### **Jules Vittone-Burnel**

Il s'est occupé de l'utilisation et de la génération du fichier lima et d'écrire le script principal. Ce script a pour rôle de formater tous les fichiers de départ pour obtenir des fichiers utilisables par la suite par le programme `evaluate.py`. L'une des grosses difficultés dans cette partie fut que

chaque fichier suivait sa propre nomenclature et ses propres règles et qu'il n'existait pas d'uniformité entre le fichier de départ fourni et les fichiers obtenus par nltk, stanford et lima.

Le problème se présenta pour l'analyse morpho-syntaxique d'abord par le fichier source qui contenait de nombreuses lignes comportant plusieurs mots pour un seul tag alors que tous les autres programmes donnaient une équivalence un mot, un tag. Ce problème fut résolu en découpant les groupes de mots d'une ligne en plusieurs lignes contenant un mot et le tag associé auparavant au groupe de mots.