Rapport de projet GATE

La tâche de ce projet consistait à annoter deux livres (un en anglais et un en français) en utilisant le logiciel GATE. Pour ce travail j'ai choisi *L'étranger*, écrit par Albert Camus et *Animal Farm* écrit par George Orwell, ils contiennent chacun onze et dix chapitres et environ 30 000 mots.

Pour la Préparation de ce travail, j'ai dans un premier temps créé un corpus pour chaque livre dans *Language Resources*, en important (populate) les fichier txt pour chaque chapitre du livre dans chaque corpus, cela m'a permis de faciliter la gestion de l'opération et le stockage.

J'ai ensuite créé un *data store* depuis l'application gate pour stocker les résultats du projet, dans lequel le corpus balisé et les documents seront stockés.

Une fois les préparatifs effectués, il est temps de commencer à annoter le texte. Il existe deux types d'annotation dans GATE(manuelle et automatique).

Pour commencer avec la partie annotation manuelle, nous devons créer des fichiers xml pour *les schémas annotations* , qui définissent les balises et les attributs du balisage. Nous devons par la suite cliquer sur *Language Ressource* , *New* et *Annotation Schéma* pour les charger dans GATE. Lorsqu'on clique sur les caractères que l'on veut marquer dans le texte, la fenêtre *Annotation editor dialog* nous montre les balises et les attributs spécifiés dans le fichier importé. Nous pouvons sélectionner les options appropriées en fonction de nos besoins.

J'ai fait 4 schémas d'annotations: LesDialogues.xml, LesTitres.xml, LesQualifieurs.xml, LesPresonnages.xml.Cela peut représenter beaucoup de travail de les marquer manuellement un par un, nous pouvons donc utiliser des expressions régulières depuis la fenêtre de recherche, qui trouveront tous les textes correspondant à nos regex, et ensuite sélectionner les balises et attributs appropriés un par un. Par exemple pour trouver les dialogues nous pouvons faire : «.*? » ou ". * ?", pour trouver les adjectifs et nom nous pouvons simplement écrire son radical pour trouver tous leur formes.

Pour la partie annotation automatique, nous devons télécharger les applications dont nous aurons besoins. Nous les trouvons depuis le plugin CREOLE, la création ou personnalisation se fait dans *Processing Resources*. Nous pouvons finalement les appliquer au texte à traiter via *application*.

Dans *application*, nous pouvons créer un pipeline conditionnel, ce qui nous permet de choisir quand une ressource de traitement est l'utilisée. Dans ce projet, j'ai défini les étapes du traitement de texte pour les deux langues en utilisant *Conditional corpus pipeline* . Tout d'abord, j'ai ajouté l'outil de reconnaissance de la langue *TextCat Language Identification* à ce pipeline, de sorte que chaque document a lang(langue de texte) dans ses propriétés, ce qui sera utilisé comme une condition pour chacune des étapes suivantes. Ensuite, en fonction de la langue, les étapes sont définies et l'ordre dans lequel elles sont effectuées est ajusté.

A l'exception des *Processing Resources* de routine comme *Tokeniser, Sentence splitter, POS Tagger*. Je vais vous présenter les *Processing Resources* que j'ai personnalisée dans mon pipeline.

Le premier est le Gazetteer, qui est un dictionnaire, il nécessite de personnaliser un fichier lst avec tous les mots que vous souhaitez trouver, puis d'écrire ce fichier lst dans un fichier def, dans lequel nous pouvons définir le type majeur, le type mineur, la langue des mots de cette liste et les balises qu'ils auront. Dans la section des ressources de traitement, nous pouvons cliquer sur *Annie gazetteer* ou *Hash gazetteer* pour importer le fichier def que nous venons de créer, personnellement je préfère *Annie gazetteer* parce qu'il permet de changer le nom de *annotation type*. Ensuite, dans la section application nous pouvons spécifier les équivalents de ce groupe d'annotation créé par ce gazetteer , et si nous l'ajoutons au pipeline nous pouvons l'exécuter sur le corpus pour obtenir les annotations dont nous avons besoin.

J'ai créé 4 fichier lst pour annoter les personnages, les locations, les références temporels et les moments. je les ai par la suite regroupé dans un annotation set nommé comme *annotations avec dictionnaires*.

Le deuxième est *flexible gazetteer*, une version avancée de ce qui précède, adaptée aux mots dont la morphologie comporte une dérivation et une inflexion, par exemple si vous voulez trouver toutes les inflexions verbales d'un verbe, toutes les formes singulières et plurielles d'un nom, etc.

La construction du fichier est la même que pour le gazetteer, il suffit de saisir le lemme morphologique original du mot dans le fichier lst, d'ajouter ce gazetteer au *gazetteerInst* dans les paramètres de *flexible gazetteer* lors de l'exécution du pipeline, et finalement éditer les tags correspondants dans les *inputFeatureNames*. Notons cependant que le *flexible gazetteer* ne peut être ajouté avant que l'analyseur morpologique ne soit ajouté au pipeline, sinon il ne reconnaîtra que les prototypes des mots que nous écrirons dans le fichier lst (tout comme le gazetteer normal).

Cependant, pour le texte français, il n'est actuellement pas possible de reconnaître sa racine, qui est l'input [Token.root] requis par le gazetteer flexible. Il m'a donc été impossible de le faire pour le texte français. Dans ce processus j'ai créé deux fichiers lst pour chaque corpus pour trouver les verbe de mouvements et les verbes du discours.

Le troisième est *JAPE Transducer* qui vous permet d'automatiser l'annotation avec un plus grand degré de liberté. Nous devons d'abord créer un fichier jape écrit en syntaxe jape. S'il est reconnu par le *JAPE Transducer*, alors le contenu du fichier est grammaticalement correct, sinon, la rupture *message* signalera l'erreur correspondante.

Un problème avec mon *JAPE Transducer* est que, dans son paramètre *inputASName*, même si j'ai correctement saisi les annotations requises pour l'élément nous avons besoin d'ajouter le contenu des *Annotations avec dictionnaires* créées ci-dessus au jeu d'annotations vierge afin qu'il reconnaisse les inputs spécifiés. Ce que je souhaites ici, c'est un contenu avec le nom de balise *PassageHistoire*, qui est le contenu qui contient les Personnages de Annotations avec dictionnaires, verbe.majorType=="VerbeMouvements". et la phrase avec les trois éléments de ReferenceTemporelles. j'ai ainsi créé trois fichiers jape qui reconnaissent les phrases contenant uniquement le premier élément, celles contenant les deux premiers éléments et celles contenant tous les éléments respectifs . J'ai récupéré les résultats (phrases annotées) dans annotation set *Annotations automatiques*.

Après que j'ai créé toutes les annotations, j'ai stocké l'annotation set *Annotations manu-*

elles des deux corpus en utilisant la fonction *save as inline.xml*, puis j'ai combiné ces fichiers en un seul fichier en respectant l'ordre pour chaque livres, j'ai ensuite écrit le code pour le convertir en html dans l'en-tête de ce fichier xml, puis j'ai changé son extension en html. Enfin dans le même répertoire des deux fichiers HTML, j'ai créé un fichier css pour modifier son format. De cette façon, les deux fichiers html peuvent être affichés sur la page correspondant au balisage, grâce à la fonction *imprimer* du navigateur, nous pouvons en faire un fichier pdf.

En résumé, l'annotation manuelle est plus détaillée et précise avec moins de travail, et l'aide de la recherche par expressions régulières. Elle est encore plus rapide et plus facile que la construction lente de dictionnaires ou de JAPE, et nous pouvons annoter avec précision un texte qui doit être annoté mais qui ne peut pas être décrit avec précision dans un langage formel en raison d'ambiguïtés. Cependant, je pense personnellement que l'annotation automatisée est plus pratique et efficace, et c'est là tout l'intérêt du logiciel GATE, car une fois que la quantité de texte à annoter est très importante, l'annotation manuelle devient une tâche tortueuse, et en raison de la fatigue humaine, l'efficacité et la précision du travail diminuent très fortement, et c'est là que les règles de correspondance prédéfinies deviennent particulièrement importantes.