Définition d'un système d'alignement SMS/français standard à l'aide d'un filtre de composition.

Richard Beaufort[1], Sophie Roekhaut[1], Cédrick Fairon[2]

[1]Multitel – Mons – Belgique

[2]CENTAL – UCL – Louvain-la-Neuve – Belgique

Abstract

The development of communication technologies has contributed to the emergence of new means of written communication which have been subject to many observations and studies. The CENTAL recently looked into one of these communication means, i.e. text messaging (SMS, textos), and gathered a corpus of about 75,000 SMS. Such a corpus enables to conduct empirical studies of the "SMS language". In order to facilitate the study of this corpus and to highlight SMS language singularities, a subset of 30,000 SMS has been manually translated into "standardized" French. This article consists of presenting the automatic method that was used for aligning each raw SMS message in its original spelling along with its transcription in standard French. This alignment is a prerequisite to further studies on SMS language vocabulary and linguistic singularities. The alignment system is based on Finite-State technologies. SMS and their translation in standard French are actually represented by Finite-State Machines (FSMs) that we align by the means of a composition filter. In the first part of this paper, we describe the formalism and the rewrite rules compiler (Ovide) which enabled the description of the filter. Afterwards, the second part, will consist, on the one hand, of laying out the type of rules used by the filter, as well as some examples of resulting alignments, and on the other hand, of evaluating the system performances. Finally, the last part, will be made up of various possible uses of the aligned corpus.

Résumé

Le développement des technologies de la communication a contribué à l'émergence de nouvelles formes de communications écrites dont le fonctionnement fait l'objet d'observations et d'études afin de parvenir à une description. Le CENTAL s'est récemment intéressé à une de ces formes de communication, le SMS (ou texto), et a récolté un corpus de 75 000 SMS dans le but de décrire objectivement le comportement du « langage SMS ». Pour pouvoir étudier ce corpus et mettre en évidence les particularités de ce langage, 30 000 SMS du corpus ont été traduits en français standard. L'article présente une méthode d'alignement automatique entre le message en langage SMS et sa traduction en français standard. Cet alignement est un préalable à des études ultérieures sur le vocabulaire et les particularités linguistiques du langage SMS. Le système d'alignement se fonde sur les technologies à états finis. Le SMS et sa traduction en français standard sont représentés sous la forme de machines à états finis (FSMs), que nous alignons au travers d'un filtre de composition. Dans un premier temps, nous décrivons le formalisme utilisé pour représenter le filtre (FSM) et le compilateur de règles de réécriture (Ovide) qui a permis la description de ce filtre. Ensuite, nous montrons le type de règles utilisées par le filtre ainsi que quelques exemples d'alignement de SMS et nous évaluons les performances du système. Enfin, nous terminons par un survol des exploitations possibles à partir des résultats obtenus par l'alignement automatique.

Mots-clés : SMS (texto), alignement automatique, Machines à états finis (FSM), composition filtrée.

1. Introduction

L'université catholique de Louvain a mené en décembre 2004 une vaste opération « Faites don de vos SMS à la science ». A cette occasion, un corpus de 75 000 SMS, écrits par quelques 3 200 participants, a été récolté (Fairon et al., 2006). Parmi les SMS récoltés, 30 000 ont été transcrits manuellement. Le corpus, disponible sur CD-Rom, est à la base de nombreuses études. L'engouement, à la fois des chercheurs et des utilisateurs, pour cette

nouvelle forme de communication écrite, a été démontré par le nombre important d'articles de presse parus sur le sujet et le succès d'outils comme le traducteur français-SMS disponible sur le Net[1]. Cet engouement a encouragé les chercheurs à poursuivre leurs études et à travailler à l'automatisation de certaines tâches, en vue d'améliorer l'accessibilité et l'exploitation des données du corpus. L'alignement du SMS avec sa traduction en français standard est une première étape d'automatisation. Dans (Fairon et al., 2006), les auteurs avaient mis en avant la difficulté d'exploiter les données pour obtenir des statistiques précises à cause des problèmes posés par l'alignement SMS/français standard:

> […] le comptage des formes par un programme informatique présuppose le découpage du texte en unités graphiques (tokens), ce qui est loin d'être évident dans le cas des SMS : plusieurs mots ou abréviations pouvant être agglutinés en une chaîne de caractères qu'il est souvent difficile de segmenter automatiquement.

Cet article propose une méthode originale d'alignement entre un SMS et sa traduction en langage naturel.

Définition (Alignement). *Etablissement de la correspondance maximale entre deux séquences, en déterminant leurs sous-séquences identiques ou similaires, ainsi que leurs sous-séquences propres.*

La notion de similarité entre deux sous-séquences est fortement dépendante du domaine. Dans le contexte de la langue, il peut par exemple s'agir d'une similarité phonétique (*cé = c'est*) ou de l'identification d'une abréviation (*svt = souvent*).

Une sous-séquence propre est une sous-séquence qui n'apparaît que dans l'une des deux séquences comparées. Elle n'a donc pu être considérée comme similaire à une autre sous-séquence, en raison des limites du domaine concerné.

De manière générale, plusieurs alignements sont acceptables pour deux séquences données.

```
Exemple:

sk          (SMS)
est-ce que  (TRAD)

Alignements possibles:

s_____k__   (ALIGN1)
_s____k__   (ALIGN2)
s___k____   (ALIGN3)
_s____k__   (ALIGN4)
____s__k__  (ALIGN5)
est-ce que  (TRAD)
```

Figure 1 : alignements possibles pour la chaîne de caractère "est-ce que"

Cependant, *dans le contexte d'une application donnée*, l'un de ces alignements est préférable, parce qu'il correspond à la *distance d'édition minimale* entre les deux séquences.

L'article est organisé comme suit. Nous commençons par présenter la notion de *distance d'édition*, et l'implémentation classique qui en est faite, avant de relever les inconvénients de la méthode, qui la rendent inutilisable dans le cadre de l'alignement entre un SMS et sa traduction. Nous proposons ensuite une modélisation de la distance d'édition à l'aide de

[1] Le traducteur français/sms peut être consulté sur le site http://cental.fltr.ucl.ac.be/demo/index.php?service=1

machines à états finis (FSMs). Après un bref rappel sur les machines à états finis, nous présentons les avantages du calcul de distance d'édition à l'aide de FSMs. Nous définissons ensuite les outils utilisés, qui permettent de calculer un alignement automatique à l'aide d'un filtre de composition. Le filtre peut être aisément modifié et recalculé, ce qui le rend utilisable pour d'autres types de tâches d'alignement. Dans un dernier point, nous proposons quelques résultats d'alignement et des statistiques obtenues à partir de l'alignement automatique du corpus ainsi que les possibilités que nous envisageons pour améliorer les résultats.

2. La distance d'édition : définition et représentation classique

La distance d'édition est nommée *distance de Damerau-Levenshtein*, du nom de ses auteurs (Damerau, 1964 ; Levenshtein, 1966), mais est plus connue sous le nom de *distance de Levenshtein*.

2.1. Définition de la distance de Levenshtein

La distance d'édition entre deux séquences X et Y mesure le nombre minimum d'opérations d'édition nécessaires pour convertir X en Y. Les opérations d'édition standard sont la substitution (*élèbe* pour *élève*), l'insertion (*élèbve* pour *élève*) et la suppression (*élèe* pour *élève*) d'un symbole, auxquelles certains systèmes ajoutent la transposition de deux symboles adjacents (*élvèe* pour *élève*). En somme, la distance d'édition, dans son acception de base, modélise les erreurs classiques qui se produisent lorsque l'on entre un texte au clavier. Généralement, chaque opération d'édition vaut 1. Le meilleur alignement est celui qui nécessite un nombre d'opérations d'édition minimal.

2.2. Inconvénients de l'implémentation classique

Wagner (Wagner, 1974) a été le premier à proposer de résoudre la distance d'édition au moyen de la programmation dynamique, à l'aide du célèbre algorithme de Viterbi (Viterbi, 1967).

Cependant, l'implémentation classique du calcul de distance d'édition pose deux inconvénients majeurs qui la rendent difficilement utilisable pour l'alignement SMS/français standard. Premièrement, il est nécessaire de recalculer la distance d'édition entre chaque mot et tous les mots d'un dictionnaire standard. Ce caractère éminemment séquentiel rend cette distance inutilisable en pratique, dès que la taille du dictionnaire atteint quelques milliers de formes. Or, un dictionnaire un tant soit peu complet compte plusieurs centaines de milliers de formes. Deuxièmement, la modélisation de la distance d'édition par programmation dynamique travaille par mots. Or, la problématique de l'alignement du SMS est impossible à résoudre en terme de mots. En effet, les nombreux phénomènes d'agglutination observés dans les SMS exigent un traitement sur l'ensemble du message plutôt qu'un traitement par mot. L'algorithme est donc difficilement utilisable pour résoudre ce problème.

3. Distance d'édition modélisée par machines à états finis

Le lecteur qui ne maîtriserait pas les concepts de base des machines à états finis est invité à se reporter à l'état de l'art dans le domaine (Roche and Schabes, 1997). Pour une plus grande clarté de l'exposé cependant, nous rappelons brièvement ce que sont les machines à états finis, avant de présenter les modèles proposés pour modéliser la distance d'édition au moyen de machines à états finis.

3.1. Les machines à états finis : définition

Une machine à états finis peut être considérée comme un graphe orienté étiqueté pourvu d'un état initial et d'un ou de plusieurs états finaux. Tandis que les transitions d'un automate (FSA) sont étiquetées par un seul symbole appartenant à un alphabet $\sum$, les transitions d'un transducteur (FST) sont étiquetées par deux symboles, l'un appartenant à l'alphabet d'entrée $\sum_1$ et l'autre, à l'alphabet de sortie $\sum_2$. Les transitions des machines pondérées (WFSA, WFST), sont quant à elles augmentées de poids.

Une séquence de transitions étiquetant un chemin allant de l'état initial à un état final est un *chemin à succès*. L'ensemble des chemins à succès d'un FSM est son *langage*. Dans ce contexte, l'automate est considéré comme un *accepteur*, qui détermine si une séquence appartient ou non au langage qu'il modélise, tandis que le transducteur définit une *relation* entre des paires de séquences, et une *transduction* d'un langage d'entrée vers un langage de sortie. Il a été démontré que les langages acceptés par les FSMs sont les *langages réguliers*, générés par les grammaires régulières de la hiérarchie de Chomsky (Chomsky, 1956).

Certaines propriétés de clôture des FSMs rendent ces outils flexibles, puissants et efficaces. Parmi celles-ci, la *composition* des transducteurs, qui généralise l'intersection des automates : étant donné deux transducteurs T_1 et T_2, la composition ($T_1 \circ T_2$) implique que le langage de sortie de T_1 et le langage d'entrée de T_2 soient définis sur le même alphabet, de manière à pouvoir calculer leur intersection. S'il y a intersection, la composition construit un transducteur T_3 qui met en relation le langage d'entrée de T_1 et le langage de sortie de T_2, réduits à l'intersection calculée. La composition est donc l'opération qui permet de combiner différents niveaux de représentation pour construire des relations complexes à partir de relations simples.

Enfin, les poids présents sur les transitions des machines pondérées rendent ces machines tout à fait appropriées pour résoudre des problèmes du type *recherche du meilleur chemin* dans un graphe de solutions. Dans la suite de cet article, la recherche du meilleur chemin d'un graphe G sera notée

$$Best(G)$$

3.2. Modélisation de la distance d'édition à l'aide de machines à états finis

Trois méthodes ont été proposées pour calculer la distance d'édition à l'aide de machines à états finis.

Les deux premières (Oflazer, 1996 ; Schulz and Mihov, 2002), très efficaces, résolvent les deux problèmes de la distance d'édition classique que nous avons précédemment relevés. Ces deux méthodes présentent cependant un inconvénient de taille : le nombre d'opérations d'édition est « précompilé » dans les deux cas, et ne dépend en aucune manière de la longueur de la séquence d'entrée. Ces méthodes manquent donc de flexibilité.

La troisième méthode (Mohri, 2003) offre cette flexibilité. Mohri propose de représenter les séquences à comparer sous la forme de deux automates A et B, et de placer entre ces deux automates un transducteur F qui représente les opérations d'édition autorisées, et agit comme un *filtre* entre A et B. Ces trois FSMs sont composés ensemble :

$$R = A \circ F \circ B$$

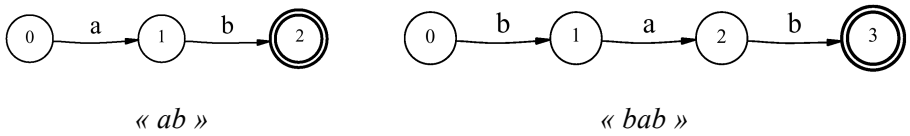
où R, le résultat de la composition, est un transducteur dont les chemins représentent toutes les suites d'opérations d'édition valides permettant de convertir la séquence d'entrée

représentée par A en la séquence de sortie représentée par B. La distance d'édition minimale D_{min} entre les séquences est dès lors simplement le meilleur chemin de R :

$$D_{min}(A,B) = Best(R) = Best(A \circ F \circ B)$$

Etant donné que la composition entre les deux automates est réalisée au travers d'un filtre, nous parlons de *composition filtrée*.

Admettons que la séquence d'entrée soit *« ab »*, et la séquence de sortie soit *« bab »*. La figure 2 montre comment se présentent ces séquences sous la forme d'automates.



« ab » *« bab »*

Figure 2 : automates pour les séquences ab *et* bab

Le principal apport de cette approche est que le transducteur *F* peut être facilement construit à partir de règles de réécriture qui décrivent les opérations d'édition autorisées. Le transducteur F_1, qui réalise la distance d'édition classique où toute opération vaut 1, est illustré par la figure 3.

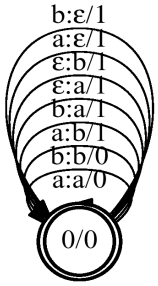


Figure 3 : transducteur F_1 défini sur l'alphabet {*a*,*b*}

La méthode d'alignement SMS/français standard que nous proposons est basée sur cette notion de *composition filtrée*. La mise en œuvre de cette méthode d'alignement a été possible grâce à la bibliothèque de machines à états finis et au compilateur de règles de réécriture dont nous disposons.

4. Présentation des outils utilisés

Les machines à états finis que nous créons et que nous manipulons dans le cadre de l'alignement SMS/français standard s'appuient sur les outils à états finis dont nous disposons : une bibliothèque de machines à états finis et un compilateur d'expressions régulières et de règles de réécriture.

4.1. La bibliothèque de machines à états finis

Nous utilisons une bibliothèque de machines à états finis, décrite dans (Beaufort, 2006), qui permet la construction, la manipulation, le chargement et la sauvegarde d'automates et de transducteurs, pondérés et non pondérés. La plupart des opérations définies sur les machines à états finis sont disponibles dans cette bibliothèque : déterminisation, minimisation, projection, inverse, complément, composition, recherche du meilleur chemin, etc.

4.2. Le compilateur Ovide

Ovide est un compilateur qui convertit des expressions régulières et des règles de réécriture, pondérées ou non, en machines à états finis équivalentes. Ce compilateur est présenté *in extenso* dans (Beaufort, 2007). Cependant, pour la clarté de l'exposé, en voici les caractéristiques principales.

Les fichiers interprétés par le compilateur *Ovide* sont divisés en différentes sections, illustrées par la figure 4.

```
[INFO]
ALPHAIN = ASCII

[CLASSIN]
EPS         \x00                                    #symbole vide
VOW         [aeiouyàáâãäåæçèéêëìíîïñòôöøùûüýîõ]    #voyelles
CONS        [bcdfghjklmnpqrstvwxz]                  #consonnes
MIN         [<VOW><CONS>]                           #minuscules
INT         [0-9]                                   #chiffres
PUNCT       [\@\'\(\-\)\{\[\]\}\,;:\!\?\.]          #ponctuations
SPACE       \s                                      #espace
SYMBOL      [&_=\+~\#\|`\\\^\@\$¨£\*\%µ\/§]         #symboles
OTHER       [¨ÃÂÈÎÏÙíðü]                            #caractères spécifiques au français standard
SMS         [<VOW><CONS><MIN><INT><PUNCT><SPACE><SYMBOL>]  #caractères SMS
LN          [<SMS><OTHER>]                          #caractères français standard

[LANGIN]
<SMS>*

[RULE]
<INT>|<SYMBOL> ?→ <MIN> /2
<PUNCT>  ?→ <PUNCT> /1

[LANGOUT]
<LN>*
```

Figure 4 : exemple de fichier Ovide

La section [INFO] précise les informations générales du fichier. Dans cette section, on trouve le nom de l'alphabet d'entrée (ALPHAIN) et éventuellement, s'il est différent, le nom de l'alphabet de sortie (ALPHAOUT). Dans notre exemple, le SMS et sa traduction travaillent sur le même alphabet, l'alphabet ASCII. Seul ALPHAIN est donc défini.

La section [CLASSIN] permet la définition de classes, qui sont des expressions régulières définies sur un alphabet. Lorsque deux alphabets sont définis, les classes de cette section ne concernent que l'alphabet d'entrée, et des classes spécifiques à l'alphabet de sortie doivent être définies dans une section [CLASSOUT]. Ce n'est pas le cas dans notre exemple. Une classe peut prendre la place de l'expression à laquelle elle correspond, soit dans la définition d'une autre classe, soit dans une autre section du fichier.

Les sections [LANGIN] et [LANGOUT] permettent de décrire des langages sous la forme d'expressions régulières. Ces langages peuvent être employés seuls, ou en combinaison avec des règles de réécriture. La section [LANGIN] définit le langage d'entrée de la machine, et la section [LANGOUT], son langage de sortie. Dans l'exemple ci-dessus, le langage d'entrée est restreint à une suite de 0, 1 ou plusieurs caractères de la classe <SMS> et le langage de sortie est restreint à une suite de 0, 1 ou plusieurs caractères de la classe <LN>.

La section [RULE] est dédiée aux règles de réécriture. Les règles de réécriture indiquent que la partie gauche de la règle (définie sur l'alphabet d'entrée) peut être transformée par la partie droite de la règle (définie sur l'alphabet de sortie). Il est également possible de contraindre l'application de la règle en fonction d'un contexte. Les différentes parties de la règle peuvent être représentées à l'aide d'expressions régulières. Un poids peut être attribué à chaque règle. Ce poids permet de quantifier l'importance donnée à chaque règle. Enfin, l'utilisation du symbole (?→) indique que la règle est optionnelle, c'est-à-dire qu'elle peut s'appliquer ou ne pas s'appliquer. La règle obligatoire utilise le symbole (→). Dans la figure 4, la première règle indique qu'un chiffre ou un symbole peut être transformé en une minuscule avec un poids de 1. Le caractère optionnel de la règle sous-entend qu'un chiffre ou un symbole peut également rester lui-même.

5. Alignement SMS/français standard par composition filtrée

5.1. Définition du filtre de composition

Pour aligner le SMS et sa traduction en français standard, nous utilisons une composition filtrée qui permet de retrouver la distance d'édition minimale entre deux chaînes de caractères représentées sous forme d'automates à états finis. Cette composition se présente donc comme suit :

$$\mathrm{ALIGN}_{\mathrm{WFST}} = \mathrm{SMS}_{\mathrm{FSA}} \circ \mathrm{FILTRE}_{\mathrm{WFST}} \circ \mathrm{TRAD}_{\mathrm{FSA}}$$

La machine $\mathrm{ALIGN}_{\mathrm{WFST}}$ est le résultat de la composition et propose tous les alignements possibles entre la machine d'entrée $\mathrm{SMS}_{\mathrm{FSA}}$ et la machine de sortie $\mathrm{TRAD}_{\mathrm{FSA}}$. La recherche du meilleur alignement consiste à sélectionner le meilleur chemin parmi l'ensemble des solutions proposées :

$$\mathit{Best}(\mathrm{ALIGN}_{\mathrm{WFST}})$$

Le filtre utilisé pour l'alignement est un filtre permissif : il permet d'augmenter la taille du langage commun entre le message en langage SMS et sa traduction. Les contraintes liées à l'écriture d'un SMS impliquent qu'il est très rare que ces messages soient écrits en français standard sans aucune modification (abréviations, ajout de ponctuation,…). Le résultat de la composition entre $\mathrm{SMS}_{\mathrm{FSA}}$ et $\mathrm{TRAD}_{\mathrm{FSA}}$ sans filtre intermédiaire est le plus souvent nul. L'introduction d'un filtre permissif modélisant les principales transformations observées dans les SMS permet de retrouver une intersection entre les deux messages.

Pour décrire ces transformations, nous utilisons un ensemble de règles de réécriture optionnelles pondérées qui permettent la transformation d'une suite d'un ou plusieurs symboles en langage SMS en une suite d'un ou plusieurs symboles en français standard. Ces règles ont été décrites à partir d'un ensemble d'observations dégagées par l'étude du corpus SMS (Fairon et al., 2006). Nous décrivons ces règles sous forme d'expressions régulières et dans un format compatible avec le compilateur *Ovide*. Le compilateur permet la transformation des règles en une machine à états finis pondérée représentant le filtre à insérer entre le SMS et sa traduction.

5.2. Modélisation des opérations d'édition à l'aide de règles de réécriture

Nous proposons quelques exemples de règles de réécriture spécifiques à l'alignement SMS/français standard et modélisant les différentes opérations d'édition.

Dans les SMS, les erreurs d'accentuation sur les voyelles sont fréquentes. Elles peuvent être modélisées par des règles de substitution (figure 5).

```
[CLASSIN]
A       [áàâäãa]        #représentations accentuées et non-accentuée de la voyelle a
E       [éèêëe]         #représentations accentuées et non-accentuée de la voyelle e

[RULE]
<A>     ?→      <A> / 1
<E>     ?→      <E> / 1
```

Figure 5 : Règles modélisant les erreurs d'accentuation (substitution)

D'autres erreurs de substitution apparaissant dans les SMS résultent de la proximité du langage SMS avec le langage oral. Cette proximité engendre des substitutions entre un ou plusieurs graphèmes qui correspondent à une même prononciation (figure 6).

```
[CLASSIN]
cs      [cs]        #prononciation [s]
ckq     [ckq]       #prononciation [k]

[RULE]
<cs>    ?→      <cs>    / 2
<ckq>   ?→      <ckq>   / 2
```

Figure 6 : Règles modélisant les substitutions entre graphies qui correspondent à une même prononciation

Les formes en langage SMS sont souvent plus brèves que leur traduction en français standard. Ces abréviations diverses (troncation du début ou de la fin d'un mot, utilisation de rébus, suppression de la ponctuation ou des consonnes finales muettes) sont liées aux contraintes d'utilisation du GSM. Pour permettre un alignement entre un mot du langage SMS et sa forme en langue standard, il est alors nécessaire d'insérer un ou plusieurs caractères. Les insertions peuvent être représentées par la règle décrite dans la figure 7, , basée sur les classes définies dans la figure 4. Cette règle indique qu'un ou plusieurs symboles du langage standard peuvent être insérés entre deux symboles SMS.

```
[RULE]
<EPS>   ?→      <LN>    /       3
```

Figure 7 : Règle modélisant l'insertion de caractères

Enfin, le caractère affectif des SMS peut inciter les utilisateurs à dédoubler les graphèmes finaux ou à insister sur certains graphèmes (Ex: viiiite, bisoussss). Pour retrouver le mot initial à partir du mot SMS, il convient alors de supprimer certains caractères du SMS. La règle modélisant les suppressions de caractères est la suivante :

```
[RULE]
<SMS>   ?→      <EPS>   /       3
```

Figure 8 : Règle modélisant la suppression de caractères

6. Résultats obtenus et exploitation possible

A partir de quelques règles dégagées sur base d'observations du corpus de SMS, nous avons créé un filtre qui a permis l'alignement des 30 000 SMS de la base de données. Après un parcours global des alignements obtenus, nous pouvons attester de l'efficacité de la méthode. Nous ne pouvons actuellement pas proposer une évaluation chiffrée du résultat global car celle-ci demanderait la comparaison du résultat avec un alignement manuel très fastidieux à réaliser. Néanmoins, nous proposons ci-dessous quelques résultats d'alignement et nous montrons également certaines erreurs non résolues par le système actuel. Nous suggérons ensuite une méthode à appliquer pour améliorer la qualité du résultat. Enfin, nous présentons quelques données statistiques dégagées à partir de cet alignement.

6.1. Résultats obtenus

Les quelques exemples d'alignement proposés ci-dessous démontrent la qualité du système. La méthode se montre efficace et parvient à proposer des solutions adéquates lorsque le message ne contient aucun séparateur de mots (voir figure 9, exemple 3) ou lorsqu'un même mot utilise plusieurs mécanismes d'écriture SMS comme l'agglutination et l'écriture phonétique (voir figure 9, exemple 2). L'impossibilité du traducteur de SMS *TiLT* à traiter ce type de messages a été mise en avant dans (Guimier de Neef et Fessard, 2007). Sur base de l'alignement effectué, des règles pourraient être dégagées pour résoudre ce type de difficultés.

```
Exemple 1 :
Je c pa kan jvien mé surmen pa avan dmain! (SMS)
Je sais pas quand je viens mais sûrement pas avant demain! (TRAD)

ALIGNEMENT :

je c___ pa_ k_an_ j__vien_ mé__ sur_men_ pa_ avan_ d_main! (SMS)
je sais pas quand je viens mais sûrement pas avant demain! (TRAD)

Exemple 2 :
Jti1 vrmt foratwa tiprinc! (SMS)
Je tiens vraiment fort à toi petit prince! (TRAD)

ALIGNEMENT :

j__ti1__ vr__m__t for__a_twa __ti__princ_! (SMS)
je tiens vraiment fort à toi petit prince! (TRAD)

Exemple 3 :
comencavamoigvb1a+ (SMS)
Comment ça va moi je vais bien à plus (TRAD)

ALIGNEMENT :

com_en__ca_va_moi_g__v____b_1__a_+___ (SMS)
comment ça va moi je vais bien à plus (TRAD)
```

Figure 9 : Exemples d'alignement

6.2. Quelques erreurs d'alignement

L'alignement des 30 000 SMS n'est cependant pas parfait. Les erreurs d'alignement peuvent être dues à la non prise en compte de certains phénomènes comme la confusion des caractères qui se trouvent sur une même touche du GSM (voir figure 10, exemple 1). D'autres erreurs d'alignement se produisent lorsque plusieurs alignements sont équiprobables pour un même mot ou groupe de mots. Le choix d'un de ces alignements est arbitraire et n'est pas toujours le

meilleur (voir figure 10, exemple 2). L'affinement des pondérations ou la description d'un contexte pour certaines règles peuvent résoudre ce type de problèmes.

```
Exemple 1 :
on pn_urrait se grouper (SMS)
on p_ourrait se grouper (TRAD)

CONFUSION n/o:

on pnurrait se grouper. (SMS)
on pourrait se grouper. (TRAD)

Exemple 2 :
es__k____e (SMS)
est-ce que (TRAD)

ALIGNEMENT ATTENDU:

e___s__k_e (SMS)
est-ce que (TRAD)
```

Figure 10 : Exemples d'erreurs d'alignement

Il arrive aussi qu'un alignement soit inadéquat à cause d'une erreur de traduction (voir figure 11, exemple 1) ou que la pertinence de l'alignement soit discutable (voir figure 11, exemple 2).

```
Exemple 1 :
ok _e_t v dormir chez m__ (SMS)
ok vais __dormir chez moi (TRAD)

ERREUR DE TRADUCTION

Ok et vais dormir chez moi (TRAD)

Exemple 2 :
b__n___ui_. (SMS)
bonne nuit. (TRAD)

MEILLEUR ALIGNEMENT?

b_____nui_. (SMS)
bonne nuit. (TRAD)
```

Figure 11 : Erreur de traduction (exemple 1) et alignement discutable (exemple 2)

6.3. Perspectives d'amélioration des résultats

Pour améliorer les résultats, nous proposons de tester une méthode itérative qui apprendrait automatiquement de nouvelles règles pondérées à partir de l'alignement automatique. Ces règles définiraient, à partir du premier alignement, les possibilités pour chaque graphème SMS d'être associé à un ou plusieurs graphèmes du langage standard.

Exemple :
o ?→ o
o ?→ au
o ?→ eau
o ?→ aux
…

Ces règles seraient pondérées à partir du système suivant :

$$P(GRAPH_{LANG}|GRAPH_{SMS}) = \frac{C(GRAPH_{LANG}, GRAPH_{SMS})}{C(GRAPH_{SMS})}$$

La probabilité d'avoir un graphème du langage standard étant donné un graphème du langage SMS consiste donc à additionner le nombre de fois qu'on rencontre l'association de ce graphème du langage avec le graphème du SMS dans le corpus et de diviser le résultat obtenu par le nombre de fois qu'on rencontre le graphème SMS.

Ces nouvelles règles formeraient un nouveau filtre de composition à appliquer entre les SMS et leur traduction pour proposer un nouvel alignement. Ce nouvel alignement serait utilisé pour calculer automatiquement de nouvelles règles et le processus serait réitéré jusqu'à qu'à ce que les résultats d'alignement saturent. La saturation des résultats serait obtenue à partir du moment où il y a peu ou pas de différences entre les alignements proposés par un nouveau filtre et ceux obtenus avec le filtre précédent.

6.4. Exploitation des résultats

A partir des résultats obtenus par l'alignement automatique, nous pouvons proposer quelques données statistiques qui permettent d'affiner les observations dégagées par l'étude du corpus de l'UCL (Fairon et al., 2006). On peut attester par exemple qu'une majorité de mots restent identiques dans le SMS et dans le langage standard (69,92%), que l'abréviation des mots est un phénomène courant du langage SMS (19,88%) alors que l'allongement de certains mots reste un phénomène relativement rare (0,63%). Le graphique ci-dessous représente ces données.

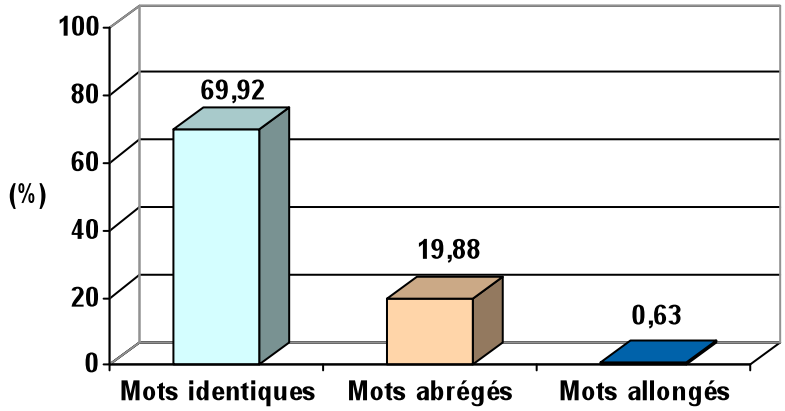


Figure 12 : Statistiques du corpus (%)

7. Conclusion et perspectives

Nous avons décrit dans cet article une méthode d'alignement SMS/français standard à l'aide d'un filtre de composition défini par des règles de réécritures compilées sous forme de machines à états finis. Notre méthode d'alignement présente des performances intéressantes qui nous incitent à poursuivre nos recherches et à chercher à améliorer les résultats (voir 6.3.).

L'apport de la méthode d'alignement que nous proposons dans cet article est multiple. Premièrement, elle permet d'affiner les études sur le langage SMS et de proposer automatiquement des statistiques sur les phénomènes observés dans le langage SMS. A partir de l'alignement, il est également envisageable de dégager des règles qui pourraient permettre de traduire automatiquement les SMS en français standard. Plusieurs études se sont déjà intéressées à cette problématique (Guimier de Neef et Fessard, 2007), mais elles se basent sur des observations de corpus et non sur des relevés automatiques. Ces études ont montré la

difficulté de traiter certains phénomènes du langage SMS qu'il semble possible de résoudre à partir de l'alignement. Cette traduction automatique se révèle non seulement utile pour la transcription des 45 000 SMS du corpus de l'UCL qui n'ont pas été transcrits manuellement, mais également pour des services tels que la synthèse vocale de SMS qui demande une traduction préalable du SMS pour obtenir un message vocal de qualité. Enfin, nous avons montré la flexibilité et la facilité de modifier les règles qui permettent de calculer le filtre de composition. Cette flexibilité nous permet d'envisager l'adaptation de la méthode pour d'autres tâches impliquant un alignement.

Références

Beaufort R. (2006). *FSM Library : description de l'API.* Rapport technique, juin'06, http://www.multitel.be/TTS.

Beaufort R. (2007). *Ovide Documentation. From regular expressions and rewrite rules to finite-state machines*, Technical report, August'07, http://www.multitel.be/TTS.

Chomsky N. (1956). Three models for the description of language. *I.R.E. Transactions on Information Theory*, IT-2(3):113–124.

Damerau F.J. (1964). A technique for computer detection and correction of spelling errors. *Communications of the ACM*, 7(3):171-176.

Fairon C., Klein J. and Paumier S. (2006). *Le langage SMS. Etude d'un corpus informatisé à partir de l'enquête 'Faites don de vos SMS à la science'*, Presses universitaires de Louvain, Louvain-la-Neuve.

Guimier de Neef E. et Fessard S. (2007). Evaluation d'un système de transcription de SMS, *Proc. Of '07*, pages 217-224.

Levenshtein V. (1966). Binary codes capable of correcting deletions, insertions and reversals. *Soviet Physics*, 10:707-710.

Mohri M. (2003). Edit-Distance of Weighted Automata. *Lecture Notes in Computer Science*, 2608:1-23.

Oflazer K. (1996). Error-tolerant finite-state recognition with applications to morphological analysis and spelling correction. *Computational Linguistics*, 22(1):73-89.

Roche E. and Schabes Y, editors. (1997). *Finite-State Language Processing*. MIT Press, Cambridge, Massachusetts.

Schulz K. and Mihov S. (2002). Fast String Correction with Levenshtein-Automata. *International Journal of Document Analysis and Recognition*, 5(1):67-85.

Viterbi A.J. (1967). Error Bounds for Convolutional Codes and an Asymptotically Optimum Decoding Algorithm. *IEEE Transactions on Information Theory*, IT-13, April 1967, pages 260-269.

Wagner. R.A. (1974). Order-n correction for regular languages. *Communications of the ACM*, 17(5):265-268.