Industrie Canada
Centre d’innovation en technologies de l’information (CITI)

Réaccentuation automatique de textes français

Michel Simard
simard@citi.doc.ca

Laval
janvier 1996

Ce document fait état de travaux de recherche réalisés dans le cadre des activités du Centre d’innovation en technologies de l’information (CITI), Industrie Canada. Les opinions exprimées dans ce document n’engagent que l’auteur.

Also available in English under the title: “Automatic Restoration of Accents in French Text”

No catalogue Co28-1/129-1996F

No ISBN 0-662-81092-9

Réaccentuation automatique de textes français

Michel Simard
simard@citi.doc.ca

Industrie Canada
Centre d'Innovation en Technologies de l'Information
1575 Chomedey
Laval (Québec)
CANADA H7V 2X2

1 Introduction

Les travaux présentés ici s'inscrivent dans le cadre du projet *Robustesse*, mené par l'équipe de traduction assistée par ordinateur (TAO) du CITI. Ce projet vise à élaborer des méthodes et des outils de traitement des langues naturelles *robustes* : plusieurs systèmes de TALN vont soit refuser de traiter des textes comportant des erreurs ou des phénomènes étrangers à leur propre ensemble de connaissances, soit afficher un comportement imprévisible dans ces circonstances. À l'opposé, un système *robuste* se comportera alors de façon prévisible et utile.

Les textes français sans accents (marques diacritiques) constituent un exemple typique et particulièrement répandu des problèmes auxquels font face les systèmes de TALN. C'est dans le contexte du courrier électronique (*e-mail*) qu'on rencontre le plus souvent ce phénomène, qui s'explique de deux façons. Premièrement, le monde de l'informatique a longtemps souffert de l'absence d'une norme suffisamment répandue pour l'encodage des caractères accentués, ce qui a entraîné toute une panoplie de problèmes de transfert et de traitement des texte français. Il n'est d'ailleurs pas rare qu'un des maillons logiciels dans la chaîne de distribution du courrier électronique "désaccentue" délibérément les caractères accentués, afin de prévenir d'éventuels problèmes. Deuxièmement, la saisie au clavier des caractères accentués demeure, encore à ce jour, un exercice ardu, voire acrobatique dans certains cas : ici, il s'agit à la fois d'une question de norme et d'une question d'ergonomie. Le résultat concret, c'est qu'un très grand nombre d'utilisateurs francophones évite systématiquement d'utiliser les caractères accentués, tout du moins pour le courrier électronique.

Si cette situation demeure tolérable en pratique, c'est parce qu'il est extrêmement rare que la disparition des accents rende un texte français incompréhensible pour un humain[1]. D'un point de vue

1. On peut quand même trouver des exemples : *Ce chantier ferme a cause des emeutes*

linguistique, l'absence d'accents en français ne fait qu'augmenter le degré relatif d'ambiguïté inhérent à la langue. À la limite, elle ralentit la lecture et suscite un certain inconfort, comme peut le faire, par exemple, la lecture d'un texte rédigé entièrement en majuscules.

Il n'en demeure pas moins que si le français sans accent est acceptable dans certaines circonstances, il ne l'est pas dans l'usage courant, notamment dans le cas des documents imprimés. Par ailleurs, l'absence des accents pose de sérieux problèmes pour le traitement automatique des textes. Qu'il s'agisse de recherche documentaire, de correction orthographique, grammaticale, stylistique, de traduction automatique, d'interface en langue en naturelle ou de quelqu'autre forme de traitement de la langue, on s'attendra en général à ce que les textes français comportent des accents. D'où l'intérêt pour des méthodes de récupération automatique des accents, ou de *réaccentuation automatique*.

En examinant le problème, on constate que la grande majorité des mots d'un texte français s'écrivent naturellement sans accents (envrion 85%), et que pour plus de la moitié des mots qui restent, la forme accentuée correcte peut être déduite de façon déterministe à partir de la forme sans accent[2]. Il en découle que la simple utilisation d'un bon dictionnaire permet de réaccentuer automatiquement un texte sans accent avec un taux de succès de près de 95% (c'est-à-dire qu'on commettra une erreur d'accent à peu près à tous les vingt mots).

Tout porte à croire qu'on peut atteindre des résultats de beaucoup supérieurs grâce à l'utilisation de *modèles de langue* plus ou moins sophistiqués, qui seront en mesure de lever les ambiguïtés résultant de l'absence d'accents, en se basant sur des considérations d'ordre linguistique. En particulier, il semble que les modèles de langue dits *probabilistes* soient particulièrement bien adaptés pour ce genre de tâche, parce qu'ils fournissent un critère de désambiguïsation *quantitatif* : lorsqu'on rencontre une forme sans accent à laquelle peuvent correspondre plusieurs formes valides (portant ou non des accents), on choisit la plus *probable,* en se basant sur le contexte immédiat et sur un ensemble d'évènements observés antérieurement (le “corpus d'entraînement”).

Notons que cette idée n'est pas entièrement originale : El-Bèze et al. exposent dans [3] une technique de réaccentuation qui s'inspire des mêmes concepts, alors que Yarowsky obtient des résultats comparables dans [6], en combinant différents critères de désambiguïsation statistiques dans un cadre unificateur (les *listes de décision*).

2 Réaccentuation automatique

Nous avons mis au point un programme de réaccentuation automatique, que nous appelons *Reacc*, basé sur un modèle de langue stochastique. Reacc accepte en entrée une chaîne de carac-

2. Dans ce qui suit, nous désignons par *mot* une occurrence dans un texte d'une unité linguistique.

tères représentant un texte français sans accent. Si la chaîne d'entrée contient des accents, on peut bien sûr la *désaccentuer* : comme à tout caractère accentué ne correspond qu'un seul caractère sans accent, ce processus est entièrement déterministe. Une autre possibilité est de conserver les accents, en prenant pour acquis qu'ils sont corrects. Dans un cas comme dans l'autre, la sortie attendue de Reacc est une chaîne de caractères qui ne diffère de la chaîne d'entrée que par les accents : on s'attend à recevoir en sortie le même texte français, mais correctement accentué.

Reacc procède donc en trois étapes successives : *segmentation*, *génération d'hypothèses* et *désambiguïsation*.

L'unité sur laquelle opère Reacc est le *mot*. L'exercice de segmentation consiste donc à prendre la chaîne d'entrée et à y localiser les frontières entre les mots, incluant les signes de ponctuation, de même que les nombres et autres expressions combinant chiffres et lettres. La segmentation repose sur un ensemble de règles décrivant des connaissances générales sur la structure des textes électroniques. Très peu de ces connaissances sont spécifiques au français. On retrouve quand même une liste d'abréviations et acronymes courants, qui sert à déterminer si un point accolé à une suite de caractères alphabétiques appartient à ce mot, ou agit comme point final[3]. On utilise aussi une liste des constructions les plus fréquentes impliquant le *tiret* et l'*apostrophe* en français, afin de déterminer s'ils agissent ou non comme frontière de mots : *l'école* versus *aujourd'hui*, *passe-montagne* versus *pensez-vous*.

L'étape suivante, la *génération d'hypothèses*, consiste à produire, pour chaque mot identifié lors de la segmentation, la liste de toutes les possibilités d'accentuation. Par exemple, si on a isolé l'unité *cote*, on veut générer les formes *cote*, *coté*, *côte*, *côté*. En fait, rien n'empêche qu'on génère aussi les formes *côtè*, *cötê*, etc. En pratique, toutefois, il importe de limiter autant que possible le nombre d'hypothèses, de façon à réduire le potentiel d'explosions combinatoires. On a donc recours à une liste de toutes les formes françaises valides, formes fléchies incluses, indexées sur leurs versions désaccentuées. En théorie, une telle liste peut contenir plusieurs centaines de milliers de formes distinctes. En pratique, on peut couper ce nombre de moitié, en excluant les formes qui ne portent pas d'accents et pour lesquelles il n'existe pas de variante accentuée valide. On peut réaliser des économies supplémentaires en excluant les formes les moins fréquentes, mais dans ce cas, il faut s'attendre à une baisse de la performance.

Une fois les hypothèses générées, il faut choisir les plus vraisemblables : c'est ce qu'on appelle la *désambiguïsation*. Nous utilisons pour ce faire un modèle de langue stochastique, appelé *modèle de Markov caché* (l'implantation que nous utilisons est le package *lm* de Foster [4]). Dans ce modèle, un texte est vu comme le résutat de deux processus stochastiques distincts. Le premier processus génère une suite de symboles qui, dans notre modèle, correspondent à des *étiquettes*

---

3. Notre heuristique exclut la possibilité qu'un point soit à la fois partie d'une abréviation et marqueur de fin de phrase. Il semble que cette approximation ne nuise pas à la performance du système.

morpho-syntaxiques (par exemple : *NomCommun-masculin-singulier*, *Verbe-Indicatif-présent-3ième-personne-pluriel*). Dans un modèle markovien d'ordre *N*, la production d'un symbole dépend uniquement des *N-1* symboles précédents. La séquence d'étiquettes produite constitue le phénomène *caché* d'où le modèle tire son nom. Le deuxième processus génère alors, pour chaque étiquette de la séquence, un autre symbole qui, cette fois-ci, correspond à une forme (un mot) du langage. Cette deuxième séquence est le résultat observable.

Les paramètres de notre modèle sont donc :

- $P(t_i \mid h_{i-1})$: La probabilité d'observer une étiquette t_i, étant données les *N-1* étiquettes précédentes (h_{i-1} désigne la suite d'étiquettes de longueur *N-1* se terminant à la position *i-1*).
- $P(f_i \mid t_i)$: La probabilité d'observer une forme f_i, étant donnée l'étiquette sous-jacente t_i.

Bien entendu, la valeur exacte de ces paramètres est inconnue, mais en pratique, on peut en faire l'estimation à partir de fréquences observées dans un *corpus d'entraînement*. Ce corpus consiste en un ensemble de phrases, à chaque mot duquel est acollée l'étiquette appropriée (en d'autres mots : un corpus dans lequel la nature du phénomène caché nous est “révélée”). La taille du corpus doit être suffisante pour assurer une estimation fiable de la valeur de chaque paramètre. À défaut d'un tel corpus étiquetté, on peut effectuer l'entraînement à partir d''un texte non-étiquetté, pour ensuite raffiner la valeur des paramètres par réestimation. On peut aussi combiner ces deux méthodes, c'est-à-dire obtenir une première estimation des paramètres à partir d'un petit corpus étiquetté, pour ensuite en faire la réestimation sur la base d'un corpus non-étiquetté de plus grande taille.

Étant donnés ces paramètres, on peut évaluer la probabilité globale d'une suite de mots $s = s_1 s_2 ... s_n$. Soit *T*, l'ensemble de toutes les séquences d'étiquettes de longueur *n* possibles :

$$P(s) = \sum_{t \in T} \prod_{i=1}^{n} P(t_i \mid h_{i-1}) P(s_i \mid t_i)$$

Bien que le calcul direct de cette équation requière un nombre d'opérations exponentiel en *n*, il existe un algorithme qui produit le même résultat en temps polynomial (voir [5]).

Notre stratégie de désambiguïsation consiste à choisir la suite d'hypothèses qui produit la version du texte dont la probabilité globale est maximale. En d'autres mots, si on représente le texte et ses hypothèses d'accentuation comme un graphe acyclique dirigé (*DAG*), le problème peut se formuler comme la recherche du chemin, allant du début à la fin du texte, dont la probabilité est maximale (figure 1).

Le calcul de ce chemin pose bien entendu des problèmes de complexité de calcul, puisque le nombre de chemins à explorer croît en général de façon exponentielle avec la longueur du texte. En pratique, toutefois, il est possible de segmenter le graphe en îlots indépendants, c'est-à-dire en sections pour lesquelles le chemin optimal est indépendant du reste du graphe. Typiquement, on considère que les *phrases* sont indépendantes les unes des autres. On peut donc segmenter le

Figure 1: Représentation d'un texte et des hypothèses d'accentuation sous forme de graphe acyclique dirigé

texte en phrases et calculer le chemin optimal pour chaque phrase[4]. Si le nombre de possibilités à l'intérieur d'une phrase demeure problématique, il existe des moyens de resegmenter celle-ci, au prix d'une légère dégradation de la précision. Dans notre implantation, chaque phrase est découpée en segments tels que le nombre de chemins à explorer à l'intérieur d'un segment n'excède pas un certain seuil (que nous appelons le paramètre *S*). Les points de coupe sont choisis au moyen d'une heuristique simple qui tend à minimiser la dépendance entre les segments : dans la mesure du possible, chaque segment doit se terminer par une suite de mots non-ambigus, c'est-à-dire pour lesquels il n'existe à la fois qu'une seule hypothèse d'accentuation et une seule analyse lexicale. On traite alors successivement les segments de gauche à droite, et on préfixe chaque segment avec les derniers mots du chemin optimal du segment précédent.[5]

Une fois la désambiguïsation effectuée, il reste à produire un résultat. Cette opération est en réalité très simple, mais quand même digne d'intérêt. En effet, un de nos principaux soucis est de préserver dans la sortie l'apparence du texte d'entrée. Il faut donc partir de chaque forme apparaissant sur le chemin optimal du graphe, retrouver la forme correspondante dans la chaîne d'entrée, et transposer l'accentuation de la nouvelle forme sur la forme originale, sans autrement en modifier l'apparence.

3 Évaluation

Pour évaluer la performance d'une méthode de réaccentuation, il suffit de choisir un texte ou un ensemble de textes français correctement accentués, de les désaccentuer automatiquement, de soumettre le tout au programme de réaccentuation, et de comparer les résultats obtenus au texte original.

Une des propriétés de Reacc que nous souhaitions évaluer était sa capacité à fonctionner avec des textes de nature variée. Pour ce faire, l'idéal aurait été de soumettre à notre programme un corpus "balancé", du même genre que le *Brown Corpus*. Comme nous ne disposions pas d'une telle res-

4. En fait, cette notion d'indépendance entre les phrases fait partie intégrante du modèle, dès lors que l'entraînement lui-même est fait à partir de phrases indépendantes.

5. Une solution alternative est d'utiliser l'algorithme de Viterbi, lequel calcule, pour un graphe tel que le nôtre, le chemin qui maximise la probabilité *conjointe* de la suite de formes et de la suite d'étiquettes sous-jacente, en temps polynomial avec la taille du graphe. Les deux solutions produisant des résultats à peu près similaires, nous avons opté pour la plus directe.

source pour le français, nous avons dû confectionner notre propre corpus, à partir de documents qui nous étaient disponibles.

Le corpus de test est donc constitué d'extraits de textes français accentués, provenant de sept sources différentes, représentées à peu près également : on y retrouve des textes du domaine militaire, des textes juridiques, des publications des Nations Unies, des textes littéraires, des revues de presse, des manuels informatiques et des extraits du Hansard canadien (journal des débats à la Chambre de Communes). L'ensemble totalise 57 966 mots (ce compte a été produit au moyen de l'utilitaire UNIX *wc*). Certaines modifications ont été apportées au texte, afin de corriger les quelques erreurs d'accentuation que nous avons pu déceler au fil des expériences.

Le générateur d'hypothèses de Reacc utilisait, pour nos tests, une liste de formes extraite du *DMF*, un dictionnaire morpho-syntaxique contenant près de 380 000 formes distinctes ([1]). En fait, ce nombre est probablement excessif. Nous avons d'ailleurs obtenu des résultats tout-à-fait satisfaisants lors d'expériences préliminaires, avec un dictionnaire ne reconnaissant que 50 000 formes environ.

Pour le modèle de langue, après différentes expériences, nous avons opté pour une approche qui privilégie la qualité des données sur leur quantité. Nous avons utilisé un modèle de Markov caché d'ordre 2, basé sur un ensemble d'environ 350 étiquettes morpho-syntaxiques. Les paramètres du modèle ont d'abord été initialisés à l'aide du DMF, c'est-à-dire qu'on a restreint d'emblée les $P(f_i \mid t_i)$ en fonction du contenu des valeurs sanctionnées par le dictionnaire. On a ensuite procédé à un entraînement du modèle sur un corpus de texte de 60 000 mots, extrait du *Hansard* canadien, étiqueté à la main ([2]). On a finalement utilisé un corpus de texte beaucoup plus volumineux (plus de 3 millions de mots), non-étiqueté, afin de réestimer les paramètres du modèle.

Outre le générateur d'hypothèses et le modèle de langue utilisés, plusieurs paramètres affectent la performance de Reacc, tant sur le plan de la qualité des résultats obtenus que sur celui du temps-machine. Néanmoins, le facteur le plus important est le paramètre S, qui limite la taille des segments sur lesquels Reacc travaille. On retrouve dans la tableau 1 les résultats obtenus pour différentes valeurs de S (une augmentation exponentielle de ce facteur se traduit en général par une augmentation linéaire de la longueur des segments traités). Les tests ont été effectués sur une machine Sun SPARCstation 10.

Un examen sommaire des résultats obtenus révèle qu'on a fort à gagner en permettant au système de travailler sur des segments plus longs. Toutefois, passée une certaine limite, la qualité des résultats tend à plafonner, alors que les temps d'exécution, eux, grimpent en flèche. Tout dépendant du genre d'application et des ressources disponibles, il semblerait qu'on puisse compter sur des résultats acceptables dès lors que S est fixé à l'entour de 16 ou 32.

Il est intéressant d'examiner où Reacc se trompe. On retrouve dans le tableau 2 une classification grossière des erreurs de réaccentuation que Reacc a commises sur notre corpus de test, lorsque

Nombre maximum d'hypothèses par segment (*S*)	**Temps-machine (secondes)**	**Nombre total d'erreurs (mots)**	**Distance moyenne entre les erreurs (mots)**
2	68	821	70
4	85	560	103
8	132	466	124
16	169	441	130
32	277	429	134
64	429	425	136
128	731	420	137

Tableau 1: Résultats des réaccentuations

S était fixé à 16. La catégorie qui arrive en tête regroupe assez libéralement les erreurs qui ont pour point en commun qu'elles résultent d'un mauvais choix sur la présence d'un accent aigu sur le *e* de la syllabe finale (par exemple : *aime* versus *aimé*). Viennent ensuite les erreurs qui découlent de "lacunes" du générateur d'hypothèses, c'est-à-dire de cas où celui-ci ne connaît tout simplement pas la forme correctement accentuée. Dans la majorité des cas, il s'agit de noms propres (près de la moitié, en fait), mais on rencontre aussi, surtout dans les textes de nature plus technique, beaucoup d'abréviations, de mots non-français et de "néologismes" (par exemple : *réaménagement*, *séropositivité*). La catégorie qui vient ensuite concerne une unique paire de mots : la préposition *à* et la forme *a* du verbe *avoir*.

Type d'erreur	**Nombre**	**Pourcentage**
Ambiguïtés *-e* / *-é*	171	38,8%
Formes inconnues	111	25,2%
Ambiguïté *a* / *à*	69	15,7%
Autres	90	20,4%
Total	441	100%

Tableau 2: Classification des erreurs d'accentuation (*S* = 16)

4 Conclusions

Nous avons présenté une méthode de réaccentuation automatique des textes français, basée sur un modèle de langue markovien caché. Cette méthode a fait l'objet d'une implantation réelle : le programme Reacc. Nos expériences ont démontré que ce programme produisait des textes d'une qualité tout-à-fait acceptable, dans des temps plus que raisonnables : on peut atteindre une moyenne d'une erreur d'accentuation aux 130 mots, en traitant plus de 20 000 mots à la minute.

Bien entendu, il y a toujours place à des améliorations. En particulier, il est certain que l'utilisation d'un modèle de langue plus fin (par exemple, un modèle d'ordre 3) ne pourrait qu'améliorer la qualité de la désambiguïsation. Compte tenu aussi de la forte proportion d'erreurs d'accentuation causées par des lacunes au dictionnaire, il serait intéressant d'examiner des façons de traiter ces "mots inconnus". À cet égard, nous avons déjà effectué certaines expériences préliminaires, qui ont produit des résultats particulièrement intéressant. En particulier, nous nous sommes intéressés à des façons de "deviner" l'accentuation d'un mot inconnu, à partir d'une modélisation stochastique de l'accentuation des mots connus. Il reste toutefois beaucoup de travail à faire de ce côté.

Par ailleurs, les méthodes que nous avons exposées ouvrent la porte à d'autres applications du même genre. Par exemple, on peut voir comment les méthodes de réaccentuation pourraient être généralisées, afin de traiter d'autres types de pertes d'information. On pense tout particulièrement aux textes dont tous les caractères accentués ont été remplacés par un caractère unique (typiquement, un point d'interrogation), ou aux textes dont le huitième bit de chaque caractère a été perdu. Dans de tels textes le *é* apparaît comme un *i*, le *è* comme un *h*, etc. Dans ces cas, au problème de l'ambiguïté lexicale s'ajoute celui du découpage, qui devient lui aussi ambigu.

Une autre possibilité intéressante est de greffer un programme du genre de Reacc à un logiciel de traitement de texte, d'une manière telle que l'utilisateur puisse taper un texte français sans se soucier des accents, qui sont alors insérés automatiquement à mesure que le texte est produit. De la réaccentuation, on passe ainisi à l'*accentuation automatique*. Un tel mécanisme pourrait faciliter sensiblement la saisie des textes français. (On sait combien les conventions de saisie des accents au clavier sont variées et pas toujours très ergonomiques.)

Une application beaucoup plus ambitieuse, se basant sur des méthodes similaires, est la *rédaction assistée par ordinateur*. Dans ce cas, plutôt que de travailler sur le texte déjà tapé par l'utilisateur, l'ordinateur s'intéresse au texte *à venir*, et essaie de prévoir ce que l'utilisateur va taper, de façon à lui éviter la saisie de grandes portions de texte.

Toutes ces applications font présentement l'objet de travaux de recherche au CITI.

Références

[1] Bourbeau, Laurent et François Pinard, 1987, *Dictionnaire Micro-informatisé du Français (DMF)*, Progiciels Bourbeau Pinard Inc., 1501 avenue Ducharme, Montréal, H2V 1G2.

[2] Bourbeau, Laurent, 1994, *Fabrication d'un corpus témoin bilingue étiqueté et annoté pour la mise au point de techniques de parsage automatique probabiliste*, Rapport technique présenté par Progiciels Bourbeau Pinard, Centre d'innovation en technologies de l'information (CITI), Laval.

[3] El-Bèze, Marc, Bernard Mérialdo, Bénédicte Rozeron et Anne-Marie Derouault, 1994, "Accentuation automatique de textes par des méthodes probabilistes", dans *Technique et sciences informatiques*, Vol 13, no 6, pp. 797 - 815.

[4] Foster, George F., 1995, *Communication personnelle*.

[5] Rabiner, L. R. et B. H. Juang, janvier 1986, "An Introduction to Hidden Markov Models", dans *IEEE ASSP Magazine*.

[6] Yarowsky, David, 1994, "Decision Lists for Lexical Ambiguity Resolution: Application to Accent Restoration in Spanish and French", dans *Proceeding of the 32nd Annual Meeting of the Association for Computational Linguistics* (*ACL-94*), pp. 88-95.