Étude de techniques de lissage pour des automates probabilistes à états finis et déterministes

Catherine Kobus, Géraldine Damnati et Lionel Delphin-Poulat

France Télécom R&D
2 avenue Pierre Marzin 22307 Lannion - FRANCE
{catherine.kobus,geraldine.damnati,lionel.delphinpoulat}@rd.francetelecom.com

ABSTRACT

In this article we propose to adapt classical smoothing techniques in order to smooth probabilistic deterministic finite state automata (PDFA), automatically inferred with the MDI (Minimal Divergence Inference) grammatical inference algorithm [6]. In a first approach, these automata are used as the first-pass language model in a speech recognition system, hence they need to be smoothed. Smoothing is necessary to deal with zero probability cases when using the automaton as a parser. We propose two types of smoothing techniques, at the state level, derived from those used to smooth *n-gram* models, called *epsilon* smoothing and *Absolute Discounting*. Finally we present the results obtained with the proposed smoothing methods.

1. INTRODUCTION

Dans cet article, nous proposons d'adapter des techniques de lissage exploitées classiquement pour des probabilités de succession de mots, à l'estimation des probabilités de transition d'automates probabilistes, à états finis et déterministes. Dans cette étude, les automates sont inférés automatiquement avec l'algorithme d'inférence grammaticale MDI (Minimal Divergence Inference) [6]. Ils ont été utilisés comme modèle de langage en première passe d'un décodeur dans un système de reconnaissance vocale. Un lissage doit leur être appliqué afin qu'ils puissent assigner une probabilité non nulle à une phrase non acceptée *a priori* par l'automate. Certaines méthodes ont déjà été proposées pour le lissage de ce type d'automates. La solution la plus simple consiste à interpoler le modèle à base d'automate avec un modèle *unigramme* ; le problème de cette méthode est qu'elle tient compte de l'*unigramme*, à faible pouvoir prédictif, même si la phrase est correctement acceptée par l'automate.
Une technique à base de correction d'erreurs a été proposée [3]. Elle consiste à rechercher dans l'automate la phrase la plus proche de la phrase cible. Cette technique fournit de meilleurs résultats que la précedente.
Un lissage par seuil a également été proposé [7]. Une masse de probabilités est retirée à chaque transition. Quand l'automate ne reconnaît pas une phrase, la probabilité est calculée en utilisant la probabilité totale décomptée. L'inconvénient de cette méthode est qu'il achève l'analyse de la phrase avec un modèle *unigramme*.
Dans [5], le modèle *n-gramme* est modélisé par un automate stochastique, dans lequel chaque état représente un historique donné v (associé à un *m-gramme*, $m < n$). Les probabilités d'apparition d'un mot étant donné l'historique sont lissées avec des techniques classiques de "discounting". Dans cet article, nous proposons de lisser les probabilités d'un état étant donné un autre état (modélisant lui-aussi un historique). L'évènement non vu n'est pas l'absence d'un mot étant donné un historique mais une succession non vue de deux états (en analogie avec une succession non vue de mots dans le modèle *bigramme*).
Dans cet article, nous rappelons le principe de l'inférence grammaticale et résumons l'algorithme MDI, utilisé pour obtenir les automates. Puis, les techniques de lissage proposées (le lissage par seuil et le lissage de type *Absolute Discounting*) [4] sont détaillées. Enfin, nous présenterons le cadre des expérimentations et les résultats obtenus en appliquant ces techniques de lissage.

2. DESCRIPTION DES AUTOMATES

2.1. Principe de l'inférence grammaticale

L'inférence grammaticale vise à identifier ou à approximer un langage à partir d'exemples d'éléments appartenant à ce langage ou de contre-exemples. Une méthode utilisée en inférence grammaticale consiste à apprendre par coeur les données d'apprentissage puis à les généraliser en fusionnant des états de l'automate. Dans un premier temps, l'arbre accepteur des préfixes est construit (l'arbre accepteur des préfixes est un automate construit à partir du corpus d'apprentissage, acceptant uniquement les chaînes du corpus, et dans lequel les préfixes communs sont fusionnés). Ensuite, tant qu'il existe des états à fusionner, ces derniers sont fusionnés puis l'automate déterminisé.

2.2. L'algorithme MDI

L'algorithme MDI vise à inférer des automates probabilistes à états finis et déterministes, en essayant de minimiser à la fois la taille de l'automate et la divergence entre la distribution obtenue à une itération donnée et la distribution de probabilités de la séquence d'apprentissage (qui correspond à celle de l'arbre accepteur de préfixes). La divergence entre deux distributions de probabilités P_A et $P_{A'}$ est calculée à partir de la *divergence de Kullback-Leibler* [2], notée $D(P_A||P_{A'})$. Cette définition de la divergence peut s'étendre aux automates A et A' induisant ces distributions de probabilités [1], elle est notée $D(A||A')$.
À une étape donnée de l'algorithme, soient A_1 la solution temporaire de l'algorithme d'inférence et A_2 le nouvel automate proposé à cette étape et dérivé de l'automate A_1 après fusion de deux états. La différence des divergences par rapport à l'automate initial A_0 est calculée :

$$\Delta(A_1, A_2) = D(A_0||A_2) - D(A_0||A_1) \qquad (1)$$

La nouvelle solution proposée A_2 est considérée comme compatible avec les données d'apprentissage si l'augmentation de la divergence relativement à la réduction de la taille de l'automate (*ie*. le nombre d'états noté $|A|$) est suffisamment petite. Soit α le seuil de compatibilité. L'automate A_2 est compatible si :

$$\frac{\Delta(A_1, A_2)}{|A_1| - |A_2|} < \alpha \tag{2}$$

L'algorithme MDI propose un nouvel automate solution tant que le critère de compatibilité est satisfait. Des techniques de lissage sont ensuite appliquées aux automates ainsi obtenus.

3. LES TECHNIQUES DE LISSAGE

3.1. Principe du lissage

Des techniques de lissage doivent être appliquées aux automates de mots afin qu'ils puissent être utilisés comme modèle de langage dans un système de reconnaissance vocale. En effet, un modèle de langage doit pouvoir attribuer une probabilité non nulle à un évènement non vu dans le corpus d'apprentissage. Le lissage consiste à retrancher une certaine masse de probabilités aux évènements observés afin de la redistribuer parmi les évènements non vus.
Dans le cas d'un modèle *n-gramme*, l'évènement observé est une succession de n mots ; dès qu'une telle succession n'a pas été observée, une probabilité de lissage est appliquée. Dans le cas d'un modèle à automate de mots, en analogie avec le modèle *bigramme*, l'évènement considéré est une succession de deux états dans l'automate. Ainsi, il est possible de faire une analogie entre les techniques de lissage pouvant être utilisées pour le modèle *bigram* et celles utilisées pour l'automate de mots.
Dans les formules de lissage, on considère les comptes des différents états et transitions. Ainsi, $c(q_i)$ désigne le compte d'un état, à savoir, le nombre de fois où l'état q_i est atteint ; $c(q_i, q_j)$ représente le compte de la transition reliant l'état q_i à l'état q_j, *ie*. le nombre de fois où la transition $q_i \rightarrow q_j$ est empruntée. Par définition, l'estimée par maximum de vraisemblance (MV) de la probabilité d'une transition particulière $q_i \rightarrow q_j$ est donnée par la formule :

$$P^{MV}(q_j|q_i) = \frac{c(q_i, q_j)}{c(q_i)} \tag{3}$$

Pour introduire cette distribution de probabilités entre les états dans l'automate, une transformation est nécessaire. A l'origine, dans l'automate, les transitions émettent les mots avec leur probabilité associée. La probabilité d'émettre le mot w_k dans l'état q_i (permettant alors d'atteindre l'état q_j) s'écrit :

$$P(w_k|q_i) = \frac{c(w_k, q_i \rightarrow q_j)}{c(q_i)} \tag{4}$$

Considérons un état q_i. Pour chacun de ses états successeurs q_j, un état intermédiaire q_{ij} est créé ; l'état q_i est connecté à l'ensemble des états intermédiaires q_{ij} ; la probabilité $P^{MV}(q_j|q_i)$ est appliquée sur la transition $q_i \rightarrow q_{ij}$. La figure 1 montre la transformation faite sur les transitions. Ces états intermédiaires q_{ij} sont eux-mêmes reliés aux états q_j successeurs de q_i, les transitions concernées émettant les mots w_k avec une probabilité associée $P'(w_k|q_i)$ égale à la probabilité d'émission initiale normalisée par la probabilité de transition totale.

$$P'(w_k|q_i) = \frac{P(w_k|q_i)}{P^{MV}(q_j|q_i)} \tag{5}$$

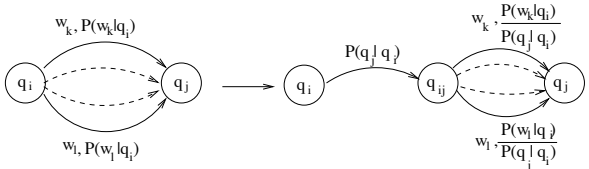

FIG. 1: Transformation de la transition

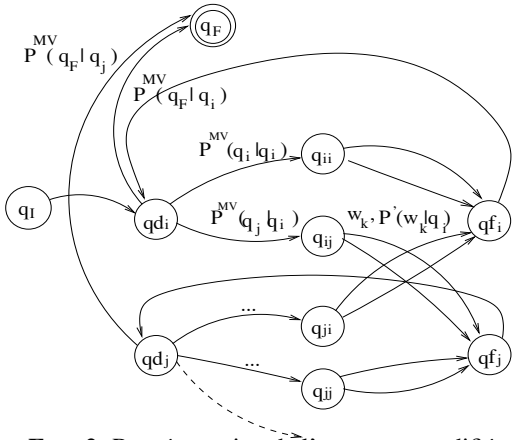

FIG. 2: Représentation de l'automate modifié

Le réseau ainsi transformé peut être vu en deux parties : l'une concernant les transitions entre les états et l'autre concernant plutôt l'aspect émission de mots. De cette manière, le lissage portera sur les probabilités $P(q_j|q_i)$ et laissera inchangée la partie de l'automate émettant les mots.
La figure 2 représente l'allure du réseau après modifications. Pour une question de commodité de représentation, tous les états q ont été dupliqués en un état qd représentant l'état q en tant qu'état de départ et en un état qf le représentant en tant qu'état d'arrivée. Ces derniers sont reliés par une transition vide. L'état intermédiaire q_{ij} se trouve donc entre l'état qd_i et l'état qf_j. L'état initial (respectivement final) est noté q_I (respectivement q_F).
Le lissage est réalisé par l'intermédiaire d'un état supplémentaire $q_{Lissage}$. Soient deux états q_m et q_n non connectés initialement. La succession $(q_m \rightarrow q_n)$ sera réalisée à l'aide de l'état de lissage avec une transition $q_m \rightarrow q_{Lissage}$ et autant de transitions $q_{Lissage} \rightarrow q_{jn}$ qu'il y a d'états intermédiaires q_{jn} permettant d'atteindre q_n.

3.2. Lissage par seuil ou lissage epsilon

Ce lissage est l'un des plus simples dans sa conception. Le lissage par seuil consiste à attribuer une même faible probabilité ϵ à toutes les successions non vues d'états. Ce type de lissage n'est pas très performant dans la mesure où il ne fournit pas un modèle discriminant mais demeure intéressant pour sa simplicité. Les formules de lissage par seuil sont les suivantes :

$$P^{\epsilon}(q_j|q_i) = \begin{cases} P^{MV}(q_j|q_i) \cdot [1 - \epsilon \cdot C_0(q_i)] & \text{si } c(q_i, q_j) \neq 0 \\ \epsilon \text{ sinon} \end{cases} \tag{6}$$

où $C_0(q_i)$ représente le nombre de non successeurs de q_i.
La figure 3 montre l'automate lissé par seuil. Tous les états de sortie qf_i sont réliés avec une probabilité ϵ à l'état de lissage q_{Eps}, lui-même relié à tous les états intermédiaires q_{ij} avec une transition vide.

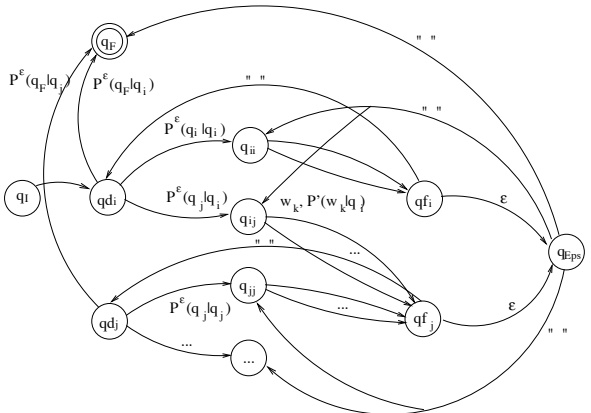

FIG. 3: Représentation de l'automate après lissage ϵ

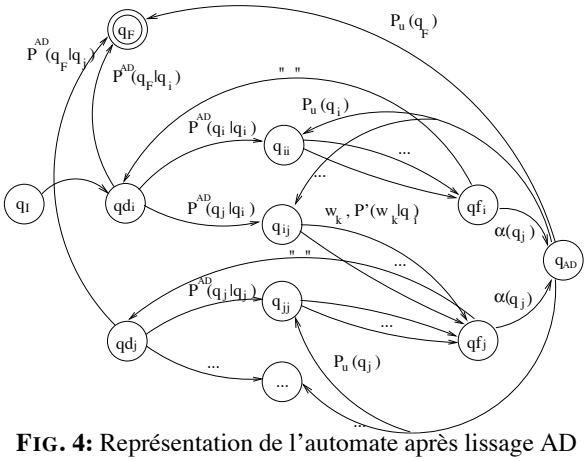

FIG. 4: Représentation de l'automate après lissage AD

3.3. Lissage de type Absolute discounting

Le lissage de type *Absolute discounting* (AD) retranche une valeur constante d au décompte des évènements observés. Les probabilités attribuées aux évènements non vus ne sont plus égales mais dépendent des états initiaux et finaux de la transition. Ce type de lissage est donc plus discriminant que le lissage par seuil et est préféré à ce dernier. Les formules de lissage de type AD se déduisent de la condition de normalisation :

$$P^{AD}(q_{ij}|q_i) = \begin{cases} P^{MV}(q_j|q_i) - \frac{d}{c(q_i)} & \text{si } c(q_i,q_j) \neq 0 \\ \alpha(q_i) \cdot P_u(q_j) & \text{sinon} \end{cases} \quad (7)$$

où $P_u(q_j)$ représente la probabilité *unigramme* de l'état q_j, elle est définie comme suit :

$$P_u(q_j) = \frac{c(q_j)}{N_T} \quad (8)$$

où N_T représente le nombre total de passages dans les états de l'automate et :

$$\alpha(q_i) = \frac{\gamma(q_i)}{\sum_{j/c(q_i,q_j)=0} P_u(q_j)} \text{ et } \gamma(q_i) = d \cdot \frac{N_{succ}(q_i)}{c(q_i)} \quad (9)$$

où $N_{succ}(q_i)$ désigne le nombre de successeurs de q_i.
La décomposition de la probabilité lissée en deux facteurs $\alpha(q_i)$ (ne dépendant que de l'état de départ) et $P_u(q_j)$ (ne dépendant que de l'état d'arrivée) permet la décomposition de la probabilité sur deux transitions, respectivement $qf_i \rightarrow q_{AD}$ et $q_{AD} \rightarrow q_{mj}$. La redistribution de la masse de probabilité réalise un compromis entre le nombre de passages dans un état q_i et la variété de ses successeurs. En ce sens, il est plus discriminant que le simple lissage par seuil. Ceci rejoint les considérations sur la perplexité faites dans la section suivante.
La figure 4 montre la forme de l'automate après lissage de type AD de topologie identique à celle de la figure 3.

4. EVALUATION

4.1. Contexte expérimental

Les tests ont été effectués sur une application de dialogue qui permet aux utilisateurs de gérer leurs portefeuilles d'actions, en effectuant des opérations de consultation, d'achat, de vente etc. Le corpus consiste en un ensemble de requêtes des utilisateurs telles que : "Je voudrais connaître le cours de l'action ...".

Les modèles à base d'automates de mots ont été inférés à partir d'un corpus d'apprentissage constitué d'environ 24900 requêtes. Le corpus de tests comporte environ 1050 phrases. Le lexique de l'application comporte 3236 mots ; tous les noms d'actions ont été regroupés dans une même classe *a priori*, si bien que l'automate manipule 1187 symboles. Plusieurs tests ont été faits en faisant varier le seuil de compatibilité α.

4.2. Calcul de la perplexité

La qualité d'un modèle de langage en tant qu'entité autonome se mesure avec la perplexité. Différentes tailles d'automates ont été obtenues en faisant varier le seuil de compatibilité α dans l'algorithme MDI. La taille, le nombre de transitions ainsi que la perplexité des automates lissés en fonction de la valeur du seuil α sont résumés dans le tableau 1. Les modèles lissés ont été obtenus en prenant $\epsilon = 0.0001$ et $d = 0.7$.
Plus le seuil α est petit, plus la perplexité diminue ; en effet, pour de grandes valeurs de α, les automates présentent peu d'états car ils sont très généralisés et donc perdent de leur précision et de leur pouvoir prédictif. Les différentes techniques de lissage améliorent la perplexité des modèles et la baisse relative de la perplexité est d'autant plus importante pour les plus grandes tailles d'automate. Avec le lissage par AD, la baisse de la perplexité va de 12% pour de petits automates jusqu'à 31% pour le plus grand. En effet, pour les petites tailles d'automates, les états sont très connectés entre eux, le lissage a donc moins d'effet que pour des automates avec plus d'états dans lesquels beaucoup de successions d'états ne sont pas observées.
La perplexité des modèles lissés par AD est meilleure que celle des modèles lissés par seuil. Ceci était prévisible dans la mesure où le lissage par AD est plus discriminant que le lissage par seuil mais l'amélioration entre les deux lissages est plus significative avec des automates présentant plus d'états, qui seraient moins connectés entre eux.

TAB. 1: Taille des automates suivant la valeur du seuil α

α	0.003	0.002	0.001	0.0007	0.0005
Nb. états	24	38	95	158	247
PP	43.15	37.67	29.65	27.39	27.40
PP lissage ϵ	38.41	33.36	24.39	21.43	21.70
PP lissage AD	37.69	32.53	22.62	19.37	18.72

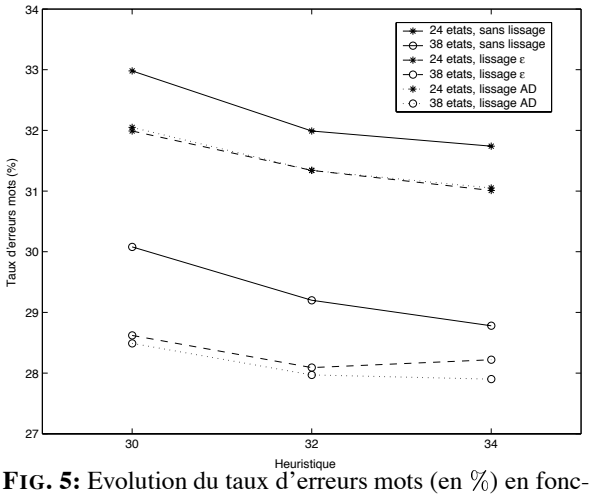

FIG. 5: Evolution du taux d'erreurs mots (en $\%$) en fonction de l'heuristique

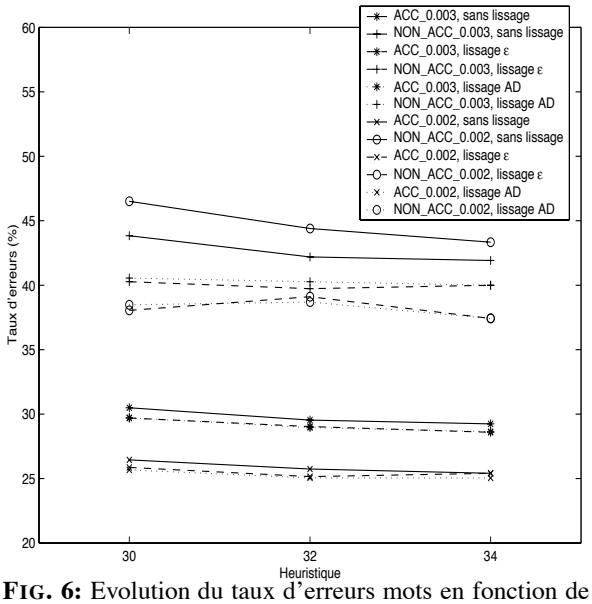

FIG. 6: Evolution du taux d'erreurs mots en fonction de l'heuristique, sur les corpus ACC_α et NON_ACC_α.

4.3. Résultats des tests de reconnaissance

Le graphique 5 montre les taux d'erreurs mots en fonction du seuil d'élagage (heuristique appliquée lors du décodage en faisceau), pour les automates lissés par seuil et pour $\alpha = 0.003$ (24 états) et $\alpha = 0.002$ (38 états). Les résultats de reconnaissance pour de plus grandes tailles d'automates nécessitent d'effectuer des optimisations afin d'obtenir des tailles de réseaux raisonnables.
Un gain est observé en termes de taux d'erreurs (jusqu'à 5% en relatif pour le plus grand automate). La faible amélioration en performances du lissage par AD (les courbes des deux types de lissage se superposent même quasiment pour l'automate à 24 états) n'est pas surprenante : en effet, le pouvoir discrimant de ce lissage (par rapport au lissage Epsilon) n'est observable que pour un nombre d'états plus important (dans les automates utilisés, les états sont presque tous connectés entre eux).
Au delà du gain observé en terme de taux d'erreurs, nous nous intéressons au gain sur les phrases non reconnues par l'automate initialement. Pour chaque valeur de α (donc pour chaque taille d'automate), le corpus de test a été divisé en deux sous-corpora ; l'un noté, ACC_α contenant les phrases du corpus, acceptées par l'automate initial et l'autre partie, notée NON_ACC_α contenant les phrases non acceptées. Le pourcentage des phrases acceptées est de 85% pour $\alpha = 0.003$ (84% pour $\alpha = 0.002$) et celui des phrases non acceptées (hors mots non vus) est de 7% pour $\alpha = 0.003$ (8% pour $\alpha = 0.002$). Les entrées restantes sont des entrées à rejeter par le système.
Le graphique 6 représente les évolutions des taux d'erreurs mots sur les corpora ACC_α et NON_ACC_α. On constate que le lissage apporte un plus grand gain pour les phrases du corpus NON_ACC_α et d'autant plus pour le plus grand automate (jusqu'à 11% de gain). De plus, il ne dégrade pas le taux sur les phrases acceptées. Les deux types de lissage ont des performances équivalentes. Il sera intéressant d'obtenir des résultats pour de plus grands automates, où les apports du lissage devraient être plus importants et le pouvoir discriminant du lissage par *Absolute Discounting* par rapport au lissage ϵ plus évident.

5. CONCLUSION

Dans cet article, nous avons présenté des techniques de lissage, analogues à celles utilisées pour le *bigramme* de mots, à des automates probabilistes, à états finis et déterministes. Ce type d'automate a été utilisé comme modèle de langage dans un système de reconnaissance vocale. Les tests effectués montrent une amélioration du taux de reconnaissance grâce au lissage, notamment sur les phrases non acceptées par l'automate initial. Des résultats doivent encore être obtenus pour de plus grands automates.
Des améliorations doivent être apportées aux modèles lissés, afin d'optimiser notamment les tailles des réseaux et de faire intervenir les mots non vus dans le lissage. Ces méthodes de lissage pourraient également s'appliquer à des automates, décrivant des structures locales dans une phrase, utilisés en combinaison avec des modèles *n-grammes* en première passe ou en seconde passe sur un graphe de mots.

RÉFÉRENCES

[1] R. Carrasco. Accurate computation of the relative entropy between stochastic regular grammars. *Theoretical Informatics and Applications*, 31(1) :437–444, 1997.

[2] T. Cover and J. Thomas. *Elements of information theory*. New York : John Wiley and sons, 1991.

[3] P. Dupont. Smoothing probabilistic automata : an error-correcting approach. In *Fifth Intel. Collo. on Grammatical Inference*, pages 51–64, 2000.

[4] H. Ney and U.Essen. On smoothing techniques for bigram-based natural language modeling. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing*, 1991.

[5] G. Riccardi, R. Pieraccini, and E. Bocchieri. Stochastic automata for language modeling. *Computer Speech and Language*, 10 :265–293, 1996.

[6] F. Thollard. *Inférence grammaticale probabiliste pour l'apprentissage de la syntaxe en traitement de langue naturelle*. PhD thesis, Université Jean Monnet, 2000.

[7] F. Thollard. Improving probabilistic grammatical inference core algorithms with post-processing techniques. In *ICML*, pages 561–568, 2001.