Damien Levy

Rapport de stage

du 9 avril 2018 au 6 juin 2018

Au Laboratoire lorrain de recherche en informatique et ses applications

à Vandœuvre-lès-Nancy

Encadré par MM. Bruno Guillaume, Yves Lepage, Jean Lieber et Emmanuel Nauer

Acquisition semi-automatique de cas de corrections de phrases en français

Licence Informatique
L3

De l’Université de Lorraine

Table des matières

INTRODUCTION :

Lorsque nous écrivons un texte en français, nous souhaitons écrire de la manière la plus juste possible. Pour ce faire, il est possible d'utiliser des outils informatiques, tels que les correcteurs orthographiques par exemple. On peut aussi avoir envie de donner une phrase en français, qui nous paraît incorrecte et vouloir sa correction. Pour pouvoir créer un programme de ce genre, on peut utiliser le raisonnement à partir de cas. Cela consiste à résoudre un problème en s'appuyant sur une base de cas, contenant des couples problème/solution. Un modèle de ce processus classique comprend plusieurs étapes : d'abord une phase d'inférence puis d'une phase d'apprentissage. La phase d'inférence se compose d'une étape de remémoration et d'une étape d'adaptation. L'étape de remémoration consiste à sélectionner un cas source jugé similaire au problème cible. L'étape d'adaptation permet de modifier la solution du cas source en solution candidate résolvant le problème cible. Enfin, l'étape d'apprentissage présente à un expert le cas nouvellement formé, qui le corrige si besoin, et le valide. Dans l'hypothèse où ce nouveau cas est considéré comme utile, il est ajouté à la base de cas.

Dans le cadre de ce stage, l'objectif est de créer une base de cas pour une application à la correction de phrases. Pour alimenter la base de cas, il sera utile de créer une base de cas minimale manuellement, afin que M. Giang puisse avancer dans la création d'un moteur d'inférences pour la correction de phrases en français. Il faudra aussi acquérir de manière semi-automatique des couples de phrases en mauvais français, accompagnées de leurs solutions en français correct. Les cas seront extrait du corpus WiCoPaCo (Wikipedia Correction and Paraphrase Corpus). Ce dernier est un historique des modifications de pages sous Wikipédia, contenant notamment des couples de phrases avant et après corrections. Et enfin, durant le processus de raisonnement à partir de cas, il faudra valider, puis ajouter les nouveaux cas créés par le programme, à la base de cas. La base de cas sera stockée dans une base de données créée par M. Ly.

Pour ce faire , je vais tout d'abord créer une base de cas minimale fondée sur mes expériences et recherches personnelles. Dans un premier temps, ces cas seront écrits dans un fichier texte, puis insérés dans une base de données. Ensuite, je vais récupérer le corpus WiCoPaCo sous forme XML, et en extraire des couples de phrases avant et après corrections. Pour finir, je vais isoler les modifications qui correspondent à des corrections de phrases. Je m'appuierai sur plusieurs critères afin de faire une présélection. Enfin, je validerai les couples obtenus dans le but d'enrichir la base de cas.

Remerciements

Description de l’entreprise

Le Loria, Laboratoire lorrain de Recherche en Informatique et ses Applications à été créé en 1997. C’est une Unité Mixte de Recherche commune au CNRS, à l’Université de Lorraine et à Inria. La mission du Loria est la recherche fondamentale et appliquée en sciences informatiques. Le Loria est ordonné en 5 départements, composé de 28 équipes dont 15 communes avec Inria.

Travail réalisé

Création de la base de cas minimale

J’ai tout d’abord fais des recherches, sur internet et grâce à mes connaissances, sur les erreurs les plus courantes dans des phrases en français. Puis, j’ai répertorié une dizaine de couples de phrases dans un fichier texte contenant uniquement les couples de phrases contenant une erreur et phrase corrigée ainsi que la source (site web présentant l’erreur par exemple). J’ai écris ce fichier afin qu’il puisse être utiliser par monsieur GIANG pour le développement du moteur de recherche à partir de cas. Par la suite, j’ai modifié le format du fichier et ajouté des informations pour les insérer dans la base de données créée par monsieur Ly. Cette base de cas contient les informations suivantes :

- phrase contenant une erreur, par exemple : « J’aime pas les pommes. »
- phrase corrigée , par exemple : « Je n’aime pas les pommes. »
- pertinence (**a définir**)
- mot ou groupe de mots contenant l’erreur : « j’aime »
- mot ou groupe de mots contenant la correction : « je n’aime »
- indice
- source
- langue : fr

Par exemple :

Je suis sur Nancy.	Je suis à Nancy.	True	sur	à	0	Wikipedia.	fr
Au jour d'aujourd'hui.	Aujourd'hui.	True	Au jour d'		0	Site LCI.	fr
J'ai été au cinéma.	Je suis allé au cinéma.	True	J'ai été	Je suis allé	1	Wikipedia.	fr
J'amènerai une bouteille.	J'apporterai une bouteille.	True	amènerai	apporterai	1	Wikipedia.	fr
Elle apporte son enfant.	Elle amène son enfant.	True	apporte	amène	1	Wikipedia.	fr
C'est de la faute de sa femme.	C'est la faute de sa femme.	True	de		0	Wikipedia.	fr
Ils croivent.	Ils croient.	True	croivent	croient	4	Site LCI.	fr
Tu es trop nul.	Tu es très nul.	True	trop	très	2	Moi.	fr
Nous avons été à Paris.	Nous sommes allés à Paris.	True	avons été	sommes allé	0	Site lefigaro.	fr
Après qu'il ait dîné.	Après qu'il a dîné.	True	ait	a	1	Site cnewsmatin.	fr
Après qu'il soit arrivé, il est allé chez sa mère.	Après qu'il est arrivé, il est allé chez sa mère.	True	soit	est	0	Site osez-ecrire-votre-roman.	fr
Lui aussi n'a pas compris pourquoi elle chante.	Lui non plus n'a pas compris pourquoi elle chante.	True	aussi	non plus	0	Site osez-ecrire-votre-roman.	fr
Ce n'est pas de ma faute.	Ce n'est pas ma faute.	True	de		0	Site osez-ecrire-votre-roman.	fr
De façon à ce que.	De façon que.	True	à ce		0	Site etudiant.aujourdhui.fr	fr
J'ai manger.	J'ai mangé.	True	manger	mangé	4	moi.	fr

Illustration 1: base de cas minimale

J ‘ai par la suite inscris de la même manière les informations extraites du corpus WiCoPaCo.

Traitement du corpus WiCoPaCo

Étude du fichier

J'ai téléchargé le fichier XML et j'ai étudié sa forme. J'ai constaté qu'il était organisé en couple de phrase. Un couple étant composé d'une phrase initial et une phrase contenant une modification de la phrase initial. La phrase initial est placé dans une balise <*before*>. La phrase contenant la modification est placé dans une balise <*after*>. Ces deux balises sont encadrés par une balise <*modif*> contenant plusieurs attributs : un id de modification unique, un id de page caractérisant la page modifiée, un id de révision avant et un autre après, un id utilisateur correspondant à l'utilisateur qui a fait la modification et un commentaire. Toutes les balises <*modif*> sont incluse dans une balise <*modifs*>. Dans les balises <*before*> et <*after*>, on trouve une balise <*m*>. Cette dernière est accompagnées par un attribut exprimant le nombre de mot modifié et contenant ce ou ces mots.

Par exemple :

```
 <modif id="31117" wp_page_id="7965" wp_before_rev_id="4952629" wp_after_rev_id="4952665" wp_user_id="0"
wp_user_num_modif="612674" wp_comment="Janvier">
    <before>** Vatican : Durant la messe du Nouvel An , consacrée au thème de la paix , célébrée dans la
basilique Saint-Pierre de Rome , le pape Benoît une 16 a appelé l' ONU à une conscience renouvelée de ses
responsabilités pour promouvoir la justice , la solidarité et <m num_words="2">la paix</m> dans le
monde .</before>
    <after>** Vatican : Durant la messe du Nouvel An , consacrée au thème de la paix , célébrée dans la
basilique Saint-Pierre de Rome , le pape Benoît une 16 a appelé l' ONU à une conscience renouvelée de ses
responsabilités pour promouvoir la justice , la solidarité et <m num_words="2">l' apéro</m> dans le
monde .</after>
  </modif>
```

Illustration 2: exemple d'un couple

Choix du langage

N'ayant que très peu eu l'occasion d'extraire des données d'un fichier XML, j'ai fais quelques recherche sur internet afin de choisir un langage de programmation adapté. Pour le choix du langage, je me suis appuyé sur plusieurs critères : la disponibilité de bibliothèques permettant l'exploitation de fichier XML, la possibilité d'écrire facilement dans un fichier et les langages utilisé par messieurs LY et GIANG. De plus, monsieur Lieber m'avait orienté vers plusieurs langage, tels que le Java, le python ou le php. Je me suis donc tourné vers python 2 avec son accord. J'ai ensuite approfondie mes recherche afin d'utiliser ce langage de manière adapté. J'utilise la bibliothèque : xml.etree.ElementTree. Cette dernière permet de représenter le fichier XML sous forme d'arbre et d'extraire les informations voulu simplement.

Conception du programme

Afin de bien comprendre l'utilisation de la bibliothèque cité précédemment, j'ai créé un script de test, que j'ai lancé sur une partie réduite du corpus WiCoPaCo. J'ai d'abord écrit un script qui extrayait uniquement les couples contenues dans la balise <*modif*>. J'ai eu quelques difficultés lors de cette étape. En effet, lors de l'utilisation des fonctions existantes pour extraire le contenu des balise <*before*> et <*after*>, uniquement le contenu précédant la balise <*m*> était extrait. J'ai additionné à ce bout de chaîne de caractère le contenu de la balise <*m*>, mais il me manquais toujours la fin de la phrase. J'ai constaté qu'il existait une fonction permettant d'extraire les balises et leur contenu. En utilisant cette méthode, je me suis aperçu qu'en utilisant cette fonction sur la balise <*m*> j'obtenais aussi la fin de la phrase. Cela m'a permis de pouvoir ajouter les phrases complètes à la base de cas. Après discussion avec messieurs Guillaume, Lepage, Lieber et Nauer, nous avons convenu d'ajouter à la base de cas le ou les mots modifiés. Ils m'ont aussi demandé d'ajouter la source du cas, un indice indiquant le lieu de début de modification dans le ou les mots modifiés et une pertinence des cas qui pourra être utilisé par le moteur de recherche. Nous avons aussi convenu que le fichier créé serait au format csv afin de faciliter l'insertion des couples dans la base de données.

Illustration 3: Utilisation du logiciel sur l'ensemble du corpus

Filtrage des données

Lors de l'étude du corpus, j'ai constaté qu'il existait un certain nombre de cas ne traitant pas de correction de français. Afin de supprimer les cas inutiles pour le logiciel de correction de phrase et d'avoir un maximum de cas pertinent, j'ai approfondie l'étude du corpus afin d'avoir une idée plus précise des cas n'étant pas adapté. Je me suis d'abord demandé si il n'était pas possible d'utiliser les commentaire présent dans les attributs des balises <*modif*>. Par exemple :

```
  <modif id="2" wp_page_id="3" wp_before_rev_id="4529726" wp_after_rev_id="4847377" wp_user_id="7343"
wp_user_num_modif="319" wp_comment="ortho">
```

Illustration 4: exemple de commentaire

```
  <modif id="1" wp_page_id="3" wp_before_rev_id="1350936" wp_after_rev_id="1409789" wp_user_id="3763"
wp_user_num_modif="15342" wp_comment="correction de numéraux">
```

Illustration 5: exemple de commentaire

```
  <modif id="4" wp_page_id="3" wp_before_rev_id="18398158" wp_after_rev_id="19601285" wp_user_id="0"
wp_user_num_modif="612471" wp_comment="">
```

Illustration 6: exemple de commentaire

Ces commentaires n'étant pas normalisé, cela rend leur utilisation pour créer un filtre compliqué. De plus, beaucoup de commentaires sont vides. Je me suis donc mis à la recherche d'autres filtres possibles. Ce corpus répertoriant tous les types de modifications faites, il contient aussi les modifications successives revenant au cas initial. Par exemple :

```
<modif id="15" wp_page_id="7" wp_before_rev_id="1599401" wp_after_rev_id="1599406" wp_user_id="0"
wp_user_num_modif="612471" wp_comment="Quelques théorèmes">
    <before>Un théorème intéressant à l' époque des mémoires d' ordinateurs de <m num_words="1">petite</m>
taille était qu ' on pouvait travailler séparément sur des sous-ensembles (« blocs ») d' une matrice en
les combinant ensuite par les mêmes règles qu ' on utilise pour combiner des scalaires dans les
matrices . Avec les mémoires actuelles de plusieurs Go , cette question a perdu un peu de son intérêt
pratique , mais reste très prisée en théorie des nombres , pour la décomposition en produit de facteurs
premiers avec le crible général de corps de nombres ( GNFS ) ( méthode Lanczos par blocs )</before>
    <after>Un théorème intéressant à l' époque des mémoires d' ordinateurs de <m num_words="1">grd</m>
taille était qu ' on pouvait travailler séparément sur des sous-ensembles (« blocs ») d' une matrice en
les combinant ensuite par les mêmes règles qu ' on utilise pour combiner des scalaires dans les
matrices . Avec les mémoires actuelles de plusieurs Go , cette question a perdu un peu de son intérêt
pratique , mais reste très prisée en théorie des nombres , pour la décomposition en produit de facteurs
premiers avec le crible général de corps de nombres ( GNFS ) ( méthode Lanczos par blocs )</after>
  </modif>
  <modif id="16" wp_page_id="7" wp_before_rev_id="1599406" wp_after_rev_id="1648108" wp_user_id="0"
wp_user_num_modif="612471" wp_comment="Quelques théorèmes">
    <before>Un théorème intéressant à l' époque des mémoires d' ordinateurs de <m num_words="1">grd</m>
taille était qu ' on pouvait travailler séparément sur des sous-ensembles (« blocs ») d' une matrice en
les combinant ensuite par les mêmes règles qu ' on utilise pour combiner des scalaires dans les
matrices . Avec les mémoires actuelles de plusieurs Go , cette question a perdu un peu de son intérêt
pratique , mais reste très prisée en théorie des nombres , pour la décomposition en produit de facteurs
premiers avec le crible général de corps de nombres ( GNFS ) ( méthode Lanczos par blocs )</before>
    <after>Un théorème intéressant à l' époque des mémoires d' ordinateurs de <m num_words="1">petite</m>
taille était qu ' on pouvait travailler séparément sur des sous-ensembles (« blocs ») d' une matrice en
les combinant ensuite par les mêmes règles qu ' on utilise pour combiner des scalaires dans les
matrices . Avec les mémoires actuelles de plusieurs Go , cette question a perdu un peu de son intérêt
pratique , mais reste très prisée en théorie des nombres , pour la décomposition en produit de facteurs
premiers avec le crible général de corps de nombres ( GNFS ) ( méthode Lanczos par blocs )</after>
  </modif>
```

Illustration 7: exemple de modification non pertinante

Dans la première balise <*modif*> on peut voir que la modification concerne le mot *petite* qui est modifié en *grd*. Dans la seconde balise <*modif*>, nous voyons que le mot *grd* est à nouveau modifié en *petite* . Ces deux couples ne sont donc pas utiles. J'ai donc continué la lecture du corpus, afin de constater que ce n'étais pas un cas isolé, mais un cas récurant. J'ai cherché d'autre similitudes dans ces cas afin de facilité leurs suppressions. C'est à ce moment là que j'ai constaté que *wp_after_rev_id* de la première modification était égale à *wp_before_rev_id* de la seconde modification. J'ai aussi observé que ces cas n'était pas forcément successif dans le corpus. J'ai donc écrit une fonction comparant les attributs *wp_after_rev_id* et *wp_before_rev_id* dans les cas suivants. Cette fonction modifie directement le fichier XML. L'utilisation de cette fonction est très longue sur le corpus WiCoPaCo. Elle a cependant permit de retirer un certain nombre de cas inutiles.

```
30598 couple suppr
comment voulez vous nommer le fichier de sortie?
le fichier sera automatiquement suivi de '.csv'
exemple_filtre
file exemple_filtre.csv open
file exemple_filtre.csv close
-------
nombre d'element traité = 174766
```

Illustration 8: Utilisation du programme avec suppression de cas

Amélioration future possible

Avec du temps supplémentaire, des filtres supplémentaire pourront être ajouté, afin d’affiner la pertinence des couples de cas présent dans la base de données. On pourra aussi faire des statistiques sur les types de modifications présentes dans le corpus afin

Conclusion

Bibliographie