Support des langues étrangères

© Bruce Wilcox, mailto:gowilcox@gmail.com www.brilligunderstanding.com
Revision 11/04/2017 cs7.61

Traduction, Mathieu Rigard pour https://utopia-french.tech 15/03/2018

Présentation des langues étrangères

ChatScript vient nativement avec un support complet en anglais. Si vous voulez utiliser une langue différente, vous avez besoin d'une variété de choses.

- Support pos-tagging (facultatif si vous n'utilisez pas de mots-clés pos)
- Support de vérification orthographique
- Concepts dans la langue (que vous utilisez)
- Substitutions LIVEDATA appropriées à la langue
- Motifs dans la langue
- Sortie dans la langue
- Des Lemmes (avec des pos-taggers étrangers ou disponibles depuis ^pos(canonical))

ChatScript a un paramètre en ligne de commande *language=* (exemple language=FRENCH) qui indique à CS la langue que vous souhaitez utiliser. Il est par défaut en ANGLAIS. Les effets de ce paramètre sont les suivants.

- Si ce n'est pas en ANGLAIS, le pos-tagging interne (autre que le marquage des balises anglaises et des numéros et dates possibles) et l'analyse syntaxique sont désactivés.
- Si le treetagger est sous licence et possède cette langue, il posera un tag.
- Le système utilisera DICT/langue.
- Le système utilisera LIVEDATA/language
- Le compilateur de script compilera automatiquement les lignes marquées pour être compilées conditionnellement avec la langue (voir les commentaires sur la langue).
- Le système stockera également le fichier de sujet de l'utilisateur avec le suffixe de langue.

ENTRÉES et SORTIES

ChatScript prend en charge l’UTF8, ce qui permet la sortie ou les modèles dans votre langue. Idem pour LIVEDATA.

ChatScript prend en charge deux types de commentaires de compilation conditionnelle. Les commentaires sur une ligne ressemblent à ceci:

```
#ENGLISH cette ligne compilera si la langue est l'anglais mais pas si la langue est GERMAN.
#FRENCH cette ligne ne compilera pas si la langue est l'anglais mais compilera si la langue est FRENCH.
```

Comme toujours, ces commentaires vont jusqu'à la fin de la ligne. L'autre commentaire est le commentaire de bloc comme ceci:

```
##<<ENGLISH ces lignes seront compilées en anglais
jusqu'à un commentaire de bloc de fermeture normal ##>>
```

À l'aide de la compilation conditionnelle, vous pouvez faire cohabiter l'anglais et les autres versions de code si vous le souhaitez.

LE DICTIONNAIRE

Le fichier de dictionnaire peut être juste une liste de mots de la langue, un par ligne. Vous devez lister toutes les conjugaisons d'un mot car il n'y a pas de support intégré pour le comprendre. Vous pouvez également ajouter des balises POS équivalentes anglaises (voir les exemples dans les dictionnaires de langues étrangères existants) si vous souhaitez utiliser des mots-clés existants liés à des balises pos.

En plus des mots normaux, il y a un fichier LIVEDATA/.../numbers.txt qui indique, pour une langue, il s'agit d'un mot numérique et de ce que signifie la signification numérique.

`:buildforeign language` peut être utilisé pour reconstruire un dictionnaire étranger compte tenu des données rawwords dans le répertoire TreeTagger (que vous n'avez pas).

POS-TAGS AND LEMMES

Si vous voulez des valeurs de POS et des lemmes réels (forme canonique d'un mot), vous aurez besoin d'un POS-tagger quel qu'il soit ou utiliser ^pos(canonical) un mot. Bien qu'il soit possible de raccorder un tagueur externe via un appel Web, il sera nettement plus lent qu'un système intégré.

ChatScript prend en charge le système TreeTagger intégré, qui prend en charge un certain nombre de langues. Cependant, vous ne pouvez l'utiliser que si vous avez une licence commerciale. Vous pouvez l'essayer en utilisant `^popen,` Comme c'est le cas dans le bot allemand, il sera lent car il doit réinitialiser TreeTagger pour chaque phrase. Le système intégré ne fonctionne pas. Une licence (par langue) est d'environ 1000 $ pour une utilisation universelle à vie. Vous pouvez me contacter si vous voulez vous organiser pour l'utiliser.

Exécuter CS dans une langue étrangère

Pour s'exécuter dans une autre langue, utilisez le paramètre de ligne de commande *language=*. Pour toute instance CS, vous ne pouvez prendre en charge qu'une seule langue à la fois (car CS ne peut pas charger plusieurs dictionnaires ou détecter automatiquement une langue). Donc, si vous voulez avoir des serveurs pour plusieurs langues, vous avez besoin de plusieurs serveurs, chacun étant lancé de manière appropriée.

Le fichier de sujet de l'utilisateur est nommé par username-botname-language (où en anglais on omet le composant -language). Cela signifie qu'un utilisateur discutant avec un serveur CS allemand a une conversation indépendante avec n'importe quelle conversation qu'il a eue avec le serveur anglais. Si vous utilisez des fichiers LTM pour stocker des données à long terme sur l'utilisateur, vous pouvez choisir si ces données sont uniques par langue ou partagées, puisque le nom du fichier est écrit dans un script.

Traduire des concepts

Voici un code intégré pour traduire les concepts à l'aide de Google Traduction. Cela nécessite que vous ayez une clé d'api pour Google Traduction (mais vous pouvez vous inscrire gratuitement et obtenir un crédit de 300 $ valable pendant 3 mois, ce qui est suffisant pour faire tout votre travail de traduction probablement). Vous dites à CS ceci comme un paramètre de ligne de commande:

```
apikey=AIzaSyAxxxx
```

Quand je veux traduire tous les concepts de niveau 0, je fais ce qui suit:

1. effacer le contenu du dossier TOPIC
2. `:build 0`
3. exécutez CS en utilisant le paramètre de ligne de commande noboot et votre apikey
4. `:sortconcept x`
5. `:translateconcept german myfilename`

Si vous lancez ^csboot et cela génère de nouvelles données conceptuelles, alors vous avez besoin de `noboot`, sinon cela n'a pas d'importance.

`:sortconcept x` localise tous les concepts actuellement définis (donc juste: build 0) et les écrit dans un fichier de niveau supérieur nommé `concepts.top` avec un concept par ligne. Ce fichier sera lu à l'étape suivante.

`:translateconcept` utilise l'apikey. Il lit chaque ligne de concepts.top (1 ligne par concept) et appelle google translate pour la langue que vous avez nommée, en enregistrant les résultats dans le chemin / fichier que vous avez donné. Actuellement, cela ne reconnaît que les noms de langue suivants: allemand, français, italien, espagnol, russe, hindi. Je pourrais en ajouter plus si nécessaire.

The resulting file will automatically prepend each line with conditional compile markers for the language you named, so you can directly add it to your bot and it will only compile when you are in that language mode.

If you want to translate concepts from your bot, then do the following:

1. erase the contents of TOPIC folder
2. `:build harry` (or whatever your bot is)
3. run CS using command line parameter `noboot` and your apikey
4. `:sortconcept x`
5. `:translateconcept french myfilename`

If you just want to translate a single concept/topic then you can call

```
:translateconcept ~myconcept french myfilename
```

It will, as a byproduct, provide the sorted english form of the concept on a single line in `cset.txt`. If you dont give a language and filename, then it will just sort your english concept and write it out.