Analyse syntaxique du français

Pondération par trigrammes lissés et classes d'ambiguïté lexicales

Richard Beaufort [1], *Thierry Dutoit* [2], *Vincent Pagel* [1]

[1] Multitel ASBL
avenue N. Copernic 1 – 7000 Mons, Belgique
+32 (0)65 37 47 56 – Fax: +32 (0)65 37 47 29
{beaufort,pagel}@multitel.be – http://www.multitel.be

[2] Faculté Polytechnique de Mons
rue de Houdain 9 – 7000 Mons, Belgique
+32 (0)65 37 47 74 – Fax: +32 (0)65 37 47 29
dutoit@tcts.fpms.ac.be – http://tcts.fpms.ac.be

ABSTRACT

In a Text-to-Speech framework, we have implemented a *n*-gram-based Part-of-Speech tagger, currently evaluated on French. Usually, such systems reduce the probability of a sentence to that of its syntactic tags, without taking the words into account, the probability of which is hard to correctly estimate from the data. Our system reintroduces the words in the probability, replacing each word by the ambiguity class it belongs to. We have tested different kinds of smoothing by interpolation, and the influence of the classes on the results of our Part-of-Speech tagger.

1. INTRODUCTION

Dans le cadre d'un synthétiseur de la parole, nous avons réalisé un analyseur syntaxique probabiliste, que nous évaluons dans un premier temps sur le français. A l'instar des systèmes actuels de ce type [bla99, kup92], notre analyseur tâche, pour une suite de mots $\{w_1, w_2, \ldots, w_n\} = W$, de déterminer la meilleure suite d'étiquettes grammaticales $\{t_1, t_2, \ldots, t_n\} = T_{MAX}$ parmi toutes les suites T_i possibles. Par la règle de Bayes, ceci se formalise :

$$\begin{aligned} T_{MAX} &= \arg\max_T P(T \mid W) \\ &= \arg\max_T P(W \mid T)\ P(T) \end{aligned} \qquad (1)$$

La plupart des systèmes probabilistes réduisent le modèle de langue, $P(T)$, à un modèle *n*-gramme lissé, et suppriment le modèle de mots, $P(W|T)$, parce qu'il est difficile à estimer à partir des données d'apprentissage.

En ce qui concerne le modèle de langue, nous nous inscrivons dans la lignée de ces systèmes : nous utilisons un modèle *n*-gramme lissé par *interpolation linéaire*. En revanche, nous réintroduisons le modèle de mots sous la forme d'un modèle de classes d'ambiguïté lexicales, $P(A|T)$.

Dans une première partie, nous présentons les différentes formes de lissage utilisées dans le modèle de langue. Nous expliquons ensuite dans une seconde partie l'origine et l'intérêt des classes d'ambiguïté lexicales pour le modèle de mots. Une troisième partie contient la description de nos corpus et de nos tests, ainsi que les résultats obtenus.

2. LISSAGE DU MODELE DE LANGUE

La majorité des auteurs font l'hypothèse que la probabilité d'une étiquette étant donné le passé ne dépend que des *n*-1 étiquettes précédentes [cha93]. Dans notre cas, *n* vaut 3 (trigramme) :

$$P(T) \approx \prod_{i=1}^{N} p(t_i \mid t_{i-n+1}^{i-1}) \qquad (2)$$

où (t_{i-n+1}^{i-1}) équivaut à $(t_{i-1}, t_{i-2}, \ldots, t_{i-n+1})$.

Cette probabilité est classiquement estimée par *maximum de vraisemblance* (MV) :

$$p_{MV}(t_i \mid t_{i-n+1}^{i-1}) = \frac{c(t_{i-n+1}^{i})}{\sum_{t_i} c(t_{i-n+1}^{i})} \qquad (3)$$

où $c(t_{i-n+1}^{i})$ représente le nombre d'occurrences d'un trigramme donné dans le corpus d'apprentissage.

Cependant, le MV de certains *n*-grammes syntaxiques est mal estimé parce qu'ils n'apparaissent que peu ou pas dans le corpus d'apprentissage.

Ceci a conduit au développement de méthodes de lissage. Traditionnellement, deux grands types de lissage se dégagent : le *backoff*, où le modèle d'ordre *n*-1 se substitue au modèle d'ordre *n* si son MV vaut zéro, et l'*interpolation linéaire*, où la probabilité du *n*-gramme est une combinaison linéaire des modèles d'ordre 0 à n (voir équation (5)). Chaque méthode de lissage peut être envisagée en version *backoff* ou en version *interpolation linéaire*.

Chen et Goodman [che98] ont montré, dans le cadre de la reconnaissance de la parole, que les méthodes d'*interpolation linéaire* ont tendance à donner de meilleurs résultats que leurs homologues en version *backoff*. Pour cette raison, nous avons restreint notre analyseur syntaxique aux versions interpolées des quatre algorithmes suivants : *Jelinek-Mercer*, *Witten-Bell*, *Absolute Discounting* et *Kneser-Ney* [che98]. Pour

pouvoir comparer les résultats à ceux d'un système de base, nous avons également implémenté le « lissage additif ».

Nous invitons le lecteur à se reporter à [che98] pour une analyse détaillée des différents algorithmes de lissage que nous avons testés. Nous n'en donnerons ici qu'une description sommaire.

2.1 Lissage additif

Pour éviter les comptages nuls, on considère que chaque *n*-gramme apparaît δ fois plus que le nombre d'occurrences présentes dans les données d'entraînement, avec $0 < \delta \leq 1$:

$$p_{ADD}(t_i \mid t_{i-n+1}^{i-1}) = \frac{\delta + c(t_{i-n+1}^{i})}{\delta \mid V \mid + \sum_{t_i} c(t_{i-n+1}^{i})} \qquad (4)$$

où $|V|$ représente le nombre de mots dans le corpus d'entraînement. Dans notre système, nous avons fixé le facteur δ à 1.

2.2 Interpolation linéaire

L'interpolation linéaire se présente de manière générale comme suit :

$$\begin{aligned} p_{INT}(t_i \mid t_{i-n+1}^{i-1}) = \quad & \lambda_{t_{i-n+1}^{i-1}} \; p_{MV}(t_i \mid t_{i-n+1}^{i-1}) + \\ & (1 - \lambda_{t_{i-n+1}^{i-1}}) \; p_{INT}(t_i \mid t_{i-n+2}^{i-1}) \end{aligned} \qquad (5)$$

L'interpolation des modèles est récursive. Elle est stoppée au modèle d'ordre 1 ou 0. Notre système s'arrête à l'ordre 0, par la distribution uniforme :

$$p_{UNIF}(t_i) = \frac{1}{|V|} \qquad (6)$$

Les méthodes testées diffèrent principalement dans la manière de calculer les facteurs λ. En outre, certaines méthodes n'utilisent pas P_{MV} dans le calcul des modèles inférieurs à l'ordre *n*, mais une autre estimation.

a) Lissage *Jelinek-Mercer*. Cette méthode implémente la forme générale de l'interpolation linéaire. Les facteurs λ sont calculés à l'aide de l'algorithme de Baum-Welch, à partir de données extérieures au corpus d'apprentissage.

La taille réduite de notre corpus d'apprentissage ne nous a pas permis d'en réserver une partie à l'entraînement seul des facteurs λ. Nous désirions néanmoins conserver ce lissage. Dans une simplification extrême, de nombreux tests nous ont permis de fixer un seul $\lambda = 0.9$ pour les modèles d'ordre 1 à 3.

b) Lissage *Witten-Bell*. Il réduit λ au nombre de mots uniques qui suivent l'historique t_{i-n+1}^{i-1} :

$$\lambda = N_{1+}(t_{i-n+1}^{i-1}\bullet) = \; |\{t_i : c(t_{i-n+1}^{i-1}\; t_i) > 0\}| \qquad (7)$$

où N_{1+} désigne le nombre de *n*-grammes qui ont une ou plusieurs occurrences dans le corpus d'entraînement, et $\bullet$ la variable (t_i) sur laquelle on somme.

c) Lissage *Absolute Discounting*. La probabilité du *n*-gramme d'ordre *n* n'est plus calculée en multipliant le MV par un facteur λ, mais en ôtant un décompte fixe $0 < D \leq 1$ au numérateur du MV :

$$P_{ABS} = \frac{\max\{c(t_{i-n+1}^{i}) - D, 0\}}{\sum_{t_i} c(t_{i-n+1}^{i})} \qquad (8)$$

En outre, pour que la distribution totale somme à 1, le facteur $(1-\lambda)$ est également affecté par ce décompte :

$$(1 - \lambda_{t_{i-n+1}^{i-1}}) = \frac{D}{\sum_{t_i} c(t_{i-n+1}^{i})} N_{1+}(t_{i-n+1}^{i-1}\bullet) \qquad (9)$$

D est calculé de la manière suivante :

$$D = \frac{n_1}{n_1 + n_2} \qquad (10)$$

où n_1 et n_2 sont le nombre total de *n*-grammes qui ont, dans la base d'entraînement, respectivement une et deux occurrences, et où *n* est l'ordre du modèle en cours d'interpolation.

d) Lissage *Kneser-Ney*. Largement inspiré du précédent, l'idée maîtresse de cette méthode est cependant que l'interpolation d'un (*n*-*i*)-gramme ne doit pas être liée à ses occurrences, mais au nombre d'étiquettes différentes que ce (*n*-*i*)-gramme suit. Tout modèle d'ordre inférieur à *n* est calculé de la manière suivante :

$$p_{KN}(t_i \mid t_{i-n+2}^{i-1}) = \frac{N_{1+}(\bullet t_{i-n+2}^{i})}{\sum_{t_i} N_{1+}(\bullet t_{i-n+2}^{i})} \qquad (11)$$

Dans notre système, nous avons implémenté une version récursive de (11), contrairement à la version *back-off* de l'algorithme original.

En reconnaissance de la parole, les résultats de *Kneser-Ney* dépassent largement ceux obtenus par les autres lissages présentés [che98].

3. MOTS ET CLASSES D'AMBIGUITE

Le modèle de mots est traditionnellement basé sur l'hypothèse que la probabilité d'un mot étant donné le passé ne dépend que de son étiquette syntaxique [cha93] :

$$P(W \mid T) \approx \prod_{i=1}^{N} p(w_i \mid t_i) \qquad (12)$$

Or, dans de nombreux cas, (12) est mal estimée parce que les mots nécessaires à son calcul ne sont que peu ou pas présents parmi les données de l'apprentissage. Ceci a amené la plupart des systèmes à ignorer le modèle de mots dans (1), réduisant de la sorte le calcul de T_{MAX} au modèle de langue, $P(T)$.

Pour notre part, nous réintroduisons le modèle de mots W sous la forme d'un modèle de classes A :

$$P(W \mid T) \approx P(A \mid T) \approx \prod_{i=1}^{N} p(a_i \mid t_i) \quad (13)$$

Le concept de *classes d'ambiguïté* a été développé par Kupiec [kup92], qui désirait réduire le nombre des paramètres d'apprentissage de son analyseur HMM. Tzoukermann et Radev [tzo96], ainsi que Kempe [kem01] ont par la suite employé les classes d'ambiguïté comme étiquettes de leurs transducteurs syntaxiques, en lieu et place des mots ; en quelque sorte, ils réunissent le modèle de langue $P(T)$ et le modèle de mots $P(W|T)$ au sein d'un seul modèle. Notre approche conserve par contre une distinction entre le modèle de langue et le modèle de mots ; les classes d'ambiguïté n'interviennent que dans le modèle de mots et sont entraînées séparément du modèle de langue.

Une classe d'ambiguïté lexicale est décrite par la liste exhaustive des catégories grammaticales qui peuvent être attribuées à certains mots de la langue : tous les mots qui partagent les mêmes catégories grammaticales – et n'en acceptent aucune autre – appartiendront à la même classe d'ambiguïté. En français par exemple, « couvent » et « président » appartiendront à la classe {NOUN,VERB} tandis que « ferme » sera dans {ADJ,NOUN,VERB} et « dure » dans {ADJ,VERB}.

Dans une classe donnée, chaque catégorie se voit attribuer une probabilité d'émission, propre à la classe et déterminée à partir du corpus d'apprentissage. Pour une classe A donnée, la probabilité d'émission d'une catégorie t donnée est déterminée en normalisant la somme des occurrences de cette catégorie pour chaque mot w_i de la classe A présent dans le corpus d'entraînement :

$$p(t \mid A) = \frac{\sum_{w_i \in A} c(w_i, t)}{\sum_{t_j \in T} \sum_{w_i \in A} c(w_i, t_j)} \qquad (14)$$

4. TESTS ET RESULTATS

4.1 Méthodologie et données

L'objectif principal de nos tests est d'évaluer l'apport que constituent les classes d'ambiguïté lexicales. En second lieu, nous voulons également établir une hiérarchie des lissages implémentés. Nous testons donc deux versions de notre analyseur : la première, sans classes d'ambiguïté, la seconde, utilisant l'information fournie par les classes. Dans les deux cas, les cinq algorithmes de lissage présentés sont utilisés, ainsi qu'une version non lissée de notre analyseur.

Nous utilisons un corpus de 66.619 mots, « Le mot et l'idée », fourni par l'AUPELF-UREF dans le cadre de l'action de recherche concertée « Synthèse vocale », étiqueté automatiquement à l'aide d'un système par règles, Vertex [mer01], et corrigé en partie manuellement. Etant donné que nous sommes loin des millions de mots préconisés par [che98], nous n'utilisons que 50 catégories grammaticales, et nous suivons la procédure dite de *10-fold-cross-validation* [koh95]. Les deux versions de l'analyseur sont de la sorte exécutées dix fois sur des données extérieures à l'entraînement.

4.2 Résultats et analyse

Sur nos dix tests, nous obtenons en moyenne 95 classes d'ambiguïté, contenant au plus 4 étiquettes syntaxiques.

a) Sans classes – avec classes. Version non lissée mise à part, l'amélioration apportée par les classes d'ambiguïté (table 1) est en moyenne de 1.06% (707 mots). Ceci dit, le lissage additif profite d'une amélioration nettement meilleure (1.47%) que les méthodes d'interpolation (±0.95%). Ceci semble montrer que l'interpolation, parce qu'elle fait constamment intervenir tous les modèles de langue de n à 1, tient compte en partie du modèle de mots. Il faut noter que les performances de l'analyseur non lissé ne sont que très légèrement améliorées par la présence des classes (0.27%, soit 180 mots). Le problème réside évidemment dans les n-grammes de probabilité nulle, qui annulent l'influence positive du modèle de classes.

b) Au sein de l'interpolation. Les quatre algorithmes ont des résultats fort similaires (table 1), que ce soit avec ou sans classes d'ambiguïté. Contrairement à ce qui a été constaté en reconnaissance de la parole, *Kneser-Ney* ne surpasse pas les résultats des autres lissages, et compte le plus d'*erreurs propres* (table 2), le phénomène étant encore amplifié en présence des classes. Ceci ne signifie pas pour autant que ce lissage soit non-performant. Mais le principe d'interpoler un (n-i)-gramme en fonction du nombre de termes différents qu'il suit impose probablement d'utiliser un corpus comptant au minimum plusieurs centaines de milliers de mots. On constate d'ailleurs que le corpus utilisé en reconnaissance [che98] dépassait le million de mots,

tandis que le nôtre n'en compte qu'un peu plus de 66.000. Nous vérifierons cette hypothèse.

Au terme de ces tests, *Kneser-Ney* ne donne pas les meilleurs résultats en analyse syntaxique, mais son principe d'interpolation la distingue des autres méthodes. Les classes d'ambiguïté lexicales, quant à elle, améliorent significativement les résultats.

Table 1: moyenne (en %) d'étiquetage correct sur les dix corpus-tests, avec et sans classes d'ambiguïté.

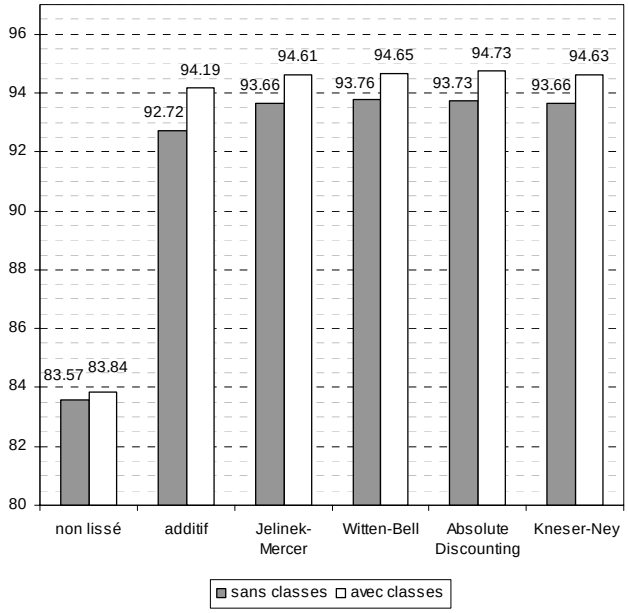

Table 2: total, sur les dix corpus-tests, des *erreurs propres* (erreurs commises par une seule méthode).

	sans classes		avec classes	
	erreurs propres	%	erreurs propres	%
Non lissé	6974	63.86	7266	67.62
Additif	574	11.91	474	12.31
Jelinek-Mercer	15	0.36	18	0.50
Witten-Bell	3	0.07	19	0.53
Abs. Discount.	4	0.10	3	0.09
Kneser-Ney	19	0.45	51	1.42

5. CONCLUSION ET PERSPECTIVES

Nous avons implémenté un analyseur syntaxique probabiliste. Comme les systèmes actuels, le modèle de langue est un *n*-gramme lissé, mais contrairement à eux, le modèle de mots n'est pas supprimé de l'estimation.

Nous avons testé quatre méthodes de lissage par interpolation : *Jelinek-Mercer*, *Witten-Bell*, *Absolute Discounting* et *Kneser-Ney*. Leurs résultats sont fortement similaires. *Kneser-Ney*, qui surpasse les autres méthodes en reconnaissance, n'apporte pas d'amélioration en analyse syntaxique, mais ce résultat est probablement dû à la petite taille de notre corpus d'apprentissage.

Nous avons réintroduit le modèle de mots sous la forme d'un modèle de classes d'ambiguïté. Nos résultats montrent que ces classes améliorent significativement l'analyse syntaxique. Cependant, le concept de *classe d'ambiguïté* est une approximation : il prétend que la distribution des catégories d'une classe donnée est uniforme pour tous les mots de la classe. Le cas idéal où chaque mot constitue sa propre classe n'est pas envisageable, comme nous l'avons vu ; mais il est possible de s'en approcher. Nous avons l'intention d'utiliser conjointement des probabilités de classes et de mots. Dans cette méthode mixte, tout mot suffisamment présent dans le corpus d'apprentissage constituera à lui seul une classe lexicale ; les autres mots seront confondus dans les classes d'ambiguïté.

BIBLIOGRAPHIE

[bla99] Black A., Caley R., Taylor P. (1999), "The Festival Speech Synthesis System". http://www.cstr.ed.ac.uk/projects/festival/

[cha93] Charniak E., "Statistical language learning", MIT Press.

[che98] Chen S. F., Goodman J. T. (1998), "An Empirical Study of Smoothing Techniques for Language Modeling", Technical Report TR-10-98, Computer Science Group, Harvard University.

[cut92] Cutting D., Kupiec J., Pedersen J., Sibun P. (1992), "A Practical Part-of-Speech Tagger", ANLP'92, pp. 133-140.

[kem01] Kempe A. (1991) "Part-of-Speech Tagging with Two Sequential Transducers", Proc. CLIN 2000, pp. 88-96.

[koh95] Kohavi R. (1995), "A study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection", IJCAI-95, pp. 1137-1143.

[kup92] Kupiec J. (1992), "Robust part-of-speech tagging using a hidden Markov model", Computer Speech and Language, (6) pp. 225-242.

[mer01] Mertens P. (2001), "Vertex, a chart parser for unification grammars". http://bach.arts.kuleuven.ac.be/~piet/vertex

[tzo96] Tzoukermann E., Radev D. (1996), "Using Word Class for Part–of–Speech Disambiguation", Proc. 4th Workshop on Very Large Corpora, pp. 1-13.