TALN 2010, Montréal, 19–23 juillet 2010

Une approche hybride traduction/correction pour la normalisation des SMS

Richard Beaufort[1] Sophie Roekhaut[2] Louise-Amélie Cougnon[1] Cédrick Fairon[1]
(1) CENTAL, Université catholique de Louvain, 1348 Louvain-la-Neuve, Belgique
(2) TCTS Lab, Université de Mons, 7000 Mons, Belgique

Résumé. Cet article présente une méthode hybride de normalisation des SMS, à mi-chemin entre correction orthographique et traduction automatique. La partie du système qui assure la normalisation utilise exclusivement des modèles entraînés sur corpus. Evalué en français par validation croisée, le système obtient un taux d'erreur au mot de 9.3% et un score BLEU de 0.83.

Abstract. This paper presents a method of normalizing SMS messages that shares similarities with both spell checking and machine translation approaches. The normalization part of the system is entirely based on models trained from a corpus. Evaluated in French by ten-fold cross-validation, the system achieves a 9.3% Word Error Rate and a 0.83 BLEU score.

Mots-clés : SMS, normalisation, machines à états finis, approche hybride, orienté traduction, orienté correction, apprentissage sur corpus.

Keywords: SMS messages, normalization, finite-state machines, hybrid approach, machine translation-like, spell checking-like, corpus-based learning.

1 Introduction

Depuis quelques années, le « *Short Message Service* » (SMS), moyen de communication qui a été rapidement adopté par les utilisateurs, offre la possibilité d'échanger des messages écrits entre téléphones mobiles. Souvent, ces messages s'écartent des conventions orthographiques. Comme l'ont montré les spécialistes (Thurlow & Brown, 2003; Fairon *et al.*, 2006), cette variabilité tient à l'usage simultané de plusieurs stratégies de codage, comme les jeux et les transcriptions phonétiques (*demain* → *2m1*, *comme* → *kom*), les squelettes consonantiques (*toujours* → *tjrs*), les séparateurs abusifs, manquants ou incorrects (*j esper* pour *j'espère* ; *j'croibi1k* pour *je crois bien que*), etc. Ces écarts sont dus à trois facteurs principaux : le faible nombre de caractères autorisé par le service sans surcoût (140 octets), les contraintes dues au clavier du mobile et, enfin et surtout, le fait que les usagers du SMS communiquent principalement entre parents ou amis, dans un registre informel. Quelles qu'en soient les causes, ces écarts entravent considérablement le fonctionnement de tout système TAL traditionnel, qui n'est pas prévu pour gérer tant de mots hors-vocabulaire. De ce fait, comme le remarquent Sproat *et al.* (2001), une normalisation SMS, c'est-à-dire la réécriture d'un SMS en orthographe conventionnelle, doit être réalisée *avant* qu'un module TAL plus standard (un système de synthèse de la parole, par exemple) ne puisse entrer en action.

La normalisation SMS a jusqu'à présent été abordée selon trois angles différents : correction orthographique, traduction automatique et reconnaissance de la parole. Chaque approche, basée sur des postu-

lats différents, gère efficacement *certains* des phénomènes présents dans les SMS. Cependant, dans l'ensemble, normaliser un SMS reste un problème complexe et non résolu : les meilleurs systèmes, en effet, descendent difficilement en dessous des 11% d'erreurs au mot. La méthode de normalisation que nous présentons ici, développée dans le cadre général d'un système de synthèse de la parole à partir de SMS[1], se situe à mi-chemin entre les approches orientées correction et traduction. Comme en correction, le système distingue différents types d'unités au sein du texte, et ne normalise *que les unités considérées comme bruitées*. Comme en traduction, les modèles de normalisation sont exclusivement appris *à partir de corpus parallèles*.

Cet article s'organise comme suit. La section 2 dresse un état de l'art du domaine. La section 3 donne ensuite une vue d'ensemble du système, tandis que la section 4 se concentre sur la manière dont nous apprenons et dont nous combinons les modèles de normalisation. Sur cette base, la section 5 évalue le système et le compare aux travaux antérieurs. La section 6, enfin, fait un point sur le travail accompli et propose quelques améliorations envisageables.

2 Travaux antérieurs

Kobus *et al.* (2008) l'ont souligné, la normalisation SMS a jusqu'à présent été formalisée au travers de trois métaphores : correction orthographique, traduction automatique et reconnaissance de la parole.

La métaphore orientée correction (Guimier de Neef & Fessard, 2007; Choudhury *et al.*, 2007; Cook & Stevenson, 2009) réalise la tâche de normalisation *mot à mot*. Partant de l'hypothèse que la plupart des mots sont corrects pour les besoins de la communication, le principe est ici de garder les mots connus en dehors du processus de normalisation. Guimier de Neef & Fessard (2007) ont proposé un système expert dont les seules ressources linguistiques dédiées aux SMS sont des lexiques d'abréviations spécifiques. Choudhury *et al.* (2007) et Cook & Stevenson (2009) ont préféré implémenter la métaphore du canal bruité (Shannon, 1948), qui pose un processus de communication dans lequel l'émetteur envoie le message voulu W au travers d'un canal de communication imparfait (bruité) tel que la séquence O observée par le destinataire est une version bruitée de W. Sur cette base, l'idée est de retrouver W caché derrière O, en maximisant :

$$\begin{aligned} W_{max} &= \arg\max P(W|O) \\ &= \arg\max \frac{P(O|W)\,P(W)}{P(O)} \end{aligned} \tag{1}$$

où $P(O)$ est ignoré parce que constant, $P(O|W)$ modélise le bruit du canal, et $P(W)$ modélise le langage de la source. Quelle que soit l'implémentation proposée, cependant, la limitation principale de cette métaphore « correction » est probablement la trop grande confiance qu'elle accorde aux frontières de mots.

La métaphore orientée traduction (Aw *et al.*, 2006) envisage le processus de normalisation comme une tâche de traduction depuis un langage source (le SMS) vers un langage cible (l'écrit normalisé). Ce point de vue se base sur l'observation que d'une part, les SMS diffèrent fortement de leur transcription normalisée et que d'autre part, la plupart des erreurs dépassent la frontière du mot et ne peuvent être gérées qu'au sein d'un contexte plus large. Partant de cette analyse, Aw *et al.* (2006) ont proposé un modèle statistique travaillant au niveau du groupe. Bien que cette approche obtienne de très bons résultats (voir section 5),

[1]Le projet Vocalise. Voir : `http://cental.fltr.ucl.ac.be/team/projects/vocalise/`

Kobus *et al.* (2008) considèrent qu'une traduction au niveau du groupe n'est pas à même de capturer la grande créativité lexicale des messages SMS. En outre, les principes de base de la traduction automatique, prévue pour gérer des correspondances multiples entre source et cible, dépassent largement les besoins de la normalisation SMS, quasi déterministe.

Sur cette base, Kobus *et al.* (2008) ont proposé de gérer cette tâche selon la métaphore de la reconnaissance de la parole (*automatic speech recognition*, ASR). Il est vrai que les SMS présentent de nombreux jeux phonétiques qui rendent parfois la forme SMS (*sré*, *mwa*) plus proche de sa représentation phonétique ([sʁɛ], [mwa]) que de sa forme normalisée (*serai*, *moi*). Or, typiquement, un système ASR tente de découvrir la meilleure séquence de mots cachée derrière un treillis de séquences phonétiques. Appliquée aux SMS, cette métaphore implique de commencer par convertir le message en un treillis de phones, avant de le transformer en un treillis de séquences de mots, à l'aide d'un dictionnaire inversé phonèmes–graphèmes. Un modèle de langue est ensuite appliqué au treillis de mots, avant d'en retenir uniquement la séquence de mots la plus probable. Un avantage indéniable de cette approche est sa capacité intrinsèque à gérer les frontières de mots. Mais l'inconvénient de l'étape de conversion graphèmes–phonèmes est qu'elle empêche d'identifier les graphèmes présents dans la séquence initiale.

Notre approche, détaillée en sections 3 et 4, partage des similitudes avec les deux premières métaphores, en essayant de combiner leurs avantages, tout en évitant leurs inconvénients : comme les systèmes de correction, nous détectons au plus tôt les unités de texte non ambiguës et nous utilisons les frontières de mots lorsqu'elles semblent *suffisamment fiables* ; mais comme les approches orientées traduction, notre processus de normalisation utilise des modèles exclusivement appris à partir de corpus parallèles. Dans notre cas, il s'agit d'un corpus SMS et de sa transcription, alignés au niveau du caractère.

3 Vue d'ensemble du système

Notre système repose entièrement sur des lexiques, des modèles de langue et des règles de réécriture compilés en machines à états finis (*finite-state machines*, FSMs) et combinés avec le texte à traiter par *composition* (○). Le lecteur qui ne maîtriserait pas les FSMs et leurs propriétés fondamentales consultera utilement la littérature de référence (Hopcroft *et al.*, 1979; Roche & Schabes, 1997). Nous utilisons nos propres outils à états finis : une bibliothèque de FSMs et son compilateur associé (Beaufort, 2008).

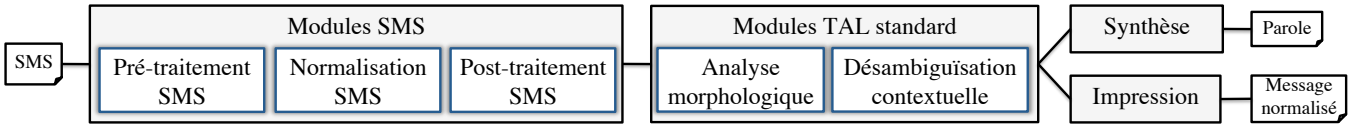

FIG. 1 – Architecture du système

Comme l'illustre la figure 1, un SMS passe d'abord au travers de trois modules SMS qui en normalisent les parties bruitées. Deux modules TAL réalisent ensuite une analyse morphosyntaxique du texte normalisé. Le dernier module, enfin, dépend du type de sortie désiré : soit un synthétiseur, qui construit le signal de parole correspondant au texte normalisé sur la base de son analyse linguistique, soit un module d'impression, qui produit le texte normalisé et lui applique les règles typographiques fondamentales (majuscule en début de phrase, présence ou absence d'espaces entre les unités du texte, etc.) en se basant sur les unités détectées par les modules de pré- et de post-traitement SMS. Cet article étant consacré à l'étape de normalisation, le reste de cette section décrit exclusivement les trois modules SMS.

Le module de prétraitement. Au sein d'un texte, ce module repère exclusivement les séquences suivantes : fins de paragraphes et de phrases, URL, numéros de téléphone, dates, unités de mesure et de temps, monnaies et, très fréquents dans le contexte des SMS, les smileys[2]. Ces séquences, identifiées à l'aide de grammaires locales, sont considérées comme des *unités non ambiguës* et évitent l'étape de normalisation. Toute autre séquence de caractères est considérée comme une *unité bruitée* et subit l'étape de normalisation. Ceci rapproche la méthode de la métaphore orientée correction.

Le module de normalisation. Les modèles et les lexiques utilisés ici sont tous appris au cours d'un entraînement détaillé en section 4. Inspirée de la métaphore du canal bruité (cf. section 2), notre approche s'en distingue néanmoins dans la mesure où le modèle dédié au bruit du canal varie selon que la séquence bruitée est connue (*known sequence*, *KN*) ou non (*unknown sequence*, *UNK*) :

$$P(O|W) = \begin{cases} P_{KN}(O|W) & \text{si } O \text{ est une séquence connue} \\ \\ P_{UNK}(O|W) & \text{sinon} \end{cases} \tag{2}$$

Ce modèle est le résultat de nombreux tests, qui ont mis en évidence le fait que distinguer les séquences connues et inconnues améliore considérablement l'efficacité du système, sans nuire aux performances. Sur cette base, notre algorithme se divise en trois étapes. La première est une composition de l'unité bruitée U avec un transducteur (*finite-state transducer*, FST) Seg dont la tâche est de différencier les séquences connues des séquences inconnues, en les étiquetant avec des symboles de l'alphabet considérés comme des *marqueurs* : *KN* et *UNK*. L'unité est ensuite divisée (*split*) en n segments O_i en fonction de ces marqueurs :

$$\{O_1, O_2, \ldots, O_{n-1}, O_n\} = \text{Split}(U \circ Seg) \tag{3}$$

Dans une seconde étape, chaque segment est composé avec le modèle de réécriture correspondant à son type : le modèle R_{KN} pour les séquences connues, et le modèle R_{UNK} pour les séquences inconnues :

$$O'_i = \begin{cases} O_i \circ R_{KN} & \text{si } O_i \text{ est une séquence connue} \\ \\ O_i \circ R_{UNK} & \text{sinon} \end{cases} \tag{4}$$

Tous les segments réécrits sont ensuite reconcaténés ensemble, de manière à récupérer l'unité complète :

$$U = \odot_{i=1}^{n}(O'_i) \tag{5}$$

où $\odot$ est l'opérateur de concaténation. La troisième étape, enfin, concerne une phrase complète S. Toutes les unités U_j de S sont concaténées ensemble et composées avec le modèle de langue lexical LM. Le résultat est un treillis pondéré de mots, dont on ne retient que la séquence de mots la plus probable S', c'est-à-dire le meilleur chemin (*best path*) du treillis :

$$S' = \text{BestPath}(\ (\odot_{j=1}^{m} U_j) \circ LM\ ) \tag{6}$$

où m est le nombre d'unités de S. Dans S', chaque unité bruitée U_j de S est associée à sa normalisation la plus probable.

Le module de post-traitement. Ce dernier module SMS n'est appliqué qu'à la version normalisée des unités bruitées, afin d'y identifier toute séquence non alphabétique et de l'isoler dans une unité distincte. A ce stade, par exemple, un point devient une « ponctuation forte ». Les grammaires locales utilisées ici sont plus complètes que celles du prétraitement, car elles peuvent – et doivent – détecter les séquences numériques et alphanumériques, les champs de données (comme les numéros de cartes bancaires), les ponctuations et les symboles.

[2]Notre liste compte environ 680 smileys distincts.

4 Les modèles de normalisation

Nos modèles de normalisation ont été entraînés sur un corpus français de 30 000 messages, collectés en Belgique, anonymisés semi-automatiquement et normalisés manuellement par l'Université catholique de Louvain (Fairon & Paumier, 2006). Ensemble, le corpus SMS et sa transcription constituent des *corpus parallèles* alignés au niveau du message. Afin de pouvoir apprendre à partir de ces corpus, nous avions besoin d'un alignement *au niveau du caractère*. Cet alignement a été obtenu de manière complètement automatique en appliquant l'algorithme itératif décrit dans (Cougnon & Beaufort, 2009). De manière succincte, cet algorithme apprend graduellement la meilleure façon d'aligner deux séquences de caractères, en affinant pas à pas la probabilité de chaque opération d'édition classique (substitution, insertion et suppression) *en fonction* des caractères à aligner. L'apprentissage réalisé sur l'alignement obtenu rapproche la méthode de la métaphore orientée traduction.

4.1 Le modèle de segmentation Seg

Ce modèle segmente une unité bruitée en une suite alternée de séquences connues et inconnues. La segmentation est donc réalisée en fonction des séquences connues, collectées au cours de l'apprentissage. Lors du parcours de nos corpus parallèles alignés au niveau du caractère, nous avons considéré comme *séquence* « la plus longue suite de caractères parcourue sans rencontrer le même séparateur de part et d'autre de l'alignement ». Par exemple, l'alignement (a) ci-dessous :

(a) `J esper_ k___tu va_` / `J'espère que tu vas` → (b) `J esper_` | `k___tu` | `va_` / `J'espère` | `que tu` | `vas`

où les soulignés (_) notent les insertions, donne selon notre définition la segmentation (b), puisque le séparateur dans « `J esper` » est différent de sa transcription, et que « `ktu` » ne contient pas de séparateur. Cette notion de *séquence* se base sur le fait que, dans les SMS, les séparateurs sont peut-être indicatifs, mais certainement pas fiables.

Donc, un premier parcours de nos corpus parallèles nous a fourni une liste de séquences connues correspondant à notre lexique, KN. Le FST de segmentation Seg est construit sur cette base :

$$Seg = (\ (\ Sep^{*}\ (Kn|Unk)\ \ (\ Sep^{+}(Kn|Unk)\ )^{*}\ Sep^{*}\ ) \circ Mrg\ ) \tag{7}$$

où :

- Kn est un FST correspondant au lexique KN, dans lequel chaque séquence connue est projetée sur le marqueur ***KN***.
- Unk est le complément de Kn[3], où les séquences inconnues sont projetées sur le marqueur ***UNK***.
- Sep est un FST correspondant à la liste des séparateurs (tout caractère non alphabétique et non numérique), projetés sur un marqueur ***SEP***.
- Mrg est un FST capable de détecter les suites de séquences connues (resp. inconnues), et de les regrouper (*merge*) sous un unique marqueur ***KN*** (resp. ***UNK***). Au cours de ce processus, tous les marqueurs ***SEP*** disparaissent de Seg, et les séparateurs entourant une séquence inconnue lui sont associés.

La figure 2 illustre le résultat de l'application de Seg à une unité bruitée.

[3]En réalité, le vrai complément de Kn accepte toutes les séquences avec séparateurs, alors que Unk ne les accepte pas.

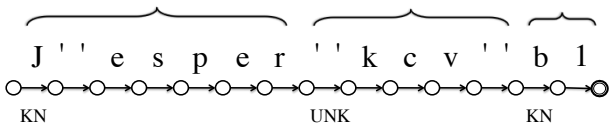

FIG. 2 – Résultat de l'application du modèle de segmentation Seg à l'unité « ***J esper kcv b1*** » (***J'espère que ça va bien***). Dans cet exemple, la séquence « *kcv* » n'a pas été rencontrée lors de l'entraînement ; elle est donc considérée comme inconnue par Seg, et étiquetée *UNK*.

4.2 Le modèle de réécriture des séquences connues R_{KN}

Ce modèle est construit au cours d'un second parcours de nos corpus parallèles, dont l'objectif est de recenser les normalisations associées aux séquences du lexique KN. Au cours de ce second parcours et contrairement au précédent, les séquences du lexique sont recherchées dans le corpus quel que soit le contexte (séparateurs ou non), afin d'assurer le recensement de *toutes* les normalisations possibles.

Chaque normalisation $\overline{kn}$ d'une séquence connue kn est pondérée comme suit :

$$p(\overline{kn}|kn) = \frac{\text{Occ}(\overline{kn}, kn)}{\text{Occ}(kn)} \tag{8}$$

où Occ(x) note le nombre d'occurrences de x dans le corpus. Le FST R_{KN} est ensuite construit comme suit :

$$R_{KN} = \ {S_{KN}}^* \ KN_R \ \ (\ {S_{KN}}^+ \ KN_R \ )^* \ {S_{KN}}^* \tag{9}$$

où :

– KN_R est un *lexique* pondéré, dans lequel chaque séquence KN est associée à la liste pondérée de ses normalisations.
– S_{KN} est un *lexique* pondéré, dans lequel chaque séparateur est associé à la liste de ses normalisations possibles. Souvent, la suppression du séparateur est l'une des normalisations possibles. Lorsque ce n'est pas le cas, cette possibilité de suppression (`DEL`) est ajoutée, et pondérée comme suit :

$$p(\texttt{DEL}|kn) = \frac{0.1}{\text{Occ}(kn) + 0.1} \tag{10}$$

4.3 Le modèle de réécriture des séquences inconnues R_{UNK}

Les deux modèles précédents étaient des expressions régulières construites à partir de lexiques pondérés. Celui-ci, par contre, correspond à une liste de *règles de réécriture pondérées* $\phi \rightarrow \psi \ / \ w$, apprises à partir de l'alignement, où le remplacement $\phi \rightarrow \psi$ se voit attribuer le poids w. Pourquoi cette différence de modélisation ? Les expressions régulières des modèles précédents avaient pour objectif de contraindre le langage accepté, et plus particulièrement la place des séparateurs, de manière à forcer le système à favoriser certains solutions. Dans le cas des séquences inconnues, nous savons que dans l'absolu, *tout doit être possible*. Il n'était donc pas nécessaire de définir un langage différent de Σ^*.

Les cibles de nos règles (ϕ) sont des séquences de 1 à 5 caractères prises du côté SMS de l'alignement, tandis que les réécritures (ψ) sont leurs normalisations correspondantes. Une contrainte importante exprimée sur les listes de règles est que les règles sont classées de la plus spécifique à la plus générale, de

sorte qu'une règle donnée n'est appliquée que si aucune règle plus spécifique et plus pertinente n'a été rencontrée plus haut dans la liste. Pour cette raison, nos règles sont classées dans l'ordre décroissant de la longueur de leurs cibles, afin que les règles aux cibles les plus longues soient choisies le plus souvent possible. Ceci réduit le nombre de normalisations proposées pour une séquence donnée, puisque les séquences les plus longues ont tendance à présenter moins de normalisations différentes.

Du large ensemble de séquences possibles de 2 à 5 caractères collectées dans le corpus, nous n'avons gardé dans notre liste de règles que les séquences qui autorisent *au moins* une normalisation *faite exclusivement de mots appartenant à la langue* : lors du recensement des séquences candidates dans le corpus, nous avons systématiquement vérifié chaque forme normalisée dans un lexique de formes françaises standard. Le lexique utilisé contient 430 000 formes fléchies et est dérivé de la base de données lexicales Morlex[4].

4.4 Le modèle de langue

Notre modèle de langue statistique est un 3-gramme de formes lexicales, lissé par interpolation linéaire (Chen & Goodman, 1998), estimé sur la partie normalisée de notre corpus d'entraînement et compilé en un FST pondéré LM_w.

A ce stade, ce FST *ne peut pas être combiné* avec nos autres modèles, parce que l'alphabet sur lequel il est défini est fait de *formes lexicales* et non de *caractères*. Ce problème est résolu en composant LM_w avec un autre FST L, qui représente un lexique associant chaque mot, considéré comme une séquence de caractères, avec le même mot, considéré cette fois comme une forme lexicale. Les formes lexicales sont ensuite supprimées définitivement du modèle de langue en ne conservant que la première projection (l'entrée) de la composition :

$$LM = \text{FirstProjection}(\ L \circ LM_w\ ) \tag{11}$$

5 Evaluation

L'efficacité et les performances de notre système ont été évaluées sur un MacBook Pro, Intel Core Duo 2,4 GHz, 4 Go SDRAM 667 MHz DDR2, tournant sous Mac OS X version 10.5.8. L'évaluation a été réalisée sur le corpus de 30 000 SMS présenté en section 4, par validation croisée en 10 blocs (Kohavi, 1995). Le principe de cette méthode d'évaluation est de diviser le corpus initial en 10 blocs de taille égale (ici, 3 000 SMS). Le système est ensuite entraîné et testé 10 fois, chaque bloc étant à son tour exclu du corpus d'entraînement, mais le seul à servir de corpus de test.

Le tableau 1 présente le nombre moyen d'entrées/sorties des 10 modèles appris au cours de la validation croisée. Si les séquences inconnues (de 1 à 5 caractères) sont beaucoup moins nombreuses que les séquences connues, leur nombre de réécritures est par contre significativement plus élevé, ce qui est dû au fait que ces séquences sont sélectionnées indépendamment des séparateurs éventuels. Malgré le grand nombre de séquences connues apprises, le système a cependant traité en moyenne 85% des séquences SMS à l'aide du modèle UNK.

Avec une vitesse moyenne de 1836,57 caractères/sec (écart type de 159,65), le système traite un SMS de 140 caractères en 76,23 ms (écart type de 22,34 ms). Le système semble donc efficace, étant donné le

[4]Voir : `http://bach.arts.kuleuven.be/pmertens/`

	KN(*)	*Réécritures KN*	*UNK*(*)	*Réécritures UNK*	*Lexèmes*(**)	*n-grammes*
Total	41 281	49 152	12 225	69 841	19 801	515 128
Rapport	1,19		5,71		26,01	

TAB. 1 – Entrées/sorties des modèles. (*) Séquences SMS. (**) Formes normalisées.

	1. Notre approche				2. Autres approches					
	Validation croisée, français				En français			En anglais		
	Copie		*Hybride*		*Guimier 2007*	*Kobus 2008*		*Aw 2006*	*Choudury 2007*(**)	Cook *2009*(**)
	$\bar{x}$	σ	$\bar{x}$	σ		1	2(*)			
Sub.	25,90	1,65	6,69	0,45		11,94				
Del.	8,24	0,74	1,89	0,31		2,36				
Ins.	0,46	0,08	0,72	0,10		2,21				
WER	34,59	2,37	**9,31**	0,78		16,51	**10,82**		41,00	44,60
SER	85,74	0,87	**65,07**	1,85		76,05				
BLEU	0,47	0,03	**0,83**	0,01	0,736		**0,8**	**0,81**		

$\bar{x}$=*moyenne,* σ=*écart type*

TAB. 2 – Performances. (*) Kobus 2008-2 correspond à une combinaison du modèle ASR de Kobus 2008-1 et d'un modèle « orienté traduction » réalisé à partir d'un ensemble de logiciels libres. (**) Scores obtenus sur des données bruitées uniquement, en dehors du contexte de la phrase.

temps considérable passé dans le modèle UNK. Sur ce point, il n'est malheureusement pas possible de proposer une comparaison avec les autres systèmes, qui ne fournissent pas cette information.

Le tableau 2, partie 1, présente les performances de notre approche (*Hybride*) et les compare à un simple copier-coller du SMS (*Copie*). Nous avons évalué le système en termes de score BLEU (Papineni *et al.*, 2001), de taux d'erreur à la phrase (*Sentence Error Rate*, SER), et de taux d'erreur au mot (*Word Error Rate*, WER), le WER se subdivisant lui-même en substitutions (Sub.), suppressions (Del.) et insertions (Ins.). Les résultats du copier-coller donnent une idée du bruit réellement présent dans le corpus SMS, et mettent en évidence le fait que notre système a encore des difficultés à réduire le SER, alors que les résultats en termes de WER et de score BLEU sont plutôt encourageants.

Le tableau 2, partie 2, reproduit les résultats des autres approches de la littérature. La plupart des résultats sont cependant difficiles à comparer aux nôtres, parce qu'ils ont été obtenus soit dans une langue différente (l'anglais), soit sur un corpus différent : c'est le cas de Kobus *et al.* (2008), qui d'une part ont combiné le corpus que nous avons utilisé à un autre corpus SMS, et d'autre part ont réalisé un seul test, basé sur un corpus d'entraînement plus important (36 704 SMS) pour un corpus de test comparable à l'un de nos blocs (2 998 SMS). Les seuls résultats véritablement comparables sont ceux de Guimier de Neef & Fessard (2007), qui ont évalué leur approche sur le même corpus que nous, mais sans validation croisée, parce que leur système expert ne nécessite pas d'apprentissage. Quoi qu'il en soit, le tableau 2 montre que notre méthode supporte très bien la comparaison avec les meilleures méthodes antérieures.

L'analyse des normalisations produites par notre système a mis en évidence trois caractéristiques importantes :

1. Les séparateurs manquants (*Pensa ms* → *Pense à mes*) ou superflus (*G t* → *J'étais*) sont globalement

bien gérés, ce qui est reflété par nos taux de suppression et d'insertion réduits.

2. Le prétraitement est utile, puisque les unités non ambiguës ne sont pas modifiées.
3. Les erreurs sont souvent contextuelles : elles concernent le genre (*quel(le)*), le nombre (*bisou(s)*), la personne (*[tu t']inquiète(s)*) ou le temps (*arrivé/arriver*). Cependant, comme le soulignent Kobus *et al.* (2008), la fréquence de ces erreurs n'est pas surprenante en français, langue dans laquelle les modèles n-grammes sont souvent incapables de modéliser cette information, hors de leur portée.

6 Conclusion et perspectives

Dans cet article, nous avons présenté une normalisation SMS basée sur des machines à états finis et développée dans le contexte d'un système de synthèse de la parole à partir de SMS. Afin d'éviter la modification erronée des unités non ambiguës, nous avons conçu une méthode hybride, entre correction et traduction. Notre algorithme de normalisation est original à deux niveaux. Premièrement, il repose entièrement sur des modèles appris. Deuxièmement, le modèle de réécriture appliqué à un segment d'unité bruitée change selon que le segment est connu ou non.

Evalué par validation croisée, le système semble efficace, et les performances en termes de score BLEU et de WER sont plutôt encourageantes. Cependant, le SER reste trop élevé, ce qui met en évidence le fait que le système a besoin d'être amélioré.

Avant tout, la normalisation devrait mieux modéliser les similarités phonétiques, au vu du grand nombre de jeux phonétiques dans les SMS. Le modèle phonétique, par exemple, devrait savoir que *o*, *au*, *eau*, …, *aux* se prononcent [o], tandis que *è*, *ais*, *ait*, …, *aient* sont souvent prononcés [ɛ]. Cependant, contrairement à Kobus *et al.* (2008), nous pensons que ce modèle doit éviter l'étape de conversion graphèmes–phonèmes, qui empêche aux étapes suivantes d'identifier les graphèmes présents dans la séquence initiale. A la place, nous proposons d'*apprendre* les similarités phonétiques à partir d'un dictionnaire de mots accompagnés de leurs transcriptions phonétiques, et de construire des règles *graphèmes–graphèmes*. Ces règles pourraient ensuite être pondérées, en apprenant leurs fréquences à partir de nos corpus alignés. Ce modèle devrait également autoriser les variations de timbre, comme [e]–[ɛ], afin d'accepter des similarités entre graphèmes fréquemment confondus en français, comme *ai* ([e]) et *ais/ait/aient* ([ɛ]).

Il serait également intéressant de tester l'impact d'un autre modèle de langue lexical, entraîné sur des phrases non-SMS. En effet, le modèle lexical présente un inconvénient majeur dans le contexte de messages SMS : il doit être appris sur des *formes standard*, ce qui, dans le contexte des SMS, implique la *retranscription* du corpus, un processus coûteux qui réduit le nombre de données d'entraînement du modèle… Le corpus qui remplacerait le corpus SMS retranscrit devrait cependant partager deux points communs avec le langage SMS : il devrait mimer la syntaxe de l'oral et être le plus spontané possible. Sur la base de ces contraintes, notre intention est de récolter des phrases de forums Internet, en sélectionnant ces forums avec soin, parce que leurs textes partagent un autre point commun avec les SMS : ils sont bruités. De ce fait, l'idée est de choisir un forum dont la philosophie est explicitement d'éviter l'utilisation du langage SMS et d'accorder de l'importance à l'orthographe et à la grammaire.

La dernière amélioration que nous proposons ici est plus orientée correction : l'idée est d'autoriser la correction orthographique *à l'intérieur* des modules TAL du système. Placées à ce stade du processus, guidées par l'analyse morphosyntaxique et en combinaison avec elle, des méthodes de correction plus sophistiquées pourraient ainsi se focaliser sur le problème non trivial des erreurs contextuelles.

Remerciements

Cette recherche a été co-financée par les projets FIRST Post-Doc « Vocalise » (convention 716619) et WIST2 « Expressive » (convention 616422) de la Région wallonne.

Références

AW A., ZHANG M., XIAO J. & SU J. (2006). A phrase-based statistical model for sms text normalization. In *Proc. COLING/ACL 2006*.

BEAUFORT R. (2008). *Application des Machines à Etats Finis en Synthèse de la Parole. Sélection d'unités non uniformes et Correction orthographique*. PhD thesis, FUNDP, Namur, Belgium. 605 pages.

CHEN S. F. & GOODMAN J. (1998). *An Empirical Study of Smoothing Techniques for Language Modeling*. Rapport interne 10-98, Computer Science Group, Harvard University.

CHOUDHURY M., SARAF R., JAIN V., MUKHERJEE A., SARKAR1 S. & BASU A. (2007). Investigation and modeling of the structure of texting language. *International Journal on Document Analysis and Recognition*, **10**(3), 157–174.

COOK P. & STEVENSON S. (2009). An Unsupervised Model for Text Message Normalization. In *Proc. Workshop on Computational Approaches to Linguistic Creativity*, p. 71–78.

COUGNON L.-A. & BEAUFORT R. (2009). SSLD : a French SMS to Standard Language Dictionary. In S. GRANGER & M. PAQUOT, Eds., *Proc. eLexicography in the 21st century : New applications, new challenges (eLEX 2009)* : Presses Universitaires de Louvain. To appear.

FAIRON C., KLEIN J. R. & PAUMIER S. (2006). *Le langage SMS : étude d'un corpus informatisé à partir de l'enquête Faites don de vos SMS à la science*. Presses Universitaires de Louvain.

FAIRON C. & PAUMIER S. (2006). A translated corpus of 30,000 French SMS. In *Proc. LREC 2006*.

GUIMIER DE NEEF E. & FESSARD S. (2007). Evaluation d'un système de transcription de SMS. In *Actes de Lexique et Grammaire 2007*, p. 217–224.

J. HOPCROFT, R. MOTWANI & J. ULLMAN, Eds. (1979). *Introduction to Automata Theory, Languages, and Computation*. Massachusetts : Addison-Wesley.

KOBUS C., YVON F. & DAMNATI G. (2008). Normalizing SMS : are two metaphors better than one ? In *Proc. COLING 2008*, p. 441–448, Manchester, UK.

KOHAVI R. (1995). A study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection. In *Proc. IJCAI'95*, p. 1137–1143.

PAPINENI K., ROUKOS S., WARD T. & ZHU W.-J. (2001). BLEU : a method for automatic evaluation of machine translation. In *Proc. ACL 2001*, p. 311–318.

E. ROCHE & Y. SCHABES, Eds. (1997). *Finite-State Language Processing*. Cambridge : MIT Press.

SHANNON C. E. (1948). A mathematical theory of communication. *The Bell System Technical Journal*, **27**, 379–423.

SPROAT R., BLACK A., CHEN S., KUMAR S., OSTENDORF M. & RICHARDS C. (2001). Normalization of non-standard words. *Computer Speech & Language*, **15**(3), 287–333.

THURLOW C. & BROWN A. (2003). Generation txt ? The sociolinguistics of young people's text-messaging. *Discourse Analysis Online*, **1**(1).