Université de Montréal

Intégration du contexte en traduction statistique à l'aide d'un perceptron à plusieurs couches

par

Alexandre Patry

Thèse présentée à la Faculté des arts et des sciences
en vue de l'obtention du grade de Philosophiæ Doctor (Ph.D.)
en informatique

26 août 2010

Université de Montréal
Faculté des arts et des sciences

Cette thèse intitulée :

Intégration du contexte en traduction statistique à l'aide d'un perceptron à plusieurs couches

Présentée par :

Alexandre Patry

a été évaluée par un jury composé des personnes suivantes :

Guy Lapalme
président-rapporteur et représentant du doyen

Philippe Langlais
directeur de recherche

Yoshua Bengio
membre du jury

Éric Gaussier
examinateur externe

Résumé

Les systèmes de traduction statistique à base de segments traduisent les phrases un segment à la fois, en plusieurs étapes. À chaque étape, ces systèmes ne considèrent que très peu d'informations pour choisir la traduction d'un segment. Les scores du dictionnaire de segments bilingues sont calculés sans égard aux contextes dans lesquels ils sont utilisés et les modèles de langue ne considèrent que les quelques mots entourant le segment traduit.

Dans cette thèse, nous proposons un nouveau modèle considérant la phrase en entier lors de la sélection de chaque mot cible. Notre modèle d'intégration du contexte se différentie des précédents par l'utilisation d'un PPC (perceptron à plusieurs couches). Une propriété intéressante des PPC est leur couche cachée, qui propose une représentation alternative à celle offerte par les mots pour encoder les phrases à traduire. Une évaluation superficielle de cette représentation alternative nous a montré qu'elle est capable de regrouper certaines phrases sources similaires même si elles étaient formulées différemment.

Nous avons d'abord comparé avantageusement les prédictions de nos PPC à celles d'IBM1, un modèle couramment utilisé en traduction. Nous avons ensuite intégré nos PPC à notre système de traduction statistique de l'anglais vers le français. Nos PPC ont amélioré les traductions de notre système de base et d'un deuxième système de référence auquel était intégré IBM1.

Mots clés : lexique global, traduction statistique, perceptron à plusieurs couches

Abstract

Phrase-based statistical machine translation systems translate source sentences one phrase at a time, conditioning the choice of each phrase on very little information. Bilingual phrase table scores are computed regardless of the context in which the phrases are used and language models only look at few words surrounding the target phrases.

In this thesis, we propose a novel model to predict words that should appear in a translation given the source sentence as a whole. Our model differs from previous works by its use of MLP (multilayer perceptrons). Our interest in MLP lies in their hidden layer that encodes source sentences in a representation that is only loosely tied to words. We observed that this hidden layer was able to cluster some sentences having similar translations even if they were formulated differently.

In a first set of experiments, we compared favorably our MLP to IBM1, a well known model in statistical machine translation. In a second set of experiments, we embedded our PPC in our English to French statistical machine translation system. Our PPC improved translations quality over our baseline system and a second system embedding an IBM1 model.

Keywords: global lexicon, statistical machine translation, multilayer perceptron

Table des matières

Figures

Tableaux

Algorithmes

Orthographe

Nous avons rédigé cette thèse en suivant la nouvelle réforme orthographique[1] . Ne soyez donc pas surpris de voir certains trémas passer du *e* au *u*, plusieurs î perdre leur accent et le mot *euristique* sans son *h* initial.

[1] http://www.orthographe-recommandee.info/

Remerciements

Cette thèse n'aurait pas été possible sans le soutien de ma tendre moitié Cindy et de nos deux enfants Noah et Liam. Je tiens aussi à remercier mes parents pour leur support autant moral que financier tout au long de mon aventure estudiantine. Finalement, la thèse que vous tenez entre vos mains aurait été très différente si ça n'avait été du support et de la supervision bienveillante du Professeur Langlais.

À ma petite tribu : Cindy, Noah et Liam.

1 Introduction

Traduire automatiquement un texte d'une langue naturelle (la langue source) vers une autre (la langue cible) est un problème sur lequel travaillent de nombreux chercheurs depuis le tout début de l'informatique (Hutchins, 2001). Même si de très bons systèmes existent pour la traduction de textes spécialisés comme les bulletins météorologiques (Chandioux, 1988 et Langlais *et al.*, 2005), le problème est loin d'être résolu pour la traduction de textes plus généraux.

Les efforts en traduction automatique se sont inscrits au sein de trois paradigmes : la traduction à base de règles, la traduction à base d'exemples et la traduction statistique. Dans les trois cas, nous cherchons à construire une base de connaissances qui sera utilisée pour traduire de nouveaux textes. Ces paradigmes se différencient par la manière dont la base de connaissances est acquise et représentée.

En traduction à base de règles, nous demandons à un spécialiste d'encoder ses connaissances dans le système. Ces connaissances peuvent prendre plusieurs formes comme des règles de transfert syntaxique et des dictionnaires bilingues.

En traduction à base d'exemples, nous parions plutôt que le système peut construire automatiquement sa base de connaissances à partir d'un bitexte (texte traduit en deux langues) assez grand. Le bitexte est mémorisé et constitue un ensemble d'exemples à partir desquels le système calcule des analogies pour traduire un nouveau texte (Nagao, 1984).

En traduction statistique, nous utilisons aussi un bitexte, mais pour estimer une fonction de densité calculant la probabilité qu'un texte cible traduise un texte

source. Les systèmes de traduction statistique cherchent ensuite à maximiser cette probabilité lors de la traduction de nouveaux textes.

Avec la disponibilité croissante de bitextes provenant par exemple de sites Internet bilingues et de transcriptions de débats parlementaires, la traduction statistique a gagné en popularité au cours des dernières années. C'est par exemple l'approche qu'utilisent les systèmes de Microsoft[2] et Google[3] .

Dans le cadre de cette thèse, nous nous intéressons à la traduction statistique à base de segments. La section 1.1 présente la traduction statistique et la section 1.2 son application dans les systèmes à base de segments. Ces systèmes offrent des performances de pointe, mais ne font qu'un usage très limité du contexte dans lequel apparait un segment à traduire. La section 1.3 présente les différentes parties de cette thèse qui visent à intégrer un contexte plus large aux systèmes de traduction statistique à base de segments.

1.1 Traduction statistique

En traduction statistique, nous cherchons la phrase cible ($\mathbf{t}$) ayant la plus grande probabilité de traduire une phrase source ($\mathbf{s}$):

$$\mathbf{t}^\star = \underset{\mathbf{t} \in \mathcal{L}_t}{\arg\max} \Pr(\mathbf{t} \mid \mathbf{s}) \tag{1.1}$$

où $\mathcal{L}_t$ est l'ensemble des phrases possibles dans la langue cible. Résoudre cette équation pose deux problèmes de taille. Il faut modéliser la fonction de densité calculant la probabilité qu'une phrase en traduise un autre ($\Pr(\mathbf{t} \mid \mathbf{s})$) et l'utiliser dans un décodeur pour chercher la phrase cible la plus probable parmi celles qui sont possibles ($\arg\max_{\mathbf{t} \in \mathcal{L}_t}$).

[2] http://microsofttranslator.com/
[3] http://google.com/translate

Pour simplifier le calcul de l'équation 1.1, nous lui ajoutons une variable cachée (δ_1^n) divisant la traduction d'une phrase en plusieurs étapes :

$$\mathbf{t}^\star = \arg\max_{\mathbf{t}\in\mathcal{L}_t} \sum_{\delta_1^n\in\Delta(\mathbf{t},\mathbf{s})} \prod_{i=1}^{n} \Pr(\delta_i|\delta_1^{i-1},\mathbf{s}) \tag{1.2}$$

où $\Delta(\mathbf{t},\mathbf{s})$ retourne l'ensemble des séquences d'étapes permettant de traduire $\mathbf{s}$ par $\mathbf{t}$. La nature des étapes dépend du système de traduction. Dans un système à base de segments, une étape correspond à la traduction d'un segment source par un segment cible.

Pour simplifier le calcul de l'équation 1.2, nous posons l'hypothèse que la séquence d'étapes la plus probable mènera à la traduction la plus probable. Plutôt que de résoudre l'équation 1.2, le décodeur cherche donc à résoudre :

$$\hat{\mathbf{t}} = \arg\max_{\mathbf{t}\in\mathcal{L}_t} \max_{\delta_1^n\in\Delta(\mathbf{t},\mathbf{s})} \prod_{i=1}^{n} \Pr(\delta_i|\delta_1^{i-1},\mathbf{s}) \tag{1.3}$$

1.2 Traduction à base de segments

Les systèmes de traduction statistique à base de segments (Koehn *et al.*, 2003) traduisent les phrases un segment à la fois et estiment la probabilité d'une traduction à l'aide des trois composantes suivantes :

Table de traduction (tm) Un dictionnaire de bisegments (segments sources et leurs traductions) où chaque entrée est associée à un ou plusieurs scores évaluant sa qualité.

Modèle de langue (lm) Un modèle markovien conditionnant la probabilité de chaque mot cible sur les $n-1$ mots le précédents où n est habituellement entre trois et cinq.

Modèle de réordonnancement (r) Un modèle associant un score à l'ordre dans lequel les segments de la phrase source sont traduits.

Ces composantes sont habituellement intégrées dans un modèle exponentiel estimant l'équation 1.3 à l'aide de :

$$\begin{aligned}\hat{\mathbf{t}} &= \arg\max_{\mathbf{t}\in\mathcal{L}_t} \max_{\delta_1^n\in\Delta(\mathbf{t},\mathbf{s})} \frac{1}{Z} \exp\left\{\prod_{i=1}^{n}\prod_{j\in\{tm,lm,r\}} f_j(\delta_i,\delta_1^{i-1},\mathbf{s})^{\lambda_j}\right\} \\ &= \arg\max_{\mathbf{t}\in\mathcal{L}_t} \max_{\delta_1^n\in\Delta(\mathbf{t},\mathbf{s})} \sum_{i=1}^{n}\sum_{j\in\{tm,lm,r\}} \lambda_j \log f_j(\delta_i,\delta_1^{i-1},\mathbf{s}) \qquad (1.4)\end{aligned}$$

où Z est un facteur de normalisation et les λ_j sont des poids configurant l'importance de chacune des composantes.

1.2.1 Exemple

Dans cette section, nous expliquons comment un décodeur utilise l'équation 1.4 pour traduire la phrase « the walls of this plant » de l'anglais vers le français. Pour simplifier l'exemple, le réordonnancement des segments n'est pas permis, le modèle de langue est un bigramme et les poids (λ_j) sont fixés à 1. Le système utilise la table de traduction et le modèle de langue présentés dans le tableau 1.1. Ces deux modèles sont des extraits de ceux utilisés au chapitre 6.

L'arbre de recherche exploré par le décodeur est présenté à la figure 1.1. Chaque chemin partant de la racine et allant jusqu'à une feuille correspond à une traduction possible. Les scores des traductions partielles et complètes sont inscrits à l'intérieur de leur nœuds. Bien que les systèmes de traduction puissent s'accommoder des bigrammes inconnus, nous les avons interdits pour simplifier l'exemple (d'où les scores $-\infty$).

Anglais	**Français**	$\log(f_{tm})$
the	la	-0.16
	le	-0.20
	les	-0.25
walls	murs	-0.41
of this	de cet	-0.99
	de cette	-0.93
plant	plante	-0.49
	usine	-1.87

Table de traduction

Bigramme	$\log(f_{lm})$
<s> la	-1.25
<s> le	-1.26
<s> les	-1.42
cette plante	-3.94
cette usine	-3.52
de cet	-2.44
de cette	-1.95
les murs	-3.73
murs de	-1.09
plante </s>	-0.10
usine </s>	-0.02

Modèle de langue.

Tableau 1.1 Modèles utilisés par le décodeur où <s> et </s> sont les marqueurs de début et de fin de phrase.

La phrase source admet donc les traductions : « les murs de cette plante » et « les murs de cette usine ». Le décodeur guidera son choix en utilisant la table de traduction et le modèle de langue. La figure 1.2 présente les étapes menant à la traduction « les murs de cette usine ». Nous voyons que le choix entre « plante » et « usine » ne se fait qu'à la dernière étape. À ce moment, la table de traduction assigne un score ne dépendant que de « usine » et le modèle de langue un score ne dépendant que de « cette » et « </s> ». Pourtant, c'est « wall » qui devrait guider le choix du décodeur. La traduction finale ne dépendra donc que du biais du corpus ayant servi à créer la table de bisegments et le modèle de langue. C'est malheureusement la mauvaise traduction qui a le plus grand score dans notre exemple.

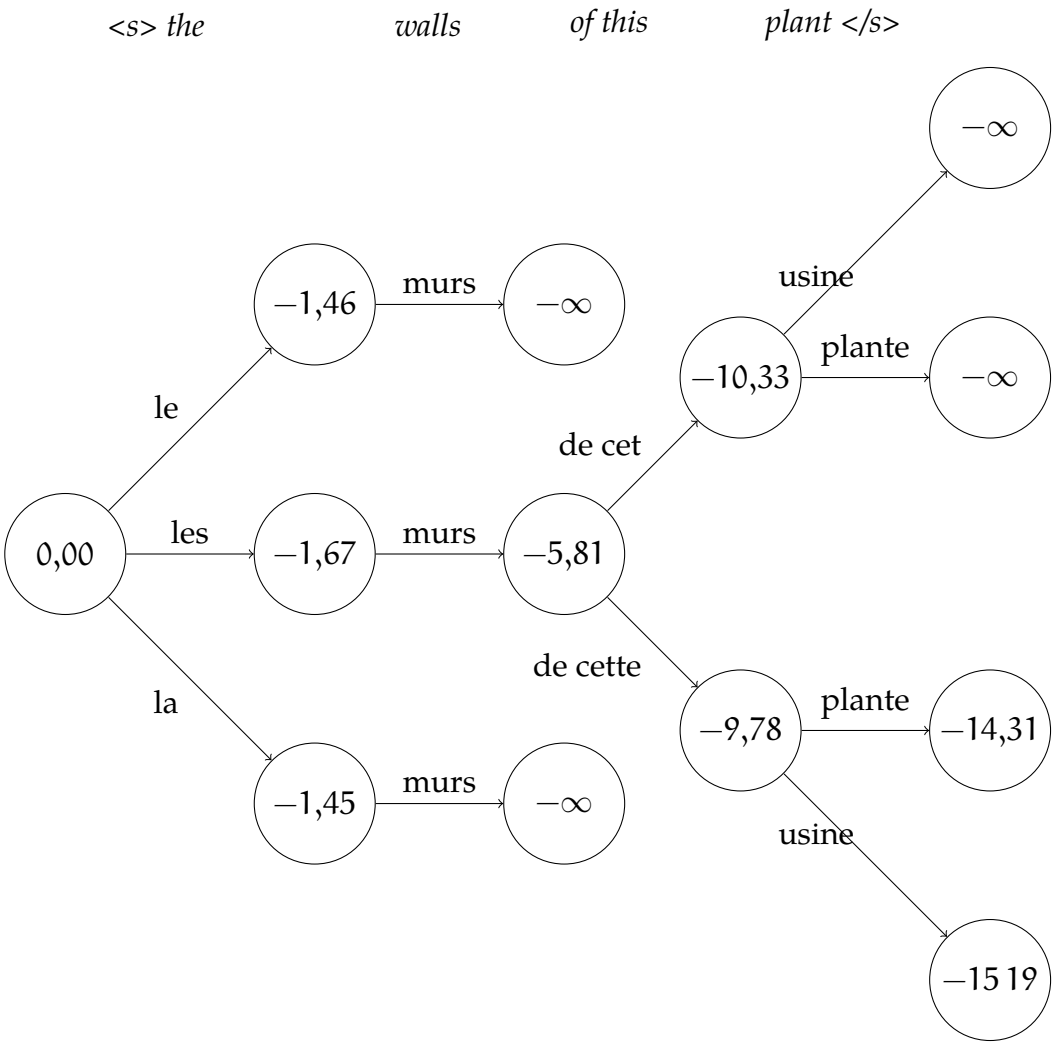


Figure 1.1 Espace de recherche exploré pour la traduction de la phrase « the walls of this plant ».

1.3 Intégration du contexte

Les systèmes de traduction à base de segments n'utilisent qu'un contexte très limité pour traduire un segment. La table de traduction ne considère que le segment source et le modèle de langue quelques mots précédant et suivant le segment cible.

Étapes			
Action	$\log(f_{tm})$	$\log(f_{lm}))$	Score
Débuter avec une phrase cible vide	0	0	0
Traduire « the » par « les »	−0,25	−1,42	−1,67
Traduire « walls » par « murs »	−0,41	−3,73	−5,81
Traduire « of this » par « de cette »	−0,93	−1,09 − 1,95	−9,78
Traduire « plant » par « usine »	−1,87	−3,52 − 0,02	−15,19

Figure 1.2 Détail de la traduction de « this plant is unique » par « cette plante est unique ». Les colonnes $\log(f_{tm})$ et $\log(f_{lm})$ présentent les scores associés à une étape et la dernière colonne présente le score cumulatif de la traduction.

Dans plusieurs cas, ce contexte limité est suffisant. Par exemple, pour traduire « plant » dans la phrase « this nuclear plant is unique », il est possible que la table de traduction contienne une entrée « nuclear plant » et si ce n'est pas le cas, le modèle de langue favorisera très probablement la traduction « usine nucléaire » à « plante nucléaire ». Il y a cependant d'autres cas, comme celui de la figure 1.1 où le décodeur aurait intérêt à modéliser un contexte plus large.

Cette thèse porte sur une meilleure intégration du contexte dans les systèmes de traduction à base de segments qui sont décrits plus en détail au chapitre 2. Le chapitre 3 présente ensuite divers travaux qui ont cherché à enrichir le contexte considéré par ces systèmes. Nous proposons ensuite au chapitre 4 de modéliser les phrases complètes à l'aide d'un perceptron à plusieurs couches. Nous évaluons notre approche au chapitre chapitre 5 et l'intégrons à un système de traduction au chapitre 6. Les principales contributions de cette thèse sont la proposition d'un nouveau modèle pour tenir compte du contexte, son intégration à un système de traduction de pointe et son évaluation.

2 Traduction statistique à base de segments

En traduction statistique, nous supposons que les connaissances nécessaires pour traduire un document peuvent être apprises à partir d'exemples, c'est à dire de documents déjà traduits. L'entrainement d'un système de traduction statistique débute donc par la sélection d'un corpus bilingue.

Il en existe heureusement plusieurs gratuits qui peuvent être utilisés pour la recherche, comme les traductions des débats parlementaires canadiens (Simard, 1998), européens (Koehn, 2005) et onusiens (Rafalovitch et Dale, 2009). Plusieurs corpus peuvent sont aussi vendus par des organisations comme ELDA[4] ou LDC[5] . Les domaines de spécialités de ces corpus sont cependant limités.

Idéalement, le corpus d'entrainement devrait être similaire aux documents que le système sera appelé à traduire. C'est pourquoi les systèmes de traductions spécialisés réalisés par le RALI ont nécessité une collecte de données à partir de documents internes (Langlais *et al.*, 2005) ou d'Internet (Langlais *et al.*, 2006 et Gotti, 2009). Il arrive aussi de disposer d'un ensemble de documents bilingues sans savoir lesquels sont en relation de traduction. Patry et Langlais (2005) et Enright et Kondrak (2007) proposent des approches simples permettant d'identifier ces relations de traduction. Finalement, il est aussi possible d'utiliser Internet comme une source de documents bilingues (Ma et Liberman, 1999, J-Y.Nie et Cai, 2001 et Resnik et Smith, 2003).

Lorsque l'ensemble des documents bilingues est établi, nous devons en extraire des bisegments (paire de segments bilingues). Pour ce faire, nous débutons par

[4] http://elda.org/
[5] http://ldc.upenn.edu/

créer un bitexte en alignant les documents au niveau des phrases (section 2.1) et au niveau des mots (section 2.2). Les bisegments sont ensuite identifiés à partir des alignements de mots (section 2.3) et intégrés au système de traduction (section 2.4) dont la qualité peut être évaluée automatiquement (section 2.5).

2.1 Alignement des phrases

Une fois les documents bilingues colligés, nous alignons les phrases qui sont en relation de traduction, comme dans la figure 2.1.

s_1 Les réunions entre le CIO et le COVAN couvrent plusieurs secteurs de la préparation des Jeux**,** comme la **structure** du COVAN**,** la **planification** et la **construction** des sites**,** le **sport,** l'héritage post-olympique**,** la **communication,** la place des médailles et les Jeux **Paralympiques.**	t_1 The meetings between the IOC and VANOC covered many areas of Games preparations**,** such as VANOC's **structure,** venue **planning** and **construction, sport,** legacy**,** **communications,** medals plaza and the **Paralympic** Games**.**
s_2 La **commission** a aussi organisé des **groupes** de travail qui ont étudié le niveau de **services** attendus par les **différents groupes** de **clients** aux Jeux **Olympiques** à **Vancouver,** à savoir notamment les **spectateurs,** les **sponsors,** les représentants des médias, les **Comités Nationaux Olympiques** ainsi que les **Fédérations Internationales.**	t_2 The **Commission** also held working **groups** that looked at the **service** levels that **different Olympic** Games **client groups** can expect during the Games in **Vancouver**.
	t_3 These groups included **spectators**, **sponsors**, the media, **National Olympic Committees** and the **International Federations.**
s_3 Un sujet **important** soulevé par le CIO pendant ces réunions est celui du désir d'accueillir plus de représentants des médias et des CNO à **Whistler** et dans ses environs**.**	t_4 One **important** subject that was raised by the IOC in these meetings was the desire to accommodate additional media and National Olympic Committee members in **Whistler** and the surrounding area**.**

Figure 2.1 Bitexte extrait de http://www.olympic.org/ où les cognats (mots de forme similaire en français et en anglais) sont en gras.

L'alignement serait simple si chaque phrase source n'était traduite que par une seule phrase cible. Ce n'est malheureusement pas toujours le cas, comme nous le voyons dans le deuxième alignement de la figure 2.1. En effet, les traducteurs peuvent couper, fusionner, omettre ou insérer des phrases.

Motifs	Motivation	Ratio
1:0, 0:1	Une phrase n'est pas traduite dans un des deux documents.	0,0099
1:1	La phrase source est traduite par une phrase cible.	0,89
1:2, 2:1	La segmentation des phrases est différentes dans les deux traductions.	0,089
2:2	Deux phrases ont été inversées par le traducteur.	0,011

Tableau 2.1 Motifs utilisés pour l'alignement des phrases ou $l : m$ signifie que l phrases sources sont alignées à m phrases cibles.

Les motifs d'alignement auxquels se limitent la plupart des aligneurs ont été proposés par Gale et Church (1993) et sont présentés au tableau 2.1. En supposant que les motifs d'alignements sont indépendants, l'alignement des phrases peut être fait par programmation dynamique en utilisant la fonction objective suivante :

$$\mathrm{Pr}_d(\mathbf{s}_1^l, \mathbf{t}_1^m) = \max \begin{cases} \mathrm{Pr}_m(\mathbf{s}_1, \cdot)\, \mathrm{Pr}_d(\mathbf{s}_2^l, \mathbf{t}_1^m) & \text{Motif } 1:0 \\ \mathrm{Pr}_m(\cdot, \mathbf{t}_1)\, \mathrm{Pr}_d(\mathbf{s}_1^l, \mathbf{t}_2^m) & \text{Motif } 0:1 \\ \mathrm{Pr}_m(\mathbf{s}_1, \mathbf{t}_1)\, \mathrm{Pr}_d(\mathbf{s}_2^l, \mathbf{t}_2^m) & \text{Motif } 1:1 \\ \mathrm{Pr}_m(\mathbf{s}_1, \mathbf{t}_1^2)\, \mathrm{Pr}_d(\mathbf{s}_2^l, \mathbf{t}_3^m) & \text{Motif } 1:2 \\ \mathrm{Pr}_m(\mathbf{s}_1^2, \mathbf{t}_1)\, \mathrm{Pr}_m(\mathbf{s}_3^l, \mathbf{t}_2^m) & \text{Motif } 2:1 \\ \mathrm{Pr}_m(\mathbf{s}_1^2, \mathbf{t}_1^2)\, \mathrm{Pr}_d(\mathbf{s}_3^l, \mathbf{t}_3^m) & \text{Motif } 2:2 \end{cases} \tag{2.1}$$

où Pr_m calcule la probabilité du premier motif d'alignement et Pr_d aligne le reste du document. Une fois la probabilité du meilleur alignement trouvé, il est simple

de récupérer les motifs d'alignements qui l'ont générée. Il ne reste qu'à définir Pr_m pour pouvoir aligner les phrases d'une paire de documents.

La longueur d'une traduction est habituellement proportionnelle à la longueur du texte original. Une approche simple et efficace pour calculer Pr_m est de modéliser la longueur des phrases sources et cibles (Brown *et al.*, 1991 et Gale et Church, 1993). En modélisant la différence du nombre de caractères entre la source et la cible d'un alignement par une loi normale de moyenne 0 et de variance 6, Gale et Church (1993) ont identifié plus de 95% des alignements dans un corpus de transcriptions de débats parlementaires.

Il reste tout de même tentant d'intégrer un dictionnaire bilingue à Pr_m. Cela demande cependant l'ajout d'un lexique bilingue pour chaque paire de langues d'intérêt. Simard *et al.* (1993) suggèrent plutôt d'utiliser les cognats, des mots de formes similaires dans la plupart des langues. Leur définition des cognats est présentée dans le tableau 2.2 et des exemples sont en gras dans la figure 2.1. En intégrant les cognats à un système basé sur la longueur des phrases, ils ont réussi à réduire le taux d'alignements erronés de 10%.

Description	Exemple
Chaînes alphabétiques débutants par les mêmes quatre lettres.	press, presse
Chaînes alpha-numériques identiques.	1999, 1999
Symboles de ponctuation identiques.	., .

Tableau 2.2 Catégories de cognats suggérés par Simard *et al.* (1993).

Moore (2002) utilise un bitexte aligné à l'aide d'un algorithme à base de longueur pour entrainer un dictionnaire bilingue probabiliste. Ils utilisent ensuite ce dictionnaire pour raffiner l'alignement initial.

En règle générale, si les traductions sont fidèles, un aligneur utilisant la longueur des phrases est suffisant. Cependant, si les traductions sont plus libres, il est préférable d'utiliser un aligneur à base de cognats (Simard *et al.*, 1998) ou de dictionnaire (Moore, 2002).

Une fois les phrases alignées, nous ne conservons pour l'alignement de mots que celles faisant partie d'un motif 1:1. C'est le motif le plus fiable et c'est celui qui sera produit par nos systèmes de traduction.

2.2 Alignement des mots

Les algorithmes d'alignement de mots les plus populaires, les modèles IBM, ont été introduits par Brown *et al.* (1993), qui présentent les fondements d'un système de traduction statistique à base de mots. Ils utilisent la formule de Bayes pour reformuler l'équation 1.1 par :

$$\begin{aligned} \Pr(\mathbf{t} \mid \mathbf{s}) &= \frac{\Pr(\mathbf{s} \mid \mathbf{t}) \Pr(\mathbf{t})}{\Pr(\mathbf{t})} \\ &\propto \Pr(\mathbf{s} \mid \mathbf{t}) \Pr(\mathbf{t}) \end{aligned} \tag{2.2}$$

où $\Pr(\mathbf{s} \mid \mathbf{t})$ est un modèle de traduction qui s'assure que le sens de la phrase cible soit le même que celui de la phrase source et $\Pr(\mathbf{t})$ est un modèle de langue qui s'assure que la traduction soit bien formée.

Pour décomposer l'évaluation du modèle de traduction, ils ont introduit les alignements au niveau des mots :

$$\begin{aligned} \Pr(\mathbf{s} \mid \mathbf{t}) &= \sum_{\mathbf{a}} \Pr(\mathbf{s}, \mathbf{a} \mid \mathbf{t}) \\ &= \sum_{\mathbf{a}} \prod_{i=1}^{l} \Pr(s_i, a_i, l \mid \mathbf{s}_1^{i-1} \mathbf{a}_1^{i-1}, \mathbf{t}) \end{aligned} \tag{2.3}$$

où a_i est la position du mot cible aligné à s_i et l la longueur de $\mathbf{s}$. Pour simplifier la notation, nous supposons que les mots sources non traduits sont alignés à t_0, le mot vide. Un exemple d'alignement au niveau des mots où la langue cible est l'anglais est présenté à la figure 2.2.

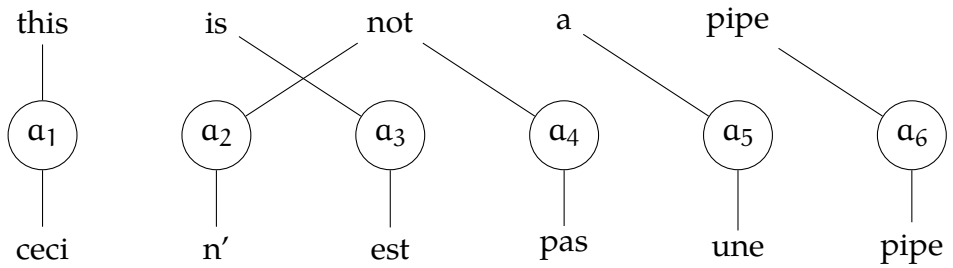


Figure 2.2 Exemple d'alignement produit par les modèles IBM1, IBM2 ou HMM.

Pour aligner les mots, nous n'avons qu'à résoudre :

$$\mathbf{a}^\star = \arg\max_{\mathbf{a}} \Pr(\mathbf{s}, \mathbf{a} \mid \mathbf{t}) \tag{2.4}$$

Nous pouvons encore décomposer la probabilité d'un alignement en trois parties :

$$\Pr(\mathbf{s}, \mathbf{a} \mid \mathbf{t}) = \Pr(l \mid \mathbf{t}) \prod_{i=1}^{l} \Pr(a_i \mid a_1^{i-1}, s_1^{i-1}, l, \mathbf{t}) \Pr(s_i \mid a_1^{i}, s_1^{i-1}, l, \mathbf{t}) \tag{2.5}$$

où la première distribution permet de choisir le nombre de mots dans $\mathbf{s}$, la deuxième leur position et la troisième les mots sources. Tous les modèles que nous présentons supposent que la distribution sur la longueur des phrases est uniforme et que la sélection d'un mot source ne dépend que du mot cible auquel il est aligné :

$$\Pr(s_i, a_i, \mid \mathbf{s}_1^{i-1}, \mathbf{a}_1^{i-1}, \ l, \mathbf{t}) \propto \prod_{i=1}^{l} \Pr(a_i \mid a_1^{i-1}, s_1^{i-1}, l, \mathbf{t}) \Pr(s_i \mid t_{a_i}) \tag{2.6}$$

Les modèles IBM ne se distinguent donc que par la probabilité qu'ils donnent aux réordonnancements des mots.

Les sections suivantes présentent une série de modèles de réordonnancements de complexités croissantes. Ils sont tous entrainés par EM (algorithme d'Espérance Maximisation (Dempster *et al.*, 1977)) à partir des modèles qui les précèdent. Ainsi, pour entrainer un modèle IBM4, nous voudrons d'abord entrainer un modèle IBM1, utiliser ses paramètres pour initialiser et entrainer un modèle HMM, ensuite un modèle IBM3 et finalement un modèle IBM4 (Och et Ney, 2003).

2.2.1 Ibm1

Le modèle IBM1 est le plus simple des modèles proposés par Brown *et al.* (1993), car il suppose que tous les réordonnancements sont équiprobables :

$$\Pr_{\text{IBM1}}(a_i \mid a_1^{i-1}, s_1^{i-1}, l, \mathbf{t}_1^m) = \frac{1}{(m+1)} \tag{2.7}$$

où on ajoute 1 à la longueur de la phrase cible m pour compter le mot vide.

Ce modèle a l'avantage d'être rapide à calculer. Cependant, comme il considère tous les réordonnancements sur le même pied d'égalité, il assignera la même probabilité de $\left(\frac{1}{5+1}\right)^6$ aux phrases des figures 2.2 et 2.3.

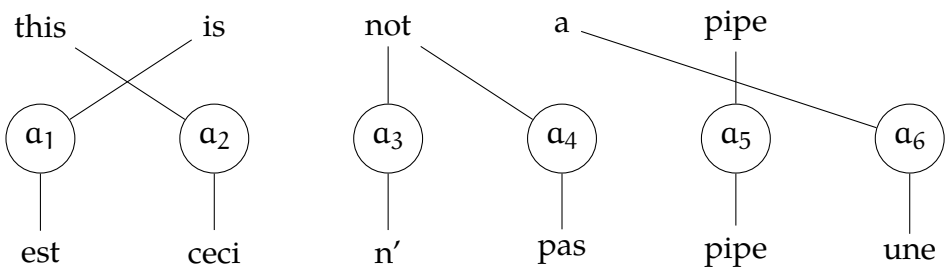


Figure 2.3 Alignement équivalent à celui de la figure 2.2 pour le modèle IBM1.

2.2.2 Ibm2 et Hmm

IBM2 suppose plutôt que la position d'un mot est déterminée par la position du mot auquel il est aligné et la longueur des phrases sources et cibles :

$$\Pr_{\text{IBM2}}(a_i \mid a_1^{i-1}, s_1^{i-1}, l, \mathbf{t}_1^m) = \Pr(a_i \mid i, l, m) \tag{2.8}$$

Cela lui permet de distinguer l'alignement de la figure 2.2 de celui de la figure 2.3.

Plutôt que de modéliser la position des mots, Vogel *et al.* (1996) modélisent la distance entre les alignements consécutifs :

$$\Pr_{\text{HMM}}(a_i \mid a_1^{i-1}, s_1^{i-1}, l, \mathbf{t}) = \Pr(a_i - a_{i-1}) \tag{2.9}$$

Cette approche est plus conciliante avec les déplacements de mots en groupes où seulement le premier mot du groupe aura une distance plus grande qu'un.

Och et Ney (2003) rapportent que ces deux modèles supplantent IBM1 et que HMM est largement préférable à IBM2.

2.2.3 Ibm3, Ibm4 et Ibm5

Les modèles IBM3, IBM4 et IBM5 regroupent les alignements associés au même mot cible dans une *tablette* π. Par exemple, le mot « not » est aligné à une seule tablette π_3 dans la figure 2.4 plutôt qu'à deux *perles* d'alignement comme dans la figure 2.2. En conditionnant l'équation 2.6 sur les tablettes plutôt que sur les alignements, nous obtenons :

$$\Pr(\mathbf{s}, \mathbf{a} \mid \mathbf{t}) \propto c_l \sum_{\boldsymbol{\pi}} \prod_{j=0}^{m} \Pr(\phi_j) \prod_{k=1}^{\phi_j} \Pr(s_{\pi_{j,k}} \mid t_j) \Pr(\pi_{j,k} \mid \pi_1^{j-1}, \tau_1^{j-1}, l, m)$$

où ϕ_j est le nombre de mots alignés à t_j, $\pi_{j,k}$ la position du $k^{\text{ème}}$ mot source aligné à t_j et τ_j contient les mots sources alignées avec t_j.

IBM3 redéfinit IBM2 en y intégrant les tablettes pour y ajouter la fertilité :

$$\Pr_{\text{IBM3}}(\pi_{j,k} \mid \pi_1^{j-1}, \tau_1^{j-1}, l, m) = \Pr(\pi_{j,k} \mid t_j, l, m) \tag{2.10}$$

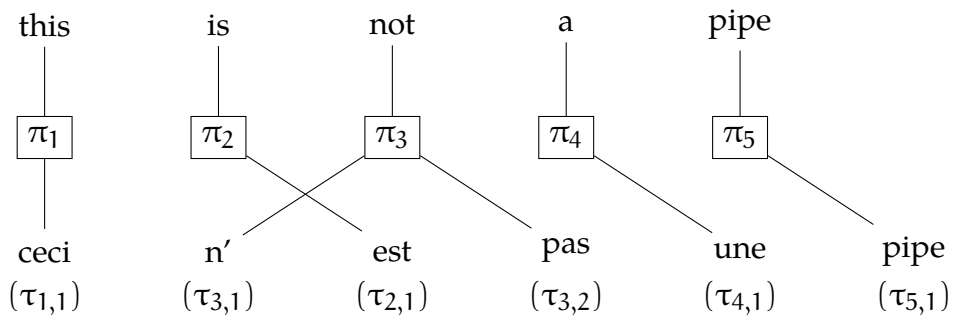


Figure 2.4 Exemple d'alignement produit par les modèles IBM3, IBM4 ou IBM5.

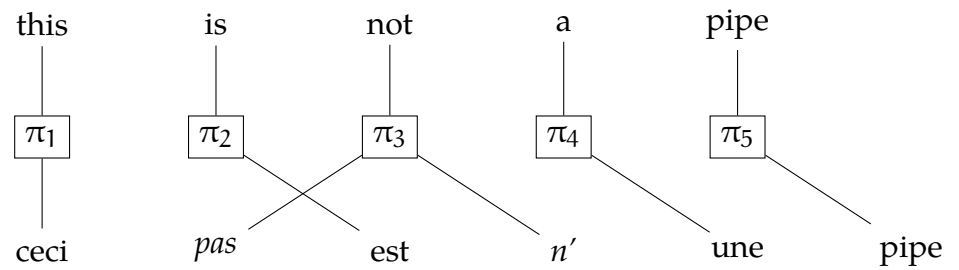


Figure 2.5 Alignement équivalent à celui de la figure 2.4 pour le modèle IBM3.

Le problème avec ce modèle est qu'il ne tient pas compte de l'ordre des mots dans une tablette donnée. Ainsi, l'alignement de la figure 2.5 aura la même probabilité que celui de la figure 2.4.

Pour tenir compte de l'ordre des mots dans chaque tablette, IBM4 introduit la notion de mot de tête (le premier mot d'une tablette) et définit la probabilité d'alignement par :

$$\Pr_{\text{IBM4}}(\pi_{j,k} \mid \pi_1^{j-1}, \tau_1^{j-1}, l, m) = \begin{cases} \Pr(\pi_{j,k} - \odot_{j-1} \mid t_{j-1}, s_{\pi_{j,k}}) & \text{Si } k = 1 \text{ (tête)} \\ \Pr(\pi_{j,k} - \pi_{j,1} \mid s_{\pi_{j,k}}) & \text{Si } k \geqslant 2 \text{ (autres)} \end{cases} \tag{2.11}$$

où la première distribution place le mot de tête par rapport à la position moyenne des mots de la tablette précédentes ($\odot_{j-1}$) et la deuxième distribution place les mots suivants par rapport au mot de tête.

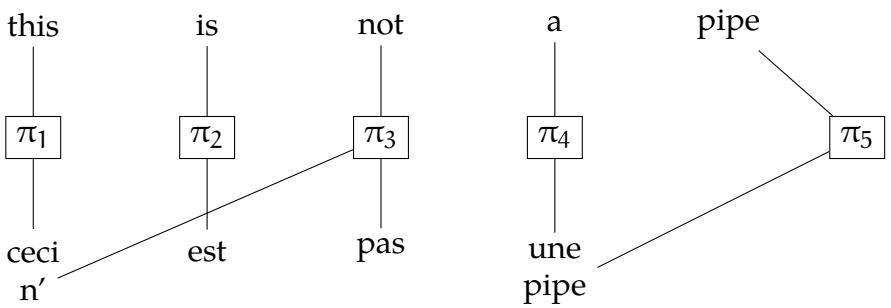


Figure 2.6 Alignement déficient permis par les modèles précédents IBM5.

Les modèles précédents sont tous *déficients* : ils assignent une probabilité non nulle à des réordonnancements impossibles comme celui de la figure 2.6 où plusieurs mots occupent la même position. Le modèle IBM5 règle ce problème en s'assurant que les différents π_j ne se chevauchent pas. En pratique, IBM5 n'est pas meilleur qu'IBM4 (Och et Ney, 2003), qui est considéré comme le modèle de référence dans la littérature.

2.2.4 Raffinement

Les modèles IBM ne permettent pas à un mot source d'être aligné à plus d'un mot cible. Les alignements idéaux sont parfois impossible sous cette restriction, comme à la figure 2.7 où nous voudrions que le mot source « pipe » soit aligné aux deux mots cibles « smoking pipe ». Ce cas particulier peut être réglé en inversant la direction de l'alignement, comme à la figure 2.8, mais cette fois-ci c'est « not » qui ne peut être aligné à deux mots.

La solution habituelle est de combiner les alignements des deux directions de traductions pour permettre des motifs d'alignements arbitraires. La figure 2.9 présente une matrice d'alignement donnant un aperçu de ces deux alignements.

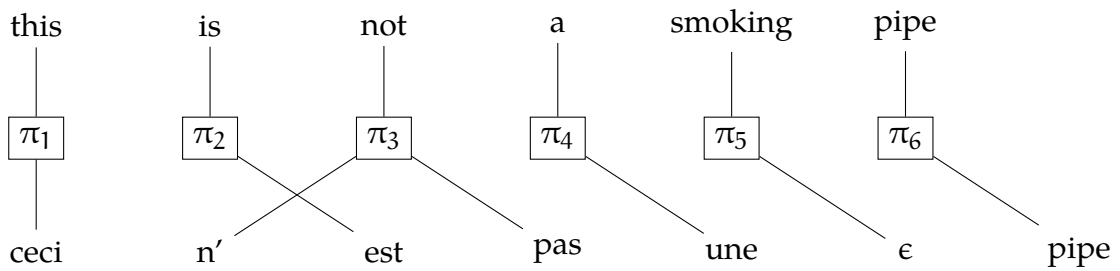


Figure 2.7 Phrase où l'alignement idéal est impossible avec les modèles IBM, ce qui les force à aligner « smoking » au mot vide ϵ plutôt qu'à « pipe ».

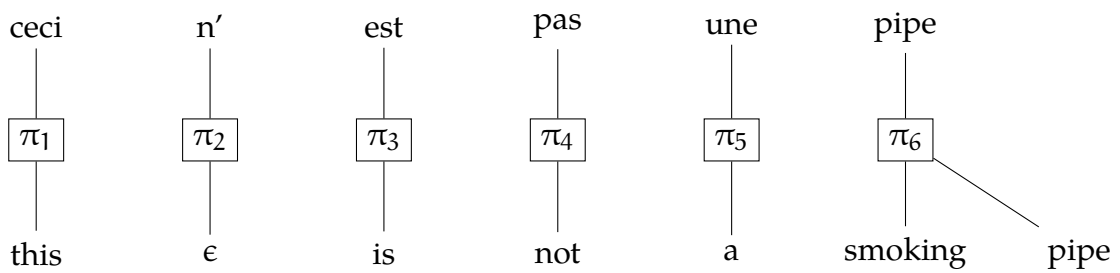


Figure 2.8 Alignement de la figure 2.7 dans l'autre direction de traduction.

Une première méthode pour combiner ces deux alignements est de ne prendre que ceux communs aux deux directions de traduction. Les alignements conservés seront de bonnes qualités, mais auront une couverture limitée. Une deuxième méthode est de prendre l'union des deux alignements, ce qui engendre une plus grande couverture, mais réduit la qualité des alignements.

La solution se trouve entre ces deux extrêmes. Och et Ney (2003) proposent plusieurs euristiques permettant de choisir quels alignements de l'union devraient être ajoutés à ceux de l'intersection. L'idée générale est de favoriser les alignements voisins dans la matrice d'alignement. La meilleure euristique dépend cependant du domaine des documents utilisés pour entrainer et évaluer le système de traduction (Wu et Haifeng, 2007).

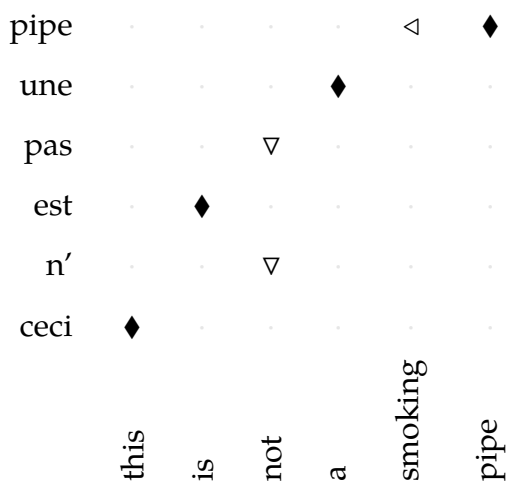


Figure 2.9 Matrice d'alignement où les ▽ indiquent les alignements spécifiques à la figure 2.7, ◁ les alignements spécifiques à la figure 2.8 et les ♦ les alignements communs.

Peu importe le modèle d'alignement de mots utilisé, Och et Ney (2003) suggèrent de combiner ses alignements des deux directions de traduction. Cette conclusion est confirmée par Koehn *et al.* (2003) lorsque les alignements doivent être utilisés par un système de traduction.

2.2.5 Autres modèles d'alignement

Les approches précédentes alignent les phrases dans les deux directions de traduction de manière indépendante. Liang *et al.* (2006) proposent un cadre statistique pour aligner dans les deux directions en même temps et favoriser les accords entre les deux modèles. En combinant ainsi deux modèles HMM, ils obtiennent des alignements de qualités comparables à un modèle IBM4.

Les meilleurs systèmes d'alignement utilisent des modèles discriminatifs qui nécessitent quelques centaines de phrases alignées à la main pour être entrainé

(Taskar *et al.*, 2005, Moore *et al.*, 2006 et Lacoste-Julien *et al.*, 2006). Les modèles discriminatifs facilitent l'intégration d'informations arbitraires dans l'alignement, ce qui permet d'injecter plus de connaissances du domaine dans le système, comme des caractéristiques de la matrice d'alignement (Niehues et Vogel, 2008) ou des propriétés grammaticales (Haghighi *et al.*, 2009). Ce sont tout de même les modèles IBM qui restent les plus utilisés.

2.3 Extraction des bisegments

L'alignement des mots n'est qu'une étape intermédiaire. Ce sont les bisegments qui pourront en être extraits qui nous intéressent. Och *et al.* (2004) suggèrent d'extraire tout les bisegments cohérents avec l'alignement de mots raffiné.

Les bisegments cohérents sont ceux dont tous les mots sources ne sont alignés qu'avec leurs mots cibles et vice versa. Le tableau 2.3 présente les bisegments cohérents avec la matrice d'alignement de la figure 2.9.

Le nombre de bisegments croît rapidement. C'est pourquoi nous nous restreignons à extraire des segments de taille limitée. MOSES limite la taille des segments à sept mots par défaut, mais Koehn *et al.* (2003) rapportent que peu de segments de plus de trois mots sont utiles à la traduction.

Il ne reste qu'à choisir l'alignement de mots à partir duquel extraire les segments. Le lien entre la qualité des alignements et la qualité des systèmes de traduction n'est pas direct (Ayan et Dorr, 2006 et Vilar *et al.*, 2006a). Il n'y a pas de consensus sur le modèle à utiliser. MOSES utilise par défaut un alignement IBM4 raffiné, Sadat *et al.* (2005) utilisent un modèle IBM2 raffiné et Vogel *et al.* (2003) utilisent entre autres un modèle HMM.

s	t
ceci	this
ceci n'est pas	this is not
ceci n'est pas une	this is not a
ceci n'est pas une pipe	this is not a smoking pipe
n'est pas	is not
n'est pas une	is not a
n'est pas une pipe	is not a smoking pipe
est	is
une	a
une pipe	a smoking pipe
pipe	smoking pipe

Tableau 2.3 Bisegments cohérents avec l'alignement de la figure 2.9.

Il pourrait être tentant d'ignorer les segments ne formant pas une unité syntaxique comme « ceci n'est pas une », mais cela diminue la qualité des traductions produites (Koehn *et al.*, 2003). Il est cependant possible de réduire considérablement le nombre de bisegments extraits sans heurter la qualité des traductions en utilisant des tests statistiques sur la cohésion entre la source et la cible de chaque paire de segments (Johnson *et al.*, 2007).

Plutôt que d'extraire les bisegments par des euristiques sur les alignements de mots, Marcu et Wong (2002) modélisent directement l'alignement des segments. Bien que ce modèle soit plus élégant, il n'a pas encore réussi à surpasser les euristiques sur les alignements de mots (DeNero *et al.*, 2006).

2.4 Décodeur

Une fois la table de traduction en main, nous pouvons traduire de nouveaux textes à l'aide d'un décodeur. Le rôle du décodeur est de trouver la meilleure traduction pour une phrase source donnée :

$$\mathbf{t}^{\star} = \arg\max_{\mathbf{t}} \sum_{\mathbf{a} \in \mathcal{A}(\mathbf{s},\mathbf{t})} \Pr(\mathbf{t}, \mathbf{a} \mid \mathbf{s}) \tag{2.12}$$

où $\mathcal{A}(\mathbf{s}, \mathbf{t})$ retourne tous les alignements permettant de traduire $\mathbf{s}$ par $\mathbf{t}$ à l'aide de la table de traduction. Bien que des travaux récents aient cherché à résoudre directement cette équation (Arun *et al.*, 2009 et Li *et al.*, 2009), nous simplifions habituellement le problème et cherchons le meilleur alignement plutôt que la meilleure traduction :

$$\hat{\mathbf{t}} = \arg\max_{\mathbf{t}} \max_{\mathbf{a} \in \mathcal{A}(\mathbf{s},\mathbf{t})} \Pr(\mathbf{t}, \mathbf{a} \mid \mathbf{s}) \tag{2.13}$$

où la probabilité d'un alignement est estimée à l'aide d'un modèle exponentiel :

$$\begin{aligned} \Pr(\mathbf{t}, \mathbf{a} \mid \mathbf{s}) &= \frac{1}{Z} \prod_{n} \exp\left(f_n(\mathbf{a}, \mathbf{s}, \mathbf{t})^{\lambda_n}\right) \\ &\propto \sum_{n} \lambda_n \log\left(f_n(\mathbf{a}, \mathbf{s}, \mathbf{t})\right) \end{aligned} \tag{2.14}$$

où Z est une constante de normalisation et les f_n sont des fonctions évaluant différents aspects de la qualité d'une traduction candidate, comme la qualité des bisegments (section 2.4.1), la fluidité de la phrase cible (section 2.4.2) ou la vraisemblance du réordonnancement des mots (section 2.4.3).

Le choix de bonnes valeurs pour λ_n est crucial. Il est habituellement fait de sorte à minimiser les erreurs de traductions sur un corpus de validation (Och et Ney, 2003).

Une fois le modèle exponentiel en main, il faut parcourir efficacement l'espace des traductions possibles, problème que nous abordons à la section 2.4.4.

2.4.1 Qualité des bisegments

Les bisegments de la table de traduction sont nombreux et de qualité variable. Par exemple, une de nos tables traduit « a bad reason » par « pas une bonne raison » ou « une bonne raison ». Dans ce cas, le mot « pas » fait toute la différence et il faut un moyen de distinguer les bonnes des mauvaises traductions.

Une méthode simple consiste à privilégier les traductions fréquentes. C'est ce que fait le score $\varphi(\mathbf{t} \mid \mathbf{s})$, qui calcule la probabilité de traduction par fréquence relative sur l'ensemble des bisegments extraits :

$$\varphi(\mathbf{t} \mid \mathbf{s}) = \frac{nb(\mathbf{t}, \mathbf{s})}{\sum_{\mathbf{s}'} nb(\mathbf{t}, \mathbf{s}')} \tag{2.15}$$

où la $nb(\mathbf{t}, \mathbf{s})$ compte la fréquence du bisegment qui lui est passé en paramètre dans l'ensemble des bisegments qui ont été extraits de l'alignement raffiné. Foster *et al.* (2006) proposent plusieurs techniques de lissage pour améliorer la robustesse de ce score aux bisegments peu fréquents.

Une autre façon d'améliorer la robustesse du système aux segments peu fréquents est de calculer un score au niveau des mots. C'est ce que fait la pondération lexicale en calculant la probabilité moyenne de chaque alignement de mots :

$$lex(\mathbf{t} \mid \mathbf{s}) = \prod_{j=1}^{m} \frac{1}{\phi_j} \sum_{i \in \pi_j} w(t_j \mid s_i) \tag{2.16}$$

où $w(t_j \mid s_i)$ est évalué par fréquence relative sur le nombre d'alignements entre t_j et s_i par rapport au nombre total d'alignements impliquant s_i. Nous calculons aussi ces deux probabilités dans la direction inverse de traduction pour ajouter les scores $\varphi(\mathbf{s} \mid \mathbf{t})$ et $lex(\mathbf{s} \mid \mathbf{t})$.

Nous ajoutons deux pénalités à ces scores. Une première qui compte le nombre de mots dans la traduction et une deuxième qui compte le nombre de segments. Un poids positif associé à la pénalité sur les mots favorisera les phrases courtes et un poids négatif favorisera les phrases longues. De la même façon, un poids négatif associé à la pénalité des segments favorisera des segments sources courts et un poids positif favorisera les segments longs.

2.4.2 Fluidité de la traduction

Les scores de la section précédente n'évaluent que la qualité intrinsèque des bi-segments. Cependant, ce n'est pas parce que « le » traduit « the » et « pipe » traduit « pipe » que « le pipe » est une bonne traduction pour « the pipe ». Le modèle de langue cherche à agencer les différentes traductions possibles pour en produire une fluide.

Pour ce faire, il calcule la probabilité d'une phrase cible en considérant ses séquences de n mots où n est l'ordre du modèle de langue. Le choix standard pour n était de trois par le passé, mais avec la disponibilité grandissante de corpus contenant plusieurs millions de phrases, les 4-grams et les 5-grams sont devenus la norme.

2.4.3 Modèle de réordonnancement

Les mots d'une phrase ne sont pas toujours traduits dans l'ordre, ce qui permet de traduire « green plant » par « plante verte ». Le modèle de langue et les bisegments devraient détecter les réordonnancements locaux menant à des traductions erronées comme « verte plante ». Il y a cependant des cas où les réordonnancements sont aussi des traductions valides mais inexactes. Comme lorsque nous traduisons

« birds eat worms » par « Les vers mangent les oiseaux ». Il y a aussi des réordonnancements qui ne peuvent être capturés par le modèle de langue, c'est souvent le cas pour les paires de langues qui ont des structures syntaxiques différentes. Nous introduisons donc un score pour évaluer la vraisemblance des réordonnancements dans la traduction.

Le score de réordonnancement le plus simple ne fait qu'accorder une pénalité constante à chaque mot séparant deux segments sources traduits consécutivement. Ce score favorise les traductions monotones tout en permettant les réordonnancements s'ils sont vraisemblables selon le modèle de langue. Cette pénalité peut aussi être lexicalisée en la conditionnant sur le dernier mot du dernier segment traduit et le premier du segment courant (Al-Onaizan et Papineni, 2006).

Plutôt que de pénaliser les sauts, Tillmann (2004) et Axelrod *et al.* (2005) modélisent l'orientation des segments. Ils définissent trois orientations possibles : deux segments sources peuvent être traduits consécutivement, en ordre inverse ou dans un ordre arbitraire.

Ces approches ne modélisent pas les réordonnancements à longue distance qui se produisent quand on traduit entre des langues qui ont des structures syntaxiques différentes. Dans ces cas, il est possible de réordonner les mots sources avant la traduction pour refléter l'ordre dans lequel ils devraient être traduits (Tillmann *et al.*, 1997). Collins *et al.* (2005) ont proposé un ensemble de règles de réordonnancements pour la traduction de l'allemand vers l'anglais et Xia et McCord (2004) un algorithme pour apprendre les règles automatiquement.

Une approche intermédiaire définit les unités qui pourront être réordonnées avant de lancer la traduction (Kuhn *et al.*, 2006) ou pendant la traduction (Yahyaei et

Monz, 2009). Ces modèles ont l'avantage de ne permettre qu'un ensemble réduit de réordonnancements tout en considérant les réordonnancements à longue distance.

Il est aussi possible de gérer les réordonnancements à l'aide d'un modèle syntaxique. Wu et Wong (1998) proposent d'utiliser une grammaire de transduction inversée qui génère la phrase source et la phrase cible en même temps. Yamada et Knight (2001) traduisent plutôt l'arbre syntaxique de la phrase source en trois étapes : le réordonnancement des noeuds, l'insertion de nouveaux nœuds et finalement la traduction de la phrase ainsi obtenue. Plus récemment, Chiang (2007) a proposé un algorithme pour apprendre automatiquement des grammaires hors-contextes synchrones et un algorithme pour utiliser ces grammaires dans un système de traduction statistique.

2.4.4 Exploration des traductions possibles

Comme la résolution de l'équation 2.13 est un problème NP-complet (Knight, 1999 et Zaslavskiy *et al.*, 2009), nous ne pouvons évaluer l'ensemble des traductions possibles pour chaque phrase source. Nous développons plutôt plusieurs traductions partielles en parallèle et ne complétons que les meilleures. Pour pouvoir évaluer une traduction partielle, nous reformulons l'équation 2.14 par :

$$\sum_{k=1}^{p} \sum_{n} \lambda_n \log \left(f_n(\mathbf{a}_k, \mathbf{a}_1^{k-1}, \mathbf{s}, \mathbf{t}^{(k)}) \right) \tag{2.17}$$

où $\mathbf{t}^{(k)}$ est la traduction partielle produite par les alignements $\mathbf{a}_1^k$. De cette façon, nous pouvons ajouter et évaluer les alignements un à la fois pendant la recherche.

L'algorithme le plus utilisé par les systèmes de traduction statistique à base de segments est la recherche en faisceau à plusieurs piles de l'algorithme 2.1 (Koehn,

2004a). Cet algorithme construit la phrase cible de gauche à droite et regroupe par pile les traductions partielles couvrant le même nombre de mots sources. Pour traduire une phrase, il ne suffit donc que d'étendre les traductions partielles de chacune des piles dans l'ordre.

1. Allouer une pile $\mathcal{P}_i$ par mot source.
2. Allouer une pile $\mathcal{P}_0$ et y ajouter une traduction vide.
3. Pour $i \in \{0, 1, ..., l-1\}$
   A. Élaguer $\mathcal{P}_i$
   B. Pour $h \in \mathcal{P}_i$
      a. Pour chaque segment source $\mathbf{s}'$ non traduit dans h
         I. Pour chaque traduction $\mathbf{t}'$ du segment $\mathbf{s}'$
            i. Copier h dans h'.
            ii. Concaténer $\mathbf{t}'$ à la traduction de h'
            iii. Marquer $\mathbf{s}'$ comme traduit dans h'
            iv. Ajouter h' à la pile $\mathcal{P}_{i+|\mathbf{s}'|}$
4. Retourner la meilleure traduction de $\mathcal{P}_l$

Algorithme 2.1 Algorithme de recherche en faisceau par pile

Utiliser cet algorithme tel quel revient à faire une recherche exhaustive et n'a aucun intérêt. Il est cependant efficace si on élague chaque pile avant d'étendre ses traductions partielles. Un premier type d'élagage vient de la programmation dynamique et recombine les traductions partielles dont les complétions possibles sont les mêmes (Nießen *et al.*, 1998). L'avantage de cet élagage est qu'il n'éliminera jamais la traduction partielle menant à la meilleure traduction. Il n'est cependant pas assez agressif pour permettre une traduction rapide. Nous ne gardons donc habituellement que les n meilleures traductions de chaque pile, ce qui accélère le

temps de décodage au risque d'élaguer la traduction qui aurait été la meilleure une fois complétée. Chiang (2007) et Huang et Chiang (2007) suggèrent des algorithmes efficaces pour identifier les n meilleures hypothèses de chaque pile.

En pratique, nous appliquons aussi deux autres types d'élagage. Un premier limite le nombre de traductions par segments dans la table de traduction et un deuxième limite la distorsion (le nombre de mots qui peuvent être sautés entre deux segments sources traduits consécutivement). Moore et Quirk (2007) proposent une méthode pour détecter au plus tôt si un saut futur sera plus grand que celui permis par la limite de distorsion.

Germann *et al.* (2001) et Langlais *et al.* (2007) ont présenté des systèmes permettant d'améliorer la meilleure traduction produite par le décodeur. Ils appliquent un ensemble d'opération pour perturber la traduction initiale et évaluent ces traductions perturbées à l'aide de modèles plus complexes qui peuvent être conditionnées sur toute la traduction plutôt que seulement des traductions partielles.

2.5 Évaluation des traductions

Une fois un système de traduction en main, nous aimerions le comparer à d'autres systèmes ou à ses versions antérieures. L'idéal est bien entendu de demander à un groupe d'experts de comparer la qualité des traductions. Cette approche est cependant coûteuse en temps et en argent, surtout si nous souhaitons évaluer le système à plusieurs étapes de sa conception. Nous évaluons plutôt la qualité des systèmes en comparant automatiquement leur sortie à une traduction de référence produite par un humain. Cela ne nécessite que quelques centaines de phrases traduites, qui peuvent être réutilisées plusieurs fois pour comparer les systèmes.

Le développement de mesures de similarité entre une traduction candidate et une traduction de référence est un domaine de recherche toujours actif. Depuis 2008, un volet des ateliers en traduction statistique (Callison-Burch *et al.*, 2008, 2009) est dédié à l'évaluation des métriques automatiques. La métrique la plus utilisée reste cependant BLEU (Papineni *et al.*, 2002), qui compte le nombre de segments d'un à quatre mots que la traduction candidate a en commun avec la traduction de référence :

$$\text{precision}_n = \frac{\sum_{\mathbf{t},\mathbf{r}} \sum_{seg_n \in \mathbf{t}} \min\left(nb(seg_n, \mathbf{t}), nb(seg_n, \mathbf{r})\right)}{\sum_{\mathbf{t},\mathbf{r}} \sum_{seg_n \in \mathbf{t}} nb(seg_n, \mathbf{t})} \tag{2.18}$$

$$\text{BLEU} = \text{bp} + \exp\left(\frac{1}{4}\sum_{n=1}^{4} \log\left(\text{precision}_n\right)\right) \tag{2.19}$$

où $\mathbf{t}$ est une traduction candidate, $\mathbf{r}$ la traduction de référence et nb une fonction retournant la fréquence d'un segment dans une phrase. En n'utilisant que le nombre de segments en commun (precision_n), les phrases vides obtiennent un score parfait. Pour compenser ce problème, nous pénalisons les traductions candidates plus courtes que la traduction de référence :

$$\text{bp} = \begin{cases} 1 & \text{Si } \sum_{\mathbf{t}} |\mathbf{t}| > \sum_{\mathbf{r}} |\mathbf{r}| \\ 1 - \frac{\sum_{\mathbf{r}} |\mathbf{r}|}{\sum_{\mathbf{t}} |\mathbf{t}|} & \text{Sinon.} \end{cases} \tag{2.20}$$

BLEU accorde un score de zéro aux mauvaises traductions et un score unitaire aux traductions parfaites. Notons aussi que cette métrique est calculée sur les documents en entier et n'est pas une moyenne sur les phrases.

Plutôt que de ne compter que les mots en communs, METEOR (Banerjee et Lavie, 2005) considère aussi leurs synonymes, leur étiquette morpho-syntaxique ou tout autre attribut. Une autre métrique populaire est TER (*Translation Edit Rate*) proposée par Snover *et al.* (2006). Cette métrique calcule le nombre minimal d'opérations

d'éditions pour produire la traduction de référence à partir de la traduction proposée par le système.

Finalement, Callison-Burch (2009) propose de remettre les humains dans le processus d'évaluation en utilisant la plate-forme *Mechanical Turk*[6] , qui fait appel à des internautes prêt à accomplir différentes tâches à très peu de frais.

2.6 Résumé

La première étape pour la conception d'un système de traduction statistique est de choisir les corpus sur lesquels l'entrainer. Pour des performances acceptables, ce corpus devrait être similaire à ce qui sera traduit. Les documents sont ensuite alignés au niveau des phrases et des mots à l'aide de méthodes statistiques. Les alignements de mots sont ensuite utilisés pour l'extraction des bisegments qui seront utilisés par le décodeur pour traduire de nouvelles phrases.

Les modèles statistiques de ce chapitre posent plusieurs hypothèses d'indépendance afin de simplifier leur calcul. Le prochain chapitre décrit certaines de ces hypothèses et présente certains travaux qui ont cherché à les relaxer.

[6] http://mturk.com/

3 Intégration du contexte

Les systèmes de traduction à base de segments posent plusieurs hypothèses d'indépendance assez fortes qui les empêchent d'avoir une vue d'ensemble sur la phrase à traduire. Par exemple, les modèles d'alignement de mots supposent que les alignements sont indépendants ou ne dépendent que de la position de l'alignement précédent. Le modèle de langue assigne toujours la même probabilité à une phrase cible, peu importe la phrase traduite. Tout comme les scores des bi-segments qui ne dépendent pas non plus de la phrase à traduire.

Dans ce chapitre, nous présentons différents travaux qui relâchent ces hypothèses d'indépendance pour ajouter des informations contextuelles aux modèles. Une première méthode présentée à la section 3.1 modélise le thème des phrases à traduire. Les thèmes permettent une modélisation assez grossière du contexte, une méthode plus fine, présentée à la section 3.2 modélise les dépendances directement au niveau des mots. Finalement, plutôt que de chercher à ajouter le contexte aux modèles existants, la section 3.3 présente de nouveaux modèles qui cherchent à prédire les mots de la traduction en considérant tous les mots de la phrase source.

3.1 Thème

La probabilité que le mot *plant* se traduise par *plante* ou *usine* devrait intuitivement être différente dans un texte de botanique et dans un texte traitant du développement industriel. Connaitre le thème d'une phrase peut donc aider à la traduire. Un problème se pose cependant lorsqu'un corpus peut contenir plusieurs thèmes. Les approches que nous présentons dans cette section s'attaquent à ce problème en utilisant le contexte pour déterminer le thème de la phrase à traduire.

La notion de thème est prise ici au sens large. Les thèmes s'imposent d'eux-mêmes lorsque les sections d'un corpus général sont bien identifiées. Ce cas est cependant assez rare et il faut souvent se résoudre à définir les thèmes à l'aide d'un algorithme de partitionnement automatique (*clustering*) ou implicitement par tout ce qui est similaire au document à traduire.

Une fois les thèmes établis, il faut les intégrer au système de traduction. Nous présentons à la section 3.1.1 comment ils ont été intégrés aux alignements de mots, à la section 3.1.2 aux modèles de langue et la section 3.1.3 aux tables de traduction. Dans tous les cas, l'idée générale est d'entrainer un modèle par thème et de les combiner selon la phrase ou le document à traduire ou à aligner.

3.1.1 Alignement de mots

Le modèle IBM1 présenté à la section 2.2.1 suppose que la probabilité de chaque alignement a_i ne dépend que de la paire de mots qui le compose :

$$\mathbf{a}_{\mathrm{ibm1}} = \underset{\mathbf{a}\in\mathcal{A}(\mathbf{s},\mathbf{t})}{\arg\max} \prod_{i=1}^{l} \Pr(s_i \mid t_{a_i}) \tag{3.1}$$

où $\mathcal{A}(\mathbf{s},\mathbf{t})$ retourne l'ensemble de tous les alignements de mots possibles entre $\mathbf{s}$ et $\mathbf{t}$. Zhao et Xing (2006) proposent de lui intégrer la notion de thème avec leur modèle de *Bilingual Topic AdMixture* (BiTAM). Leur modèle définit les thèmes par une variable cachée z qu'ils intègrent à IBM1 de trois façons.

Une première suppose qu'une phrase source ne porte que sur un seul thème :

$$\mathbf{a}_{\mathrm{bitam1}} = \underset{\mathbf{a}\in\mathcal{A}(\mathbf{s},\mathbf{t})}{\arg\max} \sum_{z\in\mathcal{Z}} \Pr(z) \prod_{i=1}^{l} \Pr(s_i \mid t_{a_i}, z) \tag{3.2}$$

La probabilité d'un alignement ne dépend donc plus de chaque alignement individuel, mais aussi du reste de la phrase source par son thème.

Dans le deuxième modèle, la phrase cible dépend aussi du thème :

$$\mathbf{a}_{\mathrm{bitam}_2} = \underset{\mathbf{a}\in\mathcal{A}(\mathbf{s},\mathbf{t})}{\arg\max} \sum_{z\in\mathcal{Z}} \Pr(z)\,\Pr(\mathbf{t} \mid z) \prod_{i=1}^{l} \Pr(s_i \mid t_{a_i}, z) \tag{3.3}$$

Finalement, le troisième modèle n'utilise aucun contexte et permet à chaque alignement d'avoir son propre thème :

$$\mathbf{a}_{\mathrm{bitam}_3} = \underset{\mathbf{a}\in\mathcal{A}(\mathbf{s},\mathbf{t})}{\arg\max} \prod_{i=1}^{l} \sum_{z\in\mathcal{Z}} \Pr(z)\,\Pr(s_i \mid t_{a_i}, z) \tag{3.4}$$

Même si ce dernier modèle n'utilise pas le contexte pour décider du thème, il produit tout de même des alignements de qualité comparable aux deux autres.

En fixant le nombre de thèmes à dix, les trois modèles améliorent les alignements d'IBM1 et donnent des résultats comparables à ceux d'IBM4 pour des alignements entre l'anglais et le chinois.

Zhao et Xing (2007) ont adapté le modèle de l'équation 3.3 aux HMM (section 2.2.2). En intégrant ces alignements à un système de traduction à base de segments, ils ont observé une amélioration des traductions. Le modèle de l'équation 3.2 a aussi été adapté aux modèles IBM2 (Civera et Juan, 2006) et HMM (Civera et Juan, 2007).

3.1.2 Modèles de langue

Les formulations de phrases sont souvent différentes d'un type de texte à un autre. Ajouter un modèle de langue entrainé sur un corpus thématique à un système de traduction général est une manière efficace de l'adapter à ce nouveau domaine (Koehn et Schroeder, 2007). Cette approche est simple si nous connaissons à l'avance le thème des documents à traduire. Cette section présente des approches de rechange lorsque le thème des phrases à traduire varie ou est inconnu.

Si le nombre de thèmes est borné et connu, Xu *et al.* (2007) proposent d'entrainer un modèle de langue par thème et de les ajouter au modèle exponentiel présenté à l'équation 2.14. Ils utilisent les mêmes modèles pour tous les thèmes, mais adaptent les combinaisons de poids. Lors de la traduction, un classificateur considère la phrase en entier pour détecter son thème et choisit ensuite la combinaison de poids qui lui correspond.

Plutôt que de regrouper les phrases par thème à priori, Zhao *et al.* (2004) génèrent des corpus thématiques adaptés pour chaque phrase à traduire. Ils produisent une première traduction en utilisant un modèle de langue général. Un moteur de recherche est ensuite utilisé pour récupérer les phrases cibles d'un grand corpus cible qui lui sont similaires. La traduction finale est produite avec un modèle de langue entrainé sur ces phrases similaires.

Dans ces deux cas, des améliorations ont été observées en faisant dépendre les probabilités du modèle de langue sur la phrase source. La première approche ne s'applique que si les thèmes sont connus d'avance et le deuxième demande la création d'un modèle de langue adapté avant de traduire chaque phrase.

3.1.3 Table de traduction

Il est préférable de produire la table de traduction avec des corpus similaires à ceux à traduire (Langlais *et al.*, 2006). Si le corpus d'entrainement est trop général pour la tâche de traduction, Hildebrand *et al.* (2005) et Lü *et al.* (2007) proposent d'utiliser un moteur de recherche d'informations pour récupérer les phrases qui sont similaires à celles à traduire et leur donner plus d'importance lors de l'entrainement. Cette approche est flexible, mais elle requiert l'entrainement d'un nouveau système de traduction pour chaque document à traduire.

Lorsque le corpus d'entrainement est divisé en thèmes et que le thème des phrases à traduire est inconnu, on peut aussi entrainer un système de traduction par thème et les interpoler selon la phrase à traduire. Pour ce faire, Lü *et al.* (2007) soumettent chaque phrase à traduire à un système de recherche d'informations pour obtenir les phrases du corpus d'entrainement qui lui sont les plus similaires. En sélectionnant le modèle du thème pour lequel le plus de phrases ont été retournées, les auteurs obtiennent de meilleures traductions qu'en considérant tout le corpus d'entrainement[7] .

Il est donc possible d'améliorer un système de traduction en découpant les données d'entrainements par thèmes et en sélectionnant le bon thème pour chacune des phrases à traduire. Encore une fois, cela demande de définir une liste fixe de thèmes à l'avance ou d'entrainer un système pour chaque document à traduire.

3.2 Modèles de désambigüisation

Les thèmes sont une approximation très grossière du contexte. Plusieurs travaux ont cherché à en utiliser une représentation plus fine en conditionnant les modèles directement sur le contenu du contexte. Par exemple, on pourrait augmenter la probabilité que le mot *plant* se traduise par *usine* si le mot *worker* est présent dans la phrase précédente.

L'idée n'est pas nouvelle et a été intégrée à CANDIDE (Berger *et al.*, 1994), un des premiers systèmes de traduction statistique. Dans ce système, un modèle d'entropie maximale considère le mot source, le mot cible et le contexte du mot cible pour

[7] Les auteurs ont testé d'autres types d'interpolation, mais elles ont toutes apportées à peu près les même gains.

évaluer les différentes traductions d'un mot source. Malheureusement, l'impact de ces modèles sur la qualité des traductions n'a pas été rapporté.

Cette idée a resurgi au cours des dernières années, mais cette fois sous le paradigme de la désambigüisation des mots. En effet, on peut considérer un mot ayant plusieurs traductions comme un mot polysémique dont les sens sont ses traductions.

La section 3.2.1 présente des expériences pour intégrer des modèles de désambigüisation des mots à un système de traduction. La section 3.2.2 montre comment ces modèles ont été généralisés pour permettre la désambigüisation de segments. Finalement, la section 3.2.3 présente comment ils ont été intégrés à l'alignement des mots.

3.2.1 Désambigüisation des mots

Un problème qui revient souvent en traitement automatique des langues est la gestion des mots polysémiques comme « plant » qui peut être une usine ou une plante selon le contexte où il est utilisé. C'est le problème que cherchent à régler les systèmes de désambigüisation, qui associent chaque usage d'un mot polysémique à l'un de ses sens dans un thésaurus tel que WORDNET (Fellbaum, 1998) pour l'anglais ou HOWNET (Dong et Dong, 2006) pour le chinois.

Carpuat et Wu (2005) proposent de contraindre la traduction d'un mot source par un mot cible du même sens. Ils utilisent un système de désambigüisation à l'état de l'art (Carpuat *et al.*, 2004) qui considère les mots entourant le mot polysémique, les séquences de deux ou trois mots l'incluant et certaines collocations syntaxiques dans lesquelles il est impliqué (par exemple un verbe et son complément d'objet).

Les étiquettes morphosyntaxiques et les lemmes de ces mots et séquences sont aussi considérés.

Les sens assignés par leur système sont tirés de HOWNET, qui proposent aussi une traduction anglaise pour chacune de ses entrées. Cette traduction est ensuite insérée dans la sortie du système de traduction. En faisant de la sorte, les auteurs rapportent une dégradation des traductions du chinois vers l'anglais. Ces résultats ne sont pas surprenants étant donné que les traductions de HOWNET sont ajoutées sans égard du contexte cible dans une étape de post-traitement plutôt que d'être intégrées au système de traduction.

Plutôt que d'utiliser un inventaire de sens défini dans un thésaurus, Vickrey *et al.* (2005) considèrent que les sens d'un mot sont ses traductions. Leur système de désambigüisation utilise un modèle de régression logistique prenant en entrée l'étiquette morphosyntaxique du mot à traduire, les quatre mots le précédant et les quatre mots le suivant.

Comme les auteurs ne disposaient pas d'un système de traduction qu'ils pouvaient adapter, ils ont évalué leur approche sur une tâche de remplissage de blancs où les blancs sont les traductions de mots ambigus.

Dans une tâche de traduction du français vers l'anglais, la qualité des mots remplaçant les blancs augmente lorsque le système de désambigüisation est intégré à un système de base composé d'un modèle de langue et d'une table de traduction. Il est cependant difficile de prédire si cette amélioration serait observée sur une tâche de traduction complète, car la traduction produite n'est pas d'aussi bonne qualité que le texte entourant les blancs.

Cabezas et Resnik (2005) considèrent aussi que les sens d'un mot sont définis par ses traductions. Leur système de désambigüisation prend en entrée des collocations de mots et d'étiquettes morphosyntaxiques dans les trois mots précédent et suivant celui à traduire. Ils considèrent aussi les mots de la phrase à traduire ainsi que ceux de la phrase la précédent et de la phrase la suivant.

Ce modèle a été intégré à un système de traduction à base de segments dans une tâche de traduction de l'espagnol vers l'anglais. Une légère amélioration de la qualité des traductions a été remarquée, mais les auteurs ne croient pas qu'elle soit statistiquement significative.

Les modèles de désambigüisation des mots ne sont pas simples à intégrer aux systèmes de traduction à base de segments, car les unités de base de ces systèmes sont les segments plutôt que les mots. La prochaine section présente comment la désambigüisation peut être appliquée directement sur les bisegments.

3.2.2 Désambigüisation des bisegments

Les scores des bisegments (section 2.4.1) sont toujours les mêmes, peu importe le contexte où ils apparaissent. Cette section présente comment les techniques de désambigüisation des mots ont été adaptées à la désambigüisation des segments.

L'idée générale reste la même. Les sens d'un segment sont définis par ses traductions et les segments sont désambigüisés en examinant leur contexte. Un classificateur associe une probabilité à chaque segment cible et cette probabilité est ajoutée au modèle exponentiel de l'équation 2.14. Les approches présentées dans cette section se distinguent par leur définition du contexte et leur classificateur.

Stroppa *et al.* (2007) n'utilisent qu'un seul arbre de décision pour tous les segments. Cet arbre prend en entrée le segment à traduire, ses étiquettes morphosyntaxiques

ainsi que les mots et les étiquettes morphosyntaxiques des deux mots précédent et suivant le segment à désambigüiser. Ils rapportent ainsi des améliorations en traduction du chinois et de l'italien vers l'anglais.

Ces résultats ont été validés sur des traductions de l'anglais vers le français par Max *et al.* (2009). Ces derniers rapportent des améliorations supplémentaires en ajoutant les collocations syntaxiques au contexte (par exemple, le complément d'un verbe contenu dans le segment).

Chan *et al.* (2007) restreignent le contexte de Stroppa *et al.* (2007) seulement au mot précédent et au mot suivant l'unité à désambigüiser et lui ajoute les mots apparaissant au moins trois fois dans la phrase source, la phrase la précédent ou la phrase la suivant. En utilisant une machine à vecteurs de support comme classificateur, ils ont amélioré les traductions du chinois vers l'anglais de HIERO (Chiang, 2007), un système de traduction hiérarchique.

Giménez et Màrquez (2007) utilisent un contexte beaucoup plus large en considérant la forme, l'étiquette morphosyntaxique, le lemme et l'étiquette de *chunking* IOB[8] des cinq mots précédent et suivant le segment à traduire. Tous les {1,2,3}-grams de cette fenêtre sont considérés en plus des lemmes de la phrase source. En considérant tout ce contexte, ils rapportent une légère amélioration sur une tâche de traduction de l'espagnol vers l'anglais lors d'une comparaison manuelle à un système ne faisant pas usage du contexte.

Carpuat et Wu (2007) considèrent aussi les {1,2,3}-grams de mots, d'étiquettes morphosyntaxiques et de lemmes, mais ils leur ajoutent les collocations syntaxi-

[8] L'étiquette IOB spécifie si un mot commence un syntagme nominal (B), le continue (I) ou ne fait pas partie d'aucun (O).

ques. Ils rapportent des traductions de meilleure qualité en intégrant ce contexte à un système de traduction du chinois vers l'anglais.

Plutôt que d'utiliser un classificateur, Gimpel et Smith (2008) ajoutent plusieurs probabilités directement dans le modèle exponentiel de l'équation 2.14. Ces probabilités sont calculées sur les mots entourant le segment, leur étiquette morphosyntaxique, diverses propriétés de l'arbre d'analyse syntaxique de la phrase et la position du segment dans la phrase source. Leurs résultats montrent que le type de contexte aidant à la désambigüisation d'un segment dépend des langues impliquées. Leur système de traduction du chinois vers l'anglais bénéficie des mots et des étiquettes morphosyntaxiques entourant le segment alors que leur système de traduction de l'anglais vers l'allemand bénéficie plus du contexte syntaxique. Ils proposent donc un algorithme pour sélectionner automatiquement les types de contextes importants et rapportent des améliorations en traduction.

La désambigüisation des bisegments à l'aide du contexte source permet donc d'améliorer les systèmes de traduction. En analysant les raisons de cette amélioration, Carpuat et Wu (2008) remarquent que le contexte permet de mieux utiliser les bisegments plus longs et moins fréquents. Bien qu'il n'y ait encore aucun consensus sur la manière d'intégrer le contexte, n'utiliser que quelques mots autour du segment source et leur étiquette morphosyntaxique aide à améliorer la qualité des traductions.

3.2.3 Alignement de mots

Les algorithmes de désambigüisation peuvent aussi aider l'alignement des mots. Ils permettent d'intégrer le contexte aux probabilités lexicales des alignements afin de les évaluer par $\Pr(s_i \mid t_{a_i}, c_{a_i})$ où c_{a_i} est le contexte de a_i, plutôt que par $\Pr(s_i \mid t_i)$.

Pour ce faire, Varea *et al.* (2002) définissent le contexte d'un alignement par les trois mots précédents et les trois mots suivants son mot cible. Comme plusieurs mots sont rares et que les approches statistiques manquent de robustesse face aux événements rares, ils les regroupent en classes découvertes par partitionnement automatique (*clustering*) et conditionnent leurs modèles sur ces classes plutôt que sur les mots. Ils entrainent ensuite un modèle d'entropie maximale pour prédire la probabilité d'un mot source étant donné un mot cible et son contexte. Ils réussissent à améliorer la qualité des alignements des modèles IBM 1 à 5.

Plutôt que de partitionner les mots en classes, Brunning *et al.* (2009) partitionnent directement les contextes. Pour ce faire, ils utilisent un arbre de décision prenant en entrée les étiquettes morphosyntaxiques du mot aligné, du mot le précédant et du mot le suivant dans les phrases sources et cibles. Un modèle IBM1 conditionné sur ces classes de contextes produit des alignements de qualités similaires à un modèle HMM. Ajouter le contexte à un modèle HMM l'améliore et améliore la sortie de leur système de traduction pour les langues arabes et anglaises.

3.3 Prédiction d'un vocabulaire actif

Les approches présentées aux sections précédentes permettent de désambigüiser la traduction d'un mot ou d'un segment en utilisant son contexte ou son thème. Cette section propose une approche différente où nous cherchons plutôt à prédire si chaque mot du vocabulaire cible devrait faire partie ou non d'une traduction. Nous pouvons comparer les mots prédits avec une grande probabilité à un vocabulaire actif qui sera privilégié lors de la traduction.

Cette approche a été proposée par Venkatapathy et Bangalore (2009) pour traduire vers ou à partir de langues où l'ordre des mots n'est pas significatif comme l'hindi

ou le japonais. Ils décrivent un système de traduction composé d'un lexique global dont les prédictions sont utilisées pour composer une phrase à l'aide d'un modèle de réordonnancement.

Leur lexique global fait ses prédictions à l'aide d'un modèle exponentiel prenant en entrée l'ensemble des unigrammes, des bigrammes et des trigrammes de la phrase à sources. Ils l'ont utilisé pour trois types de prédictions : la prédiction de mots, la prédiction de mots de contenus accompagnés de leurs mots outils et la prédiction de mots factorisés (mots accompagnés de leurs étiquettes morphologiques et de leurs mots outils).

Les auteurs ont évalué leur système sur des tâches de traduction entre l'anglais et l'hindi. En comparant les mots prédits par leur modèles à ceux choisis dans les traductions de MOSES (Koehn et Schroeder, 2007), ils rapportent une meilleure sélection lexicale par leur approche. Cependant, si on considère les traductions finales, avec le modèle de réordonnancement inclus, la sortie de MOSES est de meilleure qualité.

Mauser *et al.* (2009) utilisent un modèle similaire, mais qui ne considère que les unigrammes de la phrase source. Ils l'ont intégré à un système de traduction statistique à base de segments et rapporte des gains dans une tâche de traduction du chinois vers l'anglais.

Un modèle simple permettant aussi de prédire les mots de la phrase cible à partir de la phrase source en entier est IBM1, où la probabilité d'un mot cible est :

$$\Pr(t \mid \mathbf{s}) = \frac{1}{l+1} \sum_{i=0}^{l} \Pr(t \mid s_i) \tag{3.5}$$

Ce modèle a été utilisé avec succès pour réordonner des listes de n-meilleures hypothèses de traduction (Och *et al.*, 2004).

IBM1 considère cependant que les mots sources sont indépendants les uns des autres. C'est pourquoi Hasan *et al.* (2008) et Mauser *et al.* (2009) suggèrent de modéliser plutôt les collocations sources :

$$\Pr(t \mid \mathbf{s}) = \frac{2}{m(m+1)} \sum_{i=0}^{l} \sum_{i'=i+1}^{l} \Pr(t \mid s_i, s_{i'}) \tag{3.6}$$

Ce modèle supplante IBM1 en traduction, mais fait aussi augmenter rapidement le nombre de paramètres à optimiser.

3.4 Résumé

Les systèmes de traduction statistiques à base de segments ne considèrent que très peu de contexte pour chacune de leur décision et ce chapitre a présenté plusieurs méthodes pour y remédier.

Le contexte peut d'abord être considéré en introduisant la notion de thème. Cette approche est utile lorsque les thèmes des documents à traduire sont connus à l'avance, mais demande d'entrainer un nouveau système de traduction lorsque ce n'est pas le cas.

Une seconde approche consiste à adapter les algorithmes de désambigüisation des mots à la désambigüisation de segments. Cette approche a apporté des résultats intéressants, mais elle est lourde à mettre en place, car elle demande d'entrainer un classificateur pour chacun des segments sources.

Finalement, une troisième approche cherche plutôt à prédire un vocabulaire pertinent à la phrase à traduire. Ces prédictions peuvent être utilisées directement par un modèle de réordonnancement ou intégrées au décodeur. C'est l'approche que nous avons choisi dans le cadre de cette thèse pour intégrer le contexte à notre

système de traduction. Plutôt que d'utiliser un modèle exponentiel (Venkatapathy et Bangalore, 2009 et Mauser *et al.*, 2009) ou un modèle linéaire comme (Hasan *et al.*, 2008), nous optons pour un perceptron à plusieurs couches. Le chapitre suivant présente les perceptrons à plusieurs couches et explique ce qu'ils offrent par rapport aux autres modèles.

4 Perceptrons à plusieurs couches

Dans le cadre de cette thèse, nous nous intéressons à prédire les mots traduisant une phrase source. Les systèmes de traduction font déjà ce travail, mais en ne considérant que très peu d'information pour y arriver. Comme nous l'avons vu précédemment, la présence d'un mot cible ne dépend que du segment source traduit et de quelques mots cibles l'entourant. Dans ce chapitre, nous proposons un modèle considérant la phrase source en entier pour prédire si un mot cible doit faire partie ou non de sa traduction.

Nous présentons la tâche de la prédiction des mots cibles à la section 4.1 que nous formulons à la section 4.2 comme un problème de classification binaire probabiliste. Les sections 4.3 et 4.4 présentent les modèles de régression linéaire et logistique qui ont déjà été employés à cet effet. Nous reformulons ensuite ces deux modèles de régression en perceptrons et nous présentons les perceptrons à plusieurs couches aux sections 4.5 et 4.6. Nous résumons finalement ce chapitre en section 4.7.

4.1 Cadre théorique

Nous cherchons à prédire la présence ou l'absence de mots cibles dans la traduction d'une phrase source. Pour ce faire, nous interprétons chaque prédiction comme une épreuve de Bernouilli où la présence du mot correspond à un succès et son absence à un échec. Si nous répétons cette épreuve pour chacun des mots du vocabulaire cible, la probabilité de voir un ensemble de mots dans une traduction s'exprime par la distribution binomiale :

$$\Pr(\mathbf{t} \mid \mathbf{s}) = \prod_{t \in \mathcal{T}} \Pr(t \mid \mathbf{s})^{\mathbf{1}[t \in \mathbf{t}]} \, (1 - \Pr(t \mid \mathbf{s}))^{\mathbf{1}[t \notin \mathbf{t}]} \tag{4.1}$$

$$= \overbrace{\prod_{t \in \mathbf{t}} \Pr(t \mid \mathbf{s})}^{\text{Mots présents}} \overbrace{\prod_{t \in \mathcal{T} \setminus \mathbf{t}} 1 - \Pr(t \mid \mathbf{s})}^{\text{Mots absents}} \tag{4.2}$$

où $\mathcal{T}$ est l'ensemble des mots cibles connus, $\mathbf{s}$ la phrase source, $\mathbf{t}$ la phrase cible et $\mathbf{1}[p]$ s'évalue à 1 si le prédicat p est vrai et 0 sinon. L'équation 4.2 est valide dans la mesure où nous posons l'hypothèse, manifestement fausse, que les mots cibles sont conditionnellement indépendants les uns des autres une fois la phrase source connue.

En supposant aussi que les différentes phrases d'un document sont indépendantes les unes des autres, la probabilité que le document D_t traduise le document D_s est s'exprime par :

$$\Pr(D_t \mid D_s) = \prod_{(\mathbf{s},\mathbf{t}) \in \mathcal{A}(D_s, D_t)} \Pr(\mathbf{t} \mid \mathbf{s}) \tag{4.3}$$

où $\mathcal{A}(D_s, D_t)$ retourne l'ensemble des paires de phrases en relation de traduction dans les deux documents.

En utilisant les équations 4.2 et 4.3, il est possible d'évaluer la probabilité d'une traduction d'un document complet en ne modélisant que $\Pr(t \mid \mathbf{s})$, la probabilité d'un mot cible étant donnée une phrase source. Le reste de ce chapitre présente différents modèles pour estimer cette probabilité.

4.2 Interprétation en problème de classification

Nous cherchons à modéliser $\Pr(t \mid \mathbf{s})$ à partir d'un bitexte en utilisant des techniques d'apprentissage automatique. Comme les techniques que nous utilisons

ici manipulent des vecteurs plutôt que des phrases, nous parlerons de traits et de vecteurs de traits plutôt que de mots et de phrases. Pour alléger le texte, nous utilisons cependant « mot » et « trait » ainsi que « phrase » et « vecteur » de manière interchangeable.

Comme nous manipulons des vecteurs de taille finie, nous devons établir la liste des mots qui nous intéressent. Nous nommons cette liste le vocabulaire. En associant un indice à chacun des mots du vocabulaire, encoder une phrase consiste à construire un vecteur dont toutes les valeurs sont 0 sauf celles aux indices des mots de la phrase qui sont à 1. La section 4.6.4 présente des exemples de phrases encodées en vecteurs.

Encoder les phrases en vecteurs de traits nous permet d'utiliser les algorithmes d'apprentissage automatique de la littérature mais nous force à ignorer l'ordre des mots. Ainsi, « The box is in the pen » et « The pen is in the box » sont toutes deux encodées par le même vecteur alors qu'elles se traduisent différemment. Nous ne nous sommes pas intéressés à ce problème, mais la méthode la plus simple pour le résoudre partiellement serait de faire comme Bangalore *et al.* (2007) et d'encoder la structure locale des phrases en ajoutant des séquences de mots au vocabulaire.

Pour distinguer les phrases et leur représentation vectorielle dans les équations, nous représentons les vecteurs de traits sources et cibles par $\vec{\phi}$ et $\vec{\tau}$. Nous pouvons donc réécrire l'équation 4.2 par :

$$\begin{aligned} \Pr(\mathbf{t} \mid \mathbf{s}) &= \Pr(\vec{\tau} \mid \vec{\phi}) \\ &= \prod_{\{j|\tau_j=1\}} \Pr(\tau_j = 1 \mid \vec{\phi}) \prod_{\{j|\tau_j=0\}} 1 - \Pr(\tau_j = 1 \mid \vec{\phi}) \end{aligned} \tag{4.4}$$

Le problème ainsi posé, il ne reste qu'à estimer la probabilité qu'un mot soit présent ou absent de la traduction. Nous utilisons à cette fin un classificateur binaire

probabiliste où les deux classes sont « présent » et « absent ». Ce classificateur $f(\cdot)$ estimera la probabilité de chaque mot cible étant donnée la phrase source :

$$\vec{y} = f(\vec{\phi}) \tag{4.5}$$

où $\vec{y}$ est un vecteur de dimension J dont la j-ème composante contient la probabilité la probabilité que le j-ème mot du vocabulaire cible soit présent dans la traduction de la phrase source encodée par $\vec{\phi}$. Les sections suivantes présentent différents choix pour $f(\cdot)$.

4.3 Régression linéaire

La régression linéaire estime $f(\cdot)$ à l'aide de l'équation suivante :

$$f_{lin}(\vec{\phi}) = W\vec{\phi}' \tag{4.6}$$

où

$$\vec{\phi}' = \frac{\vec{\phi}}{\sum_{i=1}^{I} \phi_i} \quad \text{et} \quad w_{ji} = Pr(\tau_i \mid \phi_j)$$

La probabilité d'un mot cible est donc la moyenne des probabilités qu'il traduise chaque mot source. Pour nous assurer que les valeurs de $\vec{y}$ sont comprises entre 0 et 1, les valeurs de W sont restreintes entre 0 et 1 et ses colonnes somment à 1.

Les coefficients de W peuvent être optimisés à l'aide de l'algorithme EM qui cherche à maximiser la vraisemblance d'un bitexte d'entrainement. C'est ce qu'ont fait Brown *et al.* (1993) pour le modèle IBM1 et Mauser *et al.* (2009) pour leur modèle à base de déclencheurs.

4.4 Régression logistique

Les modèles de régression logistique estiment $f(\cdot)$ à l'aide de l'équation suivante :

$$f_{log}(\vec{\phi}) = \frac{exp(\Lambda\vec{\phi})}{exp(\Lambda\vec{\phi}) + exp(\vec{\Lambda'}\vec{\phi})} \tag{4.7}$$

où Λ et Λ' sont des matrices de dimension $m \times l$ dont les éléments à la position (j, i) mesurent respectivement la corrélation entre la présence ou l'absence de τ_j et la présence de ϕ_i.

Les modèles discriminants de Bangalore *et al.* (2007) et Mauser *et al.* (2009) présentés au chapitre 2 utilisent la régression logistique.

L'entrainement de ces modèles se fait à l'aide d'algorithme qui choisissent aléatoirement des valeurs initiales pour Λ et Λ' pour ensuite passer plusieurs itérations à les améliorer. Il existe plusieurs algorithmes pour définir la nature des mises à jour, la plus populaire étant le *Generalized Iterative Scaling* (Darroch et Ratcliff, 1972). Cet algorithme est cependant coûteux pour les tâches de traitement automatique des langues où il peut y avoir des centaines de milliers de paramètres à ajuster. Comme l'entrainement de ces modèles dépasse le sujet de cette thèse, nous invitons les lecteurs intéressés à consulter Malouf (2002) pour une évaluation de différents algorithmes de mise à jour pour des modèles de régression logistique utilisés en traitement automatique des langues.

4.5 Perceptrons

Les modèles de régression linéaire et logistique peuvent tous deux être interprétés comme des perceptrons, donc décrits par :

$$\vec{y} = g(W\vec{\phi}) \tag{4.8}$$

où W est une matrice de poids liant les mots sources aux mots cibles et $g(\cdot)$ est une fonction monotone que nous nommons la fonction d'activation. La fonction

d'activation est l'identité pour la régression linéaire et $\mathtt{sigmoid}$ pour la régression logistique si nous réécrivons l'équation 4.7 par :

$$\vec{y} = \mathtt{sigmoid}(W\vec{\phi}) \quad (4.9)$$

$$W = \Lambda' - \Lambda \quad (4.9a)$$

$$\mathtt{sigmoid}(x) = \frac{1}{1+e^{-x}} \quad (4.9b)$$

Les perceptrons sont simples à utiliser et à optimiser, mais leur puissance d'expression est limitée. Par exemple, si l'on ne retient que les mots prédits avec une probabilité supérieure à α, le vecteur de traits cibles est :

$$y_j = \begin{cases} 1 & \text{si } (W\vec{\phi})_j > g^{-1}(\alpha) \\ 0 & \text{sinon} \end{cases} \quad (4.10)$$

Les deux types de prédictions sont donc séparés par l'hyperplan $(W\vec{\phi})_j > g^{-1}(\alpha)$. C'est pourquoi nous disons que les perceptrons ne peuvent modéliser que les problèmes linéairement séparables.

Le perceptron à plusieurs couches est une extension naturelle qui permet au perceptrons de modéliser des problèmes non-linéairement séparables. La section suivante les décrit et motive leur utilisation.

4.6 Perceptrons à plusieurs couches

Un perceptron à plusieurs couches (PPC) est un ensemble de perceptrons connectés les uns aux autres. Dans le cadre de cette thèse, nous utilisons des perceptrons à une couche cachée :

$$\begin{aligned} &\text{perceptron 1} && \vec{h} = g_1(W\vec{\phi}) \\ &\text{perceptron 2} && \vec{y} = g_2(V\vec{h}) \end{aligned} \quad (4.11)$$

où $\vec{h}$, la couche cachée, est une représentation alternative de $\vec{\phi}$ que le PPC apprend à construire à partir des données.

Nos PPC fonctionnent donc en deux étapes. Un premier perceptron projette la phrase source vers une représentation alternative habituellement de plus petite dimension $\vec{h}$. Un deuxième perceptron utilise ensuite $\vec{h}$ pour prédire les mots de la phrase cible.

4.6.1 Motivation

Les algorithmes d'apprentissage automatique posent l'hypothèse que des entrées similaires sont associées à des sorties similaires. Pour nous, cela signifie que des phrases sources similaires devraient avoir des traductions similaires. Cette hypothèse est généralement vérifiée si nous encodons les phrases par des vecteurs représentant des sacs de mots comme à la section 4.2. En effet, les phrases ne se différenciant que de quelques mots comme « the walls of this plant » et « the walls of this factory » auront des traductions similaires.

Il y a cependant des cas où les mots sont des unités trop grossières pour bien refléter la similarité entre deux phrases. C'est ce que nous remarquons en ajoutant « the leaves of this plant » aux deux exemples précédents. Bien que cette phrase ne diffère que d'un mot de « the walls of this plant », elle n'y est pas aussi semblable que « the walls of this factory ». Comme nous en discutons à la section 4.6.4, nous espérons que la représentation alternative $\vec{h}$ permettra plus de nuances que $\vec{\phi}$.

Alors que $\vec{\phi}$ doit avoir un trait pour chaque mot du vocabulaire, la taille de $\vec{h}$ est arbitraire et permet de configurer la puissance d'expression du PPC. De plus, les valeurs de $\vec{h}$ sont continues plutôt que binaires. La représentation cachée peut

donc encoder un nombre infini de phrases et la distance entre deux phrases n'est plus déterminée par le nombre de mots qu'elles ne partagent pas.

Il est très difficile de prévoir les abstraction qui seront capturées par cette couche cachée, mais nous pouvons espérer qu'elles rapprocheront les phrases sémantiquement similaires. Cet objectif motive notre démarche mais rien ne garantie qu'il sera atteint. Nous revenons sur l'interprétation de la couche cachée aux sections 4.6.4 et 5.6.

Nous sommes cependant encouragés par les travaux de Bengio *et al.* (2003) qui ont amélioré l'état de l'art des modèles de langue en utilisant un PPC. Ce modèle de langue a été utilisé avec succès en traduction statistique (Schwenk *et al.*, 2007) ainsi qu'en reconnaissance de la parole (Schwenk, 2007).

4.6.2 Architectures

Les perceptrons à plusieurs couches que nous utilisons ont l'architecture suivante:

$$\begin{aligned} &\text{perceptron 1} \qquad \vec{h} = \tanh(W\vec{\phi}) \\ &\text{perceptron 2} \qquad \vec{y} = \mathrm{sigmoid}(V\vec{h} + \vec{b}) \end{aligned} \tag{4.12}$$

$$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

$$\mathrm{sigmoid}(x) = \frac{1}{1 + e^{-x}}$$

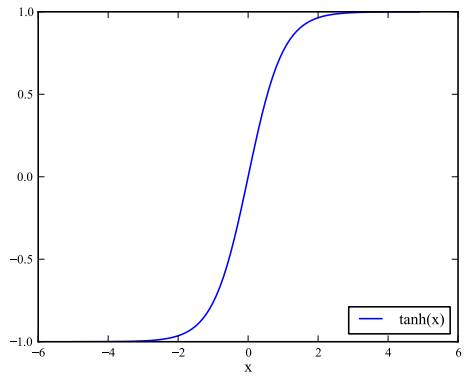
où $\vec{b}$ est un vecteur de biais.

Comme Rumelhart *et al.* (1995) le suggèrent pour la modélisation d'une distribution binomiale, la fonction d'activation du deuxième perceptron est $\mathrm{sigmoid}(\cdot)$. Cette fonction peut être interprétée comme une probabilité, car elle retourne des

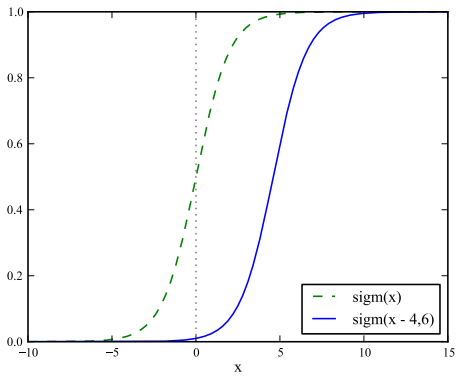
valeurs entre 0 et 1. De plus, comme nous le verrons plus loin, le calcul de sa dérivée, qui est nécessaire à l'optimisation par descente de gradient, est simple à calculer (équation 4.16).

Nous aurions pu utiliser la même fonction d'activation pour le premier perceptron, mais nous avons opté pour $\tanh$, qui est reconnue pour accélérer empiriquement la convergence de lors de l'entrainement (LeCun *et al.*, 1998).

Lorsque la phrase source est vide, toutes les valeurs de $\vec{\phi}$ sont à zéro, la sortie du premier perceptron est donc nulle et la fonction $\mathrm{sigmoid}(0)$ s'évalue à 0,5 pour chaque trait cible. Voulant que la traduction d'une phrase vide soit aussi une phrase vide, nous figeons toutes les valeurs du biais $\vec{b}$ à $\mathrm{sigm}^{-1}(0{,}01) \approxeq -4{,}6$[9]. De cette façon, tous les mots cibles sont prédits avec une probabilité de 0,01 lorsque la phrase source est vide. La figure 4.1 présente les différentes fonctions d'activation et l'effet produit par le biais.



Perceptron 1



Perceptron 2 avec et sans le biais.

Figure 4.1 Différentes fonctions d'activation.

[9] Nous ne pouvons prédire une probabilité de 0 car $\mathrm{sigmoid}^{-1}(0) \to -\infty$

1. Initialiser les valeurs de V et W à l'aide d'une distribution uniforme sur l'intervalle $[-0{,}05, 0{,}05]$.

2. Pour chaque itération:

   a. Pour chaque lot de 100 paires de phrases dans le corpus d'entrainement:

      i. Calculer les gradients de V et W à l'aide des équations 4.17 et 4.20.

      ii. Tester dans l'ordre les taux d'apprentissages 0,5, 0,05 et 0,01 et choisir le premier diminuant l'erreur (équation 4.22).

      iii. Si un des taux d'apprentissage a réussi à diminuer l'erreur, mettre à jour V et W dans la direction de leur gradient multiplié par la taille du pas d'apprentissage.

3. Retourner V et W.

Algorithme 4.1 Algorithme de descente de gradient que nous avons utilisé pour entrainer nos PPC.

4.6.3 Entrainement

Nous entrainons nos PPC en cherchant des valeurs pour V et W (équation 4.12) qui maximiseront la vraisemblance du bitexte d'entrainement (équation 4.4). Pour ce faire, utilisons l'algorithme 4.1, un algorithme de descente de gradient que nous présentons dans cette section.

L'algorithme initialise aléatoirement les valeurs de V et W. L'idée générale est ensuite de mettre à jour itérativement les valeurs de ces matrices de telle sorte que les erreurs de prédictions du PPC sur le bitexte d'entrainement diminuent. Pour

se faire, chaque mise à jour soustrait une fraction du gradient de l'erreur de ces matrices, car le gradient est la direction où l'erreur augmente le plus.

C'est comme si nous déposions une bille sur une surface irrégulière et que nous la laissions rouler jusqu'à ce quelle s'immobilise dans une crevasse. Dans cette analogie, les matrices V et W déterminent la position de la bille, l'erreur sur le bitexte d'entrainement (équation 4.13) la surface irrégulière et les gradients sont les pentes ou les directions dans lesquelles la bille roule. Tout comme notre bille peut s'arrêter dans n'importe quelle crevasse, notre descente de gradient retourne un optimum local qui n'est pas nécessairement le meilleur globalement.

Comme nous cherchons donc les valeurs de V et W qui maximisent la vraisemblance de notre bitexte d'entrainement (équation 4.4), nous définissons l'erreur par la log-vraisemblance négative du bitexte :

$$E(V, W, \mathcal{B}) = \sum_{(\vec{\phi},\vec{r})\in\mathcal{B}} \sum_{j=1}^{m} (1-\tau_j) \log\left(1-y_j\right) - \tau_j \log y_j \tag{4.13}$$

où $\mathcal{B}$ est notre bitexte d'entrainement. Notons que les valeurs de V et W maximisant l'équation 4.4 sont celles qui minimisent l'équation 4.13.

Nous initialisons les valeurs de V et W à l'aide d'une distribution aléatoire uniforme entre $-0{,}05$ et $0{,}05$ (ligne 1). Nous mettons ensuite ces matrices à jour sur plusieurs itérations de manière à réduire l'erreur sur le bitexte d'entrainement :

$$V^{(i+1)} = V^{(i)} - \gamma \frac{\partial E(V^{(i)}, W^{(i)}, \mathcal{B})}{\partial V^{(i)}} \tag{4.14}$$

$$W^{(i+1)} = W^{(i)} - \gamma \frac{\partial E(V^{(i)}, W^{(i)}, \mathcal{B})}{\partial W^{(i)}} \tag{4.15}$$

où $V^{(i)}$ et $W^{(i)}$ sont les valeurs des matrices après i mises à jour et γ est un taux d'apprentissage qui permet de contrôler l'agressivité des mises à jour (ou la vitesse de la bille). Les gradients de V et W sont données aux équations 4.17 et 4.20 :

$$\frac{\partial E}{\partial V\vec{h} + \vec{b}} = \vec{y} - \vec{\tau} \quad (4.16)$$

$$\frac{\partial E}{\partial V} = (\vec{y} - \vec{\tau})\vec{h}^{\intercal} \quad (4.17)$$

$$\frac{\partial \vec{h}}{\partial W} = \frac{\partial \tanh(W\vec{\phi})}{\partial W\vec{\phi}}\vec{\phi}^{\intercal} \quad (4.18)$$

$$\frac{\partial E}{\partial \vec{h}} = V^{\intercal}(\vec{y} - \vec{\tau}) \quad (4.19)$$

$$\frac{\partial E}{W} = \frac{\partial E}{\partial \vec{h}}\frac{\partial \vec{h}}{\partial W} \quad (4.20)$$

$$\frac{\partial x}{\partial \tanh(x)} = 1 - \tanh^2(x) \quad (4.21)$$

Comme ces formules sont déjà bien connues, nous référons les lecteurs voulant plus de détails à Bishop (1995).

Plutôt que de calculer le gradient sur tout le corpus d'entrainement, nous le calculons sur des lots de 100 paires de phrases (ligne 2.a). C'est un compromis qui est reconnu pour accélérer l'apprentissage.

Une fois la direction de la mise à jour connue, il faut choisir un taux d'apprentissage (ligne 2.a.ii), la valeur de γ dans l'équation 4.14 et l'équation 4.15. Pour ce faire, nous prenons le premier ratio parmi 0,5, 0,05 et 0,01 qui permet de diminuer la log-vraisemblance du bitexte d'entrainement. Si aucun de ces ratios n'y arrive, nous ne faisons aucune mise à jour.

Nous avons aussi expérimenté avec des taux d'apprentissage fixes et adaptés à chaque mise à jour par l'algorithme de Brent (Press *et al.*, 1992, §10.2). Peu importe la stratégie adoptée, la caractéristique qui a été déterminante dans nos expériences a été de nous assurer que chaque mise à jour diminue réellement l'erreur (ligne 2.a.iii).

Une autre caractéristique importante est l'ajout d'un terme de régularisation pénalisant les mises à jour trop agressives pour les réductions d'erreurs qu'elles engendrent (ligne 2.a.ii). Pour ce faire, nous utilisons une régularisation L1 :

$$E_{L1}(W, V, \mathcal{B}) = E(V, W, \mathcal{B}) + \nu \left(\sum_{ki} |w_{ki}| + \sum |v_{jk}| \right) \tag{4.22}$$

où ν est un ratio de régularisation que nous avons fixé à 0,1.

Le gradient de E_{L1} diminue tous les poids non-nuls de ν fois le taux d'apprentissage à chaque mise à jour. Comme plusieurs mots sources ne sont présents que dans quelques lots du bitexte d'entrainement, cela a l'effet indésirable de faire tendre leur poids vers zéro. Pour éviter cette érosion des poids associés aux mots rares, nous ne considérons la régularisation que lors du calcul du taux d'apprentissage (ligne 2.a.ii), mais nous l'ignorons lors de la mise à jour des matrices (ligne 2.a.iii).

Notre algorithme de descente de gradient (algorithme 4.1) débute donc par initialiser aléatoirement les valeurs des matrices V et W. Il les met ensuite à jour sur des lots de 100 paires de phrases en s'assurant que ces mises à jour augmentent la vraisemblance de ce lot. Une fois le nombre d'itérations maximal atteint, les matrices V et W sont prêtes à être utilisées pour faire des prédictions sur de nouveaux textes.

4.6.4 Exemple

Pour rendre notre démarche plus concrète, nous présentons un exemple comparant un perceptron (équation 4.7) à un PPC de deux unités cachées (équation 4.12). Nous avons optimisé ces deux modèles sur le corpus d'entrainement présenté à la figure 4.2.

La représentation du corpus d'entrainement sous forme vectorielle est présentée à la figure 4.3. Les parties sources et cibles sont encodées dans des matrices différentes où chaque colonne correspond à une phrase. Pour simplifier l'exemple, nous ignorons les mots outils.

Entrainement

Source	Cible
s_1 the floor$_1$ of the plant$_5$	t_1 le plancher$_5$ de l' usine$_8$
s_2 the stem$_6$ of the plant$_5$	t_2 la tige$_7$ de la plante$_6$
s_3 the leaves$_4$ and stem$_6$ of the flower$_2$	t_3 les feuilles$_1$ et la tige$_7$ de la fleur$_2$
s_4 the floor$_1$ and walls$_7$ of the house$_3$	t_4 le plancher$_5$ et les murs$_4$ de la maison$_3$

Test

Source	Cible
s_1 the leaves$_4$ of the plant$_5$	t_1 les feuilles$_1$ de la plante$_6$

Figure 4.2 Les corpus d'entrainement et de test. Les traits sont marqués par leur position dans la représentation vectorielle des phrases.

Source = Entrainement ∪ Test

	s_1	s_2	s_3	s_4	s_1
floor	1	0	0	1	0
flower	0	0	1	0	0
house	0	0	0	1	0
leaves	0	0	1	0	1
plant	1	1	0	0	1
stem	0	1	1	0	0
walls	0	0	0	1	0

Cible = Entrainement ∪ Test

	t_1	t_2	t_3	t_4	t_1
feuilles	0	0	1	0	1
fleur	0	0	1	0	0
maison	0	0	0	1	0
murs	0	0	0	1	0
plancher	1	0	0	1	0
plante	0	1	0	0	1
tige	0	1	1	0	0
usine	1	0	0	0	0

Figure 4.3 Les corpus d'entrainement et de test sous forme vectorielle. Chaque colonne correspond à une phrase de la figure 4.2.

4.6.4.1 Perceptron à l'œuvre

Le perceptron que nous avons obtenu est présenté à la figure 4.4. Nous voyons que W associe un poids nul entre les paires de mots absents des paires de phrases du bitexte d'entrainement. Ce perceptron réussit à prédire la présence de « feuilles » avec une probabilité de 0,4, mais il n'affiche aucune préférence pour « plante » ou « usine », car le mot « leaves » n'est jamais apparu dans la même paire de phrases que « usine » ou « plante ». Il ne reste donc que « plant » pour les prédire, ce qui est insuffisant et le perceptron assigne une probabilité de 0,17 aux deux mots.

Les quelques artéfacts comme les prédictions de « fleur » et de « tige » sont causés par des cooccurrences fortuites avec « leaves » et « plant » dans notre corpus d'entrainement.

4.6.4.2 Perceptron à plusieurs couches à l'œuvre

Le PPC optimisé sur le même corpus est présenté aux figures 4.5 et 4.7. Il est très difficile de prévoir le comportement de la couche cachée. Nous profitons donc de cet exemple pour expliquer comment elle se comporte dans ce cas précis. Nous ne pouvons pas généraliser ces observations, mais nous espérons tout de même clarifier nos motivations.

La première rangée de W (figure 4.5) calcule la valeur de la première unité cachée et ne distingue que le mot « plant » des autres en lui associant son seul poids négatif. C'est intéressant, car c'est le seul mot ayant deux traductions dans le corpus d'entrainement. La première unité cachée aura toujours une valeur proche de 1 sauf si la phrase source ne contient que le mot « plant ».

La deuxième rangée quant à elle associe des poids négatifs aux mots de bâtiment et des poids positifs aux mots de botanique. La deuxième unité cachée aura donc

$$\vec{y} = \text{sigmoid}(W \cdot \vec{\phi} + \vec{b})$$

$$= \text{sigmoid}\left(\overbrace{\begin{pmatrix} 0 & 4{,}2 & 0 & 4{,}2 & 0 & 0{,}6 & 0 \\ 0{,}6 & 4{,}2 & 0 & 4{,}2 & 0 & 0 & 0 \\ 0{,}6 & 0 & 4{,}2 & 0 & 0 & 0 & 4{,}2 \\ 0{,}6 & 0 & 4{,}2 & 0 & 0 & 0 & 4{,}2 \\ 7 & 0 & 2{,}6 & 0 & 0{,}8 & 0 & 2{,}6 \\ 0 & 0 & 0 & 0 & 3 & 3 & 0 \\ 0 & 2{,}6 & 0 & 2{,}6 & 0{,}8 & 7 & 0 \\ 3 & 0 & 0 & 0 & 3 & 0 & 0 \end{pmatrix}}^{W} \cdot \overbrace{\begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \\ 0 \end{pmatrix}}^{s_1} + \overbrace{\begin{pmatrix} -4{,}6 \\ -4{,}6 \\ -4{,}6 \\ -4{,}6 \\ -4{,}6 \\ -4{,}6 \\ -4{,}6 \end{pmatrix}}^{\vec{b}}\right)$$

$$= \begin{pmatrix} 0{,}40 \\ 0{,}40 \\ 0{,}01 \\ 0{,}01 \\ 0{,}02 \\ 0{,}17 \\ 0{,}23 \\ 0{,}17 \end{pmatrix} \begin{matrix} \longleftarrow \text{feuilles} \\ \longleftarrow \text{fleur} \\ \longleftarrow \text{maison} \\ \longleftarrow \text{murs} \\ \longleftarrow \text{plancher} \\ \longleftarrow \text{plante} \\ \longleftarrow \text{tige} \\ \longleftarrow \text{usine} \end{matrix}$$

Figure 4.4 Prédictions des mots cibles par un perceptron pour le corpus de test de la figure 4.3.

une valeur négative si la phrase traite de bâtiment et une valeur positive si elle traite de botanique. Cela nous mène à une représentation cachée de $(1, 0{,}54)$ pour « the leaves of the plant » (figure 4.5).

$$\vec{h} = \tanh(W \cdot \vec{\phi})$$

$$= \tanh\left(\overbrace{\begin{pmatrix} 3{,}6 & 4{,}5 & 3{,}6 & 4{,}4 & -0{,}1 & 4{,}3 & 3{,}6 \\ -3{,}1 & 3{,}5 & -0{,}9 & 3{,}5 & -2{,}9 & 3{,}1 & -1 \end{pmatrix}}^{W} \cdot \overbrace{\begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \\ 0 \end{pmatrix}}^{s_1} \right)$$

$$= \begin{pmatrix} 1 \\ 0{,}54 \end{pmatrix}$$

Figure 4.5 Prédictions de la couche cachée par un PPC pour le corpus de test de la figure 4.3.

Cette représentation cachée est ensuite passée au deuxième perceptron. La seconde colonne de ce perceptron (figure 4.7) associe les valeurs négatives de la deuxième unité cachée aux mots du bâtiment et les valeurs positives aux mots botaniques. Cela permet au PPC de préférer « plante » à « usine » pour traduire « the leaves of this plant ». C'est impossible pour le perceptron qui ne peut capturer la

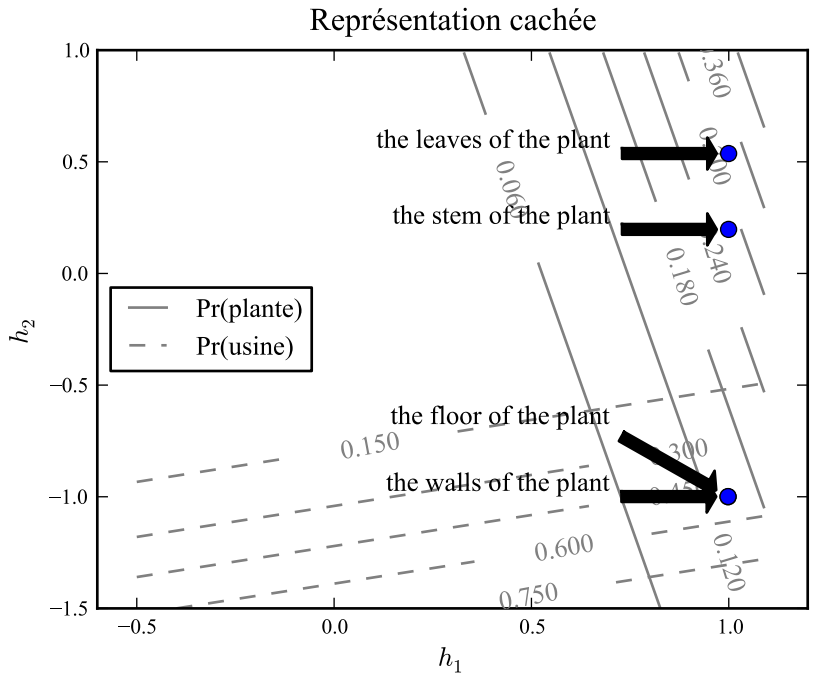


Figure 4.6 Les représentations cachées de quelques phrases. Les lignes de contour indiquent la probabilité que « plante » ou « usine » apparaisse dans la traduction.

relation entre « leaves » et « plante », car ces deux mots n'apparaissent pas dans une même paire de phrase du corpus d'entrainement.

La figure 4.6 présente la représentation cachée de trois autres phrases. Même si toutes ces phrases ne diffèrent que d'un mot, la couche cachée réussit à rapprocher les phrases qui traitent de botanique dans un groupe et celles qui traitent de bâtiment dans un autre. Les lignes de contours indiquent la prédiction du PPC pour les mots « plante » et « usine ».

Pour arriver au même résultat avec un perceptron, il faudrait ajouter d'autres traits sources comme le thème des phrases. C'est ce que nous évitons en utilisant un PPC, qui on l'espère, réussira à trouver automatiquement ces traits supplémentaires.

$$\vec{y} = \text{sigmoid}(V \cdot \vec{h} + \vec{b})$$

$$= \text{sigmoid}\left(\overbrace{\begin{pmatrix} 0{,}47 & 6{,}7 \\ 0{,}5 & 6{,}7 \\ -0{,}3 & -4{,}9 \\ -0{,}3 & -4{,}9 \\ 2 & -8{,}6 \\ 3{,}5 & 0{,}7 \\ 7 & 8{,}2 \\ 1 & -3{,}6 \end{pmatrix}}^{V} \cdot \overbrace{\begin{pmatrix} 1 \\ 0{,}54 \end{pmatrix}}^{\vec{h}} + \overbrace{\begin{pmatrix} -4{,}6 \\ -4{,}6 \\ -4{,}6 \\ -4{,}6 \\ -4{,}6 \\ -4{,}6 \\ -4{,}6 \\ -4{,}6 \end{pmatrix}}^{\vec{b}} \right)$$

$$= \begin{pmatrix} 0{,}37 \\ 0{,}38 \\ 0 \\ 0 \\ 0 \\ 0{,}33 \\ 1 \\ 0 \end{pmatrix} \begin{matrix} \longleftarrow \text{feuilles} \\ \longleftarrow \text{fleur} \\ \longleftarrow \text{maison} \\ \longleftarrow \text{murs} \\ \longleftarrow \text{plancher} \\ \longleftarrow \text{plante} \\ \longleftarrow \text{tige} \\ \longleftarrow \text{usine} \end{matrix}$$

Figure 4.7 Prédictions des mots cibles par un PPC pour les représentation cachées calculées à la figure 4.5.

4.6.5 Intégration d'un dictionnaire

Le PPC de l'équation 4.12 ne permet pas à un mot source de prédire un mot cible sans passer par la couche cachée. Il y a des cas où ce n'est pas désirable comme lorsqu'un mot n'a qu'une seule traduction. En présence d'un dictionnaire, il est inutile d'utiliser la représentation cachée pour le prédire. Cette section explique comment intégrer un dictionnaire bilingue à nos PPC.

Pour intégrer un dictionnaire bilingue à un PPC, nous modifions son architecture en y ajoutant des liaisons directes entre les mots sources et leurs traductions (U) :

$$\begin{aligned} &\text{perceptron 1} && \vec{h} = \tanh(W\vec{\phi}) \\ &\text{perceptron 2} && \vec{y} = \mathrm{sigmoid}(V\vec{h} + U\vec{\phi} + \vec{b}) \end{aligned} \quad (4.23)$$

où les poids de U sont différents de 0 seulement pour les paires de mots dans le dictionnaire.

Un dictionnaire bilingue ne contient que quelques traductions par mot. La matrice U sera donc creuse et le modèle tiendra toujours en mémoire. U peut être optimisée à l'aide de l'algorithme 4.1 en utilisant le gradient :

$$\frac{\partial E}{\partial U} = (\vec{y} - \vec{\tau})\vec{\phi}^{\intercal} \quad (4.24)$$

4.6.6 Implémentation

Nous remplaçons les vecteurs $\vec{\phi}$ et $\vec{\tau}$ dans équations 4.17, 4.20 et 4.24 par des matrices où chaque phrase tient sur une colonne. Cela nous permet de traiter les lots de 100 phrases en une seule multiplication matrice-matrice plutôt qu'en 100 multiplications matrice-vecteur. En sachant que les bibliothèques de programme d'algèbre linéaire sont beaucoup plus efficaces sur les premières multiplications

que sur les deuxièmes, nous accélérons considérablement notre application. En plus d'être efficaces, les bibliothèques de programme comme ATLAS (Whaley et Petitet, 2005) parallélisent automatiquement leurs calculs, offrant ainsi des performances de pointe sans trop d'efforts d'ingénierie.

Nos multiplications matricielles n'ont pas toutes les mêmes caractéristiques. Celles du premier perceptron (équation 4.12) et celles des équations 4.20 et 4.24 sont des multiplications entre une matrice dense et la matrice creuse $\vec{\phi}$. La complexité de ces produits n'est pas proportionnelle à la taille du vocabulaire source, mais au nombre de mots dans les phrases à traduire. Notre modèle passe donc très bien à l'échelle lorsque nous augmentons le nombre de traits sources.

L'utilisation des dictionnaires est aussi peu coûteuse en temps et en mémoire car la matrice qui les encode est creuse. La multiplication par U dans le deuxième perceptron (équation 4.23) se fait donc entre deux matrices creuses. Le calcul des traits cibles est l'opération la plus coûteuse, car il implique la multiplication de matrices denses.

Pour résumer, le cœur d'une implémentation efficace est l'utilisation d'une bonne bibliothèque d'algèbre linéaire qui supporte les calculs sur des matrices denses et creuses. Nous utilisons la librairie MKL[10] pour la multiplication de matrices denses et la librairie UBLAS[11] pour la multiplication de matrices creuses.

4.7 Défis de la prédiction de mots

La prédiction des mots cibles n'est pas un problème de classification standard comme la reconnaissance de caractères ou la classification de messages. Les principales différences proviennent du très grand nombre de traits sources et de traits

[10] http://software.intel.com/
[11] http://boost.org/

cibles combinés à leur faible fréquence. Nous expliquons dans cette section comment les modèles linéaires et les PPC gèrent ces contraintes.

4.7.1 Rareté des mots

La loi de Zipf stipule que si on ordonne les mots d'un corpus par fréquence, la position de chaque mot est inversement proportionnelle à sa fréquence. Comme nous le voyons dans la figure 4.8, cette loi est aussi observée dans nos corpus.

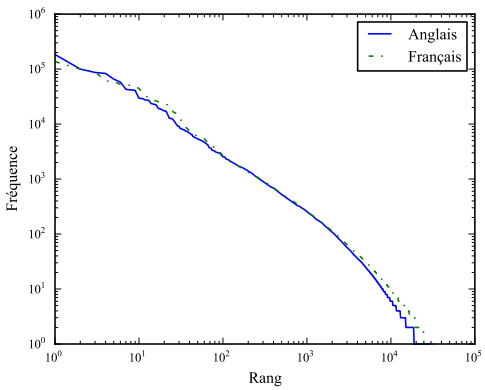


Figure 4.8 La loi de Zipf observée sur un bitexte de 100 000 paires de phrases.

La faible fréquence des mots sources complique la distinction entre les mots importants et ceux dont la présence est fortuite. Quant à celle des mots cibles, elle implique que la plupart des classificateurs apprennent à prédire les mots cibles à partir d'un très petit nombre d'exemples positifs et d'un très grand nombre d'exemples négatifs.

Comme la capacité d'apprentissage des perceptrons est limitée, ils sont adaptés au manque d'exemples pour les mots cibles peu représentés dans le corpus

d'entrainement. Les perceptrons ne peuvent modéliser que les problèmes de classification linéairement séparables, ce qui diminue les chances de surentrainement lorsqu'il y a rareté de données.

La réaction de nos PPC par rapport à la faible fréquence des mots ne nous est pas encore claire. Nous misons cependant sur la représentation cachée pour aider notre modèle. En effet, cette représentation est partagée par tous les mots du vocabulaire. Nous espérons donc que le contexte des mots peu fréquents aidera le PPC à projeter leurs phrases vers des représentations cachées similaires à celles des phrases sémantiquement proches.

4.7.2 Nombre de mots

La grande quantité de mots sources obligent nos classificateurs à bien fonctionner en haute dimension et la grande quantité de mots cibles nous oblige à entrainer plusieurs classificateurs. Pour donner une idée de ce que cela signifie, le corpus de 100 000 paires de phrases que nous avons utilisé pour les expériences du chapitre 5 contient plus de 40 000 mots français et 30 000 mots anglais.

En modélisant directement les cooccurrences dans le corpus, le nombre de paramètres possibles pour les perceptrons est dans $\mathcal{O}(l \times m)$ où l et m sont respectivement le nombre de traits sources et de traits cibles. Ces modèles comptent donc sur la faible fréquence des mots pour gérer leur quantité. Par exemple, un modèle IBM1 entrainé par GIZA++ (Och et Ney, 2003) sur notre corpus ne contient que 5 millions des 12 milliards de paramètres possibles.

De leur côté, les PPC compressent le vecteur de traits sources vers une représentation cachée en plus petite dimension. Cela leur permet donc d'avoir un nombre de paramètre dans l'ordre de $\Theta(h \times (l + m))$ où h est le nombre d'unités cachées.

La taille des matrices V et W grandit donc linéairement avec le nombre de mots. En utilisant une couche cachée de 512 unités et en ne considérant que les mots apparaissant cinq fois ou plus, le PPC entrainé sur notre corpus de 100 000 paires de phrases ne contient qu'un peu plus de 13,8 millions de paramètres. De plus, seulement deux multiplications de matrices sont nécessaires pour prédire tous les mots du vocabulaire (équation 4.12).

4.8 Résumé

En représentant les phrases par des sacs de mots, la traduction d'une phrase se résume à prédire si chaque mot connu devrait apparaître ou non dans sa traduction. Ces prédictions peuvent être faites en associant un classificateur binaire à chaque mot cible.

Nous avons proposé d'utiliser un perceptron à plusieurs couches pour produire ces prédictions. Contrairement à la régression linéaire et à la régression logistique, notre modèle tente de projeter le sac de mots à traduire dans une représentation alternative continue. Nous espérons que cette nouvelle représentation permettra de mieux représenter les phrases que les modèles de régression.

Cette représentation alternative en plus petite dimension peut sembler similaire à une analyse en composants principales. Elle s'en démarque cependant en étant entrainée pour minimiser l'erreur de reconstruction sur la phrase cible plutôt que sur la phrase source. L'utilisation d'un PPC permet aussi plus de flexibilité en intégrant par exemple des liens directs entre les mots d'un dictionnaire.

Nous profitons du prochain chapitre pour évaluer la qualité des prédictions de notre modèle et du chapitre 6 pour l'intégrer à un système de traduction statistique à base de segments.

5 Prédiction des mots cibles

Dans ce chapitre, nous nous assurons que nos PPC apprennent à bien prédire des traductions sur un corpus de taille raisonnable et nous évaluons leur qualité[12] .

5.1 Données

Nous avons évalué nos systèmes sur un sous-ensemble de la partie anglaise et française de la section des débats parlementaires des corpus rendus disponibles pour l'atelier sur la traduction statistique de 2008 (Callison-Burch *et al.*, 2008). Nous avons utilisé les 100 000 premières paires de phrases du corpus d'entrainement pour entrainer nos modèles (TRAIN), les 10 000 dernières pour les évaluer à différentes étapes de leur entrainement (TUNE) et les 2 000 paires de phrases de la section DEV2006 pour leur évaluation finale (TEST).

Nous avons segmenté les mots des bitextes à l'aide du script TOKENIZE fourni par les organisateurs de la tâche partagée[13] et nous avons converti tous les caractères en casse minuscule. Comme il y a peu d'intérêt à prédire les ponctuations et les mots outils, nous les avons supprimés des corpus[14] . Les statistiques de nos corpus ainsi traités sont présentées au tableau 5.1.

---

[12] Les résultats présentés sont différents de ceux que nous avons décrits dans Patry et Langlais (2009) car nous avons utilisé un algorithme d'apprentissage différent et nous ignorons les mots outils.

[13] http://statmt.org/wmt08/scripts.tgz

[14] Nous avons utilisé les listes de mots du projet *snowball* (http://snowball.tartarus.org/).

Corpus	Mots uniques		Mots		Phrases
	en	fr	en	fr	
TRAIN	30 032	40 247	1 229 715	1 568 083	100 000
TUNE	10 479	13 950	119 598	152 523	10 000
TEST	5 980	7 161	27 205	34 529	2 000

Tableau 5.1 Statistiques sur les corpus.

5.2 Protocole

Nous désirons évaluer l'impact du nombre d'itérations et du nombre d'unités cachées sur la qualité des prédictions de nos modèles. Nous avons donc entrainé des modèles avec 1, 2, 4, 8, 16, 32, 64, 128, 256 et 512 unités cachées sur 20 itérations.

Afin de diminuer la taille de nos PPC, nous les avons entrainés seulement sur les mots apparaissant cinq fois ou plus dans le corpus d'entrainement. Cela réduit le nombre de mots anglais uniques à 11 847 et le nombre de mots français uniques à 15 190.

Lors de nos expériences, nous comparons les différentes variations de nos modèles sur TUNE et évaluons les meilleurs sur TEST. Nous validons nos résultats en les comparant à ceux d'un modèle IBM1 (section 3.5) généré par GIZA++ (Och et Ney, 2003) en utilisant les paramètres par défaut. Comme IBM1 peut très bien s'accommoder d'un grand vocabulaire, nous n'avons pas enlevé les mots de basses fréquences pour son entrainement.

5.3 Expériences

Dans un premier jeu d'expériences, nous évaluons des PPC avec un nombre variable d'unités cachées que nous nommons PPC-γ où γ est le nombre d'unités cachées et nous les comparons à un modèle IBM1.

Nous validons d'abord notre algorithme d'entrainement à la section 5.3.1. Une fois l'algorithme d'entrainement validé, nous évaluons la qualité des prédictions de nos modèles de deux façons. Une première mesure la qualité des prédictions comme une liste ordonnée sans égard aux probabilités. Pour ce faire, nous reformulons la prédiction en un problème de recherche d'information où les phrases sources sont des requêtes et les mots du vocabulaire cible sont les documents à rechercher. Cela nous permet d'évaluer la qualité des traductions en tête de liste à l'aide de la précision à 10 (section 5.3.2) et la qualité des listes globales à l'aide de la précision moyenne (section 5.3.3). Dans un deuxième temps, nous évaluons la qualité des probabilités en ne retenant que les prédictions avec une probabilité plus grande qu'un seuil donné et nous calculons la précision et le rappel (section 5.3.4).

5.3.1 Progression de l'entrainement

Nos premières expériences vérifient si nos modèles réussissent à mieux prédire TRAIN au fil de leur entrainement. Pour ce faire, nous avons entrainés nos modèles sur 20 itérations et avons évalué la log-vraisemblance (lv) qu'ils accordent à TRAIN après chacune d'elle :

$$lv(\mathcal{B}) = \sum_{(\mathbf{s},\mathbf{t})\in\mathcal{B}} \overbrace{\sum_{t\in\mathbf{t}} \log\left(y_{t|\mathbf{s}}\right)}^{\text{Mots présents}} + \overbrace{\sum_{t'\notin\mathbf{t}} \log\left(1-y_{t'|\mathbf{s}}\right)}^{\text{Mots absents}} \tag{5.1}$$

où $\mathcal{B}$ est le bitexte sur lequel la vraisemblance est évaluée et $y_{t|\mathbf{s}}$ est la probabilité accordée par notre modèle au mot t lorsque la phrase source est $\mathbf{s}$.

La figure 5.1 présente la vraisemblance moyenne de chaque phrase de TRAIN pour quatre de nos modèles. Nous remarquons que les prédictions s'améliorent au

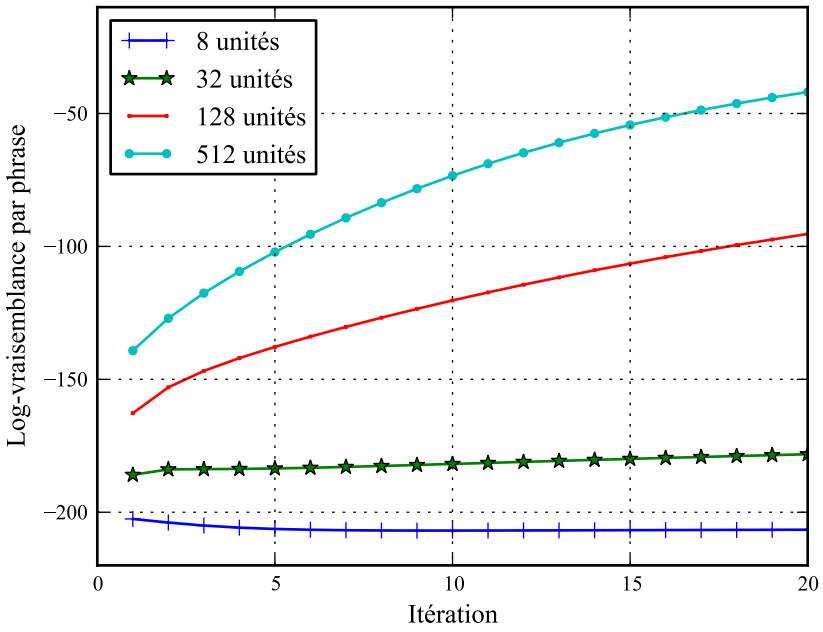


Figure 5.1 Vraisemblance moyenne des phrases de TRAIN au fil des itérations d'entrainement.

fil des itérations d'entrainement. Ce résultat est encourageant, car il montre que l'algorithme 4.1 réussit à bien optimiser nos modèles.

La log-vraisemblance s'améliore aussi avec l'ajout d'unités cachées. L'algorithme d'entrainement réussit donc à tirer profit de la puissance d'expression supplémentaire offerte par plus d'unités cachées. Seulement PPC-8 obtient des prédictions de moins bonne qualité au fil des itérations. Nous croyons que c'est dû à l'algorithme 4.1 qui n'optimise que sur 100 paires de phrases à la fois, produisant ainsi des optimums locaux de mauvaises qualités qu'il est difficile de quitter avec la marge de manœuvre plus limitée offerte par huit unités cachées.

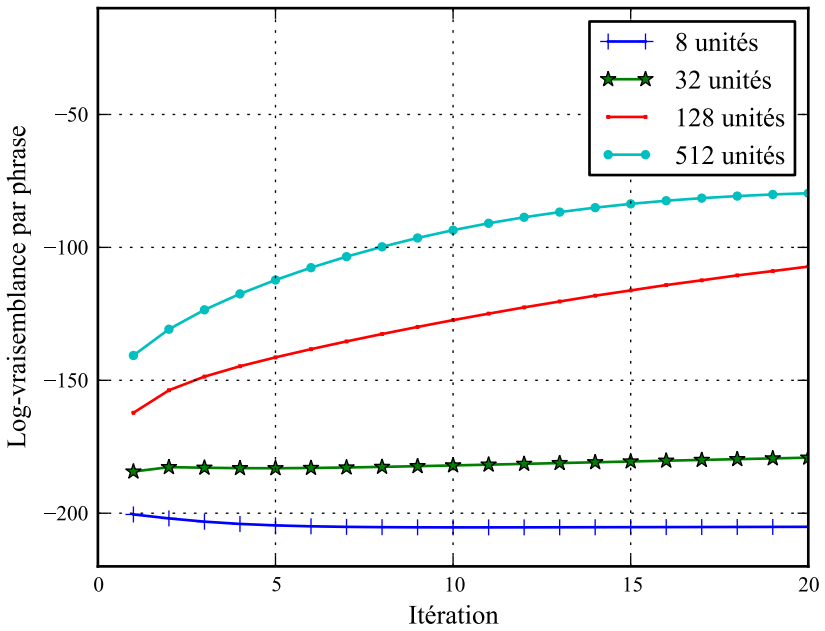


Figure 5.2 Vraisemblance moyenne des phrases de TUNE au fil des itérations d'entrainement.

Prédire TRAIN nous confirme que notre algorithme d'entrainement fonctionne. Cela ne nous garantit pas que les modèles réussiront à bien prédire les traductions de nouvelles phrases. Nous rapportons donc la log-vraisemblance moyenne sur TUNE à la figure 5.2. Nous remarquons que les prédictions sur TRAIN sont légèrement meilleures que celles de TUNE, mais nous obtenons tout de même des courbes très similaires dans les deux cas. Nos modèles semblent donc bien généraliser.

La vraisemblance des phrases est une métrique difficile à interpréter. Pour mieux saisir ce qui se passe, nous séparons les prédictions des mots présents dans la phrase (celles dont la probabilité devrait être 1) de celles des mots absents (celles dont la probabilité devrait être 0). Pour ce faire, nous calculons la probabilité

moyenne assignée par nos PPC à chacune de ces deux catégories de mots à l'aide de :

$$\mu^{+} = exp\left(\frac{1}{|\mathbf{t}|}\sum_{t\in\mathbf{t}} \log y_{t|\mathbf{s}}\right) \tag{5.2}$$

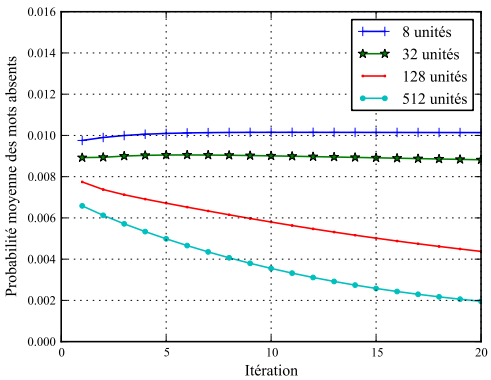
$$\mu^{-} = exp\left(\frac{1}{|\{t' \notin \mathbf{t}\}|}\sum_{t'\notin\mathbf{t}} \log\left(1 - y_{t'|\mathbf{s}}\right)\right) \tag{5.3}$$

où μ^{+} est la probabilité moyenne des mots présents et μ^{-} est la probabilité moyenne des mots absents pour une phrase donnée. La figure 5.3 présente les moyennes de ces probabilités sur les phrases de TRAIN et TUNE.

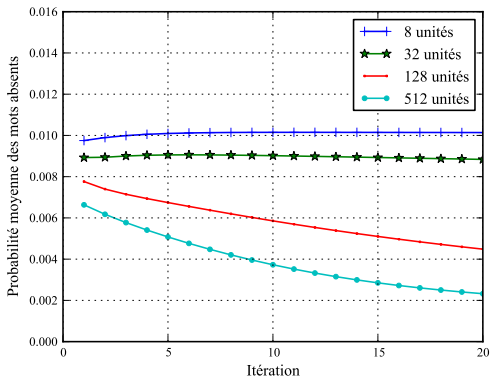
Nous remarquons dans les deux sous-figures de la première rangée de la figure 5.3 que la probabilité moyenne des mots absents est proche de zéro. C'est ce que nous espérions. En regardant la figure 5.4, nous voyons que plus le nombre d'unités cachées est grand, plus la probabilité moyenne des mots absents se rapproche de zéro. Comme le biais de nos PPC engendre une probabilité par défaut de 0,01 pour tous les mots, la probabilité de 0,002 accordée par nos modèles avec plusieurs unités cachées signifie que la couche cachée inhibe les mots absents.

La situation est différente pour la prédiction des mots présents dans la deuxième rangée de la figure 5.3. Les probabilités moyennes restent similaires entre les deux corpus pour les modèles avec peu d'unités cachées, mais elles divergent rapidement pour les modèles avec plus d'unités. Par exemple, à la vingtième itération, la probabilité moyenne des mots présents pour PPC-512 est de 42,5 % sur TRAIN alors qu'elle n'est que de 4,09 % sur TUNE.

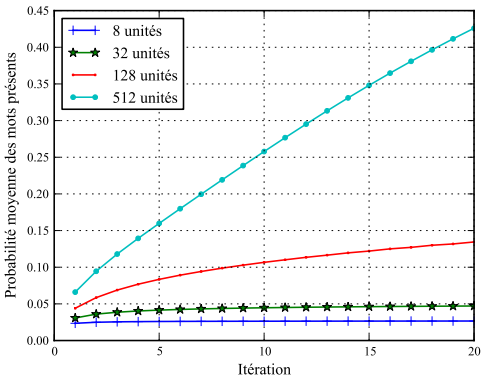
Il y a donc surentrainement, mais seulement sur les mots présents. Ce surentrainement est évident dans la probabilité des mots présents de la figure 5.4. Les différences des probabilités moyennes commencent à partir de 32 unités et grandissent



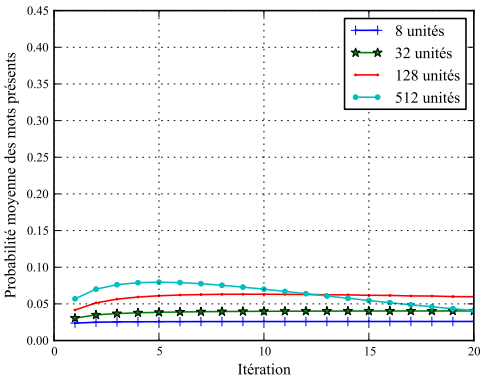
Mots absents (TRAIN).



Mots absents (TUNE).



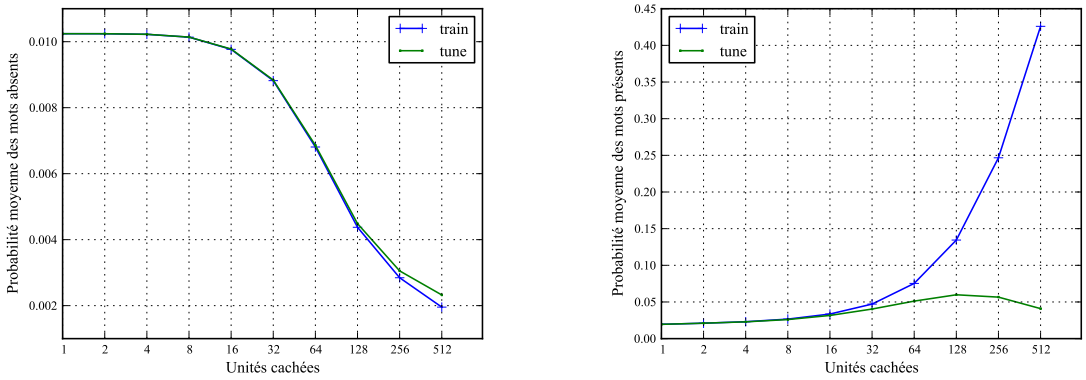
Mots présents (TRAIN).



Mots présents (TUNE).

Figure 5.3 Moyenne géométrique des probabilités des mots présents et des mots absents au fil des itérations.

exponentiellement jusqu'à 512 unités. Ce surentrainement n'est pas une si mauvaise nouvelle, car il signifie que nos modèles sont assez expressifs pour modéliser les données d'entrainement. Il nous oblige cependant à choisir avec soin le nombre d'unités cachées et d'itérations d'entrainement pour l'éviter.



Mots absents Mots présents

Figure 5.4 Probabilité moyenne des mots présents et absents par rapport au nombre d'unités cachées après 20 itérations d'entrainement.

5.3.2 Prédictions en tête de liste

Dans cette section, nous évaluons la qualité des premières prédictions de nos modèles. Pour ce faire, nous évaluons la précision à dix (P10) moyenne des phrases. La P10 d'un modèle sur une phrase donnée est le ratio des 10 traductions les plus probables se retrouvant dans la traduction de référence[15] .

La figure 5.5 présente les précisions à 10 de nos différents modèles et les compare à IBM1. Les courbes sont très similaires à celles des probabilités moyennes des mots présents de la figure 5.3, ce qui n'est pas surprenant car la P10 n'évalue que les mots présents. Même si nos modèles avec le plus d'unités cachées souffrent de surentrainement, ce sont eux qui obtiennent les meilleures P10. Seulement PPC-512 réussit à approcher IBM1 avec une P10 de 53,5 % vs 53,94 % pour IBM1. La sous-figure droite de la figure 5.5 présente la P10 de la meilleure itération pour

15 Si la traduction de référence contient moins de 10 mots, nous considérons autant de mots qu'il y en a dans la traduction de référence.

plusieurs quantités d'unités cachées. Encore une fois, nous remarquons que la P10 augmente avec le nombre d'unités cachées.

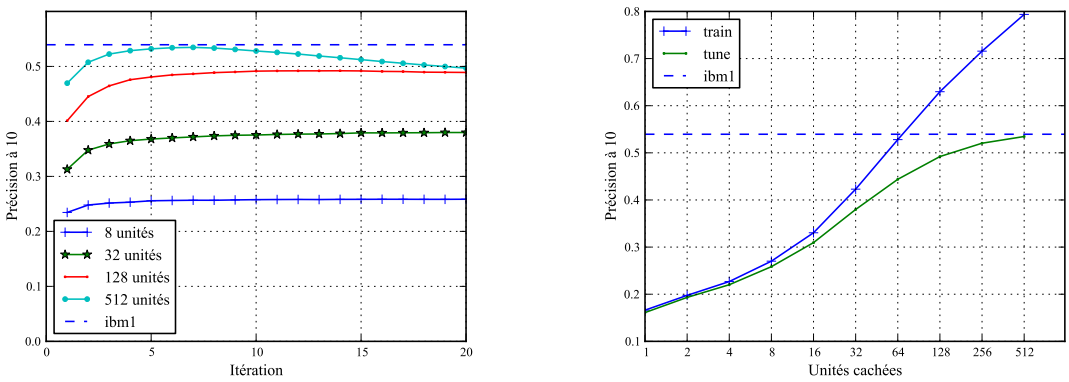


Figure 5.5 Précision à 10 moyenne par rapport au nombre d'itérations et au nombre d'unités cachées.

Les mots en tête de liste ont donc une chance sur deux d'être dans la traduction de référence. En considérant que nous ignorons les mots outils et que la traduction de référence ne contient qu'un sous-ensemble des mots pertinents à la phrase source, nous croyons que ces résultats sont intéressants. Des exemples de prédictions en tête de liste sont présentées au tableau 5.4.

5.3.3 Évaluation des listes complètes

La P10 nous informe sur la qualité des mots en tête de liste, mais elle ne nous dit rien sur la qualité globale des listes. Pour l'estimer, nous avons utilisé la précision moyenne (*Mean Average Precision* ou MAP). Nous trions les prédictions d'une phrase de la plus probable à la moins probable. Nous calculons ensuite le ratio de bonnes prédictions à chacun des points dans la liste où se trouve un mot de la phrase de référence. Nous répétons ce calcul pour chacune des phrases et calculons la moyenne.

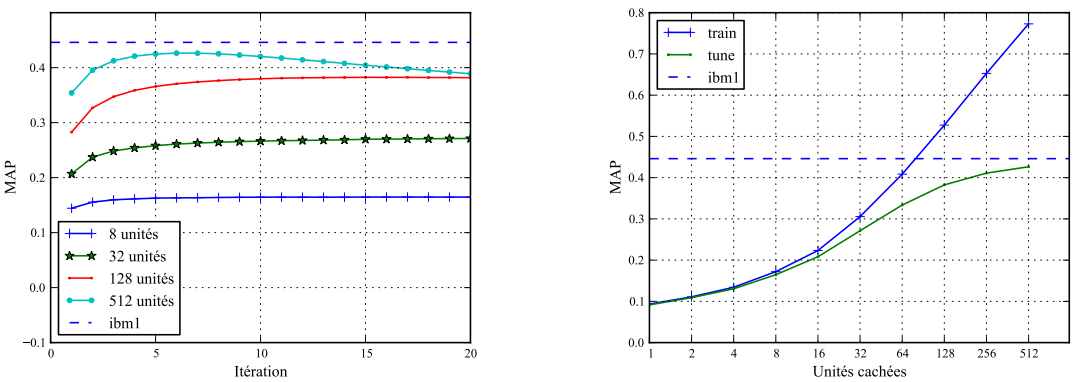


Figure 5.6 MAP sur TUNE par rapport au nombre d'itérations et d'unités cachées.

La figure 5.6 présente la MAP en fonction du nombre d'itérations et du nombre d'unités cachées. Nous remarquons que ces courbes sont très similaires à celles des mots présents de la figure 5.3. Encore une fois, notre meilleur PPC est PPC-512 avec une MAP de 42,7 % alors que celle de IBM1 est de 44,6 %. L'ordonnancement des mots du modèle IBM1 est de donc de meilleure qualité que ceux de nos modèles.

5.3.4 Sélection des mots

P10 et MAP évaluent l'ordre relatif des prédictions sans tenir compte de leur probabilité. Dans cette section, nous évaluons comment ces probabilités peuvent être utilisées pour prédire les mots qui devraient faire partie d'une traduction. Pour ce faire, nous choisissons un seuil pour chaque modèle et calculons la précision (ratios des mots prédits qui sont présents dans la référence), le rappel (ratio des mots présents dans la référence qui sont prédits) et la f-mesure (moyenne harmonique de la précision et du rappel) des prédictions retenues.

La première étape consiste donc à déterminer le seuil qui séparera les mots retenus des mots rejetés. Pour ce faire, nous choisissons le seuil donnant la meilleure

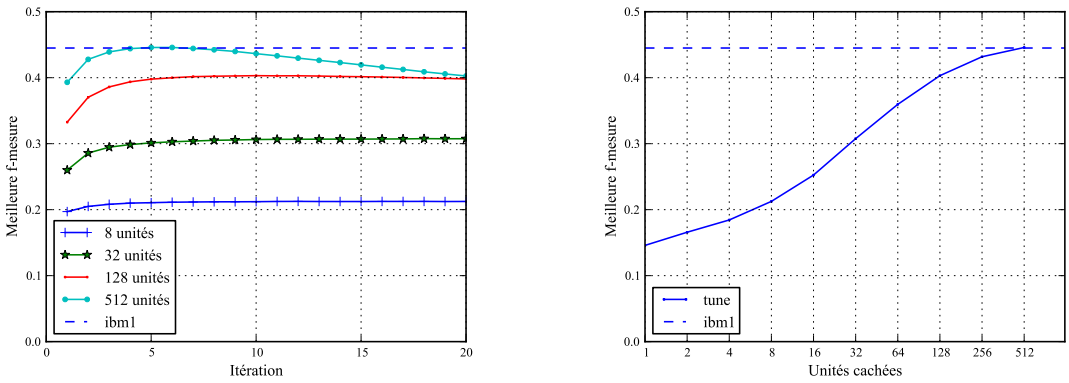


Figure 5.7 Variation de la f-mesure sur TUNE en fonction du nombre d'itérations et du nombre d'unités cachées.

f-mesure sur TUNE. La figure 5.7 présente la variation de la f-mesure en fonction du nombre d'itérations d'entrainement et du nombre d'unités cachées. Nous remarquons encore que la f-mesure augmente en fonction du nombre d'itérations et du nombre d'unités cachées. PPC-512 obtient une f-mesure de 44,6 % avec une précision de 51,1 % et un rappel de 39,6 %. IBM1 obtient une f-mesure similaire à 44,5 %, mais avec une précision de 45,1 % et un rappel 43,9 %.

La figure 5.8 présente la variation de la f-mesure en fonction du seuil. La sélection du seuil est beaucoup plus critique pour IBM1 que pour nos PPC. En effet, les aires sous les courbes de nos PPC sont plus grandes, ce qui montre que nos modèles réussissent plus facilement à prédire de grandes probabilités pour les mots présents et de petites probabilités pour les mots absents. Nous revenons sur la sensibilité d'IBM1 au choix du seuil à la section 5.7.

Le tableau 5.2 présente les résultats obtenus sur le corpus de test en choisissant les modèles pour lesquels la probabilité moyenne des mots présents est la plus élevée. Tout comme pour les expériences précédentes, les modèles avec le plus

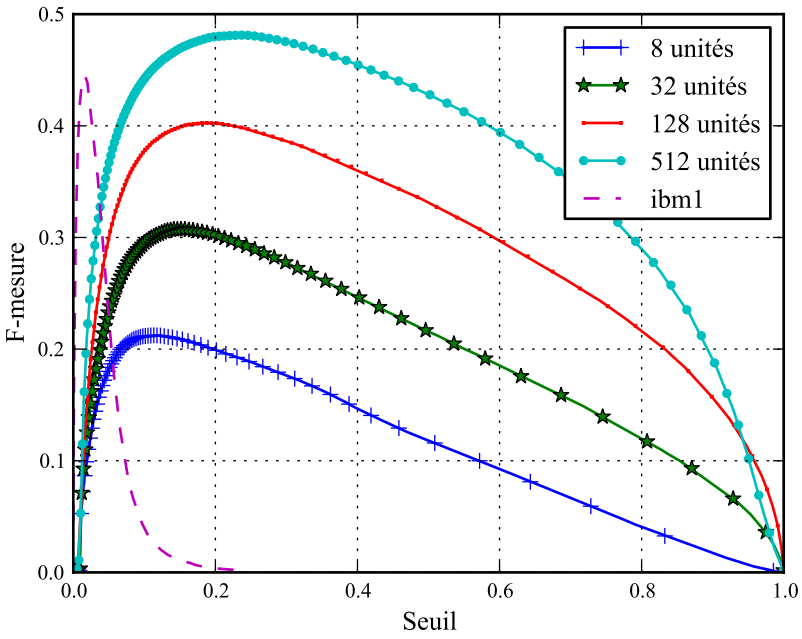


Figure 5.8 Variation de la f-mesure en fonction du seuil.

grand nombre d'unités cachées sont ceux qui s'en sortent le mieux. Bien que PPC-512 et IBM1 aient des f-mesures similaires, nos PPC favorisent la précision sur le rappel alors que IBM1 n'en favorise aucun.

Les résultats de PPC-512 sont encourageants, car ils montrent que la représentation alternative offerte par la couche cachée peut être compétitive avec IBM1, un bon système de référence. La prochaine section explore comment nous pouvons améliorer nos PPC en leur ajoutant des connexions directes tirées du modèle IBM1.

5.4 Ajout d'un dictionnaire

Dans cette section, nous ajoutons des connexions directes entre les mots en relation de traduction à nos PPC. Nous construisons a cet effet un dictionnaire bilingue à partir des dix meilleures traductions identifiées par IBM1 pour chaque

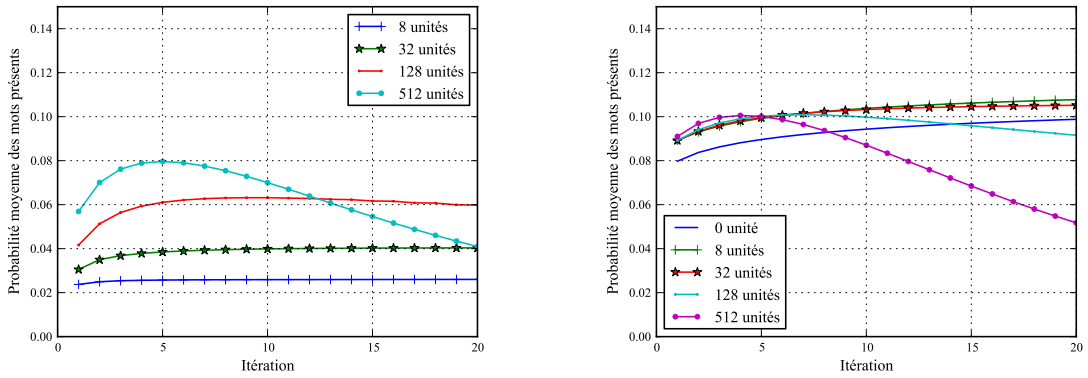
Modèle	Itération	Seuil	Précision	Rappel	P10	MAP
IBM1	–	0,015	48,3	47,1	57,6	46,4
PPC-1	2	0,075	15,9	12,2	18,6	9,97
PPC-2	5	0,079	18,8	13,7	21,8	11,8
PPC-4	8	0,093	23,5	15,0	25,1	14,2
PPC-8	20	0,115	30,0	17,4	29,2	17,7
PPC-16	20	0,131	34,0	21,5	33,9	21,9
PPC-32	20	0,144	40,6	27,0	41,1	28,0
PPC-64	14	0,174	47,8	31,7	47,2	34,0
PPC-128	9	0,182	50,4	36,5	52,4	39,0
PPC-256	7	0,201	52,3	39,2	55,1	42,0
PPC-512	5	0,204	53,1	40,6	56,2	43,6

Tableau 5.2 Résultats finaux sur TEST. Les seuils utilisés pour calculer la précision et le rappel sont ceux optimisant la f-mesure sur TUNE.

mot source. Nous nommons ces modèles PPC-γD où γ est le nombre d'unités cachées.

Une fois les connexions directes ajoutées, il faut initialiser leur poids. Nous avons mentionné à la section 4.6.2 que les unités de sortie sont associées à un biais de $-4{,}6$ pour nous assurer qu'aucun mot ne soit prédit pour une phrase vide. Ce qui signifie que si nous utilisons les probabilités de IBM1, une probabilité de 1 ne donnera qu'une sortie de $\texttt{sigmoid}(-4{,}6 + 1) \approxeq 0{,}027$. Pour mieux refléter les probabilités de IBM1, nous les avons plutôt multipliées par 9,2 afin que la sortie de `sigmoid` soit de 0,99 lorsque la probabilité de IBM1 est de 1.

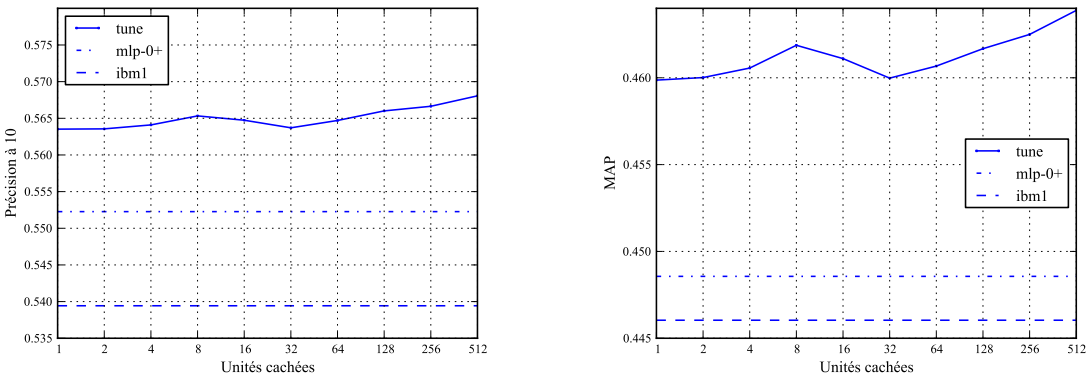
Nous avons observé les mêmes comportements lors de l'entrainement de nos modèles avec et sans dictionnaire. La plus grande différence se situe au niveau de la probabilité moyenne des mots présents, qui est élevée pour chacun des modèles avec dictionnaire, comme nous le voyons à la figure 5.9. Même notre modèle de



Sans dictionnaire Avec dictionnaire

Figure 5.9 Moyenne géométrique des probabilités des mots présents au fil des itérations sans dictionnaire (gauche) et avec dictionnaire (droite).

contrôle n'ayant aucune unité cachée s'en tire mieux que PPC-512, notre meilleur PPC sans connexion directe.



P10 MAP

Figure 5.10 P10 et MAP des modèles avec dictionnaire sur le corpus de développement.

La P10 et la MAP de nos modèles avec dictionnaire sont présentées à la figure 5.10. Nous remarquons d'abord que le modèle sans unité cachée surpasse à lui seul

IBM1. Ce qui semble indiquer, comme l'ont aussi relevé Mauser *et al.* (2009), qu'un modèle de régression logistique est préférable a un modèle IBM1. De nouveaux gains sont encore observés en ajoutant aussi peu qu'une seule unité cachée. C'est encore le PPC avec 512 unités cachées qui obtient les meilleures performances, mais suivi de très près par le modèle ne contenant que seulement huit unités cachées. Contrairement aux PPC sans dictionnaire, les résultats des PPC incluant un dictionnaire ne varient que très peu en fonction du nombre d'unités cachées.

Modèle	Itération	Seuil	Précision	Rappel	P10	MAP
IBM1	–	0,015	48,3	47,1	57,6	46,4
PPC-512	5	0,204	53,1	40,6	56,2	43,6
PPC-0D	20	0,223	57,2	42,0	59,3	46,3
PPC-8D	20	0,242	57,8	42,4	59,9	47,7
PPC-512D	4	0,230	57,4	42,8	59,9	47,8

Tableau 5.3 Résultats finaux sur TEST. Pour faciliter la lecture, la précision, le rappel, la P10 et la MAP sont rapportés en pourcentages.

Nous avons finalement sélectionné les modèles avec la meilleure f-mesure et les avons évalués sur le corpus de test. Les résultats sont présentés au tableau 5.3. Nous remarquons que les PPC favorisent la précision sur le rappel. La couche cachée de PPC-8D et de PPC-512D améliore PPC-0D sur toutes les métriques que nous avons calculées. L'amélioration la plus flagrante est sur la MAP où nous observons un gain de 1,5 %. Cela signifie que l'impact de la couche cachée ne se manifeste pas sur la qualité des prédictions en tête de liste, mais plutôt sur les autres.

5.5 Exemples de prédictions

Analyser individuellement les prédictions de nos modèles est une tâche plutôt complexe. Le tableau 5.4 présente néanmoins un exemple de prédictions que nous

croyons représentatif. Nous profitons de cette section pour décrire quelques tendances que nous avons observées en navigant parmi les résultats.

Même si PPC-64 ne contient que peu d'unités cachées, ses prédictions en tête de liste sont intéressantes. Il fait cependant plusieurs prédictions erronées comme « décharge », « déclare » ou « rejette ». De plus, les probabilités chutent rapidement à mesure que nous avançons dans sa liste de prédictions. Ajouter des unités cachées augmente le nombre de prédictions valides en tête de liste, ce que nous pouvons aussi constater dans notre exemple avec le modèle PPC-512.

Si nous regardons maintenant du côté de IBM1 et PPC-0D qui ne font pas usage de couche cachée, nous remarquons que les prédictions s'améliorent. Ces deux modèles réussissent à prédire la plupart des noms communs, mais prédisent aussi plusieurs de leurs variations comme « second » et « deuxième » pour « seconde ». Les bonnes prédictions de PPC-0D arrivent aussi plus tôt que pour IBM1. Cela est en parfait accord avec la différence en P10 observée précédemment.

Les résultats de PPC-8D sont similaires à ceux de PPC-0D, à la différence que la couche cachée a réussi à inhiber « monsieur » et « président » qui sont très fortement associés au mot anglais « president » dans le dictionnaire IBM1. Cette tendance à inhiber les mauvais mots de contenu nous a semblé assez forte lorsque nous avons comparé ces deux modèles sur plus d'exemples. Cela est en accord avec la plus grande précision moyenne que nous avons observée pour PPC-8D par rapport aux autres modèles.

5.6 Inspection de la couche cachée

Une de nos motivations pour utiliser des PPC était la représentation alternative offerte par leur couche cachée. Pour vérifier si cette représentation répondait à

Source	Madam President, there is a second reason for the European Parliament's perseverance.
Cible	Il y a une seconde raison, Madame la Présidente, à la persévérance du Parlement européen.

Prédictions									
PPC-64		PPC-512		IBM1		PPC-0D		PPC-8D	
%	Mots	%	Mots	%	Mots	%	Mots	%	Mots
99	**présidente**	99	**présidente**	9	**parlement**	97	**madame**	98	**madame**
98	**parlement**	99	**madame**	7	l'	93	**présidente**	98	**présidente**
98	**madame**	96	**parlement**	6	deuxième	89	**parlement**	91	**européen**
73	**européen**	94	**européen**	6	président	60	**européen**	90	**parlement**
15	deuxième	72	deuxième	5	**madame**	60	deuxième	61	deuxième
7	européens	34	**raison**	5	**présidente**	52	président	36	**raison**
7	c'	33	c'	4	européenne	38	**raison**	18	**persévérance**
6	décharge	12	pourquoi	4	les	23	européenne	13	**seconde**
5	cette	9	**seconde**	4	**raison**	19	monsieur	10	second
4	déclare	9	laquelle	3	**européen**	18	**persévérance**	9	européens
3	rejette	8	cette	3	monsieur	13	**seconde**	8	européennes
3	interrompue	8	second	3	**persévérance**	10	second	8	c'
3	continent	5	l'	2	d'	9	c'	8	pourquoi
3	motion	4	d'	1	c'	9	pourquoi	6	les
3	strasbourg	3	cela	1	**a**	9	laquelle	6	raisons

Tableau 5.4 Exemple de prédictions (Mots) et de probabilités (%) produites par nos modèles où les prédictions en gras apparaissant dans la traduction.

nos attentes, nous avons observé les phrases qui ont des représentations cachées similaires, comme nous l'avons fait à la section 4.6.4. Nous avons utilisé la distance L_1 pour déterminer la similarité entre deux phrases :

$$||\vec{h} - \vec{h}'||_1 = \sum_k |\vec{h}_k - \vec{h}'_k| \tag{5.4}$$

où $\vec{h}$ et $\vec{h}'$ sont les représentations cachées des deux phrases pour lesquelles il faut calculer la distance. Les représentations cachées des phrases de cette section ont été calculées avec PPC-64.

Source	We therefore hope that the Commission will introduce the measures that it feels must be adopted in this area.
Cible	Nous attendons donc de la Commission qu'elle nous présente les mesures qu'elle estime nécessaire de prendre en la matière.
Voisines	1. I therefore call upon the Commission to take action in this area. 2. Is the Commission prepared to take steps to do this? 3. In my opinion, this is really not good enough, and we expect tangible measures from the Commission. 4. What action did the Commission take? 5. Therefore, i think that the Commission should act.

Figure 5.11 Phrases exprimant une demande d'action de la part de la commission.

Le figure 5.11 présente un premier exemple de phrases ayant des représentations cachées similaires. Tout comme la phrase source, toutes les phrases voisines expriment un désir de voir la commission agir. Cette idée est cependant exprimée de manières assez diversifiées. La phrase source mentionne « measures [...] adopted in this area », alors que les voisines utilisent plutôt « take action in this area », « take step to do this » ou « should act ».

Dans l'exemple de la figure 5.12, la phrase source parle d'évoquer un aspect en utilisant « pick out one aspect ». Les quatre premières voisines traitent aussi de

Source	I would like to pick out one aspect alone.
Cible	Je voudrais encore évoquer un seul aspect.
Voisines	1. But there is one fundamental point i would like to make. 2. I should like to address one more point. 3. I am only able to mention one interesting aspect. 4. I would like to add one more point. 5. Allow me to highlight a fourth one, namely e for empathy.

Figure 5.12 Phrases traitant d'un aspect à évoquer.

ce sujet, mais un utilisant un vocabulaire plus varié. La formule « pick out » est exprimée par « make », « address », « mention » et « add » et le mot « aspect » est quelques fois remplacé par « point ». Finalement, la phrase source et les voisines de la figure 5.13 relèvent toutes un manque d'informations.

Source	This would suggest that we really do need information campaigns.
Cible	Cela prouve que nous avons vraiment besoin de campagnes d'information.
Voisines	1. We need to have more information about the full extent of the tragedy. 2. Could you provide us with information on that? 3. It has no more information than you do, on which investigations are at what stage. 4. I have no information on this subject. 5. Would it like more information about this study?

Figure 5.13 Phrases traitant d'un manque d'informations.

Nous avons aussi observé des voisines intéressantes sur les modèles avec dictionnaire. Les voisins sont cependant de moins bonne qualité dans les modèles avec plusieurs unités cachées. Cela pourrait s'expliquer par la distance L_1 qui n'est peut-être pas appropriée en trop haute dimension ou par le surentrainement dont ils souffrent.

Même si nos PPC ne surpassent pas le modèle IBM1 pour la prédiction de mots, ils réussissent tout de même à proposer des représentations alternatives intéressantes.

5.7 Longueur des phrases

Faire du sens d'un petit ensemble de mots désordonnés n'est pas très compliqué, mais faire du sens d'un grand ensemble de mots désordonnés peut s'avérer complexe. Le tableau 5.5 évalue comment nos modèles réagissent face aux phrases de longueurs différentes.

PPC-8D est le meilleur modèle selon toutes les métriques à l'exception du rappel. En comparant PPC-0D à PPC-8D, nous voyons que la couche cachée semble particulièrement être utile pour les phrases de moins de dix mots. Les performances de PPC-0D, PPC-8D et IBM1 deviennent similaires sur les phrases de 30 mots ou plus. Finalement, les métriques ont tendance à augmenter avec la longueur des phrases pour tous les modèles sauf PPC-512.

Les seuils optimaux varient aussi avec la longueur des phrases. En regardant la figure 5.8, nous remarquons que les PPC sont robustes à ces petites variations. Ce n'est cependant pas le cas pour IBM1 où ces variations diminuent énormément la f-mesure. Par exemple, si on utilise le seuil optimal des phrases de moins de 10 mots pour faire des prédictions sur les phrases de 30 mots ou plus, la f-mesure passe de 51,2 % à 18,6 %. Si on fait la même chose sur PPC-8D, la f-mesure reste à 51,2 %.

5.8 Résumé

Nous sommes en mesure d'entrainer nos PPC pour qu'ils réussissent à bien prédire le corpus d'entrainement ainsi que le corpus de test. L'usage d'un dictionnaire

$[1, 10)$ mots sources						
	MAP	P10	Seuil	Précision	Rappel	F-mesure
IBM1	46,2	45,6	0,033	45,9	43,8	47,1
PPC-512	45,7	46,4	0,193	53,2	41,3	46,5
PPC-0D	46,4	47,7	0,162	52,7	43,2	47,5
PPC-8D	47,9	48,6	0,215	56,5	42,2	48,3
$[10, 20)$ mots sources						
	MAP	P10	Seuil	Précision	Rappel	F-mesure
IBM1	46,0	60,5	0,015	51,3	45,8	48,4
PPC-512	42,7	58,3	0,211	52,8	40,0	45,5
PPC-0D	45,8	61,9	0,223	56,3	42,1	48,2
PPC-8D	47,2	62,2	0,237	56,6	42,9	48,8
$[20, \infty)$ mots sources						
	MAP	P10	Seuil	Précision	Rappel	F-mesure
IBM1	48,0	74,8	0,010	56,9	46,5	51,2
PPC-512	41,9	71,1	0,202	52,8	41,2	46,3
PPC-0D	47,4	76,3	0,253	57,6	45,8	51,0
PPC-8D	48,8	77,4	0,245	57,8	45,9	51,2

Tableau 5.5 Évaluation sur des phrases de longueurs différentes.

bilingue est cependant essentiel à l'obtention de résultats surpassant IBM1. Bien que l'entrainement de nos modèles soit plus long que celui nécessaire pour IBM1, il reste tout de même raisonnable comme le montre la figure 5.14.

Même si les modèles sans dictionnaire ne réussissent pas à surpasser IBM1 pour la prédiction de mots cibles, ils offrent tout de même une représentation alternative intéressante des phrases. Notre analyse qualitative nous indique qu'elle réussit à

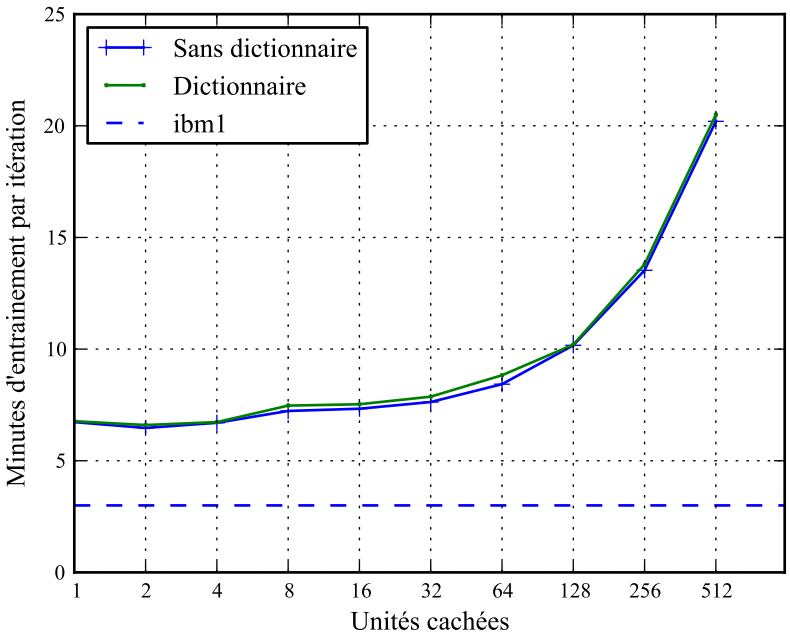


Figure 5.14 Temps pour une itération d'entrainement sur TRAIN.

rapprocher certaines phrases qui ont des sens similaires même si elles n'utilisent pas les mêmes mots.

Nos modèles sont particulièrement performants sur les petites phrases. Ils sont aussi robustes aux phrases de tailles variables, ce qui n'est pas le cas de IBM1 pour lequel les probabilités des mots sont fortement influencées par la longueur des phrases.

Maintenant que nous savons que nos modèles réussissent à prédire les mots de la phrase cible, nous profitons du prochain chapitre pour vérifier s'ils peuvent améliorer un système de traduction à base de segments.

6 Traduction

La prédiction des mots cibles n'est pas une fin en soi, mais plutôt un moyen de considérer le contexte en traduction. Dans ce chapitre, nous intégrons nos modèles à un système de traduction statistique à base de segments et évaluons la qualité des traductions résultantes.

Nous présentons notre système de traduction à la section 6.1 et décrivons deux scores permettant de lui intégrer nos modèles de prédiction à la section 6.2. Nous évaluons ensuite nos systèmes à la section 6.3 et comparons leurs résultats à ceux de la littérature à la section 6.4.

6.1 Système de traduction

Nous avons développé le système de traduction DUC (Decoder Using Context), le successeur de RAMSES (Patry, 2006 et Patry *et al.*, 2006a), notre décodeur pour la traduction statistique à base de segments. Lorsque nous avons comparé RAMSES à PHARAOH (Koehn, 2004b), le prédécesseur de MOSES, nous avons observé que les deux systèmes produisaient des traductions de même qualité lorsqu'ils étaient utilisés dans des conditions identiques (Patry *et al.*, 2006b).

DUC utilise un modèle log-linéaire (équation 2.14) pour évaluer la qualité des entrées de la table de traduction à l'aide des six scores décrits à la section 2.4.1 : deux probabilités calculées par fréquences relatives, deux probabilités lexicales, une pénalité sur le nombre de mots cibles et une pénalité sur le nombre de bisegments utilisés pour la traduction. La fluidité des traductions est évaluée dans cette étude par un modèle trigramme entrainé sur la partie cible du corpus d'entrainement à l'aide de la suite d'outils SRILM (Stolcke, 2002). Finalement, nous utilisons le même

modèle de réordonnancement que PHARAOH. Ce modèle pénalise le nombre de mots séparant chaque paire de segments sources traduits consécutivement.

Nous créons nos tables de bisegments à partir d'un corpus d'entrainement en utilisant le script `train-factored-phrase-model.perl` de MOSES (Koehn *et al.*, 2007). Ce script extrait des bisegments contenant un maximum de sept mots à partir d'un alignement IBM4 raffiné à l'aide de l'euristique *grow-diag-final* (voir la section 2.2.4 pour plus de détails).

Les poids de notre décodeur ont été ajustés pour optimiser BLEU sur des listes de 2 000 traductions candidates à l'aide de la méthode de Nelder-Mead (Press *et al.*, 1992,§10.4) telle que nous l'avons décrite dans Patry *et al.* (2007).

Afin de garder un temps de traitement raisonnable, nous n'avons considéré que les 30 meilleures traductions pour chaque segment source et nous avons limité le nombre de mots séparant des segments sources traduits consécutivement à quatre. Lors de l'exploration des traductions candidates, seulement 50 hypothèses ont été conservées par pile.

6.2 Intégration des prédictions au système de traduction

Nous avons intégré nos modèles a DUC en ajoutant deux nouveaux scores au mélange log-linéaire : un premier calculant la cote des mots présents (section 6.2.1) et un deuxième comptant le nombre de mots prédits qui apparaissent dans la traduction (section 6.2.2).

6.2.1 Cote des mots présents

Comme nos PPC optimisent la log-vraisemblance du bitexte d'entrainement, il serait naturel d'utiliser ce score dans notre décodeur :

$$lv(\mathbf{t} \mid \mathbf{s}) = \overbrace{\sum_{t \in \mathbf{t}} \log\left(y_{t|\mathbf{s}}\right)}^{\text{Mots présents}} + \overbrace{\sum_{t' \notin \mathbf{t}} \log\left(1 - y_{t'|\mathbf{s}}\right)}^{\text{Mots absents}} \tag{6.1}$$

où $y_{t|\mathbf{s}}$ est la probabilité pour le mot t estimée par nos modèles lorsque la phrase source est $\mathbf{s}$. Nous ignorons les mots n'appartenant pas au vocabulaire source ou au vocabulaire cible.

Ce score demande cependant de sommer sur tous les mots du vocabulaire cible. Nous utilisons plutôt le logarithme du produit des cotes (ou *odds* en anglais) des mots présents :

$$\log cote(\mathbf{t} \mid \mathbf{s}) = \log \prod_{t \in \mathbf{t}} \frac{y_{t|\mathbf{s}}}{1 - y_{t|\mathbf{s}}} \tag{6.2}$$

$$= \sum_{t \in \mathbf{t}} \log\left(y_{t|\mathbf{s}}\right) - \log\left(1 - y_{t|\mathbf{s}}\right) \tag{6.3}$$

La cote d'un mot est la probabilité que ce mot apparaissent dans la traduction divisée par la probabilité qu'il n'y apparaisse pas. Par exemple, une cote de trois, signifie que le mot cible à trois fois plus de chances d'apparaitre dans la traduction que de ne pas y figurer. À partir de maintenant, nous utilisons le terme *cote* pour référer à l'équation 6.3.

Une propriété intéressante de la cote est qu'elle est proportionnelle à la log-vraisemblance (équation 6.1), sans avoir à considérer les mots absents :

$$\begin{aligned} lv(\mathbf{t} \mid \mathbf{s}) &\propto \overbrace{\sum_{t \in \mathbf{t}} \log\left(y_{t|\mathbf{s}}\right)}^{\text{Mots présents}} + \overbrace{\sum_{t' \notin \mathbf{t}} \log\left(1 - y_{t'|\mathbf{s}}\right)}^{\text{Mots absents}} - \overbrace{\sum_{t} \log\left(1 - y_{t|\mathbf{s}}\right)}^{\text{Tous les mots (constante)}} \\ &\propto \overbrace{\sum_{t \in \mathbf{t}} \log\left(y_{t|\mathbf{s}}\right) - \log\left(1 - y_{t|\mathbf{s}}\right)}^{\text{Mots présents}} \end{aligned} \tag{6.4}$$

Tout comme Mauser *et al.* (2009), nous mémorisons la cote des bisegments en antémémoire avant de lancer la traduction de chaque phrase. Cela permet une intégration très efficace des modèles de prédiction, mais fausse les données lorsque le même mot apparait dans plus d'un segment.

La cote considère chaque mot comme présent ou absent. Les mots répétés ne devraient donc être comptés qu'une seule fois. Cependant, en calculant le score de chaque bisegment indépendamment des autres, un mot apparaissant dans plus d'un segment cible est compté plus d'une fois par notre système. Comme nous ne prédisons que les mots de contenu, ces répétitions restent marginales.

6.2.2 Décompte des mots prédits

Le deuxième score que nous avons intégré à notre système de traduction compte le nombre de mots prédits avec une probabilité plus grande qu'un seuil α :

$$\mathrm{pred}(\mathbf{t} \mid \mathbf{s}, \alpha) = \left|\left\{t \in \mathbf{t} \mid y_{t|\mathbf{s}} > \alpha\right\}\right| \tag{6.5}$$

où nous choisissons α de façon à maximiser la f-mesure d'un corpus de développement.

Ce score rejoint l'approche de Venkatapathy et Bangalore (2009) qui traduisent une phrase en deux étapes : la prédiction des mots cibles et leur réordonnancement. Notre approche est cependant plus permissive, car elle encourage les mots prédits sans empêcher le décodeur d'en utiliser d'autres. Cela nous permet aussi de laisser le décodeur se charger du réordonnancement en même temps qu'il sélectionne les bisegments.

Tout comme pour la cote, nous mémorisons ce score en antémémoire pour chacun des bisegments avant de débuter la traduction d'une phrase.

6.3 Expériences

Nous avons évalué nos systèmes sur des tâches de traduction de l'anglais vers le français. Pour ce faire, nous avons utilisé une partie des corpus rendus disponibles pour le quatrième atelier sur la traduction statistique (Callison-Burch *et al.*, 2009). Le corpus d'entrainement est composé de plus de 1,4 millions de phrases extraites de transcriptions de débats parlementaires européens et de 64 223 phrases extraites de fils de presse.

Le premier volet de nos expériences porte sur la traduction de documents similaires aux corpus d'entrainement (traduction du domaine). Pour ces expériences, nous avons utilisé la section TEST2007 contenant 2 000 phrases de transcriptions de débats parlementaires comme corpus de développement et la section TEST2008 contenant 2 000 autres phrases de transcriptions de débats parlementaires comme corpus de test.

Le deuxième volet de nos expériences porte sur la traduction de documents sous-représentés dans le corpus d'entrainement (traduction hors-domaine). Pour ces expériences, nous avons utilisé la section NEWS-DEV2009A contenant 2051 phrases tirées de nouvelles journalistiques comme corpus de développement et la section NEWS-TEST2009 contenant 1025 phrases comme corpus de test.

Nous avons converti ces corpus en casse minuscule avant de les utiliser pour l'entrainement et l'évaluation de nos systèmes de traduction. Lors de l'entrainement de nos modèles de prédiction, nous avons aussi enlevé les ponctuations et les mots outils comme nous l'avons fait au chapitre précédent. Ainsi, le décodeur peut produire des traductions comprenant des mots outils et des ponctuations, mais nos modèles ne se prononcent que sur les mots de contenu et ignorent les autres.

Les modèles IBM1 ont été entrainés sur tous les mots de contenus. Cependant, comme au chapitre précédent, nous n'avons entrainé nos PPC que sur les mots les plus fréquents, en l'occurrence ceux apparaissant au moins 20 fois. Les vocabulaires anglais et français passent donc respectivement de 133 541 et 143 980 à 21 915 et 28 095 mots uniques.

6.3.1 Prédictions

Avant de lancer nos expériences de traduction, nous avons évalué la qualité des prédictions de nos modèles en choisissant un seuil optimisant la f-mesure sur le corpus TEST2007. Les résultats sont présentés au tableau 6.1. Les conclusions tirées au chapitre précédent tiennent toujours sur ce nouveau corpus de plus grande taille. Les perceptrons favorisent la précision et c'est le PPC avec connexions directes qui obtient les meilleurs résultats.

Modèle	Itération	Seuil (α)	Précision	Rappel	P10	MAP
IBM1	–	0,013	40,5	37,4	50,7	36,8
PPC-0D	20	0,224	55,8	39,6	58,9	42,2
PPC-64	20	0,166	46,6	30,6	48,4	31,5
PPC-64D	20	0,254	57,9	40,3	60,2	44,2

Tableau 6.1 Évaluation des prédictions sur TEST2008 où α optimise la f-mesure sur TEST2007.

6.3.2 Traduction du domaine

Nous avons d'abord évalué nos systèmes sur des transcriptions de débats parlementaires en optimisant leurs poids sur TEST2007 et en les évaluant sur TEST2008. Les poids ainsi optimisés sont présentés au tableaux B.1 et B.2 de l'annexe B. Les résultats de l'évaluation sont présentés au tableau 6.2.

Nous remarquons d'abord que tous les modèles utilisant le contexte obtiennent de meilleures évaluations BLEU que le système de base ne l'utilisant pas. Ces différences se sont avérées statistiquement significatives à plus de 95% selon la technique de bootstrap par paires avec rééchantillonnage (Koehn, 2004b), technique que nous avons utilisée pour tous les tests statistiques de ce chapitre[16] .

Système	BLEU	
	cote	pred
DUC	30,06	30,06
+ IBM1	30,32	30,65
+ PPC-OD	30,68	30,71
+ PPC-64	**30,86**	**31,00**
+ PPC-64D	30,77	30,77

Tableau 6.2 Résultats pour les traductions du domaine. Les évaluations en gras sont significativement meilleures que les autres de leur colonne.

Pour le score pred, l'amélioration de PPC-64 est significative par rapport à tous les autres systèmes. Pour le score cote, l'amélioration de PPC-64 est significative par rapport à tous les autres systèmes sauf PPC-64D, qui est significativement meilleur que le système de base et IBM1, mais pas que PPC-OD.

Nous avons été surpris de constater que notre meilleur modèle en traduction est PPC-64 alors que c'est le pire pour la prédiction des mots. Nous ne voyons pas d'explication évidente à ces bons résultats, mais supposons que le modèle sans

[16] Ces tests ont été faits à l'aide du programme BLEUCOMPARE distribué avec le système de traduction PORTAGE (Sadat *et al.*, 2005).

connexion directe propose des mots que le système de traduction a plus de difficulté à privilégier.

Bien que le score pred obtienne une meilleure évaluation BLEU que le score cote, la différence n'est pas significative pour PPC-64. Les deux scores sont donc équivalents dans cette expérience.

En comparant les traductions de PPC-64 en utilisant le score cote à celles du système de base, nous avons remarqué qu'environ une traduction sur quatre est identique pour les deux systèmes. Les autres traductions ne divergent habituellement que de quelques mots qui ont des sens similaires, mais les modèles utilisant le contexte semblent faire les choix favorisés dans la référence. C'est par exemple le cas à la figure 6.1 où les modèles avec une couche cachée traduisent « december of last year » par « décembre de l'année dernière » comme dans la référence alors que les autres le traduisent par « décembre dernier ». Un exemple où les perceptrons proposent une meilleure traduction que les modèles de base et IBM1 est celui de la figure 6.2. Ce genre d'exemple est cependant plus rare.

6.3.3 Traduction hors-domaine

Dans cette section, nous évaluons les performances de notre système pour la traduction de corpus journalistiques. Nous avons utilisé les mêmes systèmes configurés avec les mêmes poids qu'à la section précédente, mais pour traduire le corpus NEWS-TEST2009 plutôt que des transcriptions de débats parlementaires. Les résultats de ces expériences sont présentés au tableau 6.3.

La diminution d'environ dix points BLEU par rapport aux expériences en traduction du domaine est frappante et reflète bien la difficulté de créer un système de traduction adapté à plusieurs types de textes. Les modèles utilisant le contexte

Source	it was only logical for the european council in helsinki in december last year to give the green light for the resumption of accession negotiations with malta when the accession negotiations were extended .
Référence	il n' était que logique que le conseil européen d' helsinki , en décembre de l' année dernière , ait donné son feu vert , dans le contexte de l' extension des négociations d' adhésion , à la reprise des négociations avec malte .
DUC, + IBM1, + PPC-0D	il **est** logique que le conseil européen **à** helsinki en décembre **dernier à** donner le feu vert pour la reprise des négociations d' adhésion avec malte lorsque les négociations d' adhésion ont été étendues .
+ PPC-64, + PPC-64D	il **était** logique que le conseil européen **d'** helsinki en décembre **de l' année dernière , de** donner le feu vert pour la reprise des négociations d' adhésion avec malte lorsque les négociations d' adhésion ont été étendues .

Figure 6.1 Exemple où seulement les modèles avec une couche cachée préfèrent « décembre de l'année dernière » à « décembre dernier ».

Source	the standards are scarcely comparable , let alone transferable .
Référence	les normes sont à peine comparables et encore moins transférables .
DUC, +IBM1	les normes sont **difficilement** comparables, **sans parler de valeurs** .
+ PPC-0D	les normes sont comparables **à peine** , **et encore moins transférables** .
+ PPC-64, + PPC-64D	les normes sont comparables **à peine** , **voire transférables** .

Figure 6.2 Exemple où seulement les perceptrons traduisent adéquatement « transferable ».

surpassent encore notre modèle de base et ces améliorations sont toujours statistiquement significatives.

Système	BLEU	
	cote	pred
DUC	19,05	19,05
+ IBM1	20,00	**19,92**
+ PPC-0D	19,93	**19,82**
+ PPC-64	**20,45**	**19,89**
+ PPC-64D	20,17	**19,78**

Tableau 6.3 Évaluation des traductions hors-domaines.

Le meilleur système est celui intégrant le modèle PPC-64 à l'aide du score cote. Les gains de ce système sont significatifs par rapport à tous les autres du tableau. Nous remarquons des gains un peu plus modestes pour le score pred, où le meilleur système est celui utilisant IBM1, mais les quatre systèmes intégrant le contexte à l'aide de pred obtiennent tous des évaluations comparables.

Pour estimer une borne supérieure sur les évaluations BLEU que nous pouvions espérer, nous avons adapté les poids des systèmes sur NEWS-DEV2009A. Ces nouveaux poids sont présentés au tableau B.3 et tableau B.4 de l'annexe B. Les résultats de cette expérience d'adaptation sont présentés dans le tableau 6.4.

Système	BLEU	
	cote	pred
DUC	20,58	20,58
+ IBM1	**21,00**	20,92
+ PPC-0D	**20,89**	20,89
+ PPC-64	**20,94**	**21,24**
+ PPC-64D	**20,93**	20,86

Tableau 6.4 Évaluation pour l'adaptation des poids du système de traduction.

C'est encore le système utilisant le modèle PPC-64 qui est le mieux évalué. De plus, nous remarquons que le système utilisant PPC-64 avec le score `cote`, notre meilleur système en traduction hors-domaine, n'est qu'à 0,49 point BLEU de son score en adaptation, alors que cette différence est de 1,53 pour le modèle de base. Cette différence est d'environ un point pour les autres modèles. L'intégration du contexte à notre système de traduction a donc augmenté sa robustesse pour les traductions hors-domaines de nos expériences.

6.4 Travaux précédents

Le meilleur système de traduction de l'anglais vers le français du quatrième atelier sur la traduction statistique a été évalué à 28 points de BLEU (Callison-Burch *et al.*, 2009) comparativement à notre meilleur système qui en a obtenu 21,24. Cette différence est grande, mais la tâche principale de cet atelier était la traduction de nouvelles journalistiques et les participants ont utilisé beaucoup plus de données que nous. Nous ne parlons donc plus vraiment de traduction hors-domaine pour ces systèmes.

Nous pouvons cependant nous comparer aux participants du troisième atelier qui ont traduit des nouvelles à l'aide de leurs systèmes paramétrés pour traduire des débats parlementaires. La meilleure évaluation BLEU pour la traduction du domaine de l'anglais vers le français était de 32 et la meilleure évaluation pour la traduction hors-domaine de 20. Nos systèmes obtiennent donc des performances similaires aux systèmes de pointes dans les conditions où nous les avons évalués.

Nos gains d'environ un point BLEU en traduction du domaine sont similaires à ceux observés par Mauser *et al.* (2009) (section 3.3). Leur système traduisait cependant

des nouvelles journalistiques du chinois et de l'arabe vers l'anglais. Il est donc difficile de comparer leurs résultats aux nôtres plus en détails.

De leur côté, Venkatapathy et Bangalore (2009) ont évalué leur système sur une tâche de traduction de corpus touristique de l'anglais vers l'hindi. L'hindi est une langue très différente des langues européennes entre autres parce que l'ordre des mots n'y est pas très important. Les auteurs n'ont pas réussi à surpasser l'évaluation en BLEU de MOSES. Cependant, comme BLEU est sensible à l'ordre des mots, cette métrique n'est probablement pas un bon indicateur de qualité pour cette paire de langues. Les auteurs rapportent cependant avoir surpassé MOSES pour la prédiction des mots individuels.

6.5 Résumé

Nous avons présenté deux scores permettant d'intégrer nos modèles à un système de traduction. Un premier proposé par Mauser *et al.* (2009) qui calcule la cote des mots présents et un deuxième inspiré de Venkatapathy et Bangalore (2009) qui compte le nombre de prédictions qui se retrouvent dans la traduction. Les deux scores donnent des améliorations équivalentes pour les traductions du domaine, mais la cote semble préférable pour la traduction hors-domaine.

Chose surprenante, c'est le modèle sans connexion directe qui produit les meilleures traductions, même si c'est le moins bon en prédiction de mots. Nous croyons que c'est parce que les modèles avec connexions directes proposent des mots qui sont déjà bien acquis par le système de traduction alors que le modèle n'ayant aucune connexion directe en propose de nouveaux. Cela reste cependant à vérifier.

Dans toutes nos expériences, les systèmes intégrant le contexte, même ceux n'utilisant qu'un simple modèle IBM1, ont surpassé le système de base en évaluation

BLEU, que ce soit en traduction du domaine ou en traduction hors-domaine. Alors que l'amélioration en traduction du domaine était de 0,94 en traduction du domaine, elle est de 1,4 en traduction hors-domaine. Ces améliorations sont modestes, mais comme elles ont été observées dans toutes les expériences que nous avons menées, nous croyons qu'elles ne sont pas le fruit du hasard.

7 Conclusion

Les systèmes de traduction statistique à base de segments traduisent les phrases un segment à la fois. Un problème avec les scores évaluant les segments cibles est qu'ils ne considèrent qu'un contexte très restreint. Les scores de la table de traduction ne tiennent compte que du segment source sans se soucier du reste de la phrase et le modèle de langue que des quelques mots avant et après le segment cible.

Plusieurs travaux ont cherché à conditionner ces choix de bisegments sur un contexte plus large. Nous avons divisé ces travaux en trois catégories. Les modèles thématiques modélisent les phrases ou les documents par thèmes. Les thèmes peuvent être fixés à l'avance ou adaptés à chaque document. Dans le premier cas, il faut connaitre les thèmes de tous les documents que nous voudrons traduire et dans le deuxième, il faut entrainer un système de traduction pour chaque document source. Les modèles de désambigüisation choisissent la traduction de chaque segment à l'aide d'un classificateur dédié pouvant considérer toute la phrase source. Ces modèles obtiennent des résultats intéressants, mais le grand nombre de classificateurs nécessaires peut les rendre lourds à gérer. Finalement, les modèles de vocabulaire actif cherchent à prédire les mots qui devraient apparaitre dans la traduction d'une phrase source. C'est cette dernière approche que nous avons choisie dans cette thèse.

Les systèmes de prédiction de mots décrits dans la littérature modélisent les cooccurrences de mots sources et cibles à l'aide de modèles de régressions linéaires ou logistiques. Nous avons plutôt utilisé des PPC parce qu'ils projettent les phrases

sources vers une représentation alternative (couche cachée) avant de faire les prédictions. Bien que la couche cachée d'un PPC soit difficile à interpréter, nous espérions que celle-ci rapproche les phrases sources qui ont des traductions similaires au-delà de ce qui peut être fait en ne considérant que les mots sources qu'elles ont en commun. Nous avons montré que c'est possible sur un exemple artificiel à la section 4.6.4 et nous avons observé des exemples concrets à la section 5.6.

Notre première série d'expériences vérifiait si nos modèles pouvaient prédire les mots présents dans la traduction d'une phrase source donnée. Même si la représentation alternative offerte par la couche cachée semblait intéressante, seulement nos PPC faisant aussi usage d'un dictionnaire ont réussi à surpasser le modèle de référence IBM1.

Notre deuxième série d'expériences cherchait à évaluer l'apport de nos modèles lorsqu'ils sont intégrés à un système de traduction. Tous nos systèmes utilisant le contexte ont été significativement meilleurs que notre système de base. Chose surprenante, c'est le PPC sans dictionnaire qui a apporté les meilleures améliorations, même si c'était un des modèles les moins bons à prédire les mots. Nous croyons que c'est parce qu'il se démarque plus des modèles utilisés par le décodeur de base, lui permettant ainsi d'apporter plus d'informations nouvelles au système.

La contribution la plus importante de cette thèse est la proposition d'utiliser un PPC pour prédire les mots devant traduire une phrase source. Nous avons présenté un algorithme pour entrainer nos PPC et une extension pour leur intégrer des dictionnaires bilingues. Nous avons aussi comparé favorablement nos PPC au modèle IBM1 couramment utilisé en traduction.

Notre deuxième contribution consiste à l'intégration et à l'évaluation de nos modèles de prédiction à un système de traduction statistique à base de segments.

Nos systèmes de traduction utilisant un PPC ont surpassé les systèmes n'utilisant aucun de contexte et les systèmes utilisant IBM1.

Plusieurs questions restent encore sans réponse. Nous voulons d'abord vérifier si nos résultats peuvent être reproduits sur d'autres types de corpus et d'autres paires de langues. Nous aimerions aussi évaluer l'impact de la taille du corpus d'entrainement sur la qualité des prédictions.

Nous contrôlons la taille de nos modèles en ne considérant que les mots les plus fréquents. Comme les systèmes de traduction sont déjà efficaces sur ces mots, nous croyons qu'il est possible de faire mieux en choisissant les mots pour lesquels notre système de traduction à le plus de difficulté. La manière de sélectionner ces mots reste cependant à déterminer.

Une autre partie de notre modèle qui gagnerait à être améliorée est le critère de régularisation, qui permet d'introduire de l'information à priori dans le modèle. Nous avons choisi une régularisation L1 favorisant les poids se rapprochant de zéro, car c'est un critère simple et populaire. Nous aimerions trouver un critère de régularisation qui favoriserait plutôt les poids qui rapprochent les représentations cachées des phrases sources dont les traductions sont similaires.

Nous pourrions aussi améliorer le passage à l'échelle. Comme nous en avons discuté à la section 4.6.6, la prédiction des mots cibles à partir de la représentation cachée est une opération couteuse, car nous devons prédire tous les mots du vocabulaire cible. Nous planifions de modifier notre implémentation pour limiter les mots prédits à ceux qui peuvent être produits par le système de traduction étant donnée la phrase source. Cela diminuerait grandement le temps d'entrainement tout en nous donnant l'information nécessaire pour calculer les cotes lors de la traduction.

Nous planifions aussi de nous inspirer de Collobert et Weston (2008) pour élaborer un PPC prédisant les traductions dans plusieurs langues en même temps. Dans ce nouveau modèle, la représentation cachée serait partagée pour toutes les paires de langues. Il serait donc possible d'utiliser tous les bitextes à notre disposition pour améliorer tous les systèmes de traductions. Nous espérons que cela aiderait particulièrement les traductions impliquant des langues peu dotées.

Une limitation importante de nos modèles est qu'ils traitent les phrases comme des sacs de mots et ignorent leur structure. Une solution partielle consiste à traiter les phrases comme des ensembles de segments plutôt que des ensembles de mots. Cela ne modélise cependant que la structure locale sans tenir compte des relations grammaticales et sémantiques. Nous nous interrogeons donc sur la nature du contexte à considérer pour améliorer nos prédictions et sur la manière de l'intégrer à nos modèles.

Finalement, un résultat qui nous apparait intéressant est l'observation que la couche cachée offre une représentation intéressante des phrases. Nous prévoyons donc vérifier si cette représentation pourrait être utilisée dans d'autres tâches comme la détection de paraphrases ou la désambigüisation de mots.

Bibliographie

Al-Onaizan, Y. and Papineni, K. (2006). Distortion models for statistical machine translation. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics* , pages 529–536.

Arun, A., Dyer, C., Haddow, B., Blunsom, P. and Lopez, A. et al. (2009). Monte carlo inference and maximization for phrase-based translation. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning* , pages 102–110.

Axelrod, A., Mayne, R. B., Callison-burch, C., Osborne, M. and Talbot, D. (2005). Edinburgh system description for the 2005 iwslt speech translation evaluation. In *Proceedings of International Workshop on Spoken Language Translation (IWSLT)* .

Ayan, N. F. and Dorr, B. J. (2006). Going beyond aer: an extensive analysis of word alignments and their impact on mt. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics* , pages 9–16.

Banerjee, S. and Lavie, A. (2005). METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization* , pages 65–72.

Bangalore, S., Haffner, P. and Kanthak, S. (2007). Statistical machine translation through global lexical selection and sentence reconstruction. In *ACL* .

Bengio, Y., Ducharme, R., Vincent, P. and Janvin, C. (2003). A neural probabilistic language model. *The Journal of Machine Learning Research, 3*, 1137–1155.

Berger, A. L., Brown, P. F., Pietra, S. A. D., Pietra, V. J. D. and Gillett, J. R. et al. (1994). The candide system for machine translation. In *HLT '94: Proceedings of the workshop on Human Language Technology* , pages 157–162.

Bishop, C. M. (1995). *Neural Networks for Pattern Recognition*. Oxford University Press.

Brown, P. F., Lai, J. C. and Mercer, R. L. (1991). Aligning sentences in parallel corpora. In *Proceedings of the 29th annual meeting on Association for Computational Linguistics* , pages 169–176.

Brown, P. F., Pietra, V. J. D., Pietra, S. A. D. and Mercer, R. L. (1993). The mathematics of statistical machine translation: parameter estimation. *Computational Linguistics, 19*(2), 263–311.

Brunning, J., de Gispert, A. and Byrne, W. (2009). Context-dependent alignment models for statistical machine translation. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics* NAACL '09 , pages 110–118.

Cabezas, C. and Resnik, P. (2005). Using WSD Techniques for Lexical Selection in Statistical Machine Translation. Technical Report University of Maryland, College Park.

Callison-Burch, C. (2009). Fast, cheap, and creative: Evaluating translation quality using amazon's mechanical turk. In *Proceedings of EMNLP 2009* .

Callison-Burch, C., Koehn, P., Monz, C. and Schroeder, J., editors (2009). *Proceedings of the Fourth Workshop on Statistical Machine Translation*. Association for Computational Linguistics.

Callison-Burch, C., Koehn, P., Monz, C., Schroeder, J. and Fordyce, C. S., editors (2008). *Proceedings of the Third Workshop on Statistical Machine Translation*. Association for Computational Linguistics.

Carpuat, M., Su, W. and Wu, D. (2004). Augmenting ensemble classification for word sense disambiguation with a kernel pca model. In Mihalcea, R. and Edmonds, P., editors, *Senseval-3: Third International Workshop on the Evaluation of Systems for the Semantic Analysis of Text* , pages 88–92.

Carpuat, M. and Wu, D. (2005). Word sense disambiguation vs. statistical machine translation. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)* , pages 387–394.

Carpuat, M. and Wu, D. (2007). Improving statistical machine translation using word sense disambiguation. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)* , pages 61–72.

Carpuat, M. and Wu, D. (2008). Evaluation of context-dependent phrasal translation lexicons for statistical machine translation. In *Proceedings of the Sixth International Language Resources and Evaluation (LREC'08)* .

Chan, Y. S., Ng, H. T. and Chiang, D. (2007). Word sense disambiguation improves statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics* , pages 33–40.

Chandioux, J. (1988). Meteo: An operational translation system. In *Proceedings of the 2nd Conference on RIAO* , pages 829–839.

Chiang, D. (2007). Hierarchical phrase-based translation. *Computational Linguistics, 33*(2), 201–228.

Civera, J. and Juan, A. (2006). Mixtures of IBM Model 2. In *Proc. of the 11th Annual Conf. of the European Assoc. for Machine Translation (EAMT 2006)* , pages 159–167.

Civera, J. and Juan, A. (2007). Domain adaptation in statistical machine translation with mixture modelling. In *Proceedings of the Second Workshop on Statistical Machine Translation* , pages 177–180.

Collins, M., Koehn, P. and Kuᴗerová, I. (2005). Clause restructuring for statistical machine translation. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics* , pages 531–540.

Collobert, R. and Weston, J. (2008). A unified architecture for natural language processing: deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning* ICML '08 , pages 160–167.

Darroch, J. N. and Ratcliff, D. (1972). Generalized iterative scaling for log-linear models. *The Annals of Mathematical Statistics, 43*(5), 1470–1480.

Dempster, A., Laird, N. and Rubin, D. (1977). Maximum likelihood from incomplete data via the em algorithm.. *J. Royal Statistical Society, Series B, 39*(1), 1–38.

DeNero, J., Gillick, D., Zhang, J. and Klein, D. (2006). Why generative phrase models underperform surface heuristics. In *Proceedings on the Workshop on Statistical Machine Translation* , pages 31–38.

Dong, Z. and Dong, Q. (2006). *Hownet And the Computation of Meaning*. World Scientific Publishing Co., Inc..

Enright, J. and Kondrak, G. (2007). A fast method for parallel document identification. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Companion Volume, Short Papers* , pages 29–32.

Fellbaum, C., editor (1998). *WordNet: An Electronic Lexical Database (Language, Speech, and Communication)*. The MIT Press.

Foster, G., Kuhn, R. and Johnson, H. (2006). Phrasetable smoothing for statistical machine translation. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing* , pages 53–61.

Gale, W. A. and Church, K. W. (1993). A program for aligning sentences in bilingual corpora. *Comput. Linguist., 19*(1), 75–102.

Germann, U., Jahr, M., Knight, K., Marcu, D. and Yamada, K. (2001). Fast decoding and optimal decoding for machine translation. In *Proceedings of the 39th Annual Meeting on Association for Computational Linguistics* ACL '01 , pages 228–235.

Giménez, J. and Màrquez, L. (2007). Context-aware discriminative phrase selection for statistical machine translation. In *Proceedings of the Second Workshop on Statistical Machine Translation* , pages 159–166.

Gimpel, K. and Smith, N. A. (2008). Rich source-side context for statistical machine translation. In *Proceedings of the Third Workshop on Statistical Machine Translation* StatMT '08 , pages 9–17.

Gotti, F. (2009). CSST : Livrable 2 : Nouvelle chaÃőne de traduction. Technical Report RALI (DIRO, Université de Montréal).

Haghighi, A., Blitzer, J., DeNero, J. and Klein, D. (2009). Better word alignments with supervised itg models. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP* , pages 923–931.

Hasan, S., Ganitkevitch, J., Ney, H. and Andrés-Ferrer, J. (2008). Triplet lexicon models for statistical machine translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing* EMNLP '08 , pages 372–381.

Hildebrand, A. S., Eck, M., Vogel, S. and Waibel, A. (2005). Adaptation of the translation model for statistical machine translation based on information retrieval. In *Proceedings of the European Association for Machine Translation 10th Annual Conference* .

Huang, L. and Chiang, D. (2007). Forest rescoring: Faster decoding with integrated language models. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics* , pages 144–151.

Hutchins, J. (2001). Machine translation over fifty years. *Histoire Epistémologie Langage, 23*(1), 7–31.

J-Y.Nie and Cai, J. (2001). Filtering noisy parallel corpora of web pages. In *IEEE symposium on NLP and Knowledge Engineering* , pages 453–458.

Johnson, H., Martin, J., Foster, G. and Kuhn, R. (2007). Improving translation quality by discarding most of the phrasetable. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)* , pages 967–975.

Knight, K. (1999). Decoding complexity in word-replacement translation models. *Comput. Linguist., 25*, 607–615.

Koehn, P. (2004a). Pharaoh: A beam search decoder for phrase-based statistical machine translation models. In *AMTA* , pages 115–124.

Koehn, P. (2004b). Statistical significance tests for machine translation evaluation. In Lin, D. and Wu, D., editors, *Proceedings of EMNLP 2004* , pages 388–395.

Koehn, P. (2005). Europarl: A parallel corpus for statistical machine translation. In *MT Summit X* , pages 323–330.

Koehn, P., Hoang, H., Birch, A., Callison-Burch, C. and Federico, M. et al. (2007). Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions* , pages 177–180.

Koehn, P., Och, F. J. and Marcu, D. (2003). Statistical phrase-based translation. In *NAACL '03: Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology* , pages 48–54.

Koehn, P. and Schroeder, J. (2007). Experiments in domain adaptation for statistical machine translation. In *Proceedings of the Second Workshop on Statistical Machine Translation* , pages 224–227.

Kuhn, R., Yuen, D., Simard, M., Paul, P. and Foster, G. et al. (2006). Segment choice models: feature-rich models for global distortion in statistical machine translation. In *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics* , pages 25–32.

Lacoste-Julien, S., Taskar, B., Klein, D. and Jordan, M. I. (2006). Word alignment via quadratic assignment. In *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics* , pages 112–119.

Langlais, P., Gotti, F. and Patry, A. (2006). De la chambre des communes 'e la chambre d'isolement: adaptabilit'e d'un syst'eme de traduction bas'e sur les segments. In *13th Conf'erence sur le Traitement Automatiqu des Langues Naturelles (TALN 2006)* , pages 217–226.

Langlais, P., Gotti, F. and Patry, A. (2007). A greedy decoder for phrase-based statistical machine translation. In *The 11th International Conference on Theoretical and Methodological Issues in Machine Translation* , pages 104–113.

Langlais, P., Grandrabur, S., Leplus, T. and Lapalme, G. (2005). The long-term forecast for weather bulletin translation. *Machine Translation*, pp. 83–112.

LeCun, Y., Bottou, L., Orr, G. and Muller, K. (1998). Efficient backprop. In Orr, G. and K., M., editors, *Neural Networks: Tricks of the trade* .

Li, Z., Eisner, J. and Khudanpur, S. (2009). Variational decoding for statistical machine translation. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2 - Volume 2* ACL-IJCNLP '09 , pages 593–601.

Liang, P., Taskar, B. and Klein, D. (2006). Alignment by agreement. In *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics* , pages 104–111.

Lü, Y., Huang, J. and Liu, Q. (2007). Improving statistical machine translation performance by training data selection and optimization. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)* , pages 343–350.

Ma, X. and Liberman, M. (1999). BITS: A method for bilingual text search over the web. In *Machine Translation Summit VII* .

Malouf, R. (2002). A comparison of algorithms for maximum entropy parameter estimation. In *proceedings of the 6th conference on Natural language learning - Volume 20* COLING-02 , pages 1–7. Morristown, NJ, USA.

Marcu, D. and Wong, W. (2002). A phrase-based, joint probability model for statistical machine translation. In *EMNLP '02: Proceedings of the ACL-02 conference on Empirical methods in natural language processing* , pages 133–139.

Mauser, A., Hasan, S. and Ney, H. (2009). Extending statistical machine translation with discriminative and trigger-based lexicon models. In *Conference on Empirical Methods in Natural Language Processing* .

Max, A., Malhoufi, R. and Langlais, P. (2009). Prise en compte de d'ependances syntaxiques pour la traduction contextuelle de segments. In *16'e Conf'erence sur le Traitement Automatique des Langues Naturelles (TALN'09)* .

Moore, R. C. (2002). Fast and accurate sentence alignment of bilingual corpora. In *AMTA* , pages 135–144.

Moore, R. C. and Quirk, C. (2007). Faster beam-search decoding for phrasal statistical machine translation. In *Proceedings of MT Summit XI* , pages 321–327.

Moore, R. C., Yih, W.-t. and Bode, A. (2006). Improved discriminative bilingual word alignment. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics* ACL-44 , pages 513–520.

Nagao, M. (1984). A framework of a mechanical translation between japanese and english by analogy principle. In *Proc. of the international NATO symposium on Artificial and human intelligence* , pages 173–180.

Nießen, S., Vogel, S., Ney, H. and Tillmann, C. (1998). A DP based search algorithm for statistical machine translation. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, Volume 2* , pages 960–967.

Niehues, J. and Vogel, S. (2008). Discriminative word alignment via alignment matrix modeling. In *Proceedings of the Third Workshop on Statistical Machine Translation* StatMT '08 , pages 18–25.

Och, F. J., Gildea, D., Khudanpur, S., Sarkar, A. and Yamada, K. et al. (2004). A smorgasbord of features for statistical machine translation.. In *HLT-NAACL* , pages 161–168.

Och, F. J. and Ney, H. (2003). A systematic comparison of various statistical alignment models. *Computational Linguistics, 29*(1), 19–51.

Papineni, K., Roukos, S., Ward, T. and Zhu, W.-J. (2002). Bleu: a method for automatic evaluation of machine translation. In *ACL* , pages 311–318.

Patry, A. (2006). MOOD: un cadre d'applications pour le développement de décodeurs en traduction statistique. Master's thesis, Université de Montréal.

Patry, A., Gotti, F. and Langlais, P. (2006a). MOOD: A modular object-oriented decoder for statistical machine translation. In *5th LREC* , pages 709–714.

Patry, A., Gotti, F. and Langlais, P. (2006b). MOOD at work: Ramses versus pharaoh. In *Workshop on Statistical Machine Translation, HLT-NAACL* , pages 126–129.

Patry, A. and Langlais, P. (2005). Paradocs: un système d'identification automatique de documents parallèles. In *12e Conference sur le Traitement Automatique des Langues Naturelles (TALN)* , pages 223–232.

Patry, A. and Langlais, P. (2009). Prediction of words in statistical machine translation using a multilayer perceptron. In for Machine Translation (IAMT), I. A., editor, *Proceedings of the Twelfth Machine Translation Summit* .

Patry, A., Langlais, P. and Béchet, F. (2007). MISTRAL: A lattice translation system for IWSLT 2007. In *Proceedings of IWSLT 2007* .

Press, W. H., Teukolsky, S. A., Vetterling, W. T. and Flannery, B. P. (1992). *Numerical recipes in C (2nd ed.): the art of scientific computing*. Cambridge University Press.

Rafalovitch, A. and Dale, R. (2009). United nations general assembly resolutions: A six-language parallel corpus. In *Proceedings of the MT Summit XII* , pages 292–299.International Association of Machine Translation

Resnik, P. and Smith, N. A. (2003). The web as a parallel corpus. *Comput. Linguist., 29*(3), 349–380.

Rumelhart, D. E., Durbin, R., Golden, R. and Chauvin, Y. (1995). Backpropagation: the basic theory. , pp. 1–34.

Sadat, F., Johnson, H., Agbago, A., Kuhn, R. and Martin, J. et al. (2005). Portage: A phrasebased machine translation system. In , pages 129–132.

Schwenk, H. (2007). Continuous space language models. *Comput. Speech Lang., 21*(3), 492–518.

Schwenk, H., Déchelotte, D., Bonneau-Maynard, H. and Allauzen, A. (2007). Modèles statistiques enrichis par la syntaxe pour la traduction automatique. In *Actes de la conférence sur le Traitement Automatique des Langues Naturelles* , pages 253–262.

Simard, Michel and Plamondon, P. (1998). Bilingual sentence alignment: Balancing robustness and accuracy. In *Machine Translation* , pages 59–80.

Simard, M. (1998). The BAF: A corpus of english-french bitext. In *Proceedings of LREC 98* .

Simard, M., Foster, G. F. and Isabelle, P. (1993). Using cognates to align sentences in bilingual corpora. In *CASCON '93: Proceedings of the 1993 conference of the Centre for Advanced Studies on Collaborative research* , pages 1071–1082.

Snover, M., Dorr, B., Schwartz, R., Micciulla, L. and Makhoul, J. (2006). A study of translation edit rate with targeted human annotation. In *Proceedings AMTA* , pages 223–231.

Stolcke, A. (2002). SRILM – an extensible language modeling toolkit. In *Proceedings of Intl. Conf. on Spoken Language Processing* , pages 901–904.

Stroppa, N., van den Bosch, A. and Way, A. (2007). Exploiting source similarity for smt using context-informed features. In *Proceedings of the 11th International Conference on Theoretical and Methodological Issues in Machine Translation* , pages 231–240.

Taskar, B., Lacoste-Julien, S. and Klein, D. (2005). A discriminative matching approach to word alignment. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing* HLT '05 , pages 73–80.

Tillmann, C. (2004). A unigram orientation model for statistical machine translation. In *Proceedings of HLT-NAACL 2004: Short Papers* , pages 101–104.

Tillmann, C., Vogel, S., Ney, H. and Zubiaga, A. (1997). A dp based search using monotone alignments in statistical translation. In *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics and Eighth Conference of the European Chapter of the Association for Computational Linguistics* , pages 289–296.

Varea, I. G., Och, F. J., Ney, H. and Casacuberta, F. (2002). Improving alignment quality in statistical machine translation using context-dependent maximum entropy models. In *Proceedings of the 19th international conference on Computational linguistics - Volume 1* , pages 1–7.

Venkatapathy, S. and Bangalore, S. (2009). Discriminative machine translation using global lexical selection. *ACM Trans. Asian Lang. Inf. Process., 8*(2).

Vickrey, D., Biewald, L., Teyssier, M. and Koller, D. (2005). Word-sense disambiguation for machine translation. In *HLT '05: Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing* , pages 771–778.

Vilar, D., Popović, M. and Ney, H. (2006a). AER: Do we need to "improve" our alignments?. In *International Workshop on Spoken Language Translation* , pages 205–212.

Vogel, S., Ney, H. and Tillmann, C. (1996). Hmm-based word alignment in statistical translation. In *Proceedings of the 16th conference on Computational linguistics* , pages 836–841.

Vogel, S., Zhang, Y., Huang, F., Tribble, A. and Venogupal, A. et al. (2003). The CMU statistical translation system. In *Proceedings of MT Summit IX* .

Whaley, R. C. and Petitet, A. (2005). Minimizing development and maintenance costs in supporting persistently optimized BLAS. *Software: Practice and Experience, 35*(2), 101–121.

Wu, D. and Wong, H. (1998). Machine translation with a stochastic grammatical channel. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, Volume 2* , pages 1408–1415.

Wu, H. and Haifeng, W. (2007). Comparative study of word alignment heuristics and phrase-based SMT. In *Proceedings of MT Summit XI* , pages 515–520.

Xia, F. and McCord, M. (2004). Improving a statistical mt system with automatically learned rewrite patterns. In *Proceedings of the 20th international conference on Computational Linguistics* .

Xu, J., Deng, Y., Gao, Y. and Ney, H. (2007). Domain dependant statistical machine translation. In *Proceedings of MT Summit XI* , pages 515–520.

Yahyaei, S. and Monz, C. (2009). Decoding by dynamic chunking for statistical machine translation. In *MT Summit XII: Proceedings of the twelfth Machine Translation Summit* , pages 160–167.

Yamada, K. and Knight, K. (2001). A syntax-based statistical translation model. In *Proceedings of 39th Annual Meeting of the Association for Computational Linguistics* , pages 523–530.

Zaslavskiy, M., Dymetman, M. and Cancedda, N. (2009). Phrase-based statistical machine translation as a traveling salesman problem. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP* , pages 333–341.

Zhao, B., Eck, M. and Vogel, S. (2004). Language model adaptation for statistical machine translation with structured query models. In *COLING '04: Proceedings of the 20th international conference on Computational Linguistics* .

Zhao, B. and Xing, E. P. (2006). Bitam: bilingual topic admixture models for word alignment. In *Proceedings of the COLING/ACL on Main conference poster sessions* , pages 969–976.

Zhao, B. and Xing, E. P. (2007). Hm-bitam: Bilingual topic exploration, word alignment, and translation. In Platt, J. C., Koller, D., Singer, Y. and Roweis, S. T., editors, *NIPS* .

A Notation

p	Nombre de cepts dans un alignement.
$\mathcal{A}$	Une fonction retournant les paires de phrases parallèles dans une paire de documents.
$\mathbf{a}$	Une séquence de cepts.
a	Un cept alignement un partie d'un document ou d'une phrase source à une autre partie d'un document ou d'une phrase cible.
$\mathcal{B}$	Un bitext représenté comme un ensemble de paires de phrases.
$\mathbf{1}$	Fonction retournant un si son paramètre est vrai.
δ	Une étape dans la traduction d'une phrase (e.g. la traduction d'un segment).
$\vec{h}$	La représentation de $\vec{\phi}$ apprise automatiquement par un perceptron à plusieurs couches.
lex	Score de pondération lexicale pour les entrées de la table de traduction.
φ	Score de fréquence relative pour les entrées de la table de traduction.
$\mathbf{r}$	Une traduction de référence.
r	Un mot dans une phrase de référence.
D_s	Un document source.
$\vec{\phi}$	La représentation vectorielle d'une phrase source donnée.
l	Nombre de mots dans une phrase source.
$\mathbf{s}$	Une phrase source.
Δ	L'ensemble des séquences d'étapes permettant de traduire une phrase source par une phrase cible donnée.
s	Un mot source.
D_t	Un document cible.

$\vec{\mathfrak{t}}$	La représentation vectorielle d'une phrase cible donnée.
z	Le thème d'une phrase ou d'un document.
$\mathcal{L}_t$	L'ensemble des phrases permises dans la langue cible.
m	Nombre de mots dans une phrase cible.
$\mathbf{t}$	Une phrase cible.
$\mathcal{T}$	Le vocabulaire cible.
t	Un mot cible.
$\vec{y}$	Les prédictions d'un modèle.

B Poids des décodeurs

Cette annexe présente les poids des différents scores des systèmes de traduction utilisés au chapitre 6. Les scores utilisés sont les suivants :

lv et $pred$	Score du modèle intégrant le contexte (section 6.2.1, section 6.2.2).
$\varphi(\mathbf{t} \mid \mathbf{s})$ et $\varphi(\mathbf{s} \mid \mathbf{t})$	Score évaluant la probabilité des bigsegments par fréquence relative (section 2.4.1).
$lex(\mathbf{t} \mid \mathbf{s})$ et $lex(\mathbf{s} \mid \mathbf{t})$	Score évaluant la probabilité des bigsegments par fréquence relative sur les alignements de mots (section 2.4.1).
pen_m	Pénalité sur le nombre de mots cibles (section 2.4.1).
pen_s	Pénalité sur le nombre de bisegments (section 2.4.1).
lm	Modèle de langue.
r	Modèle de réordonnancement.

Système	lv	$\varphi(\mathbf{t} \mid \mathbf{s})$	$lex(\mathbf{t} \mid \mathbf{s})$	$\varphi(\mathbf{s} \mid \mathbf{t})$	$lex(\mathbf{s} \mid \mathbf{t})$	pen_m	pen_s	lm	r
DUC	–	0,13	0,23	0,48	0,16	−1	0,35	0,4	0,15
+ IBM1	0,29	0,17	0,28	0,50	0,12	−1	0,43	0,45	0,26
+ PPC-0D	1	0,05	0,06	0,14	0,04	−0,3	0,1	0,12	0,06
+ PPC-64	1	0,01	0,12	0,16	0,03	−0,32	−0,08	0,14	0,06
+ PPC-64D	1	0 06	0 08	0 16	0 08	−0 41	−0 02	0 15	0 05

Tableau B.1 Poids des systèmes de traduction lors de l'utilisation du score ll.

Système	$pred$	$\varphi(\mathbf{t} \mid \mathbf{s})$	$lex(\mathbf{t} \mid \mathbf{s})$	$\varphi(\mathbf{s} \mid \mathbf{t})$	$lex(\mathbf{s} \mid \mathbf{t})$	pen_m	pen_s	lm	r
DUC	–	0,13	0,23	0,48	0,16	−1	0,35	0,4	0,15
+ IBM1	0 63	0 27	0 42	0 7	0 16	−1	0 5	0 65	0 37
+ PPC-0D	1	0 21	0 32	0 61	0 087	−0 84	0 13	0 54	0 28
+ PPC-64	0 93	0 29	0 37	0 66	0 12	−1	0 49	0 64	0 45
+ PPC-64D	1	0 17	0 29	0 49	0 054	−0 68	−0 013	0 52	0 33

Tableau B.2 Poids des systèmes de traduction lors de l'utilisation du score pred.

Système	lv	$\varphi(\mathbf{t} \mid \mathbf{s})$	$lex(\mathbf{t} \mid \mathbf{s})$	$\varphi(\mathbf{s} \mid \mathbf{t})$	$lex(\mathbf{s} \mid \mathbf{t})$	pen_m	pen_s	lm	r
DUC	–	0 041	0 38	0 48	0 094	−1	0 47	0 42	0 38
+ IBM1	0 4	0 12	0 18	0 35	0 17	−1	0 36	0 3	0 29
+ PPC-0D	0 62	0 11	0 25	0 49	0 12	−1	0 49	0 35	0 32
+ PPC-64	1	0 11	0 1	0 15	0 066	−0 39	0 04	0 14	0 13
+ PPC-64D	1	0 02	0 16	0 27	0 083	−0 6	0 23	0 21	0 19

Tableau B.3 Poids des systèmes de traduction adaptés aux corpus hors-domaine lors de l'utilisation du score lv.

Système	pred	$\varphi(\mathbf{t} \mid \mathbf{s})$	$lex(\mathbf{t} \mid \mathbf{s})$	$\varphi(\mathbf{s} \mid \mathbf{t})$	$lex(\mathbf{s} \mid \mathbf{t})$	pen_m	pen_s	lm	r
DUC	–	0 041	0 38	0 48	0 094	−1	0 47	0 42	0 38
+ IBM1	0 18	0 23	0 46	0 5	0 14	−1	0 12	0 42	0 38
+ PPC-0D	0 86	0 18	0 23	0 38	0 19	−1	0 33	0 43	0 4
+ PPC-64	0 85	0 19	0 47	0 29	0 21	−1	−0 14	0 46	0 37
+ PPC-64D	0 85	0 2	0 36	0 4	0 17	−1	0 48	0 49	0 43

Tableau B.4 Poids des systèmes de traduction adaptés aux corpus hors-domaine lors de l'utilisation du score pred.