Modèles de langue pour la recherche d'information

Mohand Boughanem, Institut de Recherche en Informatique de Toulouse, 118 route de Narbonne 31000 Toulouse Cedex 9, France

Wessel Kraaij, TNO TPD, 2600 AD Delft, Pays bas

Jian-Yun Nie, DIRO, Université de Montréal, CP. 6128, succursale Centre-ville, Montréal, Québec, H3C 3J7 Canada

1. Introduction

Les méthodes statistiques basées sur des corpus, au lieu de celles basées sur des connaissances préétablies, ont eu de grands succès dans la linguistique informatique [Manning99]. Ces méthodes tentent de capter, d'une manière statistique, les régularités d'une langue en observant des phrases dans un corpus d'entraînement. Les développements récents dans ce domaine ont démontré que les méthodes ainsi développées peuvent être utilisées avec succès dans l'étiquetage des catégories syntaxiques (*POS-tagging*), la reconnaissance de parole [Jelinek97], et même en traduction automatique [Brown93]. Ce succès est non seulement dû à l'avancement des méthodes utilisées, mais aussi à la disponibilité de plus en plus grande des corpus d'entraînement de différentes sortes. Par exemple, le Penn-treebank est un corpus largement utilisé pour entraîner l'étiqueteur syntaxique. Le Hansard est un corpus parallèle souvent utilisé pour entraîner des modèles de traduction statistiques.

Le domaine de recherche d'information s'est beaucoup inspiré du succès des méthodes statistiques en linguistique informatique. Dans une certaine mesure, la RI a des choses en commun avec la linguistique informatique : Les deux domaines possèdent de grandes masses de textes, ce qui permet d'entraîner des modèles statistiques.

L'idée de base des modèles de langue est de déterminer la probabilité $P(Q|D)$ – la probabilité que la requête Q puisse être générer à partir du document D. Cette formulation est similaire à l'idée derrière les modèles probabilistes formulée pour la première fois dans [Maron60]. Cependant, comme on peut voir plus loin, la façon de calculer $P(Q|D)$ dans les modèles de langue est différente de celle des modèles probabilistes traditionnelles.

Le terme « modèle de langue » est emprunté de la linguistique informatique, où l'objectif d'un modèle de langue est de capter les régularités linguistiques par une ou plusieurs fonctions probabilistes. Ainsi, nous commençons notre présentation par une brève description des méthodes développées pour la modélisation statistique de langue en linguistique informatique.

2. Modèles de langue en linguistique informatique

2.1. Idée de base

Par « modèle de langue », on désigne une fonction de probabilité P qui assigne une probabilité $P(s)$ à un mot ou à une séquence de mots s en une langue. Une fois cette fonction définie, il est possible d'estimer la probabilité d'une séquence de mots quelconque dans la langue, ou d'un point de vue générative, d'estimer la probabilité de générer cette séquence de mots à partir du modèle de la langue.
Considérons la séquence s composée des mots suivants : $m_1, m_2, \ldots, m_n$. La probabilité $P(s)$ peut être calculée comme suit :

$$P(s) = \prod_{i=1}^{l} P(m_i \mid m_1 ... m_{i-1}) \quad (1)$$

Si on utilise la règle de chaîne en théorie de probabilité pour calculer cette probabilité, il y a souvent trop de paramètres (c'est-à-dire $P(m_i|m_1 \ldots m_{i-1})$) à estimer, et ceci est souvent impossible de réaliser. Ainsi, dans les modèles de langue utilisés en pratique, des simplifications sont souvent faites. En général, on suppose qu'un mot m_i ne dépend que de ses n-1 prédécesseurs immédiats, c'est-à-dire :

$$P(m_i \mid m_1 ... m_{i-1}) = P(m_i \mid m_{i-n+1} ... m_{i-1})$$

On utilise, dans ce cas, un modèle de langue n-gramme. En particulier, les modèles souvent utilisés sont les modèles uni-gramme, bi-gramme et tri-gramme comme suit :

$$\text{Uni-gramme : } P(s) = \prod_{i=1}^{l} P(m_i)$$

$$\text{Bi-gramme : } P(s) = \prod_{i=1}^{l} P(m_i \mid m_{i-1}) = \prod_{i=1}^{l} \frac{P(m_{i-1} m_i)}{P(m_{i-1})} \quad (2)$$

$$\text{Tri-gramme : } P(s) = \prod_{i=1}^{l} P(m_i \mid m_{i-2} m_{i-1}) = \prod_{i=1}^{l} \frac{P(m_{i-2} m_{i-1} m_i)}{P(m_{i-2} m_{i-1})}$$

Ce que l'on doit estimer sont les probabilités $P(m_i)$ (un-gramme), $P(m_{i-1}m)$ (bi-gramme) et $P(m_{i-2}m_{i-1}\ m_i)$ (tri-gramme) pour la langue. Cependant, il est difficile d'estimer ces probabilités pour une langue dans l'absolue. L'estimation ne peut se faire que par rapport à un corpus de textes C. Si le corpus est suffisamment grand, on peut faire l'hypothèse qu'il reflète la langue en général. Ainsi, le modèle de langue peut être approximativement le modèle de langue pour ce corpus – $P(\bullet|C)$. Selon les fréquences d'occurrence d'un n-gramme α, sa probabilité $P(\alpha|C)$ peut être directement estimée comme suit :

$$P(\alpha) = \frac{|\alpha|}{\sum_{\alpha_j \in C} |\alpha_j|} = \frac{|\alpha|}{|C|} \quad (3)$$

où $|\alpha|$ est la fréquence d'occurrence du n-gramme α dans ce corpus, α_i est un n-gramme de la même longueur que α, et $|C|$ est la taille du corpus (c'est-à-dire le nombre total d'occurrences de mots). Ces estimations sont appelées les estimations de vraisemblance maximale (*Miximum Likelihood*, ou ML). On désignera aussi ces estimations par P_{ML}.

Nous donnons ici un exemple simple pour illustrer l'estimation de la probabilité uni-gramme ainsi que son utilisation pour calculer la probabilité d'une phrase.

Supposons un petit corpus contenant 10 mots, avec les fréquences comme montrées dans Table 1.

Table1 . Exemple d'estimation de vraisemblance maximale

Mot	le	un	prof	ML	dit	aime	de	langue	modèle	RI
Fréq.	3	2	2	1	2	1	4	2	1	2
***P*(•|*C*)**	0,15	0,1	0,1	0,05	0,1	0,05	0,2	0,1	0,05	0,1

En utilisant l'estimation de vraisemblance maximale, nous obtenons les probabilités comme illustrées dans la table (note : la fréquence totale de mots dans ce corpus est $|C| = 20$).

En utilisant ces probabilités estimées, nous pouvons calculer la probabilité de construire la séquence s = « **le prof aime le ML** » dans cette langue comme suit :

$$P(s) \approx P(s|C) = P(\textbf{le}|C)*P(\textbf{prof}|C)*P(\textbf{aime}|C)*\ P(\textbf{le}|C)*\ P(\textbf{ML}|C) = 0{,}15*0{,}1*0{,}05*0{,}15*0{,}05.$$

2.2. Lissage

Plus le corpus utilisé pour ces estimations est grand, plus on peut espérer obtenir des estimations de probabilité justes. Cependant, quelle que soit la taille du corpus d'entraînement, il y a toujours des mots ou des séquences de mots absents du corpus (par exemple le mot « **non** » dans notre corpus jouet). Pour ces mots ou séquences de mots, leur estimation de probabilité est 0. La conséquence de cette probabilité nulle est qu'on attribuera une probabilité nulle à toute séquence de mots ou des phrases contenant un mot ou un n-gramme non rencontré dans le corpus. Par exemple, la phrase s = « **le prof dit non** » aura une probabilité $P(s|C) = 0$. En d'autres termes, les modèles ainsi construits ne sauraient reconnaître que les phrases dont les n-grammes sont tous apparus dans le corpus. Ce sont donc des modèles très limités. Afin de généraliser les modèle, on voudrait assouplir cette attribution systématique de probabilité nulle aux mots ou séquences de mots non rencontrés. Cette procédure qui consiste à attribuer une probabilité non-nulle á ces éléments est appelée le lissage. Le lissage peut aussi être vu comme une façon d'éviter le surentraînement d'un modèle sur un corpus, et de doter du modèle d'une plus grande capacité de généralisation.

Le principe de lissage peut être résumé ainsi : Au lieu de distribuer la totalité de masse de probabilité sur les n-grammes vus dans le corpus d'entraînement, on enlève une partie de cette masse et la redistribue aux n-grammes non vus dans le corpus. De cette façon, les n-grammes absents du corpus vont recevoir une probabilité non-nulle.

Sur la façon d'enlever et de redistribuer une partie de masse de probabilité, il y a une série de méthodes proposées dans la littérature. Ici, nous présentons quelques-unes unes classiques.

Lissage de Laplace

Le lissage de Laplace consiste à ajouter la fréquence 1 à tous les n-grammes. Cette méthode est aussi appelée la méthode « ajouter-un ». Pour un n-gramme α, sa probabilité est estimée comme suit (où V est l'ensemble du vocabulaire d'indexes) :

$$P_{ajouter\_un}(\alpha \mid C) = \frac{|\alpha| + 1}{\sum_{\alpha_i \in V}(|\alpha_i| + 1)} \qquad (4)$$

On peut remarquer que cette méthode simple a un problème fatal : Si le corpus ne contient qu'une petite portion des n-grammes parmi tous les n-grammes possibles (et c'est souvent le cas dans la pratique, même pour un grand corpus), la majeure partie de la masse de probabilité sera distribuée uniformément sur les n-grammes non vus dans le corpus. Les n-grammes vus dans le corpus ne joueront qu'un rôle mineur dans la définition du modèle. On ne peut donc pas s'attendre à une très bonne performance (c'est-à-dire de reconnaître les phrase autorisées d'une langue correctement).

Lissage Good-Turing

L'idée de ce lissage est de modifier la fréquence d'occurrence observée de la façon suivante : Soit un n-gramme α qui apparaît r fois dans le corpus. On modifie cette fréquence à r* suivante :

$$r^* = (r+1)\frac{n_{r+1}}{n_r} \qquad (5)$$

où n_r est le nombre de n-grammes apparus r fois dans le corpus. Ainsi, l'estimation de la probabilité devient la suivante :

$$P_{GT}(\alpha) = \frac{r^*}{\sum_{\alpha_i \in C} |\alpha_i|} \quad (6)$$

Dans cette méthode, on applique une diminution (*discount*) de fréquence de l'ordre de $\frac{r^*}{r}$ au n-gramme α et cette fréquence est redistribuée sur les n-grammes non vus.

Quand la fréquence *r* est grande, le nombre n_r de n-grammes de cette fréquence est petit. Le lissage Good-Turing dans ce cas effectue une grande modification de la valeur de *r*. L'estimation Good-Turing pour les n-grammes de grande fréquence a donc tendance d'être instable. Dans l'autre bout du spectre, l'estimation de Good-Turing pour les n-grammes de faibles fréquences sont plus stables. C'est souvent pour ces derniers n-grammes que le lissage Good-Turing est recommandé.

Lissage Backoff

Le lissage « Backoff » consiste à utiliser un modèle du même ordre (par exemple, bi-gramme) si le n-gramme est observé dans le corpus, mais utiliser un modèle d'ordre inférieur (par exemple, uni-gramme) si ce n'est pas le cas. Par exemple, dans le lissage Katz, on peut combiner le modèle bi-gramme avec un modèle uni-gramme comme suit :

$$P_{Katz}(m_i \mid m_{i-1}) = \begin{cases} P_{GT}(m_i \mid m_{i-1}) & \text{si} \mid m_{i-1}m_i \mid > 0 \\ \alpha(m_{i-1}) P_{Katz}(m_i) & \text{sinon} \end{cases} \quad (7)$$

où $\alpha(m_{i-1})$ est un paramètre de normalisation.

Dans cette méthode, la diminution de fréquence utilisée dans P_{GT} est redistribuée au modèle d'ordre inférieur (uni-gramme). $\alpha(m_{i-1})$ est un paramètre qui détermine la part de cette redistribution à m_i, déterminée comme suit :

$$\alpha(m_{i-1}) = \frac{1 - \sum_{m_i : |m_{i-1}m_i| > 0} P_{GT}(m_i \mid m_{i-1})}{1 - \sum_{m_i : |m_{i-1}m_i| > 0} P_{ML}(m_i)} \quad (8)$$

Lissage par interpolation

Le lissage par interpolation, par exemple de Jelinek-Mercer, consiste à combiner un modèle avec un ou des modèles d'ordre inférieur systématiquement, plutôt que d'utiliser ce dernier seulement pour dans le cas de fréquence 0 comme dans « Backoff ». Pour une combinaison de modèle bi-gramme avec le modèle uni-gramme, on a :

$$P_{JM}(m_i \mid m_{i-1}) = \lambda_{m_{i-1}} P_{ML}(m_i \mid m_{i-1}) + (1 - \lambda_{m_{i-1}}) P_{JM}(m_i)$$

où $\lambda_{m_{i-1}}$ est un paramètre déterminé de telle manière à maximiser l'espérance des données. Ce paramètre peut dépendre du mot m_{i-1} ou il peut être attribué d'une valeur identique pour tous les mots. Sa valeur est souvent déterminée avec le processus de maximisation de l'espérance (EM).

Comme on pourra voir plus loin, dans le contexte d'utilisation des modèles de langue en RI, on se limite souvent aux modèles uni-gramme. Dans ce cas, le lissage par interpolation prend un autre sens : nous avons deux modèles uni-grammes, l'un estimée sur un document *D* et l'autre sur un grand corpus *C*. Le modèle de langue pour le document est une interpolation du modèle de langue du document $P(\bullet|D)$ avec le modèle de langue du corpus $P(\bullet|C)$.Nous décrirons ceci plus en détail dans la section 4.

Lissage Dirichlet

Dans cette méthode, la fréquence d'un mot m_i dans le document D est incrémentée de $\mu P_{ML}(m_i|D)$, où μ. est paramètre appelé pseudo-fréquence. La probabilité $P_{Dir}(w_i \mid D)$ d'un mot selon le modèle de langue du document devient la suivante :

$$P_{Dir}(m_i \mid D) = \frac{tf(m_i, D) + \mu P_{ML}(m_i \mid C)}{|D| + \mu} \quad (9)$$

où $|D|$ est la taille du document (le nombre total d'occurrences de mots), et $tf(m_i, D)$ est la fréquence du mot m_i dans D.

Les modèle de langues ont été appliqués avec grand succès pour traiter différents problèmes auxquels les approches symboliques traditionnelles ne donnent pas de résultat satisfaisant. Parmi ces problèmes, on peut citer la reconnaissance de parole [Jelinek97], l'étiquetage statistique de catégories syntaxiques des mots, et dans une certaine mesure, certains aspects de la traduction automatique [Brown93].

La question que l'on pose maintenant est : est-il possible d'utiliser les modèles de langue pour la RI? Quelle sera la performance de recherche?

Dans la prochaine section, nous allons décrire quelques approches proposées utilisant des modèles de langues pour la RI. Avant d'entrer en détail, il est important de réaliser que la RI n'a pas le même but que la modélisation statistique de langue en linguistique informatique. Le but visé des modèles de langues en linguistique informatique est de déterminer la légitimité d'une séquence de mots en une langue, ou de la générer à partir du modèle. Dans cette tâche, la séquence de mots acceptée doit être conforme aux critères linguistiques d'une langue, telles les contraintes syntaxiques et morphologiques. Pour la RI, le but est différent : on vise à retrouver les documents dont la sémantique est pertinente à celle de la requête. Dans la tâche de RI, les contraintes syntaxiques et morphologiques deviennent moins importantes, mais la notion de pertinence est centrale. Comme on peut observer, cette notion est absente dans les méthodes de modélisation de langue qu'on vient de décrire. Toutefois, les deux domaines ont certaines caractéristiques en commun :

- Ils traitent d'une langue naturelle;
- Ils disposent d'un grand volume de textes.

Ainsi, les idées et les approches développées dans la linguistique informatique peuvent être adaptées pour la RI. C'est cette intuition que les approches à la RI basées sur les modèles de langue tentent d'exploiter.

3. Modèle de langue en RI - principes

Dans les modèles traditionnels de RI, on tente de modéliser la relation de pertinence entre un document et une requête. Typiquement, dans le modèle probabiliste, on tente d'estimer la probabilité $P(R|D, Q)$ – la probabilité de pertinence d'un document face à une requête. Typiquement, cette probabilité est calculée selon des méthodes paramétriques : on suppose que la distribution des mots suit une certaine norme (par exemple, distribution Poisson) parmi les documents pertinents (et non-pertinents). En fonction des distributions des mots parmi deux ensembles (pertinent et non-pertinent) de documents échantillons, on peut estimer les probabilités des mots pour la pertinence. La probabilité $P(R|D, Q)$ pourra alors être calculée. Pour plus de détails, les lecteurs peuvent se référer au chapitre portant sur le modèle probabiliste dans ce livre.

Le principe des approches utilisant un modèle de langue est différent. On ne tente pas de modéliser directement la notion de pertinence dans le modèle; mais on considère que la pertinence d'un document face à une requête est en rapport avec la probabilité que la requête puisse être générée par le modèle de langue du document. Ainsi, on considère qu'un document D incarne un sous-langage, pour lequel on tente de construire un modèle de langue M_D. Le score du document face à une requête Q est déterminé par la probabilité que son modèle génère la requête :

$$Score(Q, D) = P(Q|M_D) \quad (10)$$

On écrira aussi $P(Q|D)$ pour représenter la même probabilité dans les descriptions plus tard.

De façon générale, une requête peut être vue comme une suite de mots : $Q = t_1 t_2 \ldots t_n$. Nous avons donc :

$$Score(Q, D) = P(t_1 t_2 \ldots t_n|M_D). \quad (11)$$

Ainsi formulé, le problème de la RI devient similaire à la modélisation statistique de langue. Il est donc possible d'utiliser ses techniques développées pour cette dernière en RI.

Le principe que nous venons de décrire est celui que la plupart des modèles de langues en RI utilisent. Il y a aussi quelques autres variantes.

Nous pouvons aussi procéder dans le sens inverse : On tente de créer un modèle de langue pour la requête, et de déterminer le score d'un document $D = t_1 t_2 \ldots t_m$ par la probabilité que le document peut être généré par ce modèle :

$$Score(Q, D) = P(t_1 t_2 \ldots t_m|M_Q) \quad (12)$$

Cependant, dans cette formulation, les documents longs, et contenant des mots fréquents vont être favorisés. Afin de résoudre ce problème, nous pouvons utiliser la loi de Bayes :

$$P(M_Q \mid D) = \frac{P(M_Q)P(D \mid M_Q)}{P(D)} \quad (13)$$

On suppose ensuite que $P(D) \approx P(D \mid C)$ et que $P(M_Q)$ est une constante *c.* Ainsi :

$$P(M_Q \mid D) \approx \frac{cP(D \mid M_Q)}{P(D \mid C)} \quad (14)$$

Dans cette nouvelle formule, nous tenons compte de la longueur du document et des mots contenus dans le document, en ajoutant le facteur $P(D|C)$.

Il est aussi possible de construire un modèle de langue pour le document $P(\bullet|M_D)$ et un autre pour la requête $P(\bullet|M_Q)$. Le score d'un document face à la requête peut être déterminé par une comparaison entre les deux modèles. C'est le principe utilisé dans la méthode d'entropie croisée.

Finalement, on peut aussi voir la relation entre une requête et un document pertinent comme celle d'une traduction : un document pertinent est une « traduction » de la requête; et on tente de déterminer la probabilité de cette traduction.

Dans les sections qui suivent, nous présentons plus en détail les méthodes spécifiques pour implanter ces principes.

4. RI comme génération de la requête par le document

4.1. Modèle de Ponte et Croft

Le premier modèle à la RI base sur la modélisation de langue est [Ponte98]. L'intuition est qu'une requête n'est pas créée de façon aléatoire, mais l'utilisateur a une idée (un modèle) sur le document idéal qui peuvent apparaître dans D et de son modèle M_D. À partir de M_D, l'utilisateur choisit (génère) les termes pour constituer sa requête. Ainsi, la requête devrait être générée à partir d'un document pertinent ou de son modèle. La probabilité de cette génération est considérée comme le score du document :

$$Score(D,Q) = P(Q \mid M_D)$$

Nous apportons quelques remarques ici :

- Une requête peut être considérée comme une suite de mots : $Q = t_1, t_2, \ldots t_n$.
- Comme dans la plupart des modèles de langues utilisées en RI, la simplification suivante a été faite : on suppose que les mots dans une requête sont indépendants.
- Dans le modèle de Ponte et Croft, ils utilisent modèle de Bernoulli multiple, c'est-à-dire que non seulement les mots présents dans la requête, mais aussi ceux absents de la requête, sont pris en compte. Dans la plupart d'autres modèles, on utilise plutôt le modèle multinomial, où on ne considère que les mots présents dans la requête.

Supposons, sans perdre la généralité, que la requête Q est composée de l'ensemble de mots $t_1, t_2, \ldots t_n$, et que les mots $t_{n+1}, t_{n+2}, \ldots t_l$ sont absents de la requête. $P(Q|M_D)$ peut être ré-exprimée comme suit :

$$\begin{aligned} P(Q \mid M_D) &= P(t_1 t_2 ... t_n \mid M_D) \times P(\neg t_{n+1}, \neg t_{n+2}, ..., \neg t_l \mid M_D) \\ &= \prod_{i=1}^{n} P(t_i \mid M_D) \times \prod_{i=n+1}^{l} P(\neg t_i \mid M_D) \\ &= \prod_{t_i \in Q} P(t_i \mid M_D) \times \prod_{t_i \notin Q} (1 - P(t_i \mid M_D)) \end{aligned} \quad (15)$$

Ponte et Croft ont proposé une estimation de la probabilité $P_{PC}(t_i|M_D)$ d'une façon similaire à l'approche Backoff. Ils combinent un modèle de langue du document avec un modèle de langue du corpus comme suit :

$$P_{PC}(t_i \mid M_D) = \begin{cases} P_{ML}(t_i \mid D)^{1-\hat{R}(t_i,d)} \times P_{avg}(t_i)^{\hat{R}(t_i,D)} & \text{si } tf(t_i, D) > 0 \\ P_{ML}(t_i \mid C) & \text{sinon} \end{cases} \quad (16)$$

où $tf(t_i, D)$ est la fréquence de t_i dans le document D

Dans cette formule, on note deux éléments principaux $P_{ML}(t_i \mid D)$ et $P_{ML}(t_i \mid C)$. Il y a aussi quelques autres éléments ajoutés pour tenir compte de certaines particularités de la RI : $P_{avg}(t_i)$ est la probabilité moyenne du terme t_i dans les documents qui le contiennent; $\hat{R}(t_i, D)$ est une fonction de « risque » déterminée comme suit :

$$\hat{R}(t,D) = \frac{1}{1+\bar{f}_t} \times \left(\frac{\bar{f}_t}{1+\bar{f}_t}\right)^{tf(t,D)} \quad (17)$$

où $\bar{f}_t$ est la fréquence moyenne du mot *t* dans les documents qui le contient. L'élément $P_{avg}(t_i)^{\hat{R}(t_i,D)}$ est ajouté pour contrer le « risque » de l'estimation $P_{ML}(t_i \mid D)$ en tenant compte de $P_{avg}(t_i)$ calculée sur le corpus. Ceci ressemble à un lissage par interpolation.

Le fait de considérer les mots absents de la requête est intéressant : il permet de faire la différence entre un document qui couvre beaucoup de sujets et un autre document qui ne couvre que le sujet de la requête, donc l'aspect spécificité du document. Cependant, on peut facilement voir que si les termes qui n'apparaissent pas dans la requête sont nombreux (ce qui est le cas en général), le calcul devient très complexe.

4.2. Modèle de Hiemstra et al. et de Miller et al.

Hiemstra et al. [Hiemstra98] utilisent le même principe de génération de la requête par le document. La formule que Hiemstra propose est la suivante :

$$\begin{aligned} &Score(D,Q) = P(D \mid Q) = P(D \mid t_1 t_2 ... t_n) \\ &= P(D)\frac{P(t_1 t_2 ... t_n \mid D)}{P(t_1 t_2 ... t_n)} \end{aligned} \tag{18}$$

Il suppose que $P(t_1 t_2 ... t_n)$ est une constante 1/*c*, et que les mots dans la requête sont indépendants. On a donc :

$$Score(D,Q) = cP(D)\prod_{t \in Q} P(t_i \mid D) \tag{19}$$

Pour $P(t_i \mid D)$, Hiemstra utilise une approche d'interpolation :

$$P(t_i \mid D) = \alpha P_{ML}(t_i \mid D) + (1-\alpha)P_{ML}(t_i \mid C) \tag{20}$$

L'estimation de vraisemblance maximale de $P_{ML}(t_i|D)$ et $P_{ML}(t_i|C)$ sont respectivement $\frac{tf(t_i,D)}{|D|}$ et $\frac{df(t_i)}{\sum_{t_j \in V} df(t_j)}$, où $df(t_i)$ est le nombre de documents contenant t_i, et V est le vocabulaire d'indexes.

Une fois on prend le logarithme, on obtient :

$$\begin{aligned} &\log(P(t_1 t_2 ... t_n \mid D)) = \log \prod_{t \in Q} P(t_i \mid D) \\ &= \sum_{i=1}^{n} \log(\alpha P_{ML}(t_i \mid D) + (1-\alpha)P_{ML}(t_i \mid C)) \\ &= \sum_{i=1}^{n} \log(1 + \frac{\alpha P_{ML}(t_i \mid D)}{(1-\alpha)P_{ML}(t_i \mid C)}) + \sum_{i=1}^{n} \log((1-\alpha)P_{ML}(t_i \mid C)) \end{aligned} \tag{21}$$

On remarque que le dernier composant de cette formule ne dépend pas de document, mais seulement de la requête et le corpus C. Ainsi, c'est une constante que l'on peut ignorer pour la fin de classer les documents.

En ce qui concerne la valeur de α, la façon la plus simple consiste à déterminer une constante pour α. Mais en principe, ce paramètre peut bien dépendre du document ou de la requête. Ainsi, une façon plus sophistiquée consiste à estimer une valeur de α en utilisant un processus d'optimisation automatique tel la maximisation de l'espérance (EM).

La probabilité a priori d'un document *D* est estimée comme suit :

$$P(D) = \frac{|D|}{|C|}$$

En somme, on a :

$$\log Score(Q,D) = \sum_{t \in Q} \log(1 + \frac{\alpha * tf(t,D) \sum_{t_i \in V} df(t_i)}{(1-\alpha)|D| df(t)}) + \log \frac{|D|}{|C|} \quad (22)$$

On pourrait estimer la probabilité a priori conditionnée sur autres propriétés des documents, par exemple la forme de URL ou nombre des liens. Cette approche s'est avérée très efficace dans une application RI particulière - le recherche des pages d'entrée (*entry page*) [Kraaij02].

Miller et al. [Miller98, Miller99] utilise une formulation similaire, à quelques détails près. La plus grande différence entre le modèle de Miller et celui de Hiemstra est la façon d'implanter le modèle : Miller utilise un modèle de Markov caché à deux états comme suit :

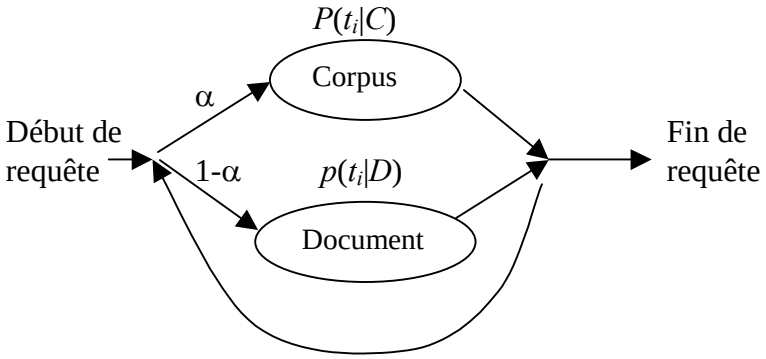

Figure 1. Modèle de Markov caché à deux états

Ce modèle correspond à la formule suivante :

$$P(Q \mid D \text{ est pertinent}) = \prod_{t_i \in Q} \alpha P_{ML}(t_i \mid D) + (1-\alpha) P_{ML}(t_i \mid C) \quad (23)$$

ce qui est similaire au modèle de Hiemstra (équation 20).

Bien que Hiemstra et Miller aient tous utilisé le lissage par interpolation, leur intention initiale n'était pas pour traiter la clairesemence de données, mais pour mieux modéliser la requête, dans laquelle certains mots peuvent être non-pertinents. Ainsi, dans la formulation de Hiemstra, un autre élément *tf*(*t*, *Q*) – la fréquence du terme *t* dans la requête *Q* - est inséré pour tenir compte de ce fait. Dans notre présentation, nous n'en avons pas décrit les détails.

5. La RI comme ratio de vraisemblance

Ng [Ng99, Ng00] a proposé une autre variante de modèle de langue basée sur le ratio de vraisemblance. Au lieu d'estimer la probabilité de pertinence d'un document pour une requête, Ng propose d'utiliser la vraisemblance comme critère de classement. L'idée qu'il avance est la suivante : Si la vraisemblance d'un document augmente après que la requête est soumise, le document a une plus forte probabilité d'être utile qu'un autre document dont la vraisemblance ne change pas ou décroît.

Ng propose la formule suivante pour mesurer cette quantité :

$$LR(D,Q) = \frac{P(D \mid Q)}{P(D)} = \frac{P(Q \mid D)}{P(Q)} \tag{24}$$

La probabilité $P(Q|D)$ et $P(Q)$ sont toutes les deux modélisées comme une distribution multinomiale. La requête Q est considérée comme une suite de mots indépendants. Ainsi, Ng utilise le lissage Good-Turing pour $P(Q)$:

$$P(Q) = \prod_{t \in Q} P_{GT}(t) = \prod_{t \in Q} P_{GT}(t \mid C) \tag{25}$$

Pour $P(t|D)$, l'estimation Good-Turing est mal adaptée à cause de la longueur limitée du document. Ainsi, Ng utilise l'interpolation suivante :

$$P(Q \mid D) = \prod_{t \in Q} P(t \mid D) = \prod_{t \in Q} \alpha P_{ML}(t \mid D) + (1-\alpha) P_{GT}(t \mid C) \tag{26}$$

La formule du score de document est donc la suivante :

$$Score(D,Q) = \sum_{t \in Q} \log\left(\frac{\alpha P_{ML}(t \mid D) + (1-\alpha) P_{GT}(t \mid C)}{P_{GT}(t \mid C)}\right) \tag{27}$$

Le paramètre α est estimé avec l'algorithme EM.

Il est assez facile de comparer le modèle de Ng et celui de Hiemstra pour montrer leur grande similarité. En effet, le modèle de Ng peut être récrit comme suit :

$$Score(D,Q) = \sum_{t \in Q} \log\left(1 + \frac{\alpha P_{ML}(t \mid D)}{(1-\alpha) P_{GT}(t)}\right) + \sum_{t \in Q} \log(1-\alpha) \tag{28}$$

Si on compare cette formule avec celle de Hiemstra, on peut voir facilement la similarité.

6. Modèle basé sur l'entropie croisée

Une variante du modèle, basé sur le ratio de vraisemblance, est de représenter la RI comme une entropie croisée (ou écart d'entropie). Nous allons montrer comment nous pouvons effectivement passer de l'équation du ratio vers une forme entropique.

Reprenons l'équation (24), on suppose comme d'habitude, l'indépendance des termes, en considérant une fonction logarithmique et après quelques transformations, cette équation peut s'écrire de la façon suivante :

$$LR(Q \mid D) = \log \frac{P(Q \mid M_D)}{P(Q \mid M_C)} \tag{29}$$

Supposons une distribution multinomiale de termes dans le document et dans le corpus, nous avons :

$$\begin{aligned} P(Q \mid M_D) &= \frac{|Q|!}{\prod_{t_i \in Q} tf(t_i, Q)!} \prod_{t_i \in Q} P(t \mid D)^{tf(t_i, Q)} \\ P(Q \mid M_C) &= \frac{|Q|!}{\prod_{t_i \in Q} tf(t_i, Q)!} \prod_{t_i \in Q} P(t \mid C)^{tf(t_i, Q)} \end{aligned} \tag{30}$$

où $|Q|$ est la taille de la requête (le nombre d'occurrences de mots). Ainsi, nous avons:

$$LR(Q \mid D) = \sum_{i=1}^{n} tf(t_i, Q) * \log \frac{\alpha P(t_i \mid M_D) + (1-\alpha) P(t_i \mid M_C)}{P(t_i \mid M_C)} \tag{31}$$

où *P(Q|Mc)* est la probabilité générative de la requête étant donné un modèle de langage estimé sur un corpus *C* et $tf(t_i, Q)$ est la fréquence du terme t_i dans a requête.

Pour chaque terme de la requête, le ratio de vraisemblance mesure le rapport entre la probabilité d'observer une requête donnée étant donné un modèle de document sur la probabilité d'observer cette requête étant donné le modèle de la collection.

Les scores dans l'équation dépendent de la longueur de la requête, ils peuvent être facilement normalisés en les divisant par la longueur de requête (comme la longueur est constante, elle n'intervient pas dans le score). La forme normalisée de l'équation 31 peut donc s'exprimer comme suit :

$$\begin{aligned} NLR(Q \mid D) &= \log \frac{P(Q \mid M_D)}{P(Q \mid M_C)} \\ &= \sum_{i=1}^{n} \frac{tf(t_i, Q)}{\sum_i tf(t_i, Q)} * \log \frac{\alpha P(t_i \mid M_D) + (1-\alpha) P(t_i \mid M_C)}{P(t_i \mid M_C)} \end{aligned} \tag{32}$$

L'étape suivante est de considérer le rapport $\frac{tf(t_i, Q)}{\sum_i tf(t_i, Q)}$ comme une estimation du maximum de vraisemblance de la distribution de probabilité représentant la requête $P(t_i|M_Q)$. L'équation 32 peut être réinterprétée comme la relation entre les deux distributions de probabilité $P(t|M_D)$ *et* $P(t|M_Q)$, normalisées par la troisième distribution $P(t|M_C)$:

$$NLR(Q \mid D) = \sum_{i=1}^{n} P(t_i \mid Q) * \frac{P(t_i \mid M_D)}{P(t_i \mid M_C)} \tag{33}$$

Le modèle mesure de combien le modèle de document pourrait coder des événements du modèle de requête mieux que le modèle de corpus. En théorie de l'information, ceci est interprété comme une différence entre deux entropies croisées, exprimée comme suit :

$$NLR(Q \mid D) = H(X \mid C) - H(X \mid D) \qquad (34)$$

où X est une variable aléatoire avec la distribution de probabilité $P(t_i)=P(t_i|M_Q)$ et C et D sont des fonctions de masse de probabilité représentant respectivement la distribution marginale (du corpus) et le modèle du document.

L'entropie croisée permet donc de mesurer en moyenne l'écart entropique sur le fait que le modèle de document correspond (suit) bien la distribution probabiliste de la requête. D'autres formes entropie croisées ont été étudiées, on y trouve notamment celle basée sur l'information de Kullback-Leibler dans [lavrenko01].

Remarquons que dans l'entropie croisée, on voit apparaître implicitement le modèle de langue de la requête M_Q. Dans la section 3, nous avons mentionné le principe d'utiliser un modèle de langue pour la requête. Si on l'utilise seul (ou pour remplacer le modèle de langue du document), l'approche souffrira encore plus du problème de l'information limitée pour la construction du modèle. Dans le cas de RI pour des requête ad hoc (comme sur le Web), typiquement, la requête est très courte. Cette approche est donc impraticable. Dans le contexte de « topic tracking » où la « requête » - un texte – est plus longue, on peut davantage utiliser cette approche. Cependant, sa performance n'est pas très bonne. Dans l'approche d'entropie croisée, comme celle utilisée dans le modèle de pertinence [Lavrenko01], le modèle de langue de la requête n'est pas utilisé seul, mais ensemble avec le modèle du corpus et du document, dans un cadre qui tente de modéliser la pertinence.

Remarquons aussi que le principe de comparaison du modèle du document et le modèle de la requête est aussi utilisé dans le cadre de « minimisation de risque » [Lefferty01, Zhai02]. La décision pour retrouver un document est basée sur une fonction de perte comparant les deux modèles. Ils ont ensuite proposé un modèle de langue à deux étapes : une première étape pour lisser le modèle de document (pour le problème de clairesemence de données), et une deuxième étape pour tenir compte de la pertinence des termes dans la requête (le modèle de requête). Les techniques spécifiques utilisées pour les deux étapes sont respectivement le lissage Dirichlet et le lissage par interpolation.

7. La RI comme traduction statistique

Une idée toujours dans la même veine que les approches basées sur les modèles de langue considère que la RI est un processus de traduction particulier [Berger99] : On considère qu'un besoin d'information est un message à transmettre. Ce message est d'abord formulé par une requête. Le processus de recherche est considéré comme un processus de traduction, qui traduit la requête en un document. Plus cette relation de traduction est probable, plus le document peut être un document pertinent à la requête. Ainsi, on tente de déterminer les documents qui maximisent la probabilité de traduction $P(D|Q)$.

Cette idée peut aussi être exprimée avec la métaphore de canal bruité :

1. L'utilisateur a un besoin d'information B;
2. À partir de ce besoin, l'utilisateur imagine un document idéal D_B;
3. Il choisit un ensemble de mots de D_B et génère une requête Q. Cette requête est transmise par un canal de transmission bruité. Le message reçu est corrompu. On tente d'estimer la probabilité que ce message corrompu est la requête.

Cette idée est directement basée sur celle de la traduction statistique où la tâche consiste à déterminer la probabilité de traduction $P(\mathbf{e}|\mathbf{f})$ pour une paire de phrase en anglais **e** et en français **f**.

Pour la traduction statistique, il existe une famille de modèles, appelés des modèles IBM, pour déterminer la probabilité de traduction. Ces modèles intègre d'abord la traduction de mots (i.e. $t(e|f)$), et éventuellement la correspondance des positions entre le mot source dans la phrase source et sa traduction dans la phrase cible ainsi que la fertilité (le nombre de mots pour traduire un mot source). Nous ne donnons pas plus de détails ici.

Appliquons le même principe à la RI, nous avons :

$$P(D \mid Q) = \frac{P(Q \mid D)P(D)}{P(Q)} \propto P(Q \mid D)P(D) \tag{35}$$

La probabilité $P(D)$ peut être calculée assez facilement (par exemple, selon les termes contenus dans le document, ou assigné une valeur identique à tous les documents). Pour le calcul de $P(Q|D)$, Berger et Lafferty utilisent un processus inspiré de la traduction statistique [Brown93], notamment le modèle IBM 1. Afin d'estimer les paramètres de ce modèle, Berger et Lafferty ont utilisé des phrases prises d'un document comme une requête qui tente de trouver ce document comme la réponse.

8. Discussions

Bien que cette classe de modèles soit toujours en cours de développement et d'investigation, les travaux et les expérimentations effectués dans ce thème ont montré que les modèles de langage fournissent un cadre intéressant et prometteur pour appréhender la RI. De plus, même si différents modèles ont été proposés, l'idée principale reste effectivement la suivante : « calculer la probabilité que la requête soit générée par le modèle de langage du document ». Ce point est évidemment partagé par l'ensemble des modèles.

Une analyse comparative des différentes approches proposées montre que les premières d'entre elles ont omis la notion de pertinence. Une justification possible consiste à considérer $P(Q|D)$ comme une version probabiliste de l'implication logique. Cette implication a été mise en évidence par Van Rijsbergen [Rijsbergen86] qui a effectivement démontré que la recherche d'information peut être vue comme le problème de calculer la probabilité qu'un document implique la requête : $P(D \rightarrow Q)$. Ceci étant, les expérimentations effectuées dans ces modèles montrent qu'ils permettent effectivement de « capturer » la pertinence. Cette pertinence a été introduite dans des modèles plus récents, on cite notamment [Lavrenko01] dans le cadre de la réinjection de la pertinence.

L'approche par traduction présente un intérêt particulier d'un point de vue théorique, parce qu'elle intègre la polysémie et la synonymie dans le modèle de RI en modélisant explicitement les relations entre les termes. Ce modèle peut être utilisé pour la RI monolingue aussi bien que pour la RI translinguistique. Bien que les résultats obtenus par ce modèle soient mitigés, l'approche est tout de même intéressante dans le cadre d'une conceptualisation des termes.

Les approches avec les modèles de langues ne sont pas totalement indépendantes des approches plus traditionnelles. Hiemstra [Hiemstra01] a montré que son modèle peut se comparer avec le modèle vectoriel. Il fournit aussi une justification probabiliste au modèle vectoriel. Son modèle a été étendu pour la recherche d'information par croisement de langue. Le modèle de [Miller99] se différencie des modèles précédents sur le plan de l'implémentation, parce qu'il utilise les modèles cachés de Markov. La contribution la plus importante du modèle de Ng [Ng99, Ng00] est qu'elle prouve que le terme de $P(Q)$ dans le dénominateur est essentiel pour normaliser les scores. Ng a également montré que son modèle de paramètres (y compris d'expansion de question) pourrait être optimisé localement (c.-à-d. sans utiliser de collection externe). Ceci

est évidemment important, dans le cas applicatif, parce que les paramètres dépendant de la collection ne peuvent pas être généralisés à toutes les collections.

Un problème important, qui a été largement étudié par les modèles classiques [Robertson94], [Singhal96] mais reste marginal dans les modèles de LM est celui de la normalisation de la longueur document. Ce point a été largement étudié par les modèles classiques [Robertson94], [Singhal96] mais reste marginal dans les modèles de LM. Bien que différentes stratégies aient été proposées [Hiemstra98a], [Miller98], aucune approche n'a permis de traiter effectivement ce problème. Une explication possible est qu'il n'y a aucune théorie pour la prise en compte de la structure interne de document. Un document est tout simplement considéré comme des ensembles de mots (« sac de mots »). En réalité, les documents longs traitent souvent plusieurs thèmes ou encore ils décrivent longuement un thème (document verbeux), ou les deux.

La plupart des méthodes présentées ici utilisent des modèles uni-gramme. Il est évident que l'hypothèse d'indépendance entre les termes de la requête est une simplification qui facilite grandement le calcul. Cependant, cette simplification dérive aussi de la réalité : les mots sont en réalité reliés. Il est assez naturel de vouloir étendre les modèles uni-grammes à des modèles d'ordre supérieur comme bi-grammes. C'est ainsi que Miller et al et [Song et Croft] ont effectué des expérimentations. Cependant, bien que théoriquement très attrayants, les modèles bi-grammes ont un plus grand nombre de paramètres à estimer, et ils doivent encore prouver leur performance en pratique.
Les études effectuées jusqu'à maintenant ont prouvé que les approches basées sur les modèles de langue peuvent produire d'aussi bons, souvent meilleurs, résultats que les approches traditionnelles comme les modèles probabilistes (Okapi) ou vectoriels. De plus, un grand avantage de ces approches réside dans son cadre théorique bien défini. Comme les études ont démontré, ce cadre est assez général pour pourvoir générer des approches traditionnelles, ou pour fournir une explication aux paramètres ad hoc ou empiriques dans ces approches. On doit donc s'attendre beaucoup de développements futurs, et on est juste au début de l'ère modèle de langue en RI.

Références

[Brown93] Peter F. Brown, Stephen A. Della Pietra, Vincent J. Della Pietra and Robert L. Mercer, The Mathematics of Statistical Machine Translation: Estimation, *Computational Linguistics*, 19(2): 263--311, 1993.

[Berger99] A. Berger and J. Lafferty, Information Retrieval as Statistical Translation, *Research and Development in Information Retrieval*, *Proc. ACM-SIGIR'99*, pp. 222-229, 1999.

[Hiemstra98] Djoerd Hiemstra, A linguistically motivated probabilistic model of information retrieval, dans Christos N and Stephanides C. (eds), *Proc. European Conference of Digital Library (ECDL98),* Sept. 1998, Springer Verlag.

[Hiemstra99] Djoerd Hiemstra and Wessel Kraaij, Twenty-One at TREC-7: Ad Hoc and Cross Language track, *TREC7*, pp. 227-238, 1998.

[Jelinek97] Frederick Jelinek, *Statistical Methods for Speech Recognition*, MIT Press, 1997.

[Kraaij02] W. Kraaij, T. Westerveld and D. Hiemstra, The Importance of Prior Probabilities for Entry Page Search, *Research and Development in Information Retrieval, Proc ACM-SIGIR'2001*

[Lafferty01] John Lafferty and C. Zhai, Language Models, Query Models, and Risk Minimization for Information Retrieval, *Research and Development in Information Retrieval, Proc ACM-SIGIR'2001*, pp.111-119, 2001.

[Lavrenko01] V. Lavrenko and W.B. Croft, Relevance-based Language Models, *Research and Development in Information Retrieval, Proc ACM-SIGIR'2001*, pp. 120-127, 2001.

[Manning99] Chris Manning and Hinrich Schütze, *Foundations of Statistical Natural Language Processing*, MIT Press. Cambridge, MA: May 1999.

[Maron60] M.E. Maron and J.L. Kuhns, On relevance, probabilistic indexing and information retrieval, *Journal of the Association for Computing Machinery,* 7: 216-244, 1960.

[Miller98] David R.H. Miller, Tim Leek and Richard M. Schwartz, BBN at TREC-7: Using Hidden Markov Models for Information Retrieval, *TREC7*, pp. 133-142, 1998.

[Miller99] David R. H. Miller and Tim Leek and Richard M. Schwartz, A Hidden Markov Model Information Retrieval System, *Research and Development in Information Retrieval Proc. ACM-SIGIR*, pp. 214-221, 1999.

[Ng99] Kenney Ng, A Maximum Likelihood Ratio Information Retrieval Model, *TREC8*, pp. 483-492, 1999.

[Ponte98] Jay M. Ponte and W. Bruce Croft, A Language Modeling Approach to Information Retrieval. *Research and Development in Information Retrieval*, *Proc. ACM-SIGIR*, pp. 275-281, 1998.

[Risbergen86] C. J. van Rijsbergen, A Non-Classical Logic for Information Retrieval, *The Computer Journal,* 29(6): 481-485, 1986.

[Robertson94] Robertson, S. and Walker, S. Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval. *Research and Development in Information Retrieval, ACM-SIGIR,* pp. 232-241, 1994.

[Singhal96] A. Singhal, C. Buckley, and M. Mitra. Pivoted document length normalization. *Research and Development in Information Retrieval, ACM-SIGIR'96*, pp. 21-29, 1996.

[Song99] Fei Song and W. Bruce Croft, A General Language Model for Information Retrieval, *Proceedings of the eighth international conference on Information and knowledge management (CIKM),* pp. 316-321, 1999.

[Zhai02] ChengXiang Zhai and John Lafferty, Two-Stage Language Models for Information Retrieval. Proc. *Research and Development in Information Retrieval, ACM-SIGIR,* pp. 49-56, 2002.