ttda - une librairie R pour l'analyse de données textuelles

Jean-Pierre Müller

Université de Lausanne – Faculté des Sciences Sociales et Politiques – BFSH #2
1015 Lausanne – Suisse
jean-pierre.mueller@dssp.unit.ch

Abstract

This contribution shows how it is possible to use "GPL" or "free" software to realize statistical analysis of textual data. R has been used as a base, as it becomes a software reference in the statistical community. A free solution was also found for lemmatization.

Résumé

Comment utiliser des logiciels avec des licences de type « GNU Public Licence » ou « libres » pour réaliser des analyses de données textuelles ? R devenant un logiciel statistique de référence, c'est naturellement qu'il a été utilisé comme élément intégrateur par la création d'un module spécifique pour le traitement de données textuelles. Une solution a été aussi trouvée pour la lemmatisation des formes. Les traitements statistiques proprement dits sont alors réalisés avec des modules connus de R.

Mots-clés : R, ispell, ttda.

1. Introduction

1.1. *Motivation*

Les logiciels d'analyse de données textuelles sont payants, pour une majorité, et sont donc difficiles à mettre à disposition des étudiants pour des travaux pratiques ou des mémoires de fin d'études. Par ailleurs, les étudiants et chercheurs utilisent de nombreux systèmes d'exploitation ; hors pour certains de ces derniers, il est extrêmement difficile de trouver une solution pour le traitement statistique de textes.

Cette contribution évoque les différents éléments ayant permis de développer une librairie de routines pour le logiciel R, d'abord pour MacOS X, puis pour les systèmes linux.

1.2. *Autres logiciels*

Il existe de nombreux logiciels qui permettent de traiter des données textuelles : SPAD-T (édité par DECISIA), Lexico (réalisé par l'équipe universitaire SYLED-CLA2T), Alceste (édité par IMAGE), Tropes (édité par Acetic), Hyperbase (Institut de Linguistique Française, UFR Lettres, Arts et Sciences Humaines, Nice), Sphinx (Le Sphinx Développement). Souvent dotés d'une interface agréable, ils autorisent de nombreux traitements et introduisent des développements spécifiques (lemmatisation, méthodes factorielles et classifications, représentations diverses du corpus). Si ces logiciels sont tous disponibles pour la plateforme Windows-Intel, seuls trois fonctionnent sous Mac OS 9 (Lexico, Alceste et Hyperbase), aucun nativement sous Mac OS X (bien qu'une version d'Alceste soit annoncée) ou Linux.

1.3. *Cahier des charges*

Le cahier des charges était relativement sommaire :

– Implémenter ou avoir accès à une partie des techniques décrites dans un ouvrage de référence comme celui de Lebart et Salem (1994).

– Traiter un corpus se présentant sous la forme d'un fichier unique, les uci (unités de contexte initial ou parties) étant marquées de manière usuelle.

– Évaluer la solution choisie et ses possibilités d'évolution.

Les choix d'implémentation ont été les suivants :

– La balise `<uci=nombre>` est utilisée pour identifier le début d'une uci dans le corpus.

– La balise `<par>` peut être utilisée pour identifier le début d'un paragraphe.

– Les mots débutant par une étoile `*` (mots étoilés) sont considérés comme des marques de variables supplémentaires. Ils peuvent être placés partout dans l'uci.

– Les accolades peuvent être utilisées sur la même ligne pour inclure des commentaires dans le corpus : `{ commentaire }`.

– Les balises doivent être séparées par des espaces ; ainsi la chaîne `<uci=3><par>` ne sera pas reconnue comme un marquage valide.

2. R comme élément intégrateur

R (Ihaka et Gentleman, 1996) est désormais un logiciel largement connu dans la communauté statistique. Reprenant la syntaxe de S, il permet de réaliser de nombreuses analyses et d'écrire des procédures spécifiques pour certains traitements. Il est librement disponible pour de nombreuses plateformes (http://cran.r-project.org/).

Parmi les autres raisons prépondérantes ayant entraîné son choix comme élément intégrateur, il faut citer :

– La disponibilité de nombreux modules d'analyse de données (`ADE-4` ou `multiv`, par exemple) pour le traitement des tableaux lexicaux.

– `grep` est intégré à R, ce qui permet des traitements efficaces des chaînes de caractères.

– Bien que R soit interprété, de nombreuses fonctions sont implémentées en C ou en Fortran, ce qui lui permet d'avoir une bonne efficacité générale.

2.1. *Lecture du corpus*

La lecture du fichier du corpus est réalisée à l'aide de la fonction :

```
ttda.get.text(textfile = "")
```

Si le paramètre `textfile` est vide, un dialogue de sélection de fichier est présenté à l'écran. Cette fonction retourne un vecteur type (chaîne de) `character` contenant les lignes du fichier où une balise `<line>` a été ajouté au début de chaque ligne. Cette fonction utilise la fonction native `readLines` de R.

2.2. *Segmentation*

La segmentation du corpus en occurrences est réalisée par la fonction :

```
ttda.segmentation(textlines,
   separators = ttda.util.separators(),
   with.hyphen = TRUE,
   upper8 = ttda.util.get.modifier()[1],
   lower8 = ttda.util.get.modifier()[2])
```

Elle retourne un vecteur de type (chaîne de) `character` contenant les occurrences et les balises du corpus. Les séparateurs utilisés par défaut sont ceux retournés par `ttda.util.separators`, dont le résultat dépend du système d'exploitation de l'ordinateur utilisé. Le paramètre `with.hyphen` précise comment traiter les tirets. Si `TRUE` est utilisé, toute suite de tirets, précédée ou suivie d'espaces, sera considérée comme caractères séparateurs. Cette fonction remplace les caractères majuscules par les minuscules correspondantes. Le remplacement des caractères majuscules accentués en minuscules accentuées est réalisé par les paramètres `upper8` et `lower8`. Par défaut, les résultats de la fonction `ttda.util.get.modifier`, qui donne la table de conversion selon l'OS, sont utilisés pour cette transformation.

La fonction native `strsplit` est utilisée pour réaliser la segmentation et la fonction `chartr` pour le passage des majuscules aux minuscules.

2.3. *Création du dataframe du corpus*

Ce tableau de données contient un enregistrement par occurrence du corpus. Cinq variables décrivent l'occurrence :

`uci` : le numéro de l'uci qui contient l'occurrence.

`uce` : le numéro de l'uce (partitions des uci en phrases) qui contient l'occurrence ou 0 si elles n'ont pas été calculées par `ttda.uce`.

`par` : Le paragraphe qui contient l'occurrence, 0 si ils n'ont pas été indiqués dans le corpus.

`line` : la ligne qui contient l'occurrence.

`graphical.forms` : l'occurrence, comme valeur d'un vecteur de type `factor`. Les `levels` de cette variable constituent un dictionnaire des formes du corpus.

Cette transformation est réalisée par la fonction :

```
ttda.forms.frame(occurences, remove.keys = TRUE)
```

à partir du résultat de la fonction `ttda.segmentation`. Le paramètre `remove.keys` précise si les balises doivent être conservées comme occurrences dans le tableau de données.

A défaut d'avoir trouvé une méthode plus élégante, cette transformation utilise des boucles sur les positions des balises.

3. Lemmatisation

La lemmatisation du texte est réalisée par `ISPELL`. `ISPELL` est un dictionnaire orthographique disponible pour les plateformes *nix et linux (attention : dans certaines distributions linux, `ISPELL` n'est qu'un script qui appelle le correcteur orthographique `aspell`). Une implémentation pour MS Windows existe, mais n'a pas été testée. `ISPELL` est couvert par la licence de l'Université de Berkeley, il est disponible sur Internet à l'adresse : http://fmg-www.cs.ucla.edu/fmg-members/geoff/ispell.html. Il commence par comparer le mot à corriger aux entrées de son dictionnaire, puis, si le mot n'y apparaît pas, combine celles-ci

avec une liste de préfixes et de suffixes. De nombreux dictionnaires et listes d'affixes sont disponibles pour les langues européennes. Une description très complète du fonctionnement d'ISPELL est donnée par Pythoud (1996).

Lemmatisation par ISPELL

`ISPELL` peut être mis sur un mode de fonctionnement particulier à l'aide du drapeau `-a`, ce qui permet de l'utiliser depuis d'autres programmes. Dans ce mode, pour chaque mot soumis, `ISPELL` retourne une ligne ayant l'un des formats suivants :

`*` : si le mot soumis a été trouvé dans le dictionnaire ; `+ <lemme>` : si le mot soumis peut être déduit de <lemme> en utilisant les règles d'affixation ; ou des lignes plus complexes débutant par les caractères `-`, `&`, `?`, `#`, si un mot composé, un ou des mots proches ou pouvant être déduits illégalement du dictionnaire ont été trouvés ou si rien n´a été trouvé.

La fonction :

`ttda.lemmatisation(forms, ...)`

réalise la lemmatisation en appelant `ISPELL` et en traitant de manière adaptée le résultat retourné par le correcteur orthographique. Il est conseillé de la placer dans une commande `transform` qui agit directement sur le dataframe.

Les paramètres suivants de `ttda.lemmatisation` précisent l'utilisation d'`ISPELL` sur la machine utilisée :

`ispell.enc` est l'encodage des caractères utilisé par `ISPELL`. Utiliser `ISOLatin1` sur Macintosh. La valeur par défaut `native.enc` devrait fonctionner pour les autres OS.

`dict` spécifie le dictionnaire à utiliser. Si le paramètre n'est pas renseigné, l'anglais est utilisé. Donner la valeur `francais` pour utiliser le dictionnaire français. Pour d'autres dictionnaires, consulter le site web de `ISPELL`.

`ispell.path` Le chemin d'accès à `ISPELL`. La valeur par defaut `/sw/bin/ispell` est valable pour l'installation faite par `fink` de `ISPELL` sous MacOS X.

4. Procédures spécifiques à l'analyse de données textuelles

4.1. *Fonctions utilitaires*

Il est conseillé de les effectuer dans une commande `transform` qui agit directement sur le dataframe.

La réduction à des formes non accentuées est réalisée par la fonction :
`ttda.remove.modifier(forms)`.

`ttda.match(forms, old.levels, new.levels)` vous permet de re-coder des formes manuellement.

4.2. *Outils d'analyse du dataframe du corpus*

La fréquence des formes peut être obtenue à l'aide de la commande R : `summary()`.

La fonction `ttda.hapax(forms)` permet d'obtenir la liste des hapax.

La fonction `ttda.pareto()` permet d'obtenir des graphiques des proportions des formes (ou occurences) selon leurs fréquences.

La fonction

```
ttda.concordances(forms, theform, uci, indices = NULL,
     n.before = 3, n.after = 3, sort = "text", out = "vector")
```

permet d'obtenir les concordances des formes spécifiées en paramètre.

4.3. *Création du tableau lexical*

La fonction `ttda.TLE(uc, forms, uc.names = NULL)` permet de construire un tableau lexical à partir du dataframe du corpus. Le paramètre `uc.names` permet de nommer les parties dans le tableau retourné.

4.4. *Outils d'analyse du tableau lexical*

Les deux fonctions :

```
ttda.get.suplvar(tle)
```

```
ttda.remove.suplvar(tle)
```

agissent sur le tableau lexical. La première permet d'obtenir une copie des colonnes du tableau lexical provenant de mots étoilés (variables illustratives). La seconde permet de supprimer ces mêmes colonnes du tableau lexical.

Les trois fonctions

`ttda.f.speci.exact()`, `ttda.speci.pos()`, `ttda.speci.neg()`

permettent d'étudier les formes spécifiques d'une partie, en particulier d'une uc. La première fonction permet de calculer la probabilité d'avoir un effectif supérieur ou inférieur à celui observé, les deux dernières d'extraire du tableau des probabilités les formes de spécificité négative ou positive.

R fournit tous les outils nécessaires pour faire des AFC, ACM, et autres techniques statistiques, en particulier les packages : `ade4`, `multiv`, `vegan`, `tree`, `rpart`, `mva`, `class`, `cluster`.

5. Résultats et limitations

La qualité de la lemmatisation dépend très fortement du dictionnaire utilisé ou simplement disponible. Le dictionnaire et le fichier d'affixes de `Francais-GUTenberg`, également développés par Pythoud (1999) donnent de bons résultats sur des textes français tout-venant. La lemmatisation réalisée vous surprendra peut-être, les lemmes étant parfois de forme féminine, `directeurs` étant lemmatisé `directrice`.

Deux problèmes sont toutefois posés par `Francais-GUTenberg` :

1) La lemmatisation des verbes n'est pas parfaite, car il utilise au moins cinq lemmes par verbe : les premières personnes du singulier du présent, de l'imparfait, du passé simple et du futur de l'indicatif, ainsi que l'infinitif. La solution pourrait être la création d'un fichier de congruences de ces lemmes.

2) Le problème de l'élision n'est pas réglé de manière satisfaisante dans la version actuelle de la librairie. Les formes élidées ne sont pas reconstituées par `ISPELL` si l'apostrophe est traitée comme caractère séparateur dans `ttda.segmentation` et, sinon, la forme élidée est perdue ou non traitée par `ISPELL` (par exemple, `s'attendre` est lemmatisé `attendre`,

`l'âme` comme `âme,` mais `j'assaillisse` est considéré comme un lemme valide)[1].

La taille du corpus pouvant être traité dépend de la mémoire disponible, toutes les opérations étant réalisées en mémoire vive. De nombreuses méthodes ne sont pas encore intégrées: segments répétés et unités caractéristiques, par exemple.

La relative simplicité de la programmation dans R permet d'envisager l'adaptation rapide de ces techniques dans `ttda`. L'existence de librairies permettant d'accéder à des données stockées dans des bases de données SQL permet d'envisager des procédures de traitement de corpus importants et les fonctions d'accès à Internet d'utiliser des données extraites de pages HTTP.

6. Disponibilité de ttda et exemples

La librairie `ttda` est disponible à l'adresse http://wwwpeople.unil.ch/jean-pierre.mueller/. Deux corpus sont fournis avec la librairie. `anmat.txt` est constitué de 330 annonces matrimoniales parues entre 1960 et 1990. `fed_paper.txt` contient les 85 essais dus à Alexander Hamilton, John Jay, ou James Madison parus d'octobre 1787 à mai 1788, dont le but était de convaincre les habitants de New-York d'accepter la future constitution américaine. Ils sont connus sous le nom de « The Federalist papers ». Des exemples complets de traitement de ces fichiers de données sont donnés dans le manuel français de la librairie.

Références

Ihaka R. et Gentleman R. (1996). R: A language for data analysis and graphics. *Journal of Computational and Graphical Statistics*, vol. (5/3) : 299–314.

Lebart L. et Salem A. (1994). *Statistique textuelle*. Dunod.

Pythoud C. (1996). *Problèmes de la correction automatique de l'orthographe lexicale du français à travers une étude de cas: le correcteur orthographique Ispell et le dictionnaire Français IREQ.* Mémoire de licence, Section de linguistique, Université de Lausanne, http://www2.unil.ch/ling/cp/Francais-GUTenberg-v1.0.tar.gz

Pythoud C. (1999). *Francais-GUTenberg v1.0 - Manuel de référence*. http://www2.unil.ch/ling/cp/reference.pdf

[1] Entre deux maux, vous êtes donc invités à choisir celui qui vous gêne le moins !