

La représentation des documents par réseaux de neurones pour la compréhension de documents parlés

Killian Janod

▶ **To cite this version:**

Killian Janod. La représentation des documents par réseaux de neurones pour la compréhension de documents parlés. Intelligence artificielle [cs.AI]. Université d'Avignon, 2017. Français. NNT : 2017AVIG0222 . tel-01824741

HAL Id: tel-01824741
https://theses.hal.science/tel-01824741

Submitted on 27 Jun 2018

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



UNIVERSITÉ D'AVIGNON ET DES PAYS DE VAUCLUSE

THÈSE

présentée à l'Université d'Avignon et des Pays de Vaucluse
pour obtenir le diplôme de DOCTORAT

SPÉCIALITÉ : Informatique

École Doctorale 536 « Sciences et Agrosciences »
Laboratoire d'Informatique (EA 4128)

La représentation des documents par réseaux de neurones pour la compréhension de documents parlés

Application au cas de la classification thématique

par **Killian Janod**

Soutenue publiquement le 27/11/17 devant un jury composé de :

Dr	Gilles Adda	CNRS-LIMSI	Rapporteur
Pr	Frédéric Béchet	LIF	Rapporteur
Pr	Fabrice Lefèvre	LIA	Président du jury
Dre	Véronique Moriceau	MCF, IRIT	Examinateur
Dr	Benjamin Lecouteux	MCF, LIG/GETALP	Examinateur
Dr	Xavier Bost	Ingénieur R&D Orkis	Membre invité
Pr	Georges Linares	LIA	Directeur
Dr	Richard Dufour	MCF, LIA	Co-Encadrant
Dr	Mohamed Morchid	MCF, LIA	Co-Encadrant



Laboratoire d'Informatique d'Avignon

Remerciements

Mes premiers remerciements iront aux membres de mon jury qui ont accepté de participer à ma soutenance. Merci au président du jury Pr Fabrice Lefèvre, aux rapporteurs Dr Gilles Adda et Pr Frédéric Béchet ainsi qu'aux examinateurs Dre Véronique Moriceau et Dr Benjamin Lecouteux.

Je remercie d'abord mon directeur de thèse, Pr Georges Linares et mon premier co-encadrant, Dr Richard Dufour, dont les interventions et la direction ont rendu ces travaux possibles. Je veux remercier aussi Dr Mohamed Morchid, car s'il n'avait pas rejoint l'équipe d'encadrement, je ne me serais jamais dépêtré de mes problématiques initiales.
Je remercie également André Capurro, de m'avoir fait confiance et d'avoir financé cette aventure. Mais aussi Philippe Leroy, de m'avoir fait découvrir une approche du métier de programmeur que l'on n'apprend pas à l'université. Je veux aussi remercier tous les membres d'Orkis : Angélique, Carole, Catherine , Christine, Emmanuelle, Bernard, Bertrand, Daniel, Éric, Jean-Marc, Sylvain, Sébastien et Pierre, pour leur accueil chaleureux et tous ces moments de convivialité partagés dans les locaux d'Orkis. Je sais que vous avez réservé le même traitement à mes successeurs et amis Titouan et Xavier.

Je veux remercier Michel et iSmart de m'avoir donné le temps supplémentaire nécessaire pour terminer la rédaction de ces travaux et de m'avoir intégré dans leur équipe. Je sais qu'ensemble nous ferons quelque chose de bien. Je veux remercier toute l'équipe administrative du LIA, Simone, Dominique, Michèle et tous les autres, vous avez été un maillon indispensable à la réussite des ces travaux. Je voudrais aussi remercier toutes les personnes du LIA qui sont devenues des collègues, Corinne, Fabrice, Jean-François, Stephane, Mickael R., Driss , etc. parce que sans eux et leur convivialité un séjour au LIA ne serait pas le même. Merci aussi à Bassam pour ces moments d'échanges récurrents, principalement autour de sujets de TP python, mais aussi de m'avoir fait encadrer le projet de certains étudiants si particuliers. Merci aussi à Imedh, Adrien, Adrian, JV, Romain, Hugo, Cedric, Elvys, Matthieu, Moez, Zakaria, Waad, Mayeul, Mathias et tous ceux que qui ne sont pas cités dans ces quelques lignes, mais que je n'oublie pas, pour ces trop longues balades dans les couloirs du CERI pour un (pas si bon) café afin de relâcher un peu la pression tout en martyrisant cette pauvre trottinette. J'en profite aussi pour remercier Afsana, de m'avoir mis régulièrement face à la cruelle réalité de ma prise de poids et pour sa bonne humeur au quotidien. Je ne remercierais jamais assez ma moitié

Edwina, dont l'amour et le soutien indéfectible m'ont porté tout ce temps. Je n'aurais jamais eu l'optimisme et la force nécessaire pour arriver jusqu'au bout sans toi. Je veux remercier tout particulièrement Manu, avec qui les débats et les intenses réflexions ont été pour moi une source d'inspiration importante de la licence au doctorat, merci aussi pour ton soutient et les échanges fréquents durant ces derniers mois d'écriture. Je remercie aussi Marion pour ce temps passé à faire de la relecture et pour m'avoir, avec tant de pragmatisme, cloué à ma chaise au lieu de profiter du soleil. Merci aussi à Gaël pour tout, et surtout pour rien ! Je voudrais remercier aussi ma famille Françoise, Nanie, Chloé, Audrey et Pascal qui ont toujours eu foi en moi et m'ont aidé à avancer avec détermination. Je voudrais remercier Smirchat de me supporter depuis toutes ces années malgré mon manque cruel de coordination. Comme dirait l'autre « 5,4,3 SPIKE !.. »

Je pense aussi à Q et Mika, même si nos contacts sont moins fréquents. Depuis toujours, votre amitié m'a aidé à me forger et à définir l'homme que je veux être. Mika, j'espère que tu as encore la tête dans les nuages, au sens propre. Q, Bonne chance pour ton propre manuscrit, à l'heure où j'écris ces lignes, tu es toi aussi en pleine rédaction.

Résumé

Les méthodes de compréhension de la parole visent à extraire des éléments de sens pertinents du signal parlé. On distingue principalement deux catégories dans la compréhension du signal parlé : la compréhension de dialogues homme/machine et la compréhension de dialogues homme/homme. En fonction du type de conversation, la structure des dialogues et les objectifs de compréhension varient. Cependant, dans les deux cas, les systèmes automatiques reposent le plus souvent sur une étape de reconnaissance automatique de la parole pour réaliser une transcription textuelle du signal parlé. Les systèmes de reconnaissance automatique de la parole, même les plus avancés, produisent dans des contextes acoustiques complexes des transcriptions erronées ou partiellement erronées. Ces erreurs s'expliquent par la présence d'informations de natures et de fonction variées, telles que celles liées aux spécificités du locuteur ou encore l'environnement sonore. Celles-ci peuvent avoir un impact négatif important pour la compréhension. Dans un premier temps, les travaux de cette thèse montrent que l'utilisation d'autoencodeur profond permet de produire une représentation latente des transcriptions d'un plus haut niveau d'abstraction. Cette représentation permet au système de compréhension de la parole d'être plus robuste aux erreurs de transcriptions automatiques. Dans un second temps, nous proposons deux approches pour générer des représentations robustes en combinant plusieurs vues d'un même dialogue dans le but d'améliorer les performances du système la compréhension. La première approche montre que plusieurs espaces thématiques différents peuvent être combinés simplement à l'aide d'autoencodeur ou dans un espace thématique latent pour produire une représentation qui augmente l'efficacité et la robustesse du système de compréhension de la parole. La seconde approche propose d'introduire une forme d'information de supervision dans les processus de débruitages par autoencodeur. Ces travaux montrent que l'introduction de supervision de transcription dans un autoencodeur débruitant dégrade les représentations latentes, alors que les architectures proposées permettent de rendre comparables les performances d'un système de compréhension reposant sur une transcription automatique et un système de compréhension reposant sur des transcriptions manuelles.

Table des matières

1 Introduction **11**
- 1.1 Contexte général . . . 11
- 1.2 Présentation de l'entreprise Orkis . . . 12
- 1.3 Compréhension de la parole . . . 12
  - 1.3.1 Les interactions homme/machine . . . 13
  - 1.3.2 Les interactions homme/homme . . . 14
- 1.4 Problématique . . . 16
- 1.5 Organisation du manuscrit . . . 17

I État de l'art sur les représentations de données textuelles et les réseaux de neurones artificiels **19**

2 Représentations des documents textuels pour le traitement automatique **21**
- 2.1 Introduction . . . 21
- 2.2 Les opérations de prétraitement . . . 22
- 2.3 Les modèles en sac-de-mots . . . 23
- 2.4 Les modèles de thématiques latentes . . . 26
  - 2.4.1 L'analyse sémantique latente (LSA) . . . 26
  - 2.4.2 L'allocation latente de Dirichlet (LDA) . . . 28
- 2.5 Un changement de paradigmes : Les modèles *d'embeddings* de mots . . . 31
- 2.6 Conclusion . . . 33

3 Réseaux de neurones artificiels profonds **35**
- 3.1 Introduction . . . 35
- 3.2 Les origines des réseaux de neurones artificiels . . . 36
- 3.3 Les réseaux stochastiques à base d'énergie . . . 39
- 3.4 Les réseaux *feedforward* . . . 42
- 3.5 Les réseaux récurrents . . . 45
- 3.6 Les méthodes multiarchitectures . . . 49
  - 3.6.1 La fonction cout classification temporelle connexionniste (CTC) . . . 49
  - 3.6.2 La fonction d'activation relu . . . 50
  - 3.6.3 Des algorithmes de descente de gradient adaptatifs . . . 50
  - 3.6.4 Les méthodes de régularisation . . . 51

3.7 Conclusion 53

II Contribution : Projection neuronale d'informations pour la classification thématique de documents parlés 55

4 Problématique et cadre expérimental 57
4.1 Introduction 57
4.2 Tâche d'Identification de Thématique (TI) 58
4.3 Les systèmes de référence 61
4.4 Conclusion 65

5 Représentations distribuées des mots pour la compréhension de documents bruités 67
5.1 Introduction 67
5.2 Les Architectures Word2vec 69
5.3 Intégration de l'information de position 71
5.3.1 Le CBOW avec pondération (LL-CBOW) 72
5.3.2 Le Skip-gram avec pondération (LL-SG) 75
5.4 Analyses des modèles proposés et application à la compréhension de la parole 77
5.5 Conclusion 83

6 Utilisation d'autoencodeurs empilés pour le débruitage non supervisé de documents parlés 85
6.1 Introduction 85
6.2 Fonctionnement des autoencodeurs 87
6.2.1 Les concepts fondamentaux 87
6.2.2 Autoencodeur débruitant (DAE) 89
6.2.3 Autoencodeur empilé (SAE) 90
6.3 Protocole expérimental 91
6.4 Résultats et discussion 93
6.4.1 Analyse des autoencodeurs simples 93
6.4.2 Analyse des autoencodeurs profonds 94
6.5 Conclusion 95

III Contribution : Exploiter plusieurs visions d'un document : des réseaux de neurones multivues 97

7 Représentations multivues par compression pour la classification thématique de documents 99
7.1 Introduction 99
7.2 Modèle *Author-Topic* (AT) 100
7.3 Systèmes multivues : compresser et combiner l'information 103
7.3.1 Compression de l'information en c-vecteurs 104

7.3.2 Compression avec un autoencodeur débruitant (DAE) 104
7.3.3 Compression avec des sous-espaces thématiques latents (LTS) . . . 106
7.4 Classification à l'aide de la distance de Mahalanobis 106
7.5 Protocole expérimental 107
7.6 Résultats 109
7.6.1 Résultats des représentations AT. 109
7.6.2 Approches multivues sur une tâche de classification de documents parlés 110
7.6.3 Approche multivues sur une tâche de classification de documents écrits 113
7.7 Conclusion 116

8 Supervision de l'apprentissage des autoencodeurs pour améliorer la compréhension de la parole. 119
8.1 Introduction 119
8.2 Autoencodeur profond supervisé (SDAE) 122
8.2.1 Intérêt du SDAE 122
8.2.2 Particularités de l'entrainement du SDAE 123
8.3 Expériences 124
8.3.1 Protocole expérimental 124
8.3.2 les DAE simples 124
8.3.3 Le SDAE 125
8.3.4 Le FSDAE 125
8.4 Résultats 125
8.5 Discussion 128
8.6 Conclusion 129

9 Utilisation de caractéristiques supervisées dans un autoencodeur hétérogène. 131
9.1 Introduction 131
9.2 Autoencodeur débruitant avec caractéristiques spécifiques à la tâche (TDAE) 133
9.2.1 Spécificités du TDAE 133
9.2.2 Méthode d'entrainement du TDAE 135
9.3 Expérimentations 135
9.3.1 Perceptrons multicouches homogènes 136
9.3.2 TDAE 136
9.3.3 MLP-AE 136
9.3.4 Analyse en composantes principales (PCA) 136
9.4 Résultats expérimentaux 137
9.5 Conclusion 138

10 Bilan et significativité 141
10.1 Confusion et Significativité 141
10.2 Comparaison globale 144

11 Conclusion et perspectives 149

11.1 Bilan des contributions 150
11.2 Perspectives de ces travaux 152

Liste des illustrations **153**

Liste des tableaux **157**

Bibliographie **159**

Bibliographie personnelle **183**

Annexes **185**

A Les Références de la frise chronologique **187**

B Apprentissage par Gradient Tree Boosting **189**
B.1 Les arbres de décisions 189
B.2 Méthode de boosting 190

Chapitre 1

Introduction

1.1 Contexte général

Les vidéos sont devenues le premier média sur internet. Le nombre de vidéos vues et publiées chaque jour est en constante augmentation. Les experts estiment [1] que 75% du trafic internet transporte de la vidéo. Ce média est devenu accessible au grand public en grande partie grâce à l'essor de plateformes de publication en ligne, comme YouTube, Dailymotion et Viméo, mais aussi à la démocratisation des *smartphones* qui ont rendu la consommation et la création de vidéos de plus en plus facile. Les données vidéos sont très employées à des fins professionnelles, comme les films promotionnels qui sont aujourd'hui un outil majeur des régies publicitaires et des services markétings. Le marché montant des MOOC [2] repose sur l'exploitation de vidéos, elles sont aussi utilisées par les marques traditionnelles pour communiquer avec leurs clientèles, par exemple pour proposer des manuels d'entretien sous forme de vidéos. Aujourd'hui, près de 400 heures de vidéo sont déposées sur la plateforme YouTube chaque minute [3]. Ces quantités de plus en plus importantes font de l'organisation, de la structuration et de la manipulation de contenu un véritable challenge pour les systèmes d'information. Dans une vidéo, deux médias distincts sont principalement porteurs d'information. Le premier média est celui des images. Récemment, les systèmes d'annotations d'images se sont nettement améliorés (Karpathy et al.) ; le second média est celui de l'audio. Le domaine de la compréhension de la parole propose un cadre de recherche pour extraire et traiter l'information parlée de documents audios. Dans cette thèse, nous abordons cette problématique sous l'angle particulier de la classification thématique de documents parlés, par une approche basée sur le traitement de la parole. Ce travail est réalisé en collaboration avec l'entreprise Orkis dans le cadre d'un financement CIFRE.

Ce chapitre est organisé de la manière suivante : la Section 1.2 présente brièvement l'entreprise Orkis et ses motivations quant au financement d'une thèse CIFRE. La Sec-

1. http ://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/complete-white-paper-c11-481360.html
2. Cours en ligne ouvert à tous.
3. http ://tubularinsights.com/youtube-300-hours/

tion 1.3 présente de manière générale le domaine de recherche de compréhension de la parole. La Section 1.4 situe la problématique principale abordée dans ces travaux. Enfin l'organisation du manuscrit est détaillée dans la Section 1.5.

1.2 Présentation de l'entreprise Orkis

Orkis est une entreprise française fondée en 1991. Son siège social est situé à Aix-en-Provence. Elle est spécialisée dans la gestion de ressources numériques (Digital assets management, DAM). Pionnière française dans la gestion de photothèques, elle propose depuis plusieurs années des solutions modernes pour la gestion de vidéos professionnelles. Ces vidéos sont de types très variés. On peut trouver, par exemple, des enregistrements de réunions, des émissions de télévision, des enregistrements d'essais produits, des témoignages clients, des conférences de presse, etc. La compréhension de la parole est devenue un axe de développement stratégique pour Orkis. En effet, les fruits de la recherche dans ces domaines intégrés à une solution de DAM mature et éprouvée, lui permettent de proposer des services visant à simplifier et automatiser le traitement et l'indexation de documents audios et vidéos. Ces services apportent un avantage concurrentiel considérable dans le marché actuel du DAM. C'est dans ce contexte, influencé par les besoins applicatifs en traitement de la parole de la société Orkis, que les travaux présentés dans ce manuscrit sont abordés.

1.3 Compréhension de la parole

La **compréhension de la parole (Spoken language understanding, SLU)** est l'action de rechercher et relier dans le signal de parole des éléments de sens. C'est une problématique particulièrement difficile, car le signal transporte des informations de natures et de fonction variées, telles que l'ambiance, le locuteur ou le bruit environnemental , etc. Pour traiter cette variabilité, les systèmes de SLU reposent généralement sur un composant de reconnaissance automatique de la parole, qui produit des transcriptions potentiellement imparfaites. Pour des raisons expérimentales, afin de se prémunir des particularités de la transcription automatique et se concentrer sur les propriétés particulières du langage parlé, certains travaux reposent sur des transcriptions manuelles (Turmo et al., 2008). Une approche récente consiste à ne pas utiliser de transcriptions, mais seulement les motifs récurrents dans le signal (Lee et al., 2015a). Pour rester les plus proches des cas applicatifs réels, nous nous concentrons uniquement sur des méthodes qui utilisent une transcription automatique.

Le domaine de recherche de compréhension de la parole est devenu très actif à partir des années 90, en partie grâce au financement de projets par une agence du département de la Défense des États-Unis, chargée de la recherche et du développement des nouvelles technologies destinées à un usage militaire (Defense Advanced Research Projects Agency, DARPA). Ces projets de recherche ont rendu possible le développement de grands corpus de données, par exemple, la production du corpus ATIS (Air travel

Information Service) (Price, 1990). Ce corpus a permis aux chercheurs d'évaluer et de comparer leurs systèmes automatiques de compréhension du langage. L'évaluation des systèmes automatiques est réalisée en mesurant leurs performances sur une tâche appelée SLU (comme le domaine puisqu'elle en est la première incarnation). Dans cette tâche, le système doit reconstruire le scénario de vol, en s'aidant des informations de vols contenues dans une base de données, sachant un ensemble de phrases décrivant un trajet en avion. Les premières approches employées pour aborder ce problème s'appuyaient sur des règles de grammaire (Seneff, 1992),(Ward et Issar, 1994),(Dowding et al., 1994) construites manuellement pour analyser ces phrases et en déduire des informations définies à l'avance, telles que l'aéroport de départ, l'aéroport d'arrivée ainsi que l'heure de départ. Aujourd'hui, les approches statistiques dominent cette thématique (Schwartz et al., 1997),(Wang et Acero, 2006),(Mesnil et al., 2015). Des approches hybrides ont aussi été étudiées (Wang et al., 2002). Le sens du terme SLU s'est ensuite largement élargi (De Mori et al., 2008). Il ne fait plus simplement référence à une tâche unique, mais à un ensemble de tâches qui partagent l'objectif d'extraire des informations sémantiques du signal de parole. Ces tâches peuvent être divisées en deux catégories : la compréhension de la parole pour l'interaction homme/machine et la compréhension de la parole pour les interactions homme/homme.

1.3.1 Les interactions homme/machine

Dans cette catégorie, nous appelons le système automatique l'agent. Dans les situations les plus simples, l'utilisateur humain est en interaction directe avec l'agent par la parole, au moins dans le sens *humain* $\rightarrow$ *agent*. L'agent traite l'interaction avec l'utilisateur et réalise une action en conséquence. Le champ des actions possibles est large. Il peut répondre oralement, afficher un retour sur un écran ou encore réaliser une action telle que envoyer un message, prendre un rendez-vous, acheter un billet d'avion, etc. Dans le cadre d'interactions homme/machine, le format des échanges et le type de réponse permettent de considérer des sous-groupes de tâche de compréhension du langage.

Les systèmes de recherche vocale (Voice search, VS)
Dans ce type d'application, l'utilisateur envoie une requête parlée, dans un cadre fonctionnel précis. Le système doit répondre par une information structurée extraite d'une base de données (Wang et al., 2008). Les services de renseignements téléphoniques automatiques sont des exemples classiques d'application de VS . L'utilisateur appelle le service parce qu'il cherche une adresse ou un numéro de téléphone. Pour résoudre les problèmes de VS, les approches usuelles reposent sur la composition d'un système de reconnaissance automatique de la parole (SRAP) et d'un moteur de recherche d'information(Information Retrieval, IR) (Natarajan et al., 2002),(Wang et al., 2008).

Les systèmes de réponses aux questions parlées (Spoken Question Answering, SQA)
Ces tâches sont une évolution des VS. Dans celles-ci, l'utilisateur pose une question à l'oral en langage naturel et le système génère une réponse lisible à partir d'une collection de données non structurées. Les systèmes de référence reposent principalement sur

des approches issues des recherches en réponse aux questions textuelles (Bernard et al., 2009),(Comas et al., 2010).

Les systèmes de dialogue (Dialogue system, DS)
Ces systèmes regroupent des éléments qui dépassent le cadre de la pure compréhension du langage. Un DS est composé d'au moins un SRAP, un gestionnaire de dialogue, et un système de génération de parole. Contrairement aux documents parlés employés par les méthodes déjà présentées, un dialogue est composé de plusieurs tours de parole. Dans ce contexte interactif, l'agent peut demander des confirmations ou des informations supplémentaires quant à la requête de l'utilisateur. Ce procédé ouvre des champs applicatifs plus larges, mais demande en même temps au système de gérer des complexités supplémentaires. Parmi elles, on trouve la séquentialité des interactions, ainsi que la possibilité que le but de l'utilisateur et la base de connaissances de l'agent évoluent en fonction du temps. Les premiers systèmes de dialogue utilisent des graphes pour organiser l'historique du dialogue (McTear, 1998). Les systèmes état de l'art actuel utilisent des méthodes statistiques (Williams, 2008).

Les interactions homme/machine ont la particularité de contraindre l'interaction à la forme de requêtes-réponses. Dans ces modes de fonctionnement, l'information parlée est courte et peu contextualisée. Il est important de noter que la recherche sur les DS tend à généraliser ce mode de fonctionnement. Certaines tâches (Kim et al., 2017) utilisent des données plus contextualisées, issues de conversations entre humains, pour l'entrainement de système de dialogue homme/machine. Malgré cela, la forme contrainte des interactions reste la principale différence entre les systèmes de dialogue homme/machine et les systèmes de traitement automatique de données parlées dans un contexte d'interaction homme/homme.

1.3.2 Les interactions homme/homme

Les récents progrès en reconnaissance de la parole et en compréhension des interactions homme/machine ont ouvert la porte à de nouvelles approches de la SLU. La parole est le principal moyen de communication des humains. Pourtant, l'exploitation des médias parlés a donné lieu à assez peu d'applications grands publics. Depuis les années 90, de nombreux corpus de données ont été collectés. Ces corpus ont permis à des tâches de SLU, dans le cadre d'interactions entre humains, d'être intensivement étudiées. Ces tâches peuvent être différenciées en plusieurs groupes. Les principaux sont présentés ci-dessous.

La reconnaissance d'entités nommées (Name entity recognition, NER)
La NER consiste à rechercher un segment de document (dans ce cas parlé), qui fait référence à un ensemble de catégories prédéfinies. Ces catégories peuvent être des noms de personnes, des noms de lieux, des dates, des valeurs, des unités, des noms d'entreprises, etc.
La NER, dans le cadre de la parole repose sur les mêmes méthodes que dans le cadre textuel, soit le plus souvent une combinaison entre une grammaire *ad hoc* et un modèle statistique (Favre et al., 2005),(Chieu et Ng, 2002). Cependant, les erreurs de trans-

cription automatique ont un impact fort sur les résultats de ces systèmes (Béchet et Charton, 2010). La robustesse du système est donc un critère important pour répondre à une tâche de NER (Favre et al., 2005), dans un contexte d'interactions parlées.

La recherche de documents parlés (Spoken content retrieval, SCR). Cette tâche consiste à retrouver et à soumettre à l'utilisateur des documents parlés pertinents, qui correspondent à un besoin exprimé, sous la forme d'une requête structurée ou en langage naturel. Ce domaine est à mi-chemin entre les thématiques de recherche d'informations (RI) et la SLU. En effet, la résolution des problématiques de SCR nécessite des moyens d'indexation et de recherche d'informations dans des transcriptions automatiques. Le système doit donc faire face aux particularités liées au traitement des données transcrites automatiquement, comme la présence importante d'erreurs ou d'hypothèses multiples, etc. Les premières solutions proposées pour la SCR reposent sur de la détection de mots dans des hypothèses de phonèmes, pour compenser le faible vocabulaire des SRAP (Glavitsch et Schäuble, 1992). Avec l'amélioration des SRAP maintenant capables de supporter un large vocabulaire, ces méthodes ont ensuite été combinées aux prédictions d'un SRAP (Jones et al., 1996). La campagne d'évaluation TREC (Garofolo et al., 2000) a créé une dynamique importante pour la recherche autour de données issues de journaux télévisés et radiophoniques. Les données ne provenant pas d'un contexte aussi contrôlé, en particulier avec des documents courts, rendent le fonctionnement des systèmes de SCR beaucoup plus sensible à l'utilisation d'un SRAP. Des stratégies ont été proposées pour pallier ce problème. Récemment, des représentations thématiques de transcriptions sont proposées pour augmenter la robustesse aux erreurs des systèmes automatiques. Cette stratégie permet d'obtenir de meilleurs résultats que les représentations classiques, sur ces tâches plus complexes (Hu et al., 2009),(Chen, 2009),(Hazen).

L'Identification de thématique (Topic identification, TI) La TI consiste à retrouver le sujet (ou thématique) principal abordé dans un segment audio. Dans une tâche de TI, les segments sont thématiquement homogènes, ou dominés par un sujet principal et la liste des thématiques possibles est préalablement déterminée (Hazen, 2011). (Gorin et al., 1997) proposent un système de routage d'appels qui correspond à la fois à un agent de DS et à un système de TI. Dans leur cas, le système dirige l'appel de l'utilisateur vers le service adapté, en fonction du thème détecté dans l'énoncé du problème et par les confirmations de l'utilisateur. Les premiers travaux en TI (Fiscus et al., 1999) traitent un corpus d'émissions radiodiffusées et télédiffusées où les erreurs sont rares et dont le traitement est proche des transcriptions manuelles. L'approche usuelle, pour résoudre cette tâche consiste à extraire des informations sémantiques à partir des cooccurrences des mots produits automatiquement. Elle repose sur la projection des transcriptions dans des espaces vectoriels, pour extraire de l'information sémantique. Les espaces de projections abordés ci-dessous sont décrits plus en détails dans le Chapitre 2. Une majorité des travaux dans ce domaine emploie la représentation TF.IDF. Par exemple, (Gemello et al., 2011) ont construit cette représentation à partir d'un ensemble d'hypothèses de mots produit par un SRAP. Celle-ci est ensuite classifiée automatiquement pour déterminer la bonne thématique. Dans (Wintrode et Kulp, 2009), une pondération particulière est rajoutée aux scores TF.IDF, pour tenir compte de la confiance accordée aux mots par le SRAP, afin de construire un système robuste. D'autres travaux utilisent

des projections des transcriptions dans des espaces sémantiques ou conceptuels, pour compenser les erreurs de transcription automatique. Dans les travaux de (Lane et al., 2007), les documents sont projetés dans un espace de concepts pour réduire la dimension des représentations et filtrer l'information utile avant le processus de classification automatique. Plus récemment, une méthode pour créer des ensembles conceptuels latents (LDA) a été utilisée pour créer une représentation robuste (Tür et al., 2008), (Morchid et al., 2014b). Une approche alternative consiste à baser l'identification de la thématique sur les hypothèses phonétiques produites, par le système de transcription automatique, pour renforcer l'information acoustique et être ainsi moins affecté par les erreurs de transcription. Cette méthode est appliquée parfois directement (Hazen et al., 2007) ou en construisant des espaces de concepts de phonèmes (Bost et al., 2013). Toutes ces méthodes ont permis de créer des représentations de plus en plus robustes. Elles restent néanmoins étroitement liées aux performances du SRAP lorsqu'elles traitent des mots ou perdent l'information lexicale quand elles emploient les hypothèses phonétiques.

Nous avons présenté dans cette section des tâches fréquemment documentées dans la littérature ainsi que les différentes solutions proposées pour aborder ces problématiques. On peut remarquer que l'amélioration des performances des SRAP a permis de rapprocher les approches de SLU et du traitement du langage écrit. En revanche, les particularités des données parlées nécessitent l'utilisation de méthodes robustes dédiées.

1.4 Problématique

Les orientations de la société Orkis font de la robustesse des méthodes de compréhension et de structuration de documents parlés l'axe principal de cette thèse. En effet, leurs cadres applicatifs impliquent un contrôle très faible sur le type de parole, le vocabulaire employé ainsi que les conditions acoustiques. Même si des solutions sont mises en place pour restreindre ces variabilités, les risques que le système automatique rencontre un segment audio, avec une importante quantité d'erreurs de reconnaissance automatique de la parole, sont importants.

L'objectif attendu est de créer une représentation des données, avec lesquelles les performances du système de SLU utilisant les transcriptions automatiques, soient très proches des performances obtenues à partir de transcriptions manuelles. Pour réaliser cet objectif, nous étudierons deux pistes principales :

— Créer des espaces de projections des données homogènes réduisant l'impact des erreurs.

— Exploiter plusieurs vues d'un même document pour augmenter la robustesse du système en environnement bruité.

Les propositions qui en découlent seront évaluées dans le cadre de la classification thématique de transcriptions de documents audios.

1.5 Organisation du manuscrit

Dans la Partie I, nous présentons les cadres théoriques nécessaires pour aborder les contributions de ces travaux. Dans un premier temps, les méthodes de représentation des données issues d'informations textuelles ou transcrites sont introduites. Ensuite, les réseaux de neurones artificiels profonds, leurs nombreuses architectures et leurs propriétés respectives sont décrits. La Partie II est dédiée aux propositions à base de représentations latentes neuronales, n'utilisant qu'une seule vue du document pour le traitement de données issues de SRAP. Enfin, la partie III détaille les propositions qui utilisent des combinaisons multivues, pour réduire les effets des erreurs de transcription.

Première partie

État de l'art sur les représentations de données textuelles et les réseaux de neurones artificiels

Chapitre 2

Représentations des documents textuels pour le traitement automatique

Sommaire

2.1 Introduction 21
2.2 Les opérations de prétraitement 22
2.3 Les modèles en sac-de-mots 23
2.4 Les modèles de thématiques latentes 26
2.5 Un changement de paradigmes : Les modèles *d'embeddings* de mots 31
2.6 Conclusion 33

2.1 Introduction

Un document textuel sous sa forme brute est difficilement exploitable par un système de classification automatique. Pour pallier ce problème, il est nécessaire d'ajouter une étape préalable d'indexation qui va lier le texte à une représentation exploitable normalisée appelée **modèle vectoriel**. Ces représentations sont construites selon un niveau de granularité, en associant un vecteur à chaque mot, ensemble de lettres (character ngram) (Mcnamee et Mayfield, 2004) phrase (Khoo et al., 2006) ou encore document (Salton et Buckley, 1988), etc. Le niveau de granularité est intrinsèquement lié à la tâche traitée. L'utilisation de ces espaces permet de calculer facilement la similarité entre deux documents. L'extraction des descripteurs qui composent l'espace vectoriel est un élément crucial de tout système de traitement automatique du langage naturel. Le choix de ceux-ci est basé le plus souvent sur des intuitions linguistiques ou des connaissances liées à la tâche traitée. Chaque nouvelle application nécessite donc à nouveau un travail

d'analyse pour déterminer les procédures à appliquer et les informations adéquates à extraire. Le reste du chapitre présentera les représentations vectorielles utiles pour la suite du manuscrit. La Section 2.2 présente les opérations de prétraitement usuelles. Ensuite, la Section 2.3 introduit les représentations fréquentistes. Puis, la Section 2.4 présente les modèles thématiques. Enfin, la Section 2.5 aborde les *embbeding* pour la représentation des mots.

2.2 Les opérations de prétraitement

Le prétraitement de données textuelles est constitué d'un ensemble d'étapes de segmentation, d'uniformisation et de nettoyage du texte. La première étape de prétraitement indispensable est celle de *tokenisation* (Grefenstette et Tapanainen, 1994). Dans cette étape un texte est segmenté en unités élémentaires dites token. Un token peut être un mot ou une séquence de mots. Cette étape n'est pas complexe techniquement en français et en anglais où les mots sont le plus souvent séparés naturellement par la présence d'un espace ou d'une ponctuation. Cependant, le choix des règles de tokenisation impacte fortement les performances finales et nécessite une expertise sur les données traitées.
Ensuite, un processus de normalisation est appliqué. En effet, après tokenisation, il persiste des situations ambigües qui impactent le système et nécessitent une prise de décision (Grefenstette et Tapanainen, 1994). Parmi elles, on trouve par exemple le traitement des nombres qui offre au moins 3 possibilités : la première solution consiste à regrouper sous une même balise (<NOMBRE>) les nombres rencontrés. Ce procédé facilite la généralisation et réduit le vocabulaire des documents en contrepartie d'une perte d'information. La seconde méthode décompose les nombres en chiffres, laissant ainsi l'interprétation à la suite de traitement automatique. La troisième solution conserve la forme du nombre intacte, elle conserve ainsi toute l'information. En contrepartie, la prise en compte de celle-ci lors de calcul de similarité interdocuments est plus complexe. Des décisions similaires sont prises lors du traitement de la ponctuation, de dates, de liens hypertextes, d'acronymes, de fautes d'orthographe, de mots composés, d'émojis, etc.
Ensuite, tous les mots vides peuvent être supprimés du corpus. Les mots vides sont l'ensemble des mots très fréquents qui sont peu porteurs d'information sémantique, par exemple les déterminants ou les articles, etc. .
Enfin, des étapes de *racinisation (stemming)* ou de *lemmatisation* peuvent être employées. Ces deux prétraitements suppriment les flexions des mots, réduisent le vocabulaire à traiter et augmentent les occurrences des mots pour faciliter leur généralisation par le modèle appris.

L'algorithme de *racinisation (Hull et al.)* le plus commun est l'algorithme de Porter (Porter, 1980 ; Willett, 2006), initialement introduit pour l'anglais. Il a été adapté ensuite pour de nombreuses langues [1] (Willett, 2006). Cet algorithme supprime le préfixe et le suffixe de chaque mot pour en déduire une forme courte qui n'est pas un mot existant, la racine. Par exemple, la racine des mots *programme, programmation,*

1. http ://snowball.tartarus.org/

programmeur est *«programm»*.

L'autre méthode usuelle est la lemmatisation qui remplace chaque mot par son *lemme (ou forme canonique)* correspondant. Le plus souvent l'infinitif ou la forme au masculin singulier sont utilisés comme lemme. Le processus de lemmatisation est plus complexe que la racinisation parce qu'il nécessite des informations morphosyntaxiques pour déterminer la bonne forme à utiliser. Cependant cette méthode obtient de meilleurs résultats au prix d'une complexité supérieure (Fautsch et Savoy, 2009) à la racinisation.

Il est important de noter que l'augmentation de la taille des corpus textuels disponibles et les capacités de modélisation toujours plus importantes réduisent l'intérêt de ces méthodes. En particulier, les modèles thématiques (Schofield et Mimno, 2016) et les *embeddings* de mots (c.f. Chapitre 5) sont capables de déduire des relations, sémantiques ou grammaticales entre les flexions mots des documents.

2.3 Les modèles en sac-de-mots

La représentation la plus commune, introduite initialement en Recherche d'Information, est appelée représentation en sac-de-mots (bag-of-words, BOW) (Salton, 1989). Dans cette modélisation un vecteur est associé à chaque entité textuelle (par exemple une phrase, un paragraphe ou un document). Ces vecteurs sont de taille $|V|$ où V est l'ensemble des mots (termes) du vocabulaire connu du système. V contient la liste des mots qui apparaissent au moins une fois dans au moins un des documents du corpus utilisé. À chaque dimension du vecteur est donc associé un mot. Chacune d'elles contient un poids décrivant l'importance du mot dans cette entité. Le plus souvent, l'entité utilisée est le document. En associant un vecteur pondéré à chaque document d d'un corpus D, on construit une matrice appelée la matrice documents-termes M_t^d pour R documents avec un vocabulaire de taille $|V|$.

$$\mathbf{M}_t^d = \begin{pmatrix} p_{0,0} & p_{0,0} & \cdots & p_{0,v-1} & p_{0,v} \\ p_{1,0} & p_{1,0} & \cdots & p_{1,v-1} & p_{1,v} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ p_{R-1,0} & p_{R-1,0} & \cdots & p_{R-1,v-1} & p_{R-1,v} \\ p_{R,0} & p_{R,0} & \cdots & p_{R,v-1} & p_{R,v} \end{pmatrix} \tag{2.1}$$

Où $p_{i,j}$ correspond à la pondération du mot j (parmi V) dans le document i (parmi R). Les pondérations utilisées reposent sur le compte des occurrences de mots. Dans le cas le plus simple, la pondération est binaire, $p_{i,j} = 1$ quand le mot correspondant est présent dans le document et $p_{i,j} = 0$ sinon. Cette forme de pondération est appelée sac-de-mots binaire (Binary bag-of-words, BBOW). Dans le cas non binaire, la fonction de pondération standard dite "fréquence des termes et fréquence inverse de document" (Term Frequency-Inverse Document Frequency, TF.IDF) (Sparck Jones, 1972 ; Salton et Buckley, 1988) est utilisée. Cette fonction est composée de deux scores : la fréquence du terme (TF) qui dénote l'importance du mot dans le document et la fréquence inverse de

documents (IDF) qui indique à quel point un mot est commun. Le descripteur TF.IDF est calculé comme présenté dans l'équation suivante :

$$tf.idf(d_i, t_j) = \#(t_j, d_i).log\frac{R}{\{d_i \in D : t_j \in d\}} \tag{2.2}$$

où $\#(t_j, d_i)$ correspond au nombre d'occurrences du terme t_j dans le document d_i et $\{d \in D : t_j \in d\}$ est le nombre de documents dans lesquels le terme t_j apparait au moins une fois. Cette fonction est basée sur deux intuitions : plus un mot est fréquent dans un document, plus il est représentatif du contenu et plus un mot apparait dans un grand nombre de documents, moins celui-ci est discriminant. Plusieurs variantes du TF.IDF sont disponibles dans la littérature (Singhal et al., 1996 ; Salton et Buckley, 1988). Une normalisation est fréquemment utilisée en réponse aux défauts de la formulation initiale de l'équation 2.2 qui favorise les documents longs. Cette normalisation est définie dans l'équation 2.3.

$$\frac{tf.idf(d_i, t_j)}{\sqrt{\sum_{s=1}^{|V|}(tf.idf(d_i, t_s))^2}} \tag{2.3}$$

Avec cette représentation, chaque document du corpus est projeté dans un vecteur dont la taille est déterminée par le vocabulaire rencontré dans le texte utilisé lors de l'apprentissage, alors que seulement un sous-ensemble restreint de mots est présent dans chaque document. On génère donc de grands vecteurs creux, dont la taille complexifie les traitements et limite la capacité de généralisation. (Bellman, 1961 ; Bengio et al.). En réponse à cette faiblesse, des méthodes ont rapidement été introduites pour filtrer les mots les moins utiles et réduire la dimensionnalité des représentations (John et al.). Ce filtrage est réalisé via une fonction qui détermine l'importance des mots. Ainsi, seuls les mots de plus grande «importance» sont conservés.

La fonction d'importance la plus simple s'appuie sur l'IDF. Une méthode limitée, mais fréquente dans la littérature, est d'ignorer tous les mots n'apparaissant au moins x ($1 < x < 5$ selon la tâche) fois dans le corpus (Ittner et al., 1995),(Dumais et Chen, 2000),(Dagan et al., 1997),(Baker et McCallum, 1998). D'autres travaux ont montré qu'il est possible sur certaines tâches de ne conserver que 10% des mots les plus fréquents sans provoquer de dégradation trop importante des résultats (Yang et Pedersen). D'autres fonctions communes mesurant l'importance pour un terme t_j dans une classe c sont définies dans le Tableau 2.1 et présentées ci-dessous. La mesure d'information mutuelle compare la force de la probabilité de l'évènement joint (le mot j est présent et le document appartient à la classe i) à la probabilité des deux évènements indépendamment. Plus le rapport est grand (proche de 1) plus les évènements sont liés statistiquement et donc le mot j est utile pour trouver la classe i. La mesure du gain d'information est issue des travaux sur la catégorisation par des arbres de décision. Cette mesure est une combinaison de l'information mutuelle des valeurs possibles pour une variable catégorique et l'ensemble des catégories. Elle est utilisée principalement pour déterminer sur quelle variable l'arbre doit prendre une décision en priorité. De la même manière, elle peut permettre de choisir les termes influant pour la classification. La mesure d'impureté de

Table 2.1 – *Liste non exhaustive de fonctions d'importance pour le filtrage des mots où t_k est le terme du vocabulaire à évaluer en fonction de la classe c_i, $DF_{c_i}(t_k)$ est le nombre de documents associés à la classe c_i contenant le terme t_k et d est un facteur d'amortissement.*

Fonction		Forme mathématique
Fréquence de documents (DF)		$\frac{1}{idf}$
Gain d'information (IG)	(Caropreso et al., 2001) (Lewis, 1992)	$\sum_{c\in\{c,\bar{c}\}}\sum_{t\in\{t_j,\bar{t_j}\}} P(t,c)\cdot log\frac{P(t_j,c)}{P(t_j)\cdot P(c)}$
Impureté de Gini	(Dong et al.)	$\sum_{c\in\{c,\bar{c}\}}\left(\frac{DF(t_j)}{DF_c(V)}\right)^2$
Information mutuelle	(Dumais et al., 1998) (Lewis et Gale, 1994)	$\frac{P(t_j,c_i)}{P(t_j)\cdot P(c_i)}$

gini informe sur la dispersion d'un terme dans les différentes classes. Plus cette métrique est proche de 1 et plus le terme est dispersé dans les différentes classes. En sélectionnant les mots dont la dispersion est faible, nous sélectionnons les mots qui sont représentatifs d'une ou d'un petit groupe de classes en particulier.

En utilisant ces fonctions, la taille du vocabulaire peut être réduite de deux manières :

— Soit en conservant les b meilleurs mots.

— Soit en conservant tous les mots ayant un score supérieur à un seuil s.

Il est dans tous les cas nécessaire de déterminer b et s en fonction de la tâche. Les scores produits par ces métriques d'importance peuvent aussi être utilisés en combinaison des scores TF.IDF comme descripteur dans le sac-de-mots directement (Dong et al.).

Les représentations en sac-de-mots, basées sur des modèles statistiques, font toujours office de références quand il s'agit de traiter des données textuelles, à la fois pour leur performance et pour leur simplicité. Bien qu'elles soient historiquement les représentations les plus étudiées, elles ont plusieurs limites. Malgré les méthodes de réduction, le vocabulaire reste large et les représentations sont creuses (une majorité de dimensions à zéro). De plus, ces représentations ne modélisent dans les documents que les variables visibles dans les données d'apprentissage. Ainsi, les notions sous-jacentes de sémantique, de thème et les dépendances statistiques inter et intradocuments ne sont pas encodables dans ces représentations. Les informations multiples d'un terme comme la polysémie et les relations intermots comme la synonymie, ou l'appartenance à un même champ lexical sont ignorées. Pour répondre à ces pertes d'informations, différentes représentations ont été introduites reposant principalement sur des modèles thématiques ou neuronaux. La Section suivante présente les modèles thématiques les plus communs. Puis la Section 2.5 présente les modèles d'*embeddings* de mots.

2.4 Les modèles de thématiques latentes

Les modèles de thématiques latentes tentent de modéliser l'information qui n'est pas directement visible (latente) dans les documents à l'aide des distributions observées. Ces méthodes ont été introduites entre autres pour deux raisons principales : capturer l'information sémantique d'un corpus de documents et réduire la taille des représentations vectorielles. Les deux modèles les plus présents dans la littérature sont introduits dans cette section, l'analyse sémantique latente (latent semantic analysis, LSA) dans la Sous-Section 2.4.1 et l'allocation latente de Dirichlet dans la Sous-Section 2.4.2.

2.4.1 L'analyse sémantique latente (LSA)

La LSA est un paradigme formulé à l'origine sous le nom de *latent semantic indexing* (LSI) dans le cadre de la recherche d'information (Deerwester et al., 1990),(Berry et al., 1995),(Dumais, 1994),(Landauer et Dumais, 1997),(Foltz et Dumais, 1992) (Hofmann, 1999a),(Landauer et al., 1997),(Story, 1996). LSA décompose en valeurs singulières (Singular Value Decomposition, SVD) (Abdi et Williams, 2010), (Golub et Reinsch, 1970) la matrice documents-termes (cf. Section 2.3), pour en déduire trois matrices :

$$\mathbf{M}_t^d = U\Sigma V \tag{2.4}$$

La première matrice U, la matrice de vecteurs singuliers gauches, représente un espace de concepts (latents). Ces concepts sont composés d'un ensemble de mots pour en modéliser les relations. La seconde matrice Σ contient les valeurs singulières. Celle-ci nous informe de l'importance de chaque vecteur singulier pour sélectionner l'information à conserver. Enfin la troisième, V, contient la projection des documents dans l'espace de concepts. Ce fonctionnement est représenté dans la Figure 2.1

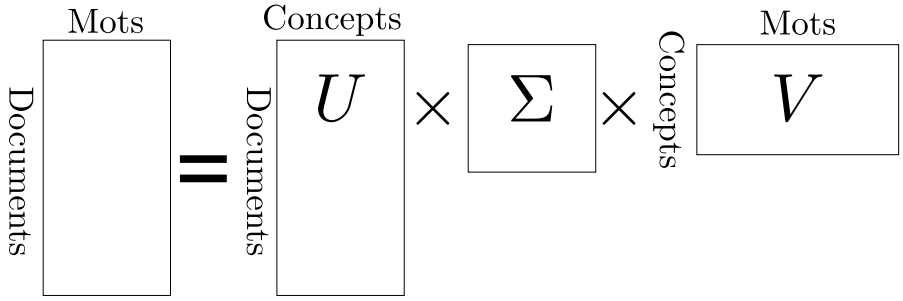


Figure 2.1 – *La décomposition matricielle de la méthode LSA*

La méthode LSA a été utilisée avec succès lors de nombreuses tâches de traitement automatique du langage, telles que la production de résumés automatiques (Yeh et al., 2005) ou la détection de mots hors vocabulaire (Lecouteux et al., 2009).
Elle est également employée dans d'autres domaines de recherche, comme le traitement de documents multimodaux (Pham et al., 2007) ou l'annotation d'images (Pham et al., 2007).

Cependant, l'approche LSA a deux limites importantes : l'interprétation des vecteurs singuliers et des concepts est compliquée ; la polysémie n'est pas modélisée. De plus, la méthode LSA ne construit pas de modèles génératifs. En effet, les modèles génératifs produisent de meilleures représentations dans de nombreuses situations (Baroni et al., 2014).
Pour dépasser ces limitations, une évolution probabiliste a été introduite : **l'Analyse sémantique latente probabiliste (Probabilistic latent semantic analysis, pLSA)** (Hofmann, 1999c,b). L'objectif de la pLSA est d'extraire l'information sémantique sous-jacente par le biais de l'information de cooccurrence des mots dans un cadre probabiliste. Pour modéliser cette information, pLSA utilise trois ensembles de variables aléatoires :

— les documents : $d_i \in D$, une variable aléatoire visible et D représente le corpus de documents.

— les mots : $t_j \in G$, une variable aléatoire visible avec G l'ensemble des mots du document.

— les concepts : $z \in T$, une variable latente (ou cachée) qui représente la structure sous-jacente de thématique. Pour faire référence à cette variable, on parle aussi de thèmes ou de *topics*.

Cette modélisation est représentée par le schéma en *Plate notation* dans la Figure 2.2. Cette notation expose les variables aléatoires et leur dépendance conditionnelle sous la forme d'un graphe. Les noeuds sur fond gris représentent les variables observées et les noeuds blancs les variables cachées inférées par le modèle. Les rectangles expriment les répétitions de variables ; le nombre de répétitions est indiqué en indice du rectangle (G et D).
La pLSA fait partie des modèles génératifs, elle réalise une approximation des distributions de probabilité qui aurait généré les données d'apprentissages observées (considérées comme des variables aléatoires). Ces modèles s'opposent aux modèles discriminants qui ne modélisent que la distribution jointe entre les variables cibles et la variable d'entrainement utile.

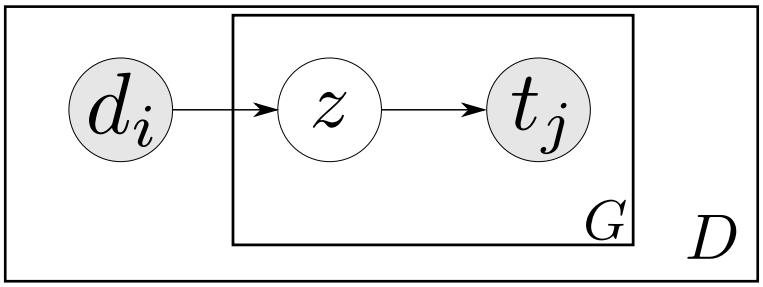


Figure 2.2 – *Le modèle pLSA sous forme de Plate notation.*

Pour faire le parallèle avec l'analyse factorielle du modèle LSA, on peut considérer les matrices de décompositions de l'équation (2.4) comme présentées dans les schémas 2.3.

La méthode pLSA a été utilisée lors de plusieurs tâches de traitement automatique du langage naturel écrit, parmi lesquelles, le résumé multidocuments (Hennig et Labor,

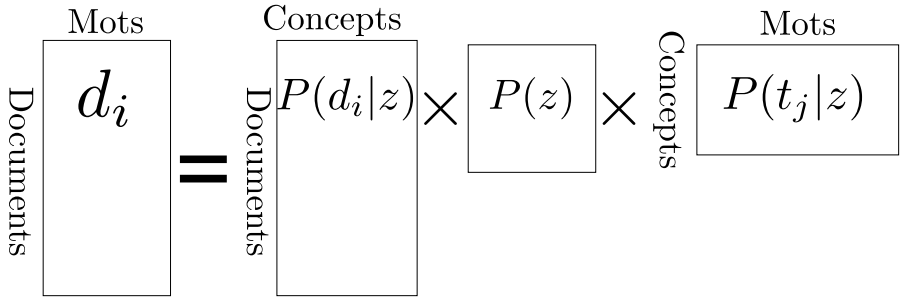


Figure 2.3 – *Le modèle pLSA d'un point de vue d'analyse factorielle*

2009), (Harashima et Kurohashi, 2010) ou la création de modèles de langue en reconnaissance automatique de la parole (Mrva et Woodland, 2004). Elle a été appliquée également dans des domaines connexes au traitement du texte, notamment pour la création de modèles acoustiques (Smaragdis et al., 2006), le traitement d'image (Bosch et al., 2006), de vidéo (Choudhary et al., 2008) et celui de données multimodales (Lienhart et al., 2009).

Malgré ces qualités, les modèles pLSA restent peu utilisés pour deux raisons principales :

— La formulation des modèles pLSA ne permet pas la projection dans les espaces de concepts de nouveaux documents.

— Le nombre de paramètres dans le modèle augmente linéairement avec le nombre de documents dans le corpus d'entrainement, ce qui implique que la pLSA est sensible au surapprentissage.

Ces limitations sont pénalisantes dans un cadre applicatif réaliste.

2.4.2 L'allocation latente de Dirichlet (LDA)

L'introduction de l'allocation latente de Dirichlet (latent Dirichlet allocation, LDA) (Blei et al., 2003 ; Blei, 2012) apporte des solutions aux limitations de pLSA : le nombre de paramètres du système est fixé par le nombre de concepts modélisables lors de l'apprentissage et LDA est conçue pour permettre de modéliser des nouveaux documents (Griffiths et Steyvers, 2004 ; Heinrich ; Blei et Lafferty, 2009)

La Figure 2.4 présente LDA sous forme de *plate notation*. On peut y voir les différentes variables aléatoires qu'elle modélise, avec θ qui modélise la distribution multinomiale suivie par les concepts z dans les documents et Φ qui modélise la distribution multinomiale suivie par les mots sur les concepts. Φ et θ suivent une distribution à priori tirée par concept et par document respectivement, selon des processus de Dirichlet de paramètres β et α respectivement.

Les modélisations réalisées par une LDA sont donc dépendantes des trois variables déterminées à priori : α, β et K. Les deux premières β et α influent sur les processus de Dirichlet. Plus elles sont grandes, plus les distributions générées sont proches d'une distribution uniforme ; lorsqu'elles sont proches de 0, les distributions sont au contraire

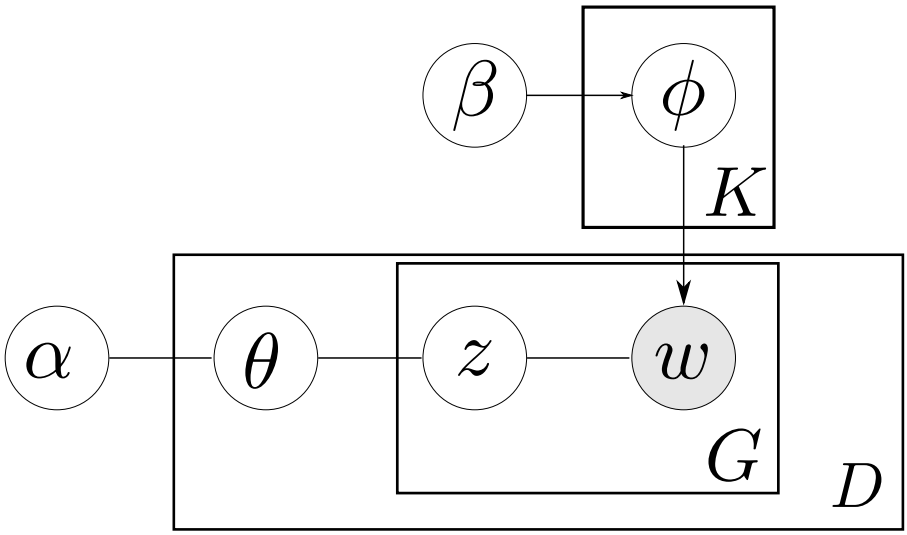


Figure 2.4 – *Le modèle LDA sous forme de Plate notation.*

plus éparses. Pour obtenir des documents orientés thématiquement et des concepts très différents, il est d'usage d'avoir β et α proches de 0 . Le troisième paramètre K est le nombre de concepts à modéliser. Il influe sur la complexité calculatoire du modèle ainsi que sur le niveau de spécificité (granularité) des concepts appris.

L'estimation des paramètres du modèle selon un corpus d'apprentissage donné est complexe. Des algorithmes d'apprentissage ont été introduits pour répondre à cette problématique. Ils sont pour la plupart des optimisations ou des variations de deux algorithmes principaux. Le premier est basé sur les méthodes d'inférences variationnelles (Blei et al., 2003) et le second est une méthode d'échantillonnage inspirée des chaines de Markov appelées *Gibbs samplings* (Griffiths et Steyvers, 2002). Certaines variations sont introduites pour répondre à des besoins applicatifs particuliers, par exemple pour gagner en vitesse d'apprentissage (Asuncion et al., 2009 ; Teh et al., 2006 ; Porteous et al., 2008)ou distribuer l'apprentissage sur plusieurs machines (Newman et al., 2007). D'autres exploitent au mieux la puissance de calcul des cartes graphiques (Yan et al., 2009) ou permettent de traiter les documents pour mettre à jour le modèle au fur et à mesure de leurs disponibilités au lieu de considérer un corpus fini pour l'apprentissage.

Durant cet apprentissage, deux distributions de probabilités sont apprises : $P(\phi|\beta)$ et $P(\theta|\alpha)$. Elles modélisent la distribution des mots dans les topics et des topics dans les documents selon un corpus d'apprentissage. L'analyse de ces distributions permet l'interprétation de la modélisation apprise par LDA. En effet, $P(\phi|\beta)$ estime les mots les plus importants pour un concept permettant ainsi d'interpréter un sens ou un topic associé au concept. De son côté, $P(\theta|\alpha)$ renseigne sur la répartition des concepts dans les documents, ce qui permet d'interpréter les topics.

Les modèles LDA ont été introduits pour le traitement automatique du langage écrit. (Wei et Croft, 2006a) formulent une représentation des documents à base de LDA pour la recherche de documents. Une méthode de retour de pertinence et une méthode d'exten-

sion de requêtes (Deveaud et al., 2013) sont introduites pour une formulation similaire. L'approche LDA a été utilisée pour exploiter l'information du corpus *freebase*[2] dans l'objectif d'obtenir d'excellents résultats sur une tâche de recherche d'entités nommées dans un corpus de *tweets* (Ritter et al., 2011). Elle a également été employée pour des tâches de résumé automatique de documents par extraction, où la répartition des phrases dans les topics peut être exploitée pour déterminer les phrases les plus pertinentes pour le résumé (Arora et Ravindran, 2008). C'est aussi le cas en analyse de sentiments (Li et al., 2010), où LDA permet de déterminer des topics négatifs et positifs caractérisant un commentaire produit.
L'impact fort des méthodes LDA en traitement automatique du langage a entrainé leur exploitation dans d'autres domaines de recherche. Des travaux ont par exemple, employé LDA pour analyser et chercher des erreurs dans du code source (Lukins et al., 2008 ; Rama et al., 2008 ; Zibran, 2016).
LDA est également utilisée en traitement automatique de la parole, par exemple pour déterminer des domaines acoustiques et entrainer des modèles adaptés en conséquence (Doulaty et al., 2015 ; Lecouteux et al., 2009 ; Sheikh et al., 2016). Elle permet également d'extraire des descripteurs latents afin de différencier des timbres de voix (Nakano et al., 2014) ou encore de caractériser et rechercher des documents audios (*Audio retrieval*) (Hu et al., 2014 ; Kim et al., 2010) En traitement automatique de l'image des méthodes exploitent des mots visuels, pour extraire des espaces latents construits par une LDA une structure latente, dans l'objectif de classifier les images (Rasiwasia et Vasconcelos, 2013 ; Friedlander et al., 2012 ; Fei-Fei et Perona, 2005). Certaines applications bimodales (Liao et al., 2014) reposent sur LDA pour représenter une structure latente commune (image - texte) afin d'améliorer la classification.

LDA a quand même plusieurs limitations. La première est liée à la difficulté d'évaluation du modèle appris. En effet, le moyen le plus commun pour l'évaluation d'un apprentissage repose sur le calcul de la perplexité (Blei et al., 2003). Celui-ci indique la capacité du modèle à représenter des données nouvelles. Cette métrique ne renseigne pas sur sa qualité pour une tâche applicative. Une autre limitation est liée à la sélection d'un nombre de concepts (K) à utiliser. Il n'est pas possible à priori de déterminer un nombre de concepts optimal pour une tâche.

De nombreuses variations du modèle ont été introduites pour traiter différents types de données et de tâches, ou pour lever certaines limites du modèle. On peut trouver par exemple une formulation pour traiter l'évolution des concepts en fonction du temps (Wang et McCallum, 2006), une autre pour supprimer l'indépendance entre les concepts et mesurer leurs corrélations (Blei et Lafferty, 2005) et orienter la dépendance (Li et McCallum). Certaines variations ont été introduites pour modéliser des informations supplémentaires comme la *labelled LDA* (Ramage et al., 2009) qui peut modéliser une variable supplémentaire pour orienter les concepts ou le modèle *Author-topic* (AT) qui modélise un jeu d'auteurs et leurs sujets d'écritures (Rosen-Zvi et al., 2004). Ce modèle est détaillé dans le Chapitre 7. Enfin, les processus hiérarchiques de Dirichlet (Teh et al., 2003) proposent de déterminer automatiquement un ensemble de concepts hiérarchisés dépendants des données. Dans cette variation, le nombre de topics n'est plus un

2. https ://developers.google.com/freebase/

paramètre fixé, mais dépend du niveau de granularité voulu et doit être choisi dans la hiérarchie proposée par le modèle.

Malgré ces adaptations, certaines limitations persistent. En particulier, le système ne permet pas de déterminer le bon nombre de concepts selon la tâche, et il ne modélise pas les corrélations entre les mots.

2.5 Un changement de paradigmes : Les modèles *d'embeddings* de mots

Avec l'avènement de l'ère de l'apprentissage profond, les systèmes d'apprentissage automatique issus de la recherche en intelligence artificielle (Artificial inteligence, AI) ont montré que les méthodes à bases de réseaux de neurones artificiels (c.f. Chapitre 3) sont capables d'extraire des représentations contenant des informations de haut niveau à partir de représentations très proches des données initiales (brut), si un corpus conséquent est disponible. Les descripteurs ainsi produits ont des propriétés qui ne sont pas présentes chez les descripteurs décrits plus haut.
En traitement de la parole, une méthode *d'embeddings* a été introduite pour extraire une représentation améliorant la mesure de similarité intersegments parlés (Chung et al., 2016).
Un *embedding* est la projection d'un vecteur (souvant large) représentant une variable catégorique, tel que la présence de mots, le genre, la couleur dans un nouvel espace de taille contrôlé qui modélise les relations entre les différentes catégories. En traitement de l'image, un *embedding* a été présenté pour créer un espace de représentation commun mots-images, afin de faciliter la recherche d'images par requête textuelle (Kottur et al., 2016).
Ce principe a été appliqué en traitement du langage écrit (Collobert et al., 2011), où les mots sont projetés dans un modèle vectoriel (et pas seulement les documents) de taille réduite via un réseau de neurones artificiels. Ce procédé est appelé un *embedding* de mots. Ces mêmes travaux ont montré qu'un espace commun peut être utilisé pour plusieurs tâches et que la répartition géographique dans cet espace possède d'intéressantes propriétés qui permettent de manipuler les relations entre les mots. À la suite de ces travaux, deux méthodes majeures ont été introduites pour apprendre des représentations des mots de manière non supervisée. La plus utilisée est *Word2vec* qui utilise des réseaux de neurones et la seconde méthode est *Glove* qui s'appuie sur de l'analyse factorielle. Ces deux méthodes sont présentées ci-dessous.
Les modèles ***Word2vec***, introduits dans (Mikolov et al., 2013b), permettent de construire sans supervision des *embeddings* de mots efficaces et de taille contrôlée. Dans ces travaux, deux réseaux de neurones artificiels simples sont proposés. Ils sont adaptés pour être le plus simples possible algorithmiquement, de manière à pouvoir apprendre sur de très grande quantité de données. L'utilisation de grands corpus est nécessaire pour la modélisation d'informations pertinentes. Contrairement au modèle LDA, la rapidité des calculs de cette méthode rend la taille des corpus bien moins problématique.
Les modèles ainsi appris ont obtenu de bons résultats sur plusieurs tâches. D'autres expé-

riences (Mikolov et al., 2013a,b) sur ces mêmes modèles ont montré que l'espace de mots construit par les réseaux de neurones capture des relations sémantiques et syntaxiques entre les mots. Mais aussi que ces relations sont manipulables avec des opérateurs d'additions et de soustractions et que les similarités entre les mots peuvent être mesurées efficacement par une distance cosinus. Les détails techniques concernant ces réseaux sont présentés dans le Chapitre 5.
Les nouvelles possibilités offertes par cette méthode d'*embeddings* ont rapidement été étendues à de nombreuses tâches de traitement automatique du langage. Le plus souvent, ils sont utilisés pour la préinitialisation de réseaux de neurones profonds (Guggilla et al. ; Kim, 2014 ; Dai et Le, 2015). Ils peuvent aussi être utilisés directement pour l'extraction de descripteurs. Par exemple en traduction automatique, il est possible de créer un espace vectoriel de mots commun aux deux langues afin d'utiliser les capacités de modélisation des réseaux *Word2vec* (Gouws et al., 2015 ; Wolf et al., 2014). Ou encore en reconnaissance de la parole, ces modèles ont été utilisés pour retrouver les probabilités d'apparition de mots hors vocabulaire d'un modèle de langage (Sheikh et al., 2015). Enfin, en recherche d'information la similarité entre les mots des modèles *Word2vec* peut être employée pour permettre au modèle de généraliser des similarités par proximité des mots dans l'espace *Word2vec* (Ganguly et al., 2015).
Les *embeddings* de mots à base de *Word2vec* ont plusieurs limitations. La première est liée au besoin de corpus de données propres de plusieurs millions de mots. Des grands corpus spécialisés ne sont pas toujours disponibles, même si aucune annotation n'est nécessaire. La seconde est due aux représentations des contextes pour l'apprentissage qui ignorent l'information de structure des documents. Celle-ci est importante pour la modélisation des relations, en particulier pour les relations grammaticales. Une proposition pour lever en partie ce verrou est présentée dans le Chapitre 5. La troisième limitation est liée à la construction de représentations de documents à partir de représentations des mots. Cette problématique est un sujet de recherche actif. Les méthodes les plus simples utilisent la somme ou la moyenne des mots, (Tai et al., 2015 ; Mikolov et al., 2013b) mais obtiennent des performances sous-optimales. D'autres méthodes plus complexes reposent sur des réseaux de neurones pour traiter les mots séquentiellement pour produire une représentation unique du document (Guggilla et al. ; Kim, 2014 ; Dai et Le, 2015). Enfin, l'interprétation des résultats de la méthode *Word2vec* est très complexe. En effet, la raison qui permet à ces réseaux de neurones de modéliser des représentations sémantiques et syntaxiques n'est pas complètement expliquée [3].
Une autre méthode a été introduite pour réaliser un objectif similaire, mais qui ne repose pas sur des réseaux de neurones. Cette méthode, *Glove*, est basée sur l'analyse factorielle de matrice de cooccurrences des mots pour construire des représentations compactées des mots (Pennington et al., 2014). La matrice de cooccurrences M est de dimension $V \times V$ avec V qui est le nombre de mots dans le vocabulaire du système automatique. La valeur $M_{i,j}$ représente le nombre de fois où le mot_i et mot_j apparaissent dans un même contexte. Ensuite, une analyse factorielle est appliquée pour réduire les dimensions de M et en extraire l'information utile. Les représentations extraites avec *Glove* modélisent linéairement des relations sémantiques entre les mots. Ces relations sont très

3. (Goldberg et Levy, 2014) Y. Goldberg et O. Levy, 2014. word2vec explained : deriving mikolov et al.'s negative-sampling word-embedding method. preprint arXiv :1402.3722.

similaires à celles modélisées par *Word2vec*. Glove et *Word2vec* obtiennent des résultats proches sur plusieurs tâches (Muneeb et al., 2015 ; Soutner et Müller, 2015). Certains travaux apportent un début d'explication à la similarité des résultats de ces méthodes et *Word2vec* (Levy et al., 2015 ; Levy et Goldberg, 2014b). Ils ont montré que globalement les deux méthodes optimisent la même fonction objectif, mais utilisent des paramètres différents. Comme ces deux méthodes tentent de représenter les mêmes informations, elles subissent donc les mêmes limites. Par exemple, les deux méthodes ne sont pas capables de gérer l'apparition de mots qui ne sont pas contenus dans leur vocabulaire d'origine.

2.6 Conclusion

Dans ce chapitre nous avons présenté les trois approches les plus influentes pour la représentation des documents textuels : l'approche sac-de-mots, l'approche thématique latente et l'approche par embeddings de mots.

Les sac-de-mots sont encore à ce jour utilisés par la communauté du traitement automatique du langage pour leur simplicité d'utilisation et leur robustesse. En contrepartie, ils produisent des représentations creuses et larges qui freinent la généralisation des modèles et ne représentent que l'information superficielle. Les modèles thématiques latents apportent une solution à ce problème, en proposant une représentation de taille contrôlée, modélisant la structure sous-jacente sous la forme de mélanges de topics qui est plus informative et généralise mieux aux nouvelles données. Les modèles *d'embeddings* ont été introduits pour extraire automatiquement et avec peu d'aprioris des représentations pertinentes. Ces représentations ont la particularité de construire un espace où la distance intermots porte une information de similarité sémantique et grammaticale qui est exploitable par un système d'apprentissage automatique.

Ces représentations sont indispensables dans un contexte d'apprentissage automatique. Dans ces travaux, nous utiliserons principalement des réseaux de neurones artificiels pour réaliser ces apprentissages. Le prochain chapitre va présenter les réseaux de neurones profonds et les différentes architectures les plus communes ainsi que leurs particularités.

Chapitre 3

Réseaux de neurones artificiels profonds

Sommaire

3.1 Introduction **35**
3.2 Les origines des réseaux de neurones artificiels **36**
3.3 Les réseaux stochastiques à base d'énergie **39**
3.4 Les réseaux *feedforward* **42**
3.5 Les réseaux récurrents **45**
3.6 Les méthodes multiarchitectures **49**
3.7 Conclusion **53**

3.1 Introduction

L'introduction des neurones artificiels et leurs applications sont survenues dans les années 60. Depuis 2009, les réseaux de neurones artificiels profonds [1] sont devenus les outils majeurs de l'apprentissage automatique, et ce en grande partie grâce à l'implémentation de réseaux de neurones sur carte graphique (graphical processing unit, GPU) (Chellapilla et al., 2006). L'utilisation de GPU a permis d'entrainer plus rapidement des réseaux plus profonds sur des volumes de données plus importants. De nombreuses bibliothèques logicielles (Bergstra et al., 2010a ; Chollet ; Collobert et al. ; Chen et al., 2015) [2]

ont vu le jour pour simplifier l'utilisation des ressources de calcul GPU.

L'étude des réseaux de neurones profonds est un domaine de recherche en plein essor avec notamment les récentes introductions d'architectures (Goodfellow et al., 2014), de

1. Les termes : réseau, réseau de neurones, NN et ANN font systématiquement référence, dans ces travaux, aux réseaux de neurones artificiels. Un rapprochement avec les neurones naturels sera précisé.

2. (Abadi et al.,) M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, et al. Tensorflow : Large-scale machine learning on heterogeneous distributed systems. preprint arXiv :1603.04467

méthodes de régularisation (Ioffe et Szegedy, 2015), des fonctions d'activation (Clevert et al., 2015), des méthodes d'optimisation (Kingma et Ba, 2015), etc.
Ils sont, par ailleurs, le moteur d'une recherche intensive dans de nombreux domaines applicatifs. Ils sont devenus incontournables, notamment en traitement du langage naturel (Wang et al., 2016 ; Mikolov et al., 2013a ; Xiong et al., 2017a) [3], en traitement d'images (Song et Xiao, 2016 ; Lee et Osindero, 2016 ; Yang et al., 2016) ou encore pour le traitement de documents multimédia (Wu et al., 2015). Ils sont aussi devenus source d'innovation dans de nombreux autres domaines applicatifs, par exemple : le traitement automatique de données médicales (Cire**c**san et al., 2013 ; Roth et al., 2015), la réalisation de prédictions météorologiques (Salman et al., 2015) et l'amélioration de la sécurité routière (Koesdwiady et al., 2016).

Ce chapitre présente les principaux réseaux de neurones artificiels employés et les variations les plus utiles présents dans la littérature. Ces contributions marquantes sont situées dans la frise chronologique présentée Figure 3.1. Le chapitre est organisé de la manière suivante : La Section 3.2 présente les publications à l'origine de cette thématique. La Section 3.3 introduit les différents réseaux génératifs stochastiques à base d'énergie. La Section 3.4 détaille les architectures dites «FeedForward». La Section 3.5 définit les réseaux récurrents. La Section 3.6 aborde les publications qui concernent des avancées structurelles communes à plusieurs types d'architecture.

3.2 Les origines des réseaux de neurones artificiels

La première définition des neurones artificiels ou neurones formels est réalisée dans (McCulloch et Pitts, 1943). Un schéma du neurone formel est montré dans la Figure 3.2a. Dans cette définition, un neurone a un nombre d'entrées variable. Ces entrées sont pondérées, *simulant* les dendrites des neurones naturels. Toutes les entrées sont additionnées et une fonction d'activation (les plus courantes sont présentées dans la Figure 3.2b) est appliquée sur la somme pour produire une valeur de sortie.

La première application du neurone formel est le Perceptron (Rosenblatt, 1958). Il est présenté dans la Figure 3.3a. Il est composé d'un unique neurone et d'une fonction d'activation *heavyside* (c.f. Figure 3.2b). Le Perceptron permet de réaliser une classification binaire. En plus du neurone, une méthode d'apprentissage supervisée du Perceptron est introduite. Elle permet, par un processus itératif, d'estimer ses poids pour déterminer un hyperplan. Cet hyperplan est optimisé pour séparer deux classes. Par contre, le Perceptron est incapable de résoudre un problème qui n'est pas linéairement séparable. Cette faiblesse fut mise en évidence par le problème du "Xor" : un Perceptron ne peut pas modéliser la fonction "ou exclusif". Un exemple du problème "Xor" est schématisé dans les Figures 3.3b et 3.3c . C'est cette limitation qui ralentira le développement des réseaux de neurones artificiels pendant plusieurs années.

Une variante de la méthode d'apprentissage du Perceptron est proposée avec les

3. (Bahdanau et al., 2014) D. Bahdanau, K. Cho, et Y. Bengio, 2014. Neural machine translation by jointly learning to align and translate. preprint arXiv :1409.0473.

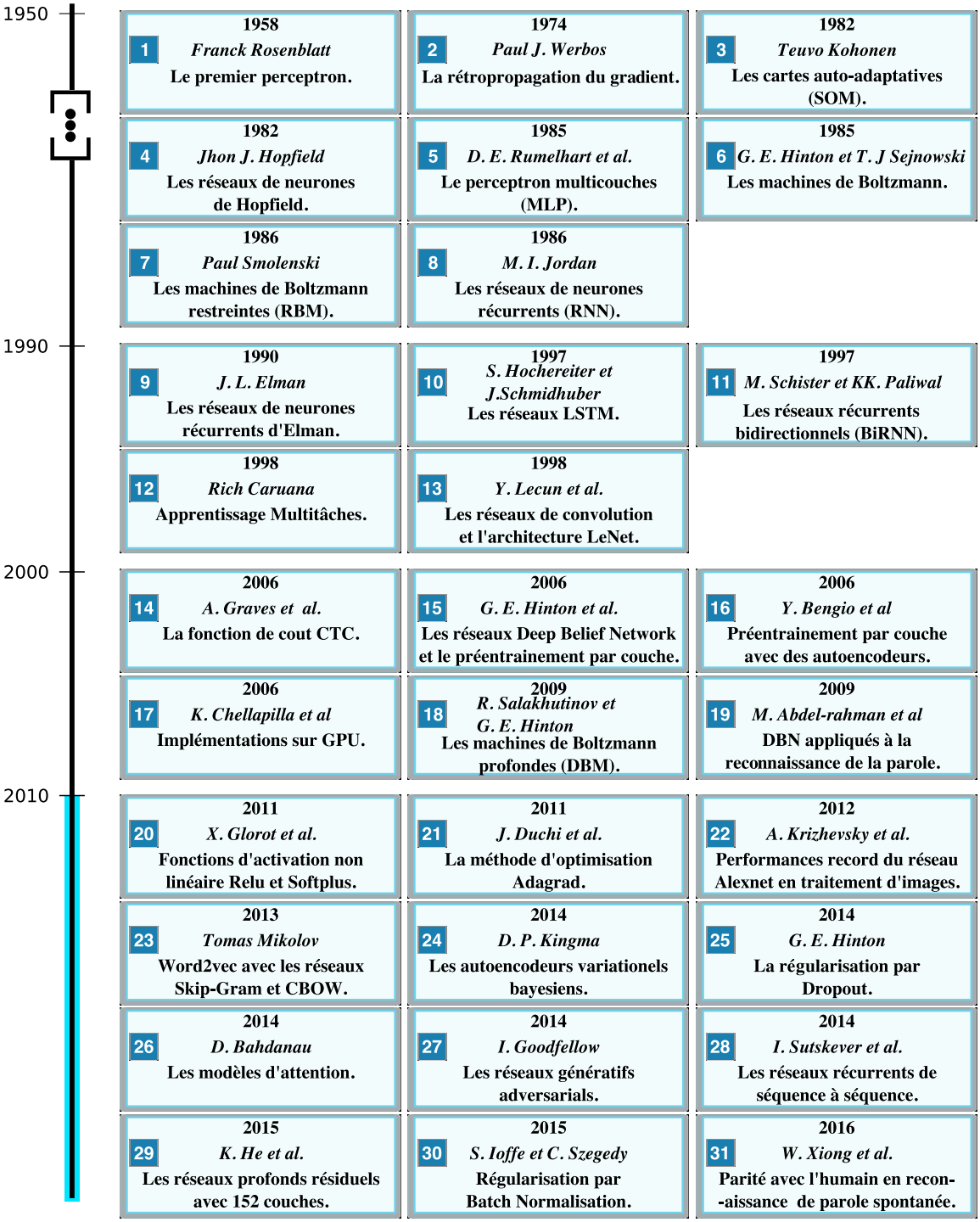
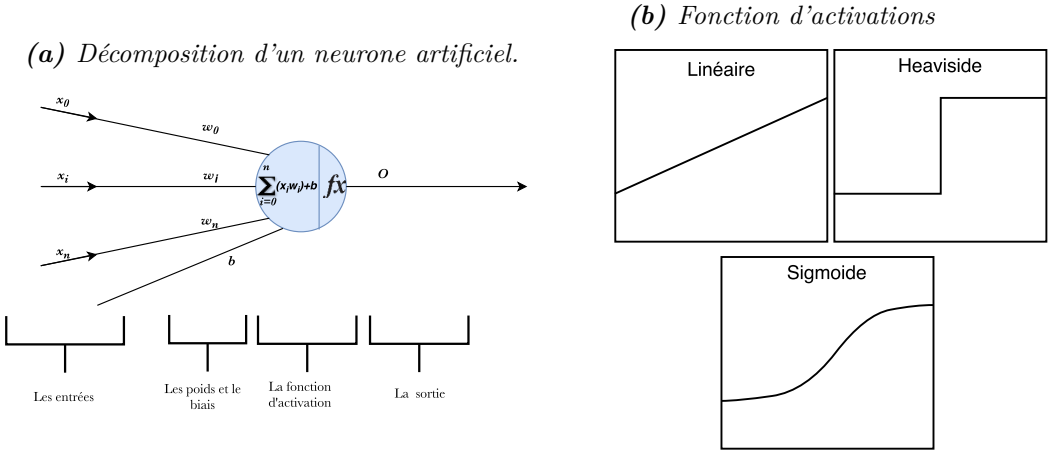


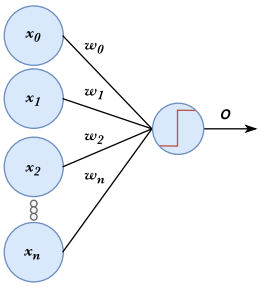
Figure 3.1 – *Chronologie des évènements les plus marquants de l'apprentissage profond. Une table de correspondance des publications est disponible en annexe A.*



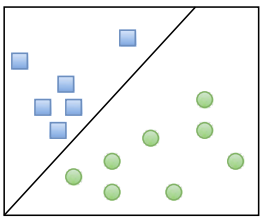
(a) *Décomposition d'un neurone artificiel.*

(b) *Fonction d'activations*

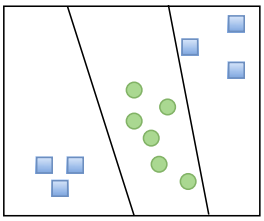
Figure *3.2* – *Schémas d'un neurone artificiel (a) et des fonctions d'activations historiques (b)*



(a) *Architecture d'un Perceptron.*



(b) *Exemple d'un hyperplan séparant des données appris par un Perceptron.*



(c) *Le problème "'Xor" sans solution pour un Perceptron.*

Figure *3.3* – *Architecture d'un Perceptron (a) et exemple classification avec un Perceptron (b et c)*

neurones "Adaline" (Widrow et al., 1960). Ce neurone, en plus de prédire une classe, est capable d'attribuer un score de classification. Il peut donc indiquer si l'appartenance à la classe est sure ou non. C'est dans (Werbos, 1974) que la méthode de rétropropagation de gradient est proposée pour l'apprentissage de réseaux de neurones artificiels.

Des réseaux de neurones capables de réaliser l'extraction de descripteurs par le biais d'un apprentissage non supervisé sont introduits en 1982. Ce sont les cartes auto adaptatives de Kohonen (Kohonen, 1982) (self-organised map, SOM). Dans ces réseaux, les neurones sont disposés en grille. Chaque zone de cette grille de neurones apprend à réagir à un type de données particulier. Une fois entrainée, une carte auto adaptative peut servir à discrétiser, à réaliser une quantification vectorielle, ou réduire la dimensionnalité des données d'entrées (Chihi, 1998 ; Nasrabadi et Feng, 1988).

Sur ces bases, les réseaux de neurones ont ensuite évolué avec différents types de topologies. Ces topologies peuvent être catégorisées en trois groupes :

1. Les *réseaux stochastiques à base d'énergie* capturent les dépendances intervariables en associant une valeur d'énergie à chaque configuration de variables et en mini-

misant l'énergie totale du réseau.

2. Les *réseaux FeedForward (FFNN)* sont organisés par couche, sans former de cycle. Ils sont optimisés par rétropropagation de gradient pour apprendre des modèles principalement discriminants.
3. Les *réseaux de neurones récurrents (RNN)* sont dérivés des réseaux *feedforward* et sont introduits principalement pour modéliser les données temporelles et séquentielles.

Chacune de ces catégories est présentée ci-dessous.

3.3 Les réseaux stochastiques à base d'énergie

Cette famille de réseaux de neurones utilise des méthodes d'apprentissage basées sur l'énergie d'un système. L'énergie mesure les dépendances entre des variables. Le réseau doit donc, en fonction de variables connues, déterminer l'état des variables inconnues qui vont minimiser l'énergie du système. L'apprentissage de ces réseaux consiste à trouver la fonction d'énergie qui associe des valeurs faibles aux situations correctes et des valeurs fortes aux situations incorrectes. De nombreuses méthodes d'apprentissage de ces réseaux sont inspirées de la physique. Parmi ces méthodes, on trouve par exemple le recuit simulé (Ackley et al., 1985) ou le recuit par échantillonnages préférentiels (Neal, 2001). La méthode du recuit (non simulé) est un traitement thermique utilisé en métallurgie pour modifier les contraintes internes des matériaux. Des algorithmes efficaces (Hinton, 2002) reposants sur l'énergie du système, mais qui ne sont pas issus de la physique, ont aussi été proposés (Carreira-Perpinan et Hinton, 2005).
Le premier réseau à énergie (non stochastique) défini dans la littérature est le réseau de Hopfield (Hopfield, 1982). Ce réseau est constitué d'un groupe de neurones binaires, dont les états valent 0 ou 1. Ils sont tous reliés entre eux par une connexion bidirectionnelle. Ainsi, si w est la matrice de poids et i et j deux neurones du réseau, alors $w_{ij} = w_{ji}$. Chaque neurone est lié à une observation. Cette architecture a des capacités de mémoire associative prouvées, mais elle ne peut modéliser qu'une faible quantité d'information pour un nombre important de neurones. Elle a principalement été remplacée par des réseaux stochastiques, les machines de Boltzmann (Ackley et al., 1985). Dans ces réseaux (c.f. Figure 3.4), deux types de neurones sont présents.

— Les neurones visibles, qui reçoivent les informations de l'environnement. Ils sont représentés en pratique par des vecteurs binaires.

— Les neurones cachés, qui représentent des variables aléatoires dont les distributions modélisent les régularités présentes dans les données.

Dans une machine de Boltzmann, tous les neurones partagent une connexion deux à deux. Les poids de ces connexions sont déterminés par recuit simulé. Les machines de Boltzmann ont d'excellentes capacités de représentation et de génération, mais la complexité algorithmique de leur apprentissage est exponentiellement proportionnelle au nombre de neurones. Cette limitation les rend difficilement applicables à des cas pratiques dans cette version.

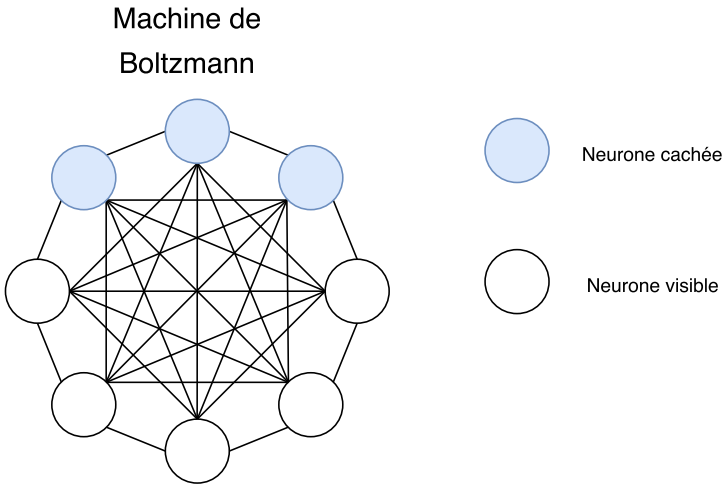


Figure *3.4* – *Présentation des machines de Boltzmann*

En 1986, les machines de Boltzmann restreintes (RBM) sont introduites sous le nom harmonium (Smolensky). Cette architecture est présentée dans la Figure 3.5. C'est une forme spécifique des machines de Botlzmann composée de deux couches de neurones. La première couche contient les neurones visibles et la seconde les neurones cachés. Contrairement aux machines de Boltzmann standards, il n'y a pas de connexion entre les neurones d'une même couche. Cette restriction permet aux neurones cachés d'être indépendants entre eux pour un ensemble de neurones visibles donné. Cette indépendance permet de réduire les échantillonnages nécessaires à l'apprentissage du réseau. Les paramètres de cette architecture sont optimisés avec la divergence contrastive (Hinton, 2002), qui permet d'entrainer un modèle stochastique génératif efficacement. Elle est utilisée avec succès dans plusieurs applications, principalement pour de l'extraction de descripteurs (WhyeTeh et Hinton, 2001 ; Dahl et al. ; Salakhutdinov et al., 2007). Les RBM peuvent être adaptées en modèle discriminant (Larochelle et Bengio, 2008).

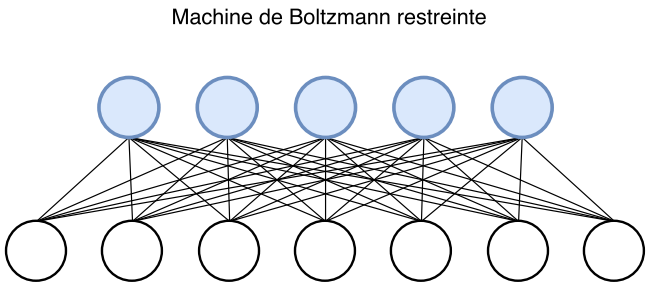


Figure *3.5* – *Architecture des machines de Boltzmann restreintes.*

Une architecture profonde, issue des RBM, est introduite en 2006, appelée «réseaux de croyance profonds» (Deep belief network, DBN) (Hinton et al., 2006). L'architecture de ce modèle génératif est présentée dans la Figure 3.6a. Le DBN est composé d'une couche de neurones visibles et de plusieurs couches cachées de variables latentes stochastiques. Les deux dernières couches cachées sont reliées par des connexions symétriques,

les autres couches cachées sont liées par des connexions dirigées. Chaque couche cachée est préentrainée au préalable par une RBM qui utilise la projection de la couche précédente comme vecteur d'entrée. L'apprentissage des DBN est aussi non supervisé. Ils sont utilisés principalement pour extraire des représentations de haut niveau des données d'entrée ou pour initialiser un réseau discriminant. Ils obtiennent de bonnes performances dans des domaines variés : en traitement de la parole (Mohamed et al., 2009 ; Sainath et al., 2011), en traitement d'image (Lee et al., 2009), en compréhension du langage (Sarikaya et al., 2014) , pour prévoir les taux de changes (Chao et al., 2011), etc.

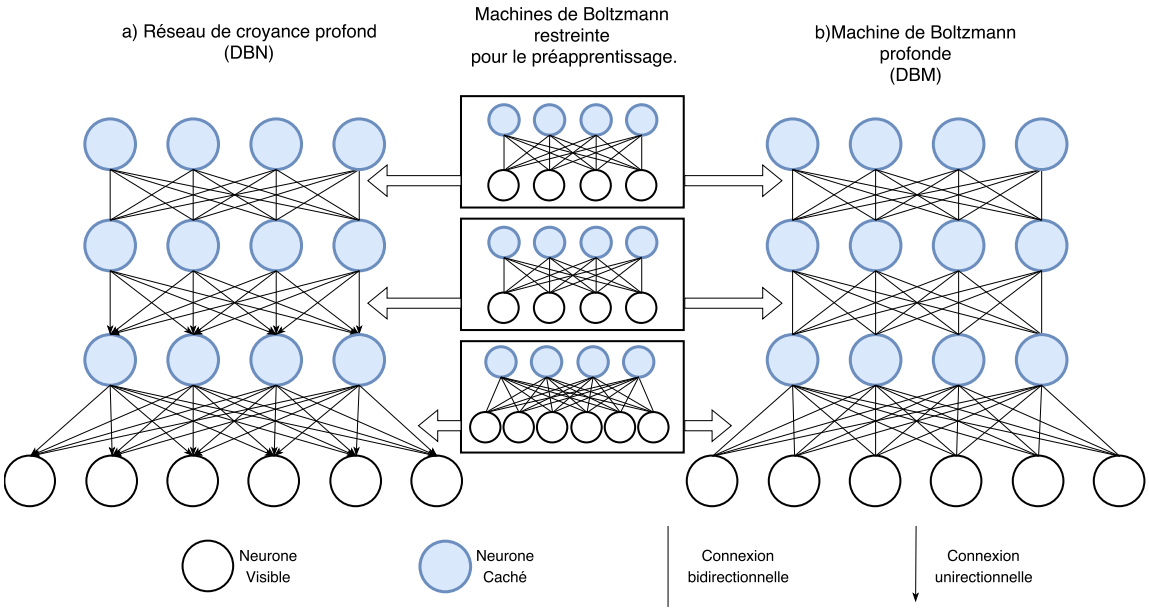


Figure *3.6* – *Architecture des réseaux de croyance profonds et des machines de Boltzmann profondes.*

Une dernière architecture, la machine de Boltzmann profonde (Deep Botlzmann Machine, DBM) (Salakhutdinov et Hinton, 2009) est proposée comme une évolution des DBN. Ils sont présentés dans la Figure 3.6b. De la même manière que les DBN, les DBM sont des réseaux de neurones génératifs stochastiques. Ils sont composés de plusieurs couches cachées qui sont préentrainées, une par une, via des RBM. À la différence des DBN qui utilisent des connexions dirigées, toutes les connexions d'une DBM sont bidirectionnelles. Cette différence permet au DBM de modéliser et d'utiliser l'information des couches basses du réseau pour déterminer des représentations des couches de haut niveau plus robustes. Les DBM ont obtenu de bonnes performances en dépassant systématiquement leurs homologues DBN et RBM, notamment sur des tâches de traitement de données multimodales (Srivastava et Salakhutdinov, 2012), de modélisation des thèmes (Srivastava et al., 2013), de reconnaissance d'expression faciale (He et al., 2013), pour le traitement de requêtes parlées (Zhang et al., 2012), etc.

Ces réseaux stochastiques génératifs sont capables d'apprendre des modèles génératifs puissants de manière non supervisée. Il leur est plus compliqué de produire des modèles

déterministes. Pour compenser cette faiblesse, ils sont souvent utilisés avec des réseaux *feedforward*.

3.4 Les réseaux *feedforward*

Le terme *feedforward* fait référence à un ensemble de réseaux de neurones dont les connexions interneuronales forment un graphe acyclique. Le Perceptron présenté Figure 3.3a est un exemple de réseau *feedforward*. Dans cette section, les cinq principales architectures *feedforward* sont présentées :

1. les Perceptron multicouches (MLP)
2. les autoencodeurs (AE)
3. les réseaux de convolutions (CNN)
4. les réseaux génératifs adverses (GAN)
5. les réseaux variationnels (VNN).

Les Perceptrons multicouches (MLP) (Rumelhart et al., 1985), dont un est visible dans la Figure 3.7, sont des réseaux supervisés utilisables pour la classification, la régression ou la réduction de dimensionnalité. Dans cette architecture, les neurones sont regroupés par couches. On retrouve une couche d'entrée qui reçoit les informations que le réseau doit traiter et une couche de sortie qui contient la prédiction du réseau. La fonction d'activation de la couche de sortie peut être choisie en fonction de la tâche que le réseau doit accomplir. La fonction d'activation linéaire est utilisée pour apprendre des modèles de régression linéaire. La fonction d'activation sigmoïde est utilisée pour des régressions logistiques ou des classifications binaires. La fonction d'activation *Softmax* (Bishop, 1995) est utilisée pour des classifications multiclasses. Une fonction de cout est utilisée pour calculer l'erreur entre la sortie produite et la sortie attendue. Entre l'entrée et la sortie, un nombre variable (supérieur ou égal à 1) de couches cachées avec une fonction d'activation non linéaire est utilisé. Les fonctions les plus courantes sont présentées dans la Figure 3.2b. Chaque neurone d'une couche est connecté à tous les neurones de la couche qui le précèdent et qui le succèdent directement. Par contre, il n'y a pas de connexions entre les neurones d'une même couche. Les poids de ces connexions sont appris par rétropropagation du gradient calculé entre la couche de sortie et la prédiction attendue.

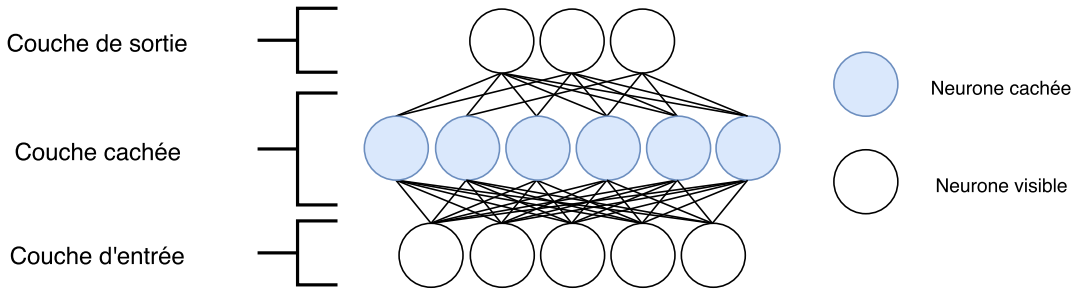


Figure 3.7 – *Présentation d'un Perceptron multicouche.*

Le MLP est capable d'approximer la fonction «OU exclusif», il n'est pas contraint aux mêmes limitations que le Perceptron simple. Le MLP est un «approximateur universel» (Cybenko, 1989) à condition qu'il y ait suffisamment de neurones cachés.
Une version non supervisée du MLP est introduite (Rumelhart et al., 1985) : l'autoencodeur (AE). Lors de l'apprentissage de ce réseau, la supervision est réalisée avec les informations d'entrée. Il apprend à reconstruire le vecteur d'entrée en passant par une couche cachée de taille inférieure. Le fonctionnement des autoencodeurs est présenté en détail dans le Chapitre 6. Ils sont principalement utilisés dans deux cas : le premier à des fins de débruitage et de réduction de dimensions et le second comme préapprentissage des poids des couches cachées de MLP profonds (Bengio et al., 2007a) . Le préentrainement d'un réseau profond à l'aide d'autoencodeur réduit les possibilités de converger vers un optimum local lors de l'étape d'apprentissage globale (Erhan et al., 2010).

Une autre méthode est régulièrement utilisée pour générer des descripteurs ou pour initialiser des poids de réseaux de neurones. Ce sont les méthodes d'*embeddings* (plongement). Elles sont devenues des méthodes d'extraction de descripteurs de variables qualitatives très populaires ; en particulier grâce aux méthodes d'embeddings de mots *Word2vec* (Mikolov et al., 2011, 2013b,b). *Word2vec* propose deux réseaux de neurones capables de créer des espaces multidimensionnels dans lesquels sont projetés les mots et qui modélisent les relations sémantiques et grammaticales. Ils ont été largement utilisés et adaptés pour traiter des données textuelles (Wang et al., 2016 ; Ferreira et al., 2015 ; Levy et Goldberg, 2014a ; Clinchant et Perronnin, 2013), mais aussi pour des données multimédias [4] (Bengio et Heigold, 2014). L'intérêt des représentations distributionnelles est explicité dans le Chapitre 2 et le fonctionnement de la méthode *Word2vec* est présenté en détail dans le Chapitre 5.

Récemment deux types d'architectures prometteuses ont été introduites pour entrainer, par rétropropagation du gradient, des modèles génératifs puissants : les réseaux adverses génératifs et les autoencodeurs variationnels.

Les réseaux adverses génératifs (Generative adversarial network, GAN) (Goodfellow et al., 2014) combinent deux réseaux de neurones. Le premier réseau G est un réseau génératif qui génère un vecteur x à partir d'une représentation latente issue de documents du corpus d'entrainement. Le second réseau D est un réseau discriminant qui prédit la probabilité que l'échantillon d'entrée soit généré par G au lieu de provenir du corpus d'entrainement. Les deux réseaux optimisent leurs poids pour trouver un équilibre dans un jeu de minmax. Ils ont été utilisés principalement pour générer des images, à partir de contraintes de formes ou de couleurs (Zhu et al., 2016) ou à partir de descriptions textuelles (Reed et al., 2016).

Les autoencodeurs variationnels (VAE) sont introduits comme un lien entre l'inférence variationnelle (Fox et Roberts, 2012) et les réseaux de neurones profonds (Kingma et Welling, 2014 ; Gal et Ghahramani, 2015). Ils considèrent un ensemble de variables aléatoires latentes z, pour capturer les variations des variables observées x. Leurs distri-

4. (Kiros et al., 2014) R. Kiros, R. Salakhutdinov, et R. S. Zemel, 2014. Unifying visual-semantic embeddings with multimodal neural language models. preprint arXiv :1411.2539.

butions jointes sont définies par :

$$p(x,z) = p(x|z)p(z) \tag{3.1}$$

Où la distribution de probabilités à priori de $p(z)$ suit une loi normale et $p(x|z)$ est un modèle observé dont les paramètres sont calculés par le réseau de neurones en fonction de z. La projection non linéaire de z vers x rend impossible l'inférence de la distribution à postériori de $p(z|x)$. Dans ce but le VAE utilise une approximation variationnelle $q(z|x)$ dont la distribution suit une loi normale. Les paramètres de cette loi normale (moyenne et variance) sont la sortie d'une fonction non linéaire apprise aussi par le réseau de neurones. Le modèle génératif $p(x|z)$ et le modèle d'inférence $q(z|x)$ sont appris par rétropropagation de manière à maximiser la borne inférieure de la vraisemblance de $p(x)$. Ils ont été utilisés avec succès pour générer des phrases (Bowman et al., 2016), pour générer des images (Gregor et al., 2015) ou encore pour appliquer des rotations sur des images 3D (Kulkarni et al.).

Pour modéliser des évènements indépendamment de leurs positions dans des données (un phonème dans un signal acoustique, une forme dans une image), l'utilisation d'un MLP est inadaptée. C'est d'autant plus vrai quand ces données sont encodées sur plusieurs canaux dépendants, par exemple les 3 canaux rouge, vert et bleu composant une image ou les 13 descripteurs par unité de temps des MFCC pour le signal audio. En effet, il serait nécessaire au MLP de rencontrer ces évènements dans tous les contextes possibles et de modéliser chacun d'entre eux. Cette contrainte implique l'utilisation de plus de paramètres dans le réseau et de données d'apprentissage. Pour pallier cette faiblesse, les réseaux de convolutions ont été introduits (LeCun et al., 1998) (LeCun et al.). Ils imitent le fonctionnement du cortex visuel animal. Les réseaux de convolutions sont particulièrement efficaces pour modéliser des dépendances locales sur plusieurs dimensions. Ils peuvent, par exemple, modéliser des n-grammes de mots dans le cadre de textes ou des formes dans le cadre d'images. Un exemple de réseaux de convolutions est présenté dans la Figure 3.8. Ils sont composés de trois groupes de couches différentes. Des couches de convolutions, des couches de regroupement (pooling) et un groupe final de couches MLP. Les couches de convolutions et de pooling peuvent être alternées plusieurs fois pour une détection des relations de plus en plus complexes. Les couches de convolutions sont composées de plusieurs filtres de convolutions. Chaque filtre de convolution est appliqué sur une fenêtre glissante parcourant tous les descripteurs des données d'entrées et s'active pour détecter un patron particulier. La sortie d'un filtre de convolution s'appelle une carte de caractéristiques. Une méthode de regroupement est ensuite appliquée sur chacune des cartes de caractéristiques. La méthode de regroupement est utilisée pour renforcer l'information d'activation du filtre de convolution en sacrifiant une partie de l'information positionnelle. Cette mécanique permet de limiter le nombre de paramètres à optimiser dans le réseau. Les résultats de la dernière couche de pooling sont cumulés et utilisés comme vecteurs d'entrés des couches MLP pour la classification. Les couches de convolutions ont permis d'obtenir des performances à l'état-de-l'art en traitement automatique de la parole (Lei et al. ; Xiong et al., 2017b), en traitement automatique de l'image (He et al., 2016) ainsi qu'en classification textuelle (Moschitti et al., 2014).

La capacité des réseaux de convolutions à modéliser des dépendances locales, même

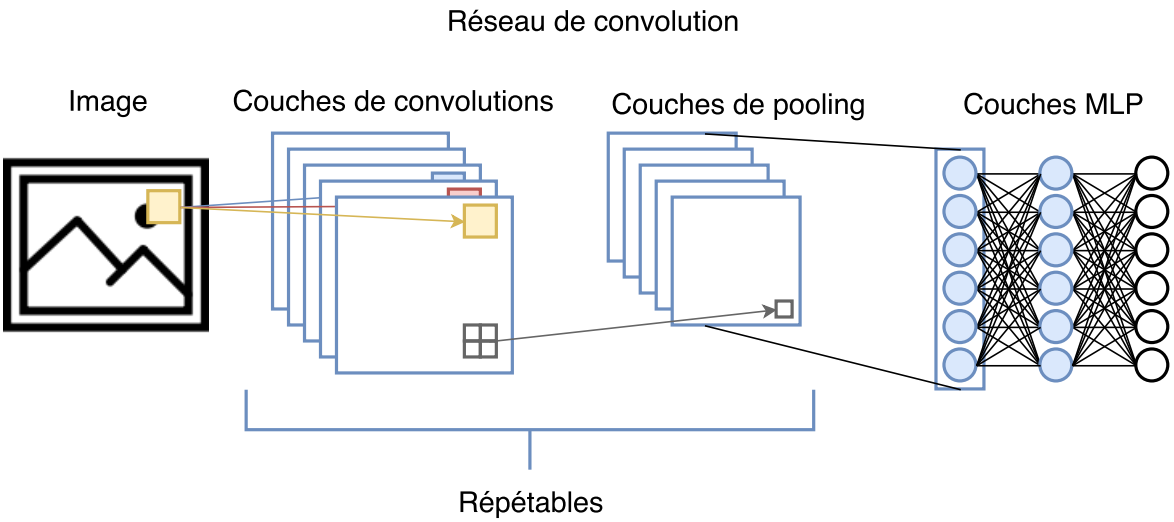


Figure ***3.8*** – *Schéma d'un réseau de convolutions.*

complexes, ne permet pas de traiter efficacement des données séquentielles. Pour répondre à cette problématique, un ensemble de réseaux récurrents est proposé.

3.5 Les réseaux récurrents

Dans cette section, nous abordons dans un premier temps les réseaux de Jordan et d'Elman qui ont introduit les bases des réseaux récurrents. Ensuite, nous décrivons deux évolutions : les réseaux bidirectionnels et les réseaux à longue mémoire à court terme (LSTM) qui apportent une solution aux limites des deux précédents. Enfin, la méthode d'attention est présentée. Elle permet à un réseau récurrent de filtrer l'information utile à un temps t.

Les réseaux récurrents sont définis comme des réseaux contenant au moins un cycle par opposition aux réseaux *feedforward*. Ils sont introduits avec les architectures des réseaux de Jordan (Jordan, 1986) et ceux d'Elman (Elman, 1990). Ces deux architectures sont schématisées dans la Figure 3.9 (a et b). Dans les réseaux de Jordan, la prédiction au temps t dépend de la prédiction réalisée au temps $t-1$ et de l'entrée du réseau au temps t. Dans les réseaux d'Elman, la prédiction au temps t dépend de l'état de la couche cachée au temps $t-1$ et de l'entrée du réseau au temps t. Ces architectures permettent aux réseaux de neurones de modéliser de l'information séquentielle. La méthode nécessaire à l'apprentissage de ces réseaux a été inventée indépendamment par (Robinson et Fallside, 1987 ; Werbos, 1988). Cette méthode, la rétropropagation du gradient dans le temps, consiste à dérouler le réseau dans le temps, comme présenté en bas du schéma 3.9 et à appliquer une descente de gradient sur ce réseau déroulé. Ces réseaux sont particulièrement utilisés en modélisation du langage naturel (Mikolov et al., 2010) où la séquentialité est particulièrement importante.

Les réseaux bidirectionnels ont été introduits dans les travaux de (Schuster et Pali-

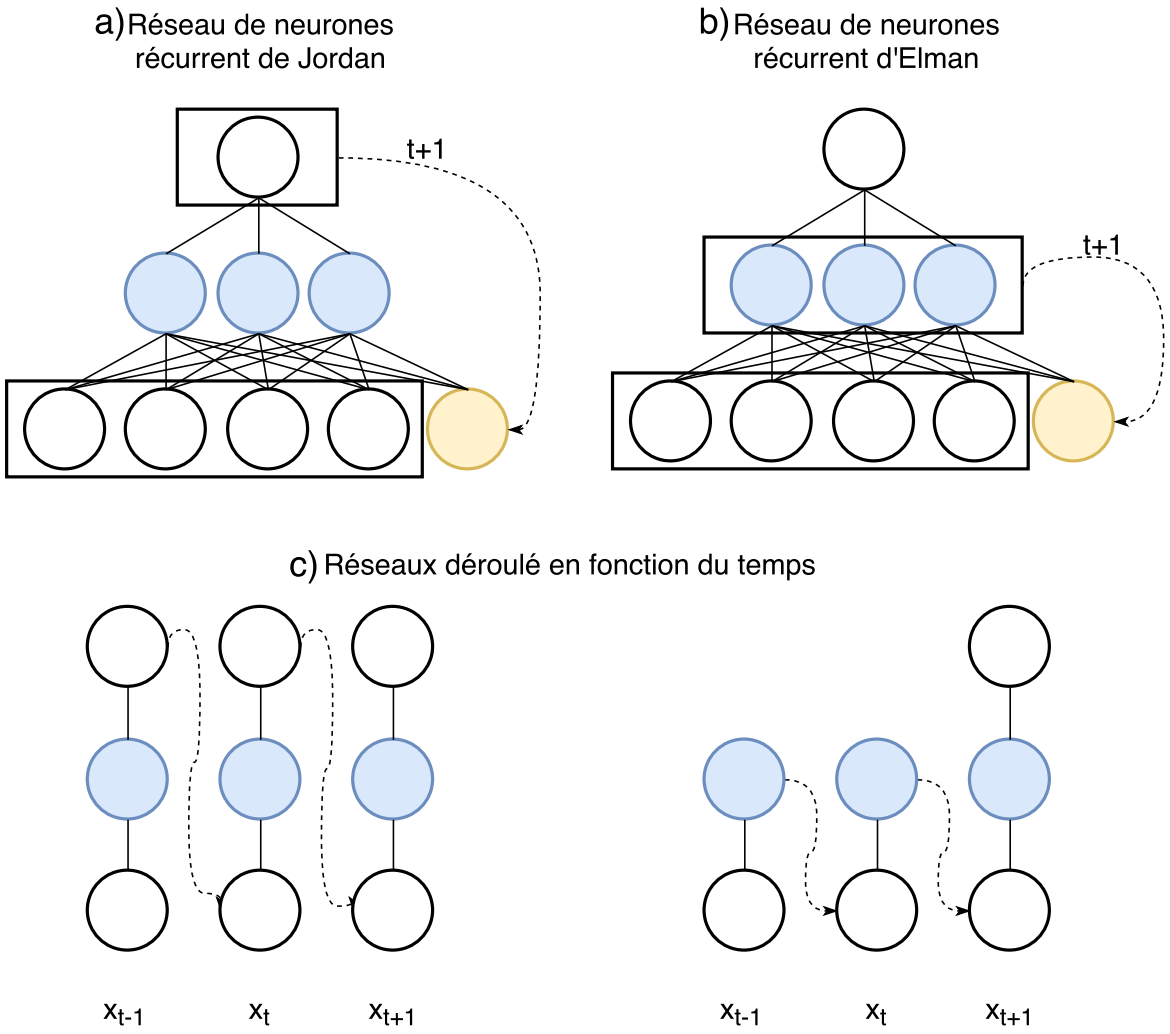


Figure *3.9* – *Architecture des réseaux de neurones récurrents Elman et Jordan et un exemple de réseau déroulé.*

wal, 1997) comme une évolution des réseaux récurrents. Dans le cadre de données temporelles, les réseaux bidirectionnels exploitent l'information passée et future pour réaliser de meilleures prédictions. L'architecture d'un réseau bidirectionnel est schématisée dans la Figure 3.10. Il est composé d'une couche dite "forward" qui applique une récurrence dans le sens temporel et une couche dite "backward" qui applique la récurrence en sens inverse. La sortie du réseau est produite en utilisant les deux couches cachées pour le même temps t. Ces réseaux récurrents atteignent leur limite lorsqu'ils traitent des séquences longues comme c'est le cas des phrases sous forme textuelle ou sous forme de signal. En effet, le gradient diminue au fil du temps et n'impacte que faiblement les premières itérations.

Afin de pallier ce problème, les travaux de (Hochreiter et Schmidhuber, 1997) proposent des cellules à longue mémoire à court terme (Long short term memory, LSTM).

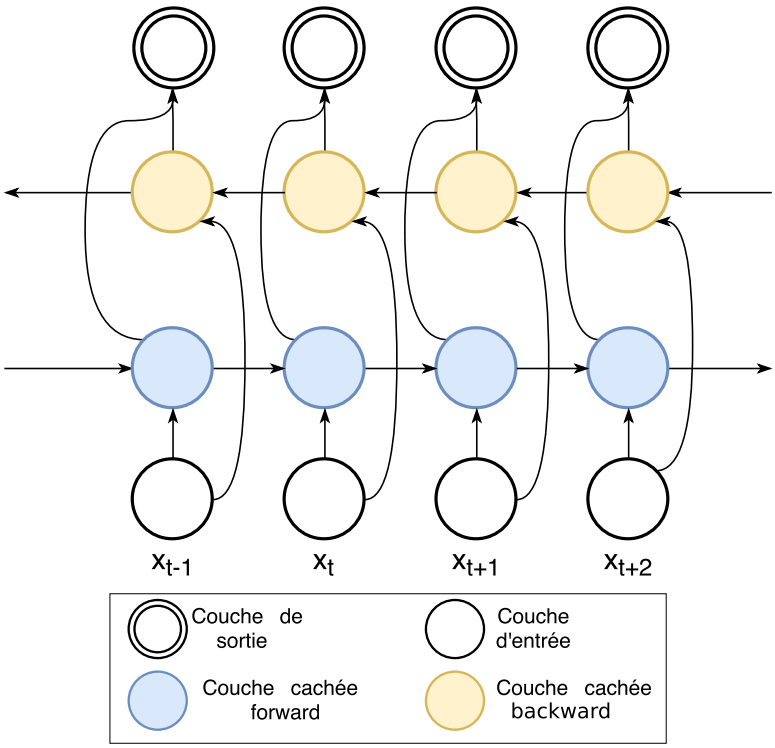


Figure *3.10* – *Principe de fonctionnement d'un réseau récurrent bidirectionnel.*

Cette architecture permet de limiter l'effet de «disparition du gradient» pour pouvoir modéliser des relations temporelles plus longues. L'architecture d'une cellule LSTM est schématisée dans la Figure 3.11. On voit, dans ce schéma, qu'à l'intérieur d'une cellule LSTM, deux couches cachées sont transmises en fonction du temps c_t et h_t. c_t représente la mémoire du réseau qui évolue à chaque temps t pour être utilisée par la cellule au temps $t+1$ et h_t représente les informations prédites par la cellule LSTM. Lorsque la cellule LSTM fait partie d'un réseau, la couche h_t est transmise à la couche supérieure dans le réseau pour poursuivre le traitement. On trouve aussi trois "portes" qui contrôlent le flux d'information, une porte d'entrée, une porte d'oubli et une porte de sortie. Les trois portes sont indépendantes, elles fonctionnent comme un ensemble de poids appliqué à chaque élément du vecteur passant par la porte. Les poids des portes sont déterminés par une couche cachée avec une fonction d'application sigmoïde qui utilise en entrée les vecteurs x_t et h_{t-1}. La porte d'oubli permet de sélectionner la mémoire de la cellule à effacer, en donnant à certaines zones un score de zéro. Elle contrôle donc l'état de la mémoire c_t. La porte d'entrée permet quant à elle de filtrer l'information utile contenue dans les observations x_t et la prédiction précédente h_{t-1} qui doit être introduite dans l'état de la cellule (c_t). La porte de sortie filtre l'information prédite par la cellule. Ces portes permettent de transférer plus d'informations pertinentes entre les temps t_i et d'apprendre des dépendances de plus longs termes que ne parviennent pas à modéliser les réseaux Jordan et Elman.

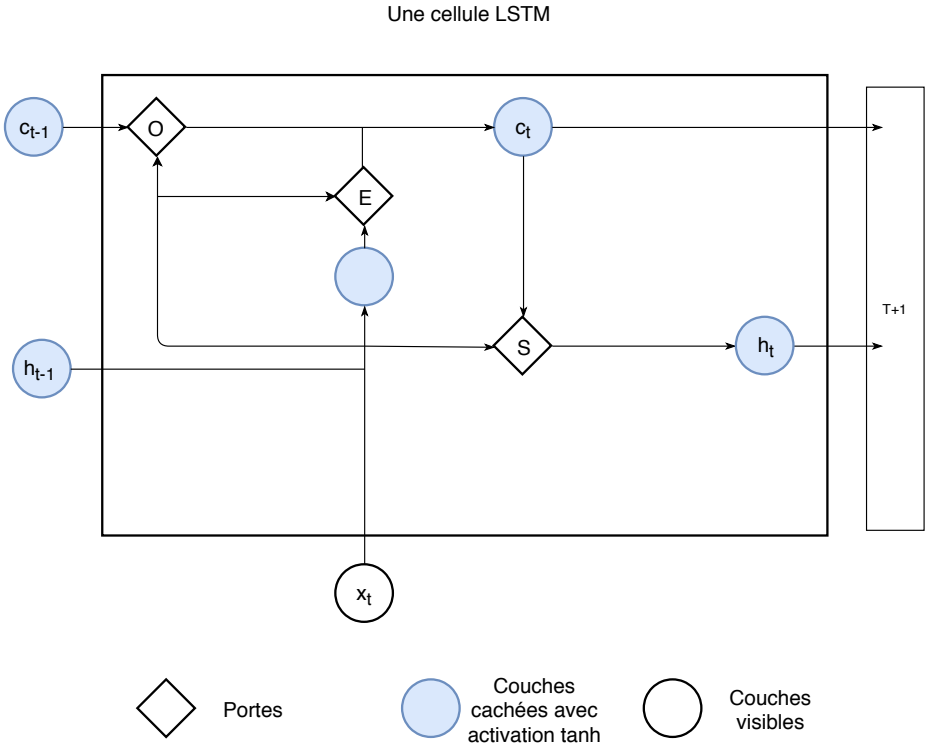


Figure *3.11* – *Schéma d'une cellule LSTM.*

Une méthode inspirée des mécaniques de portes présentes dans les LSTM a été introduite pour permettre aux réseaux de neurones de sélectionner l'information nécessaire à utiliser en fonction du temps t. C'est la méthode des modèles d'attention [5], schématisée dans la Figure 3.12. La méthode d'attention se traduit par une couche qui contient un MLP avec une couche de sortie Softmax (ou parfois sigmoïde) qui génère un filtre à appliquer aux représentations de l'information. Ce filtre localise les informations utiles : il conserve l'information utile en la multipliant par un poids proche de 1 et réduit l'information inutile avec un poids proche de 0. Dans la formulation initiale, le filtre ne dépend que de l'état du réseau au temps $t-1$. Cette méthode a un effet positif sur le traitement des séquences longues, car la localisation de l'information facilite sa mémorisation dans une couche cachée et son utilisation par le réseau. Les modèles à base d'attention sont utilisés le plus souvent pour la prédiction de séquence. Ils ont apporté des améliorations en traduction automatique (Luong et al., 2015), en génération d'image (Gregor et al., 2015), en traitement d'images (Xu et al., 2015a), en traitement de langage (Chopra et al., 2016 ; Chorowski et al.), etc.

Une grande majorité des architectures les plus courantes en apprentissage profond ont été présentées jusqu'ici. Cependant, certaines publications ont un impact dans toutes les catégories présentées ci-dessus. La prochaine section va s'attacher à présenter ces travaux-là.

5. (Bahdanau et al., 2014) D. Bahdanau, K. Cho, et Y. Bengio, 2014. Neural machine translation by jointly learning to align and translate. preprint arXiv :1409.0473.

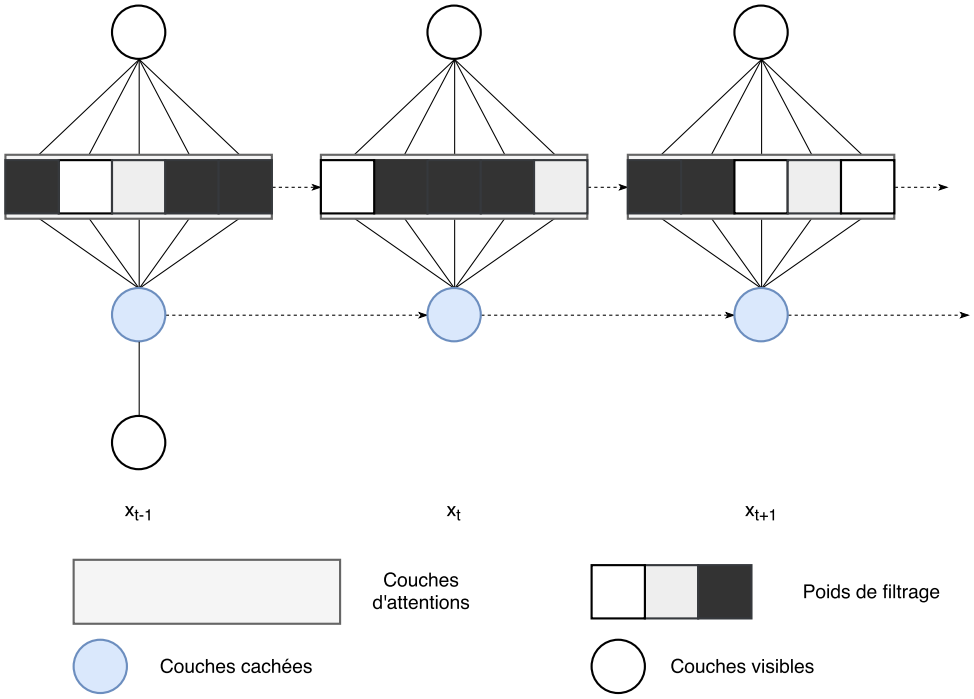


Figure ***3.12*** – *Schéma d'un réseau de neurones récurrent avec attention.*

3.6 Les méthodes multiarchitectures

Dans cette section, plusieurs évolutions seront présentées. Elles proposent toutes des solutions qui sont utilisées dans les réseaux de neurones à l'état-de-l'art et sont communes à plus d'un type de topologies. Nous parlerons d'abord d'une fonction de cout utilisée pour l'inférence de séquences. Puis de la fonction d'activation *relu*, introduite en 2011 et popularisée par les CNN. Ensuite, une descente de gradient adaptative est présentée, l'Adagrad. Enfin, deux méthodes de régularisations qui jouent un rôle important dans les performances des réseaux à l'état-de-l'art sont présentées.

3.6.1 La fonction cout classification temporelle connexionniste (CTC)

La fonction de cout de classification temporelle connexionniste (CTC) (Graves et al., 2006) est capable de traiter des séquences de données non segmentées et d'intégrer l'alignement automatique dans le réseau. Cette fonction est utilisée en particulier pour entrainer des modèles acoustiques et pour simplifier les systèmes de reconnaissance automatique de la parole. En effet, des SRAP neuronaux de bout en bout (ou *end to end*, un réseau de neurones pour réaliser une chaine complète) ont été proposés [6] (Graves et

6. (Hannun et al., 2014) A. Hannun, C. Case, J. Casper, B. Catanzaro, G. Diamos, E. Elsen, R. Prenger, S. Satheesh, S. Sengupta, A. Coates, et al., 2014. Deep speech : Scaling up end-to-end speech

Jaitly ; Amodei et al., 2016). Ces SRAP exploitent des réseaux de neurones récurrents pour générer une hypothèse de phrase directement à partir du signal. Ils permettent ainsi de simplifier le processus de reconnaissance automatique de la parole, en proposant des alternatives efficaces aux modèles combinant mixture de gaussiennes et chaines de Markov cachées (HMM). Pour apprendre une telle tâche, le réseau a besoin de "comprendre" les erreurs qu'il a commises. Le réseau va donc prédire pour chaque frame temporel une probabilité pour chaque phonème (ou mot) plus le vide. Une fois les prédictions réalisées un algorithme ("dynamic time warping") va déterminer le meilleur chemin parmi les hypothèses en fonction de l'information de référence et en ignorant les répétitions et les blancs. Lors de la rétropropagation du gradient les chemins qui ne sont pas sélectionnés sont pénalisés et les autres favorisés. Grâce à cette fonction de cout, un réseau de neurones peut apprendre à prédire des informations étalées sur un ou plusieurs frames temporels (comme les mots) en reposant seulement sur un signal parlé et sa transcription.

3.6.2 La fonction d'activation relu

Les réseaux de neurones rectifieurs ont introduit l'utilisation de la fonction d'activation *rectified linear unit* (relu) (Glorot et al., 2011). C'est une alternative aux fonctions non linéaires, sigmoïde et tangente hyperbolique. Elle est définie par :

$$f(x) = max(0, x) \tag{3.2}$$

La fonction d'activation relu, en plus d'être plus proche du fonctionnement réel du cerveau, permet au gradient d'être plus efficace [7] et réduit la dépendance aux préentrainements, en particulier pour les réseaux de neurones profonds. Cette fonction produit des couches creuses (dont une partie vaut 0). Ces réseaux aux représentations creuses ont obtenu de meilleurs résultats qu'en utilisant des fonctions sigmoïdes ou tangentes hyperboliques sur de nombreuses tâches (Glorot et al., 2011 ; Maas et al., 2013 ; Zeiler et al., 2013 ; Nair et Hinton, 2010 ; Szegedy et al., 2015 ; Moschitti et al., 2014). En contrepartie, les neurones avec fonction d'activation relu ont tendance à "mourir", c'est-à-dire rentrer dans un état où le neurone sera systématiquement inactif sans possibilité de sortir de cet état. Les neurones morts réduisent la capacité de modélisation du réseau. La mort de neurones arrive le plus souvent lorsque le gradient est trop important et qu'il provoque un changement trop rapide des paramètres du réseau. Plusieurs fonctions d'activations sont introduites pour contrer cet effet (He et al., 2015 ; Clevert et al., 2015), parmi celles-ci la softplus qui est visible dans la Figure 3.13 et qui est définie ainsi :

$$f(x) = log_e(1 + e^x) \tag{3.3}$$

3.6.3 Des algorithmes de descente de gradient adaptatifs

La méthode de descente de gradient classique présente plusieurs faiblesses :

recognition. preprint arXiv :1412.5567.

7. En réduisant l'effet du problème dit de *Vanishing gradient*

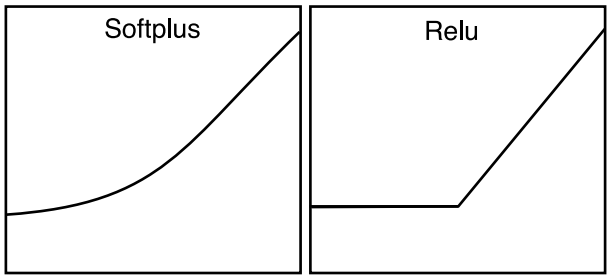


Figure *3.13* – *Fonction d'activation relu et softplus (Glorot et al., 2011).*

1. Le **choix du taux d'apprentissage** est une opération sensible avec cette méthode. Un taux trop petit peut piéger le système dans un minimum local alors qu'un taux trop grand risque de faire diverger l'apprentissage.
2. Ce taux d'apprentissage est global, ce qui signifie que tous les neurones utilisent le même taux alors que toutes les données ne suivent pas forcément la même distribution et donc ne nécessitent pas d'adapter le réseau de la même manière.
3. La gestion des **Points-col** qui apparaissent quand le gradient a une pente positive sur une dimension et une pente négative sur une autre. La présence d'un point-col provoque des plateaux qui peuvent bloquer l'apprentissage (Dauphin et al., 2014).

L'Adagrad (Duchi et al., 2011), à base de gradient adaptatif est une méthode introduite pour pallier ces problèmes. Cette méthode introduit un gradient différent pour chaque dimension du vecteur de sortie qui dépend du gradient actuel et des précédents. Cela permet à l'apprentissage d'être plus robuste au choix du taux d'apprentissages de départ et de converger plus rapidement vers de meilleures solutions. Plusieurs méthodes dérivées de l'Adagrad sont aussi couramment utilisées pour obtenir des performances état-de-l'art. Les plus communes sont :

— RMSprop (Tieleman et Hinton, 2012) adapte le gradient en fonction de son moment de second ordre.

— Adadelta [8] propose une adaptation du gradient dépendante de son moment de second ordre et de celui de l'état de la couche du réseau corrigé.

— Adam (Kingma et Ba) propose une variante d'adaptation où le gradient dépend de ces moments de premier et second ordre ainsi que du temps.

3.6.4 Les méthodes de régularisation

Les méthodes de régularisation sont des variations introduites dans l'apprentissage d'un réseau pour forcer des comportements et améliorer les capacités de généralisation du modèle appris. Deux méthodes de régularisation sont devenues particulièrement populaires. La première, le *Dropout* (Srivastava et al., 2014), est une méthode qui consiste à retirer de l'apprentissage une partie des neurones du réseau (en général 15 à 50% par

8. (Zeiler, 2012) M. D. Zeiler, 2012. Adadelta : an adaptive learning rate method. preprint arXiv :1212.5701.

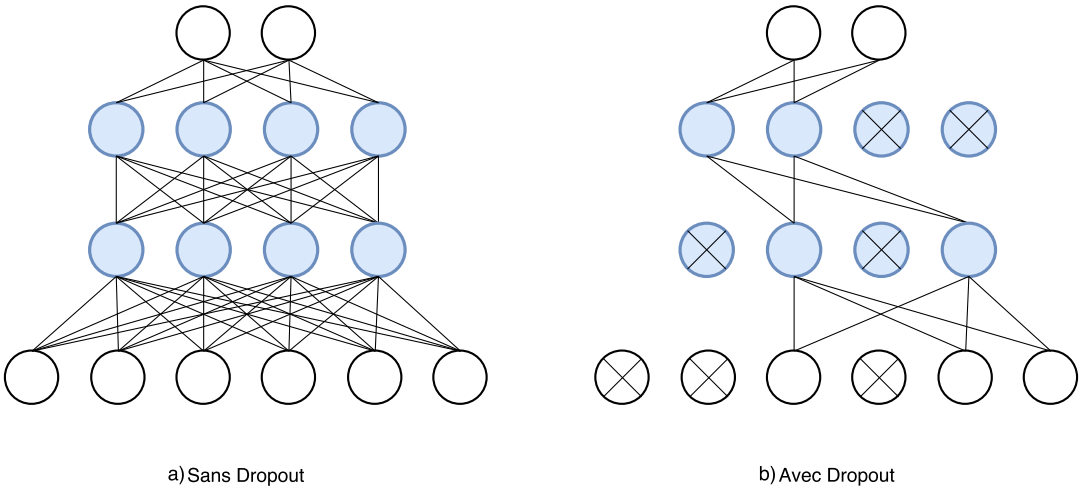


Figure *3.14* – *Schéma de deux réseaux de neurones avec et sans dropout.*

couche) lors de l'entrainement comme montré par la Figure 3.14. Le *Dropout* s'attaque au problème de coadaptation des neurones dans le réseau. La coadaptation est une situation où plusieurs neurones d'une même couche sont utilisés pour modéliser une information. D'après les travaux de (Srivastava et al., 2014) les coadaptations dites complexes apprises par un réseau ne sont pas toujours nécessaires et introduisent deux problèmes :

1. la diminution des capacités de modélisation du réseau. Si plusieurs neurones modélisent la même information, alors ils sont perdus pour en modéliser de nouvelles.
2. une tendance au surapprentissage. En effet des coadaptations simples généralisent mieux que les coadaptations complexes.

Le *Dropout* est donc une solution peu couteuse en calcul qui va permettre d'améliorer les capacités de modélisation et de généralisation du réseau. En contrepartie, le nombre d'itérations nécessaires à l'apprentissage est augmenté.

La seconde méthode, plus récemment introduite, est la normalisation par lot (batch normalization, BN) (Ioffe et Szegedy, 2015 ; Cooijmans et al., 2017). Elle peut être utilisée lorsque les vecteurs d'entrée sont transmis en lots dans le réseau pour accélérer la vitesse de traitement. Cette normalisation s'attaque au problème de **changement de covariable interne (internal covariate shift)** du réseau. Celui-ci est lié à la distribution des neurones activés dans une couche. En effet, lors de la descente du gradient, les poids d'activation des neurones changent et donc leur probabilité d'activation est aussi modifiée. La couche suivante du réseau doit compenser ce changement en plus d'adapter sa modélisation. La BN permet de lisser cet effet. Ainsi le réseau n'a plus besoin d'apprendre à compenser à chaque itération. L'application de la normalisation est réalisée à chaque dimension des vecteurs d'entrée ou des vecteurs cachés d'un même batch selon la fonction suivante :

$$\tilde{x} = \frac{x - E[x]}{\sqrt{Var(x)}} \tag{3.4}$$

Où x est une dimension de la matrice contenant les lots d'entrées, $\tilde{x}$ la même dimension normalisée. L'espérance $E(x)$ et la variance $Var(x)$ sont calculées sur la batch en cours

lors de l'apprentissage. Pour permettre au réseau de réaliser des prédictions, les espérances et les variances de chaque dimension sont calculées sur un corpus d'entrainement indépendamment.
Réduire l'effet de changement de covariable interne lors de l'adaptation des poids d'un réseau lui permet de mieux généraliser, de converger plus rapidement et dans certaines conditions d'obtenir un effet similaire au dropout (Ioffe et Szegedy, 2015 ; Cooijmans et al., 2017).

3.7 Conclusion

Les réseaux de neurones offrent un cadre de travail souple et robuste dont les intérêts ont été largement démontrés expérimentalement. Ils ont permis de franchir des étapes importantes pour le développement de l'apprentissage automatique, notamment en reconnaissance automatique de la parole (Yu et al., 2010), et en traitement de l'image (Krizhevsky et al.) où des performances de systèmes automatiques sont comparables à celles obtenues par les humains dans certaines conditions (He et al., 2016 ; Xiong et al., 2017b). Les très grands corpus rendus disponibles récemment par exemple le Yahoo news feeds dataset [9], le Google audio set [10] et le Google video set [11] couplés aux puissantes capacités de modélisation des réseaux profonds qui évoluent rapidement, laissent présager l'exploration de nouveaux domaines de recherche ainsi que de nouvelles applications intéressantes.

Dans cette partie, nous avons introduit les problématiques de compréhension de la parole (SLU) et les solutions proposées dans la littérature. Nous avons vu qu'une grande partie de ces méthodes repose sur une étape de transcription automatique de la parole. Les méthodes classiques, pour créer des descripteurs efficaces, sac-de-mots (TF.IDF) ou probabilistes (LDA), afin d'exploiter automatiquement les transcriptions issues de la reconnaissance automatique de la parole, ont été présentées dans le Chapitre 2. L'utilisation de ces représentations dans des tâches de compréhension de la parole est efficace dans des contextes précis tels que dans le cas de la parole lue. Dans des contextes plus complexes et variés, les performances sont drastiquement dégradées, notamment à cause d' erreurs introduites par la reconnaissance automatique de la parole. Nous avons présenté dans ce chapitre des réseaux de neurones artificiels profonds ainsi que leurs différentes applications. Parmi ces architectures, nous avons abordé les autoencodeurs qui ont été employés avec succès pour construire des représentations robustes en milieux bruités. La Partie II présente l'utilisation de réseaux de neurones pour construire des représentations de meilleure qualité, facilitant la classification de documents parlés. La Partie III introduit l'utilisation de réseaux de neurones pour fusionner plusieurs sources d'informations d'un même document (contexte multivues) afin de compenser l'impact négatif des SRAP sur les systèmes de classification thématique de dialogue.

9. https ://yahoo.tumblr.com/post/137282204964/yahoo-releases-the-largest-ever-machine-learning
10. https ://research.google.com/audioset/
11. https ://research.google.com/youtube8m/

Deuxième partie

Contribution : Projection neuronale d'informations pour la classification thématique de documents parlés

Chapitre 4

Problématique et cadre expérimental

Sommaire

4.1 Introduction 57
4.2 Tâche d'Identification de Thématique (TI) 58
4.3 Les systèmes de référence 61
4.4 Conclusion 65

4.1 Introduction

Le signal audio est porteur de types d'informations variés, ce qui rend son interprétation complexe pour un système automatique. Pour résoudre un problème de compréhension du langage (SLU), un système automatique doit interpréter les informations transmises par la parole. L'extraction de ces informations parlées est le plus souvent réalisée par un système de reconnaissance automatique de la parole (SRAP). Un SRAP produit des hypothèses de transcription dissociées des autres informations acoustiques. Les résultats de transcription sont bien souvent imparfaits. Ces imperfections s'expliquent par la présence de dysfluences ou de conditions acoustiques sous-optimales. Les transcriptions erronées introduisent des risques d'erreurs de compréhension. Les performances d'un système de SLU sont donc fortement dépendantes des résultats de transcriptions automatiques. Les contributions de ce manuscrit proposent d'exploiter des représentations neuronales de transcriptions pour améliorer les performances des systèmes de SLU. Dans ce chapitre, nous présentons, dans un premier temps, une tâche d'identification de thématique (TI) qui sera utilisée, tout au long du manuscrit, pour évaluer des systèmes automatiques. Dans un second temps, des systèmes de référence sont introduits.

4.2 Tâche d'Identification de Thématique (TI)

La tâche d'identification de thématique a été introduite dans la Section 1.3.2 consiste à déterminer la thématique principale d'un segment audio parmi un ensemble prédéterminé de thématiques. Dans ces travaux, nous utilisons une tâche de TI construite à partir des données du projet DECODA [1]. Pour simplification nous appellerons cette tâche «TID».
Les données utilisées dans TID sont constituées d'un ensemble de conversations téléphoniques issues de la centrale d'appel de la Régie Autonome des Transports Parisiens (RATP), qui est en charge des transports publics dans la ville de Paris. Ces conversations ont deux acteurs : un agent RATP (humain) et un client. Le corpus est composé de 1 242 conversations soit environ 74 heures de signal de parole. L'ensemble des conversations du corpus est annoté et transcrit manuellement. Les annotations manuelles comportent des informations thématiques et syntaxiques. Les détails sur les annotations sont accessibles dans (Bechet et al., 2012). Dans la suite de ce document, nous manipulerons deux types de transcription, les transcriptions manuelles notées «TRS» et les transcriptions produites par un système automatique notées «ASR».
Dans le cadre expérimental étudié, nous nous intéresserons particulièrement aux informations thématiques. Chaque conversation a été manuellement annotée par un agent humain parmi les 8 labels suivants :

1. Problème d'itinéraire
2. Objets trouvés
3. Horaires
4. Carte de transport
5. État du trafic
6. Prix du ticket
7. Infractions
8. Offres spéciales

L'hypothèse qu'une conversation puisse appartenir à une unique classe est discutée dans (Hazen, 2011). Cette hypothèse peut être acceptable si la classification correspond à la thématique principale abordée dans une conversation. Par exemple, le schéma 4.1 montre une conversation entre un agent et un client où le thème principal est un problème d'itinéraire. Chaque conversation a été associée à un thème principal, des thèmes secondaires sont également présents, mais ils ne sont pas considérés dans le cadre de ce travail. La répartition des conversations dans les différentes classes est répertoriée dans le Tableau 4.1. [2]
La répartition des dialogues dans les différentes classes est très inégale. Certaines classes sont jusqu'à dix fois moins présentes que la plus représentée. On peut remarquer dans ce tableau que le corpus est découpé en trois parties. Ce découpage est réalisé pour la tâche

1. http ://decoda.univ-avignon.fr/
2. L'utilisation des termes, labels et classes dans ce manuscrit fera toujours référence à ces 8 labels. Alors que les termes thème et topic feront référence à ceux appris par les modèles statistiques. c.f. Chapitre 2

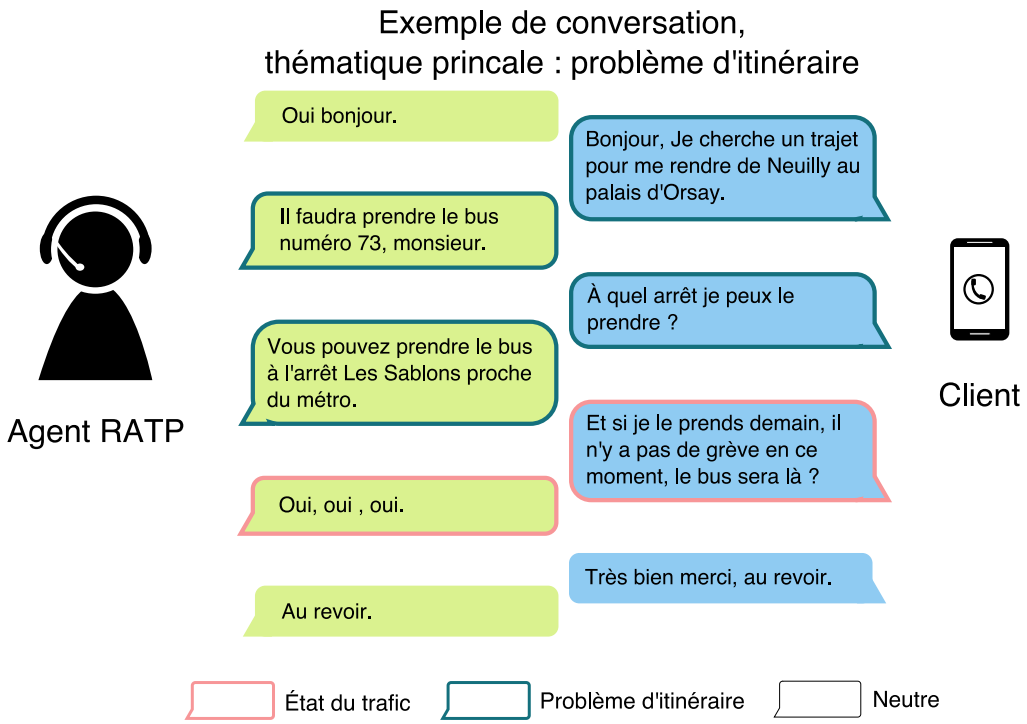


Figure *4.1* – *Exemple de dialogue, transcrit manuellement, extrait de la tâche TID attribué à la catégorie "problème d'itinéraire" par un expert, mais qui contient aussi le thème "État du trafic".*

Table *4.1* – *Description des classes dans la tâche TID.*

	Nombre d' échantillons		
Classes	**Entrainement**	**Developpement**	**Test**
Problème d' itinéraire	145	44	67
Objets trouvés	143	33	63
Horaires	47	7	18
Carte de transport	106	24	47
État du trafic	202	45	90
Prix du ticket	19	9	11
Infractions	47	4	18
Offres spéciales	31	9	13
Total	**740**	**175**	**327**

de classification. Le corpus d'entrainement est utilisé pour entrainer les systèmes automatiques. Le corpus de développement est utilisé pour optimiser les méta paramètres (nombre de couches cachées, nombre de neurones, taux d'apprentissage ...). Le corpus test sert lui à comparer les systèmes proposés.
Les données du projet DECODA ont la particularité d'être directement issues d'une centrale d'appel. Les dialogues ne sont donc pas simulés. Elles représentent donc en matière de qualité et de cohérence un cadre applicatif réel.
La qualité du signal audio est très variable et dépend du type de téléphone et du lieu de l'appel. De plus, les discours ne sont ni préparés ni cadrés. De nombreux phénomènes propres à la parole spontanée sont présents. On entend parfois des gens en condition de stress ou en colère dont la prononciation n'est plus classique. Le vocabulaire aussi n'est pas contraint, beaucoup parmi les mots employés par les clients sont hors du vocabulaire du SRAP. Leur présence rend les conversations difficiles à transcrire et donc leur interprétation encore plus difficile.

Les classes utilisées pour l'annotation n'ont pas été choisies pour faciliter la séparation sémantique. Elles proviennent de l'ontologie utilisée par la RATP. Cette ontologie est choisie pour répondre à un besoin applicatif. Par conséquent, d'un point de vue sémantique, certaines classes ont une intersection forte. Enfin, la répartition des documents dans les différentes catégories est très inégale. Toutes ces difficultés cumulées rendent les classes difficiles à distinguer les unes des autres pour un système automatique.
Attribuer à chaque document une classe thématique revient donc à une tâche de classification utilisant les informations extraites du signal parlé. L'évaluation de cette tâche se fait par la métrique de précision multiclasse P, qui évalue la proportion de labels correctement attribués par le système comme présenté dans l'équation (4.1).

$$P = \frac{\sum_{i=1}^{c} Dj_i}{\sum_{i=1}^{c} D_i} \times 100 \tag{4.1}$$

Où Dj_i représente le nombre de documents correctement classifiés dans la classe i et D_i

le nombre de documents dans la classe i avec c le nombre de classes. Elle est exprimée en pourcentage.

4.3 Les systèmes de référence

Nous allons maintenant présenter le processus de labellisation automatique et les systèmes standards qui serviront de référence pour évaluer l'apport de nos contributions. On peut représenter le processus de labellisation par un système en trois étapes comme illustré dans le schéma 4.2.

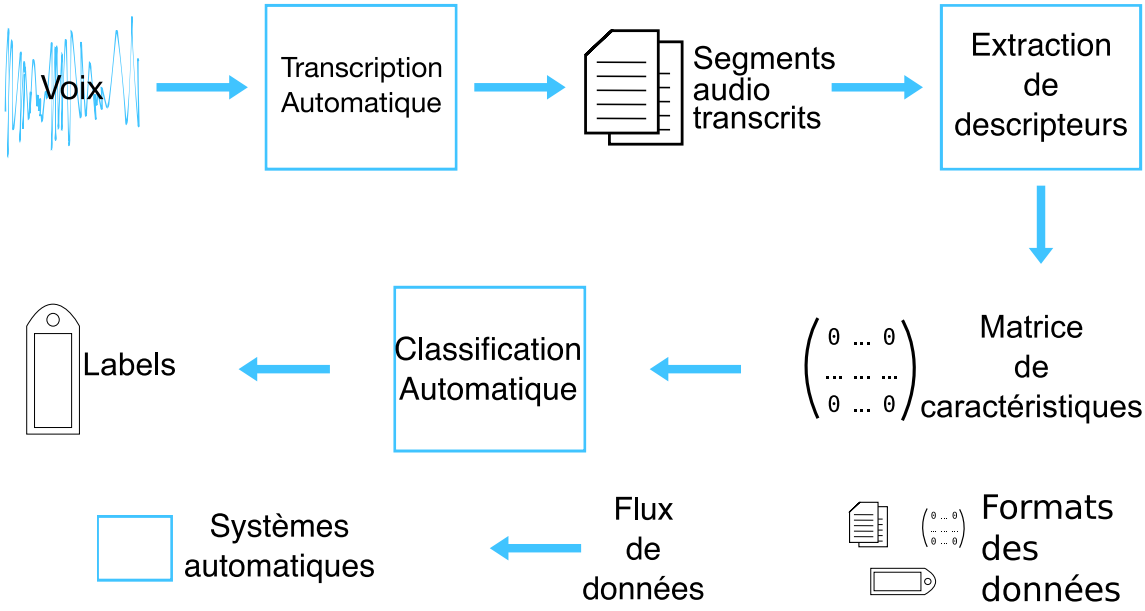


Figure *4.2* – *Processus simplifié de labellisation de conversation pour la tâche TID.*

La première étape correspond à la transcription automatique de la parole. La deuxième étape correspond à l'extraction des descripteurs nécessaires pour la troisième étape, la classification. (Lee et al., 2015b) expliquent que ce processus se résume souvent à l'enchainement du système de reconnaissance de la parole et du système de recherche d'informations avec des systèmes simples dans l'étape deux. Les contributions qui seront présentées dans ce manuscrit interviennent dans l'étape d'extraction des descripteurs. En ajoutant de «l'intelligence» dans cette étape, nous pouvons créer des représentations de l'information plus adaptées à la classification et moins sensibles aux erreurs introduites par le SRAP. Ces étapes sont détaillées dans la section suivante.

La reconnaissance automatique de la parole.

La première étape de reconnaissance automatique de la parole permet de passer d'une représentation en signal sonore de bas niveau, à une suite d'hypothèses de trans-

cription (mots ordonnés chronologiquement). La complexité des signaux audios rend la segmentation et l'extraction de concept de haut niveau bien plus efficace sur des données textuelles que sur le signal directement (De Mori et al., 2007). Il est donc nécessaire de recourir à un SRAP pour pouvoir abstraire la variabilité et la complexité du signal. En contrepartie, sur certains types de documents, le SRAP introduit une importante quantité d'erreurs qui rendent sous optimal l'exploitation des prédictions du système.

Dans le cadre des expériences sur le corpus de la tâche TID, la transcription automatique de la parole est réalisée avec le SRAP LIA-Speeral (Linarès et al., 2007). LIA-Speeral utilise des modèles acoustiques triphone appris par un modèle de mélange de gaussiennes utilisant 230 000 gaussiennes. Les paramètres du modèle sont estimés par «maximum à postériori» (MAP) sur 150 heures de parole en condition téléphonique.
Le vocabulaire du SRAP contient 5 782 mots. Le modèle de langue (ML) trigramme utilisé est obtenu en interpolant deux modèles. Le premier est un ML générique appris à partir de données textuelles comme Wikipedia, des dépêches AFP et les transcriptions manuelles de campagnes d'évaluation françaises. Le second est appris à partir des transcriptions manuelles issues du corpus d'entrainement de DECODA.

Un SRAP est évalué au moyen de la métrique du Taux d'erreur mot (word error rate, WER). Le WER évalue le nombre d' erreurs introduites par le système comparé au nombre de mots correctement transcrits. Il est exprimé en pourcentage.

Dans ces conditions, le système LIA-Speeral obtient un WER de 33,8% sur le corpus d'entrainement, 45,2% sur le développement et 49,5% sur le test. Ces taux d'erreurs sont fréquents sur ce type de données et représentatifs des problématiques peu contrôlées réelles. Des résultats comparables sont obtenus dans (Garnier-Rizet et al.). Dans ces conditions d'évaluation, environ un mot transcrit sur deux est une erreur. C'est une mesure du bruit introduit par le SRAP. Les systèmes qui reposent sur le SRAP vont donc devoir être capables de gérer ces informations manquantes, superflues ou trompeuses. Ces forts taux d'erreurs s'expliquent notamment par les spécificités des locuteurs, des environnements bruités, mais aussi des déformations linguistiques dues à la conversation spontanée (hésitations, reprises, répétitions, erreurs de grammaire, paroles simultanées, …). Notons qu'une «stop list» de 126 mots outils[3] (déterminants, articles, etc.) sont supprimés avant le calcul du WER et ne sont plus utilisés ensuite. L'absence de ces mots peut avoir un impact sur les WER présentés ci-dessus, mais nous n'avons pas mesuré cet impact.

Pour faciliter la suite de la chaine de traitement, il serait envisageable de réduire le WER en tentant d'améliorer le SRAP. Cependant, conserver un taux d'erreur mots important est plus représentatif des cas applicatifs réels, où les caractéristiques acoustiques environnementales ne sont pas contrôlées. C'est le cas notamment avec les problématiques de l'entreprise Orkis qui finance ces travaux. Dans leur système informatique, le système d'extraction d'informations de documents parlés est utilisé sur des données envoyées par les utilisateurs de tous niveaux d'expertise. Il est donc évident que le SRAP produira des erreurs sur au moins un sous-ensemble de ces documents.

3. http ://code.google.com/p/stop-words/

L'extraction d'informations.

Le second processus de traitement correspond à l'étape de prétraitement des résultats du SRAP et d' extraction d'informations. Les caractéristiques utilisées pour représenter un document peuvent être les mots employés, leurs relations dans la phrase ou la structure de la conversation. Le choix des caractéristiques pertinentes à produire à partir de documents textuels est un élément déterminant pour tout système d'extraction d'informations. Ce problème est encore plus complexe dans le cadre de documents parlés, puisque les mots exploités sont les hypothèses issues d'un SRAP avec des erreurs de transcription. Malgré ces difficultés, de bons résultats ont été obtenus sur des tâches d'analyse de la parole (Melamed et Gilbert, 2011), d'identification de thématiques (Lagus et Kuusisto, 2002 ; Hazen, 2011) et de la segmentation (Eisenstein et Barzilay, 2008 ; Purver, 2011). D'autres tâches d'analyses de conversations parlées sont détaillées dans la Section 1.3.

L'objectif de cette étape est de retenir et de mettre en forme les informations les plus pertinentes et les plus utiles possible pour déterminer une thématique. Ceci permet de faciliter le travail du classifieur. Dans les systèmes initiaux, deux méthodes différentes sont utilisées. Les deux méthodes commencent par une étape de filtrage des mots vides, basée sur la «stop list» de 126 mots (c.f. Section 4.3).

La première méthode représente les documents sous forme de sac-de-mots binaires (BBOW, c.f. Chapitre 2) avec un vocabulaire d'environ 7000 mots sans les mots outils. Cette représentation génère des vecteurs très larges et principalement creux. Elle est simple à mettre en place, mais donne généralement des résultats limités.

La seconde méthode (TF.IDF) réalise un prétraitement classique pour la recherche d'informations. D'abord un score de discrimination est attribué à chaque mot dans chaque thématique. Ce score repose sur la combinaison TF.IDF.GINI comme décrite dans le Chapitre 2.3. Ensuite, les 100 mots les plus discriminants par classe sont sélectionnés et cumulés. Si des mots discriminants sont présents dans plus d'une classe parmi les huit, ils ne sont comptés qu'une fois. Le vocabulaire d'environ 7000 mots est donc réduit à un ensemble de 707 mots discriminants qui sera le vocabulaire conservé pour la classification. Ensuite, pour chaque document est attribué un vecteur sac-de-mots de la taille du vocabulaire discriminant, où chaque mot est lié à son score TF.IDF.GINI. Cette représentation est moins large et plus informative que la précédente, mais elle est toujours principalement creuse.

Les classifieurs

La troisième étape de classification attribue un label à chacun des documents. Pour la classification nous avons choisi d'utiliser un perceptron multicouche (MLP), présenté dans le Chapitre 3, pour ses performances générales et sa vitesse d'apprentissage sur GPU. La sélection des bons métaparamètres pour le réseau tels que sa taille, sa profondeur, le type de lot entre autres sont cruciaux pour son efficacité.

Les descripteurs issus du corpus d'entrainement sont utilisés pour entrainer des MLP

avec différents métaparamétres. Les meilleures caractéristiques du MLP sont déterminées via leurs performances sur la partie développement du corpus. La performance du système est ensuite évaluée sur le test. Ce principe permet aux scores de précision d'être plus représentatifs des performances et des capacités de généralisation du modèle qui sont particulièrement importantes dans un cadre applicatif pour TID. La configuration retenue du MLP est décrite dans le Tableau 4.2.

Les précisions des deux systèmes présentés ci-dessus sont répertoriées dans le Tableau 4.3. À titre de comparaison, le tableau contient aussi les résultats d'un SVM utilisant des caractéristiques TF.IDF.GINI proches décrites dans (Morchid et al., 2014c) ainsi que les résultats d'un système MLP qui utilise un oracle (transcription manuelle) au lieu du SRAP.

Table *4.2* – *Métaparamètres sélectionnés pour le MLP appliqué sur TID.*

	Nom	Taille (en neurones)	Activation
Architecture	Entrée	7 400 (taille du vocabulaire)	-
	C. Cachée 1	512	Tanh
	C.Cachée 2	256	Tanh
	Sortie	8 (nombre de labels)	Softmax
Générale	Fonction de cout	Descente de Gradient	
	Entropie Croisée	Adam	

Table *4.3* – *Précisions des méthodes simples sur TID et les résultats MLP TRS dans le cas où des transcriptions idéales (manuelles) remplacent le SRAP.*

Classifieur	Entrée	Précision	intervalle de confiance ($p < 0.05$)
MLP	BOW - Binaire	59,0%	+/- 5%
MLP	BOW - TF.IDF.GINI	**77,1%**	+/-4,7%
SVM	BOW - TF.IDF.GINI	73,5%	+/-4,9%
MLP TRS	BOW - TF.IDF.GINI	83.4%	+/-3,7%

Il est intéressant de voir dans ces résultats que le MLP qui repose sur le SRAP est au moins 6,3% moins bon que le système qui repose sur un oracle. Ce qui confirme que les erreurs introduites par le SRAP ont un impact négatif sur les performances globales du système. En contrepartie, la chute en performance est relativement faible au regard des 49,5% de *WER*. On peut en déduire que les mots porteurs de sens sont majoritairement correctement transcrits, ce qui permet aux classifieurs de détecter le plus souvent le bon label. Comme attendu, les caractéristiques plus élaborées TF.IDF.GINI réalisent de meilleurs résultats que des sacs de mots binaires. Les intervalles de confiances montrent que la taille du corpus de test rend les différences de performances faiblement significatives. Notamment lorsque sont comparé les résultats du classifieur SVM de la littérature et les résultats du classifieur MLP que nous proposons. Pour compenser cet effet les résultats présentés dans la suite du manuscrit correspondent aux meilleurs résultats de plusieurs évaluations et dans le Chapitre 9.

4.4 Conclusion

Les performances globales obtenues sur cette tâche peuvent être améliorées selon trois axes.

Le premier axe est l'amélioration des performances du SRAP. En effet, la comparaison des performances entre le SRAP et l'oracle montre clairement qu'une meilleure transcription automatique améliorerait les résultats du système. Cet axe est un domaine de recherche en plein essor qui mobilise beaucoup de chercheurs depuis 1930 (Juang et al., 2004) et encore aujourd'hui (Xiong et al., 2017b). Malgré les très bonnes performances obtenues récemment (Xiong et al., 2017b), dans un contexte industriel/applicatif réel il est fréquent de travailler avec des données non maitrisées lexicalement et acoustiquement. Rencontrer des transcriptions de mauvaise qualité est donc inévitable. Il est nécessaire pour des systèmes de compréhension de la parole de considérer ces situations.

Le second axe porte sur l'étape de classification en remplaçant le MLP par un classifieur plus efficace ou plus robuste. On pourrait par exemple évaluer les récentes «Deep Forest» (Zhi-Hua Zhou, 2017) ou créer une architecture de réseau de neurones très profonde. Cet axe, bien que fortement susceptible d'améliorer les performances du système, est aussi le plus intensément étudié par la communauté et nécessite plus de ressources pour produire des systèmes de plus en plus complexes (Esteve et al., 2015 ; Xiong et al., 2017b ; Chan et al., 2016 ; Simonyan et Zisserman, 2016 ; He et al., 2016) pour une tâche où la quantité des données disponibles est relativement faible et le cout d'annotation important.

La troisième approche possible est de travailler au niveau de l'extraction de caractéristiques. Les réseaux de neurones fournissent un cadre de travail intéressant qui permet de créer de nouvelles représentations des documents pour améliorer les capacités de généralisation du modèle, sélectionner l'information pertinente, essayer de combler la différence entre représentation issue d'un SRAP et représentation transcrite manuellement. C'est cet axe d'amélioration qui est étudié dans ce manuscrit.

Le prochain Chapitre va aborder l'utilisation des caractéristiques vectorielles abstraites des mots produites par des réseaux de neurones *Word2vec*. Ces descripteurs apportent une modélisation sémantique nouvelle et des possibilités de généralisation qui peuvent améliorer l'identification de thématiques et rendre le système moins sensible aux erreurs du SRAP.

Chapitre 5

Représentations distribuées des mots pour la compréhension de documents bruités

Sommaire

5.1 Introduction 67
5.2 Les Architectures Word2vec 69
5.3 Intégration de l'information de position 71
5.4 Analyses des modèles proposés et application à la compréhension de la parole 77
5.5 Conclusion 83

5.1 Introduction

Les représentations classiques en sac-de-mots avec TF.IDF (présentés dans le Chapitre 2), encodent les documents dans un espace vectoriel où chaque mot est une dimension de l'espace. Cette forme peut difficilement modéliser les relations entre les mots et leurs contextes. De plus, l'espace ainsi construit a autant de dimensions que de mots présents dans le vocabulaire. Ce nombre de dimensions élevé est un frein pour les systèmes d'apprentissage automatique (Bengio et al.). Proposées comme une alternative, les représentations distribuées des mots appelées «*embeddings* de mots» sont rapidement devenues extrêmement populaires. Elles utilisent l'information de cooccurrence pour créer un codage des mots en les projetant dans un espace multidimensionnel de taille contrôlée. Elles introduisent une distance intermots qui porte une information de relation sémantique ou grammaticale. Par exemple, en projetant les mots dans un espace produit par une représentation distribuée, le mot «hélicoptère» peut être plus proche du mot «aviation» que du mot «héliographe». Cette information est impossible à modéliser dans un sac-de-mots.

Les méthodes les plus courantes d'estimation de représentations distribuées sont définies pour être simples, rapides et non supervisées. L'intuition derrière ces choix est qu'un modèle simple capable de traiter de grandes quantités d'exemples sera plus efficace qu'un modèle complexe qui nécessiterait des temps de calcul exorbitants pour traiter les mêmes données. Elles nécessitent par contre de grandes quantités de données pour apprendre des distributions efficaces des mots.
L'algorithme le plus populaire dans la littérature pour réaliser des représentations distribuées est une méthode à base de réseaux de neurones appelée *Word2vec*. Elle a déjà prouvé son efficacité sur de nombreuses tâches. Pour une liste des applications de *Word2vec* et des alternatives , le lecteur peut se référer au Chapitre 2. Dans les travaux présentés ci-dessous, des projections *Word2vec* sont intégrées à un processus de compréhension de la parole et évaluées sur la tâche TID (c.f. Chapitre 4).

L'utilisation de représentations distribuées pour une tâche applicative apporte un certain nombre de difficultés. En effet, apprendre une représentation distribuée nécessite une grande quantité de données. Les données utilisées pour l'apprentissage doivent être proches des données d'application. Un grand nombre de documents spécifiques à une tâche n'est pas toujours disponible. C'est notamment le cas pour TID dont le nombre de données dédiées à l'apprentissage est limité. Il est donc nécessaire d'utiliser des ressources textuelles supplémentaires génériques, mais suffisamment proches, pour apprendre une représentation distribuée efficace. De plus une fois l'apprentissage réalisé, représenter des documents à partir des projections des mots est une opération complexe.

Il est apparu durant nos travaux que pour construire ces représentations, *Word2vec* ne prend pas en compte l'information d'ordonnancement des mots dans la phrase. Une évolution des modèles *Word2vec* est proposée et évaluée dans ce chapitre pour combler cette lacune. Les réseaux de neurones *Word2vec* utilisent, pendant la phase d'entrainement, chaque mot et sa fenêtre de contexte relative (mots qui l'entourent). Ces deux informations sont représentées par un sac-de-mots binaires comme présenté dans le Chapitre 2. C'est cette représentation binaire qui ignore l'information d'ordre d'apparition des mots. Elle est aussi responsable de l'absence de différenciation (même importance) des mots dans le contexte.

Les travaux présentés dans ce chapitre proposent une évolution de l'implémentation *Word2vec* (Mikolov et al., 2013b), dans laquelle une pondération originale est utilisée pour modéliser l'information liée à la position des mots. Cette pondération est basée sur les contextes continus (Rubino, 2011) qui attribuent un score aux mots en fonction de leur distance relative. Ils ont été introduits comme descripteurs de documents textuels pour une tâche de traduction automatique. Ces contextes continus ont également été utilisés (Bigot et al., 2013) afin de modéliser les contextes proches pour de la détection d'entités nommées. Cette approche permet d'introduire, en entrée du réseau de neurones, l'information de position relative des mots dans une fenêtre de contexte. Cette approche est motivée par l'hypothèse de distributivité (Harris, 1954) qui implique que le sens des mots peut être induit par son contexte proche. Nous en avons déduit que les contextes proches doivent avoir un impact plus important sur la représentation latente que les plus distants. Cependant, il est également important que les occurrences très distantes, malgré leur importance réduite, ne soient pas ignorées. On remarquera que cette méthode

permet de réduire l'impact du métaparamètre de la taille du contexte, puisque le réseau peut déterminer la distance utile du mot via la pondération. En contrepartie, la sélection d'une bonne fonction de pondération peut jouer un rôle crucial dans la qualité du modèle.

La pondération proposée est évaluée de deux manières. D'abord, par la tâche d'analogie *Semantic-Syntactic Word Relationship test* (Mikolov et al., 2013b) (présentée dans la Section 5.4) qui permet l'évaluation qualitative d'un modèle *Word2vec* . Ensuite, les modèles d'*embeddings* avec et sans position relative des mots, sont évalués quantitativement en mesurant leur impact sur la tâche TID décrite dans le Chapitre 4.

Ce chapitre est organisé de la façon suivante : tout d'abord, la Section 5.2 explique le fonctionnement de l'approche à base de réseaux de neurones *Word2vec*. Ensuite, la Section suivante 5.3 détaille la fonction de pondération proposée et l'introduction de la pondération dans les réseaux de neurones. Enfin, les expériences et les résultats sont présentés dans la Section 5.4 avant de conclure ce Chapitre dans la Section 5.5.

5.2 Les Architectures Word2vec

Word2vec est une méthode qui regroupe deux réseaux de neurones définis dans (Mikolov et al., 2013b). L'objectif de ces réseaux de neurones est de construire des *embeddings* de mots (c.f. Chapite 2) en utilisant des mots et leur contexte respectif. Construire des *embeddings* revient à créer un espace multidimensionnel où chaque mot a une position. Une des propriétés particulières de cet espace est que le vecteur entre deux mots contient une information sur la relation sémantique et grammaticale entre ces deux mots. La représentation latente des mots est apprise en maximisant la vraisemblance L qu'un mot soit prédit à partir de son contexte. Les deux architectures proposées dans *Word2vec* sont le *Skip-gram* (SG) et le sac-de-mots continus (continuous bag-of-words, CBOW). Ils sont présentés dans le schéma 5.1. Les deux modèles utilisent des représentations en sac-de-mots binaires comme vecteurs d'entrée et de sortie.

Le modèle **Skip-gram (SG)** est entrainé à prédire le contexte pour un mot donné. La couche d'entrée du SG est un vecteur de la taille du vocabulaire ne contenant que le mot au centre du contexte. Elle est projetée dans la couche cachée du réseau pour produire une représentation dense. Ce vecteur dense est la représentation du mot unique utilisé en entrée. Celle-ci est ensuite projetée à son tour dans la couche de sortie pour générer une prédiction. Cette prédiction est corrigée indépendamment par chaque mot contenu dans la fenêtre de contexte.
Ce réseau maximise la vraisemblance suivante :

$$L = \frac{1}{T} \sum_{t=1}^{T} \sum_{j=t-c, j \neq t}^{t+c} \log p(w_j | w_t) \tag{5.1}$$

où c est un métaparamètre qui définit la taille de la fenêtre de contexte pour chaque mot. T est la taille des données d'entrainement. Le modèle estime une matrice M de dimension $|V| \times n$ où n est la taille de la couche cachée du réseau. Cette matrice de

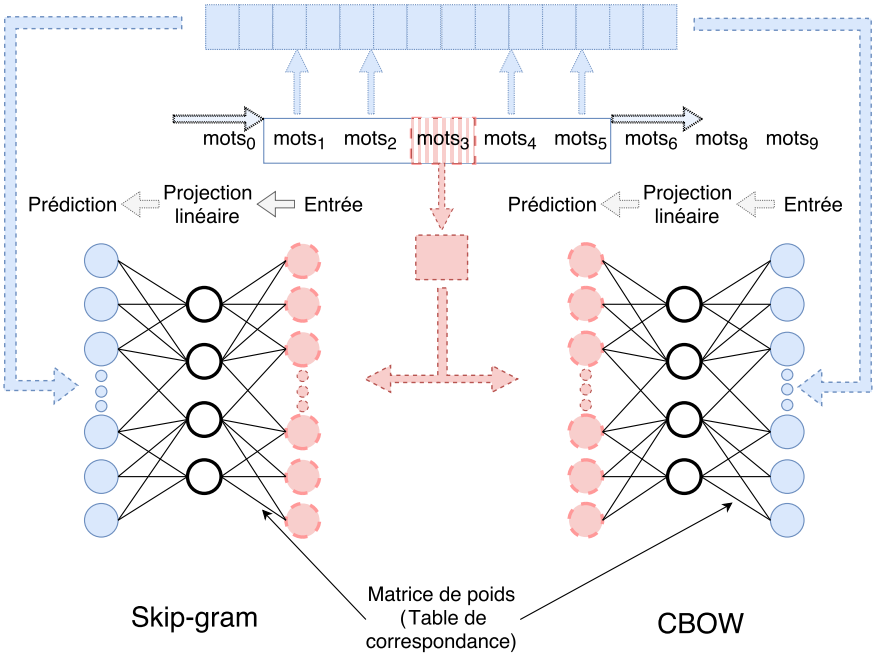


Figure *5.1* – *Les architectures Word2vec.*

poids sert de table de correspondance. Chaque mot est donc lié à un vecteur dense de taille n appelé v_{w_i}. La probabilité $p(w_0|w_i)$ pour le mot w_0 est calculée par la couche de sortie du réseau dont la fonction d'activation est une approximation de la fonction *Softmax* qui est définie dans (Mikolov et al., 2013a). Durant les phases d'apprentissage, le SG réalise une prédiction et une rétropropagation pour chaque mot dans la fenêtre de contexte.

Le modèle **Continuous Bag-of-Words (CBOW)** apprend à prédire le mot w_0 en fonction du groupe de mots qui l'entoure $\{w_{-c},\ldots,w_{-1},w_1,\ldots,w_c\}$. Durant l'étape d'apprentissage, un sac-de-mots binaires est soumis à la couche d'entrée. Ce sac-de-mots contient l'ensemble des mots de la fenêtre de contexte. Cette couche est projetée dans la matrice de poids global M pour devenir la couche cachée. La couche cachée est ensuite projetée dans la couche de sortie pour prédire un mot. La prédiction du réseau est ensuite comparée au mot central du contexte. L'erreur de prédiction est ensuite rétropropagée dans le réseau. Durant l'apprentissage, les matrices de poids du modèle sont adaptées pour maximiser la vraisemblance L définie comme :

$$L = \frac{1}{T}\sum_{t=1}^{T} \log p(w_t|w_{t-c}...w_{t+c}) \tag{5.2}$$

Il est important de noter que le réseau *Skip-gram* introduit un mécanisme pour sauter (*Skip*) des mots aléatoirement. Ce mécanisme va tronquer les bords de la fenêtre de contexte c d'un facteur de réduction $r \in \mathbb{N}$, avec $0 \leq r \leq \frac{|c|}{2}$, choisis par tirage aléatoire pour chaque fenêtre selon une loi uniforme. Toutes les fenêtres de contextes rencontrées sont donc réduites d'un nombre de mots différents. Ce processus est illustré dans le

Schéma 5.2. On peut y voir des exemples de segments d'une phrase et la fenêtre de contexte produite en fonction d'un tirage aléatoire du facteur de réduction.

Ce mécanisme est utilisé pour réduire le temps de calcul lors de l'apprentissage puisque chaque mot supprimé du contexte en diminue le nombre de rétropropagations qui sont les opérations les plus consommatrices en temps de calcul.

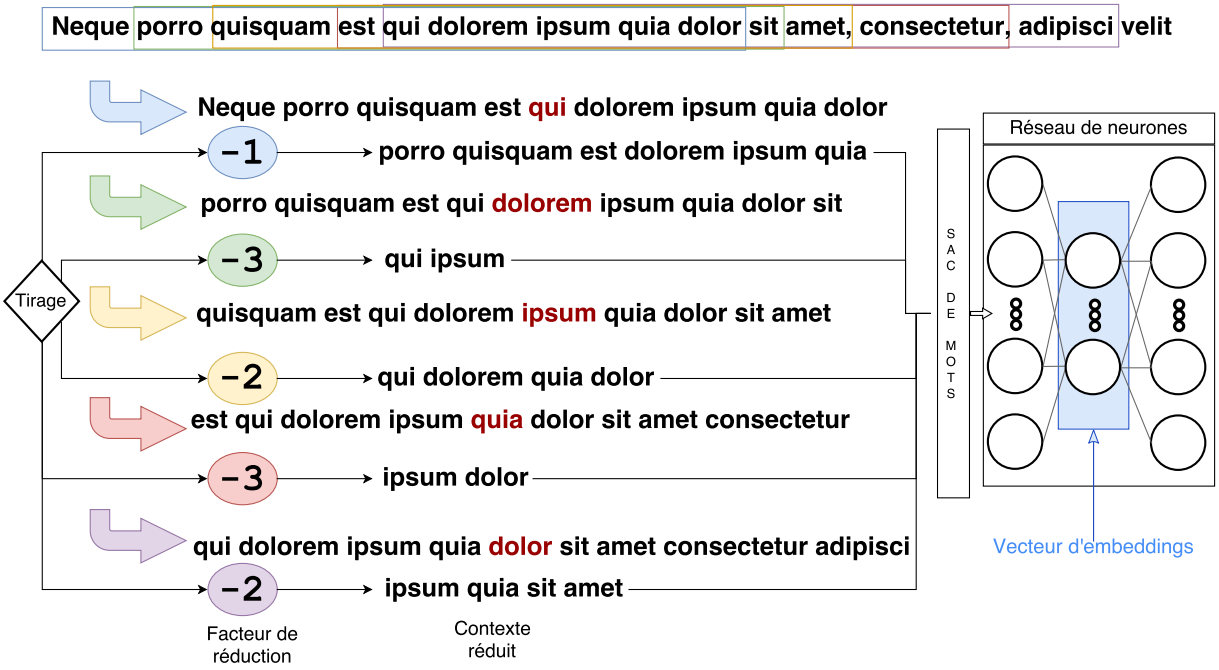


Figure 5.2 – *Exemple du facteur de réduction appliqué à des fenêtres de contexte dans un réseau Skip-gram.*

Le principe de fonctionnement des réseaux SG et CBOW a deux défauts principaux. Le premier, viens du facteur de réduction qui, en réduisant le contexte aléatoirement, ignore des relations rares ou distantes. La seconde difficulté est liée à l'utilisation de sacs-de-mots binaires en entrée et sortie du réseau qui ignore la position des mots dans une fenêtre de contexte. La position d'un mot dans une phrase joue un rôle important sur le sens qu'il porte et la fonction qu'il remplit. La représentation produite est donc potentiellement sous-optimale puisqu'une partie de l'information contenue dans le contexte est ignorée. Une variation des réseaux *Word2vec* est introduite dans la section suivante pour pallier ces problématiques.

5.3 Intégration de l'information de position

La solution que nous proposons tire avantage de la position des mots dans leur contexte pour améliorer les représentations distribuées apprises. Pour cela une fonction de pondération log-linéaire δ remplace les caractéristiques binaires en entrée du réseau

pour les CBOW ou en sortie du réseau pour les *Skip-gram*. Le contexte est pondéré avec $\delta(w)$ pour chaque mot w :

$$\delta(w) = \frac{\alpha}{\gamma + \beta \log(d(w_i))} \tag{5.3}$$

Dans cette équation, $d(w_i)$ correspond à la distance en nombre de mots qui sépare w du centre de contexte et le i$^{\text{éme}}$ mot dans le contexte. α, γ et β sont les paramètres servant à modifier ou adapter la fonction de pondération.
Lors de l'apprentissage par un réseau de neurones *Word2vec* d'une fenêtre de contexte, tous les mots sont traités sans distinction. Mais si le nombre d'occurrences de ce contexte est suffisamment important, la mécanique du facteur de réduction agit comme une pondération aléatoire globale dépendante des tirages du facteur de réduction. La Figure 5.3 compare la pondération locale appliquée par la fonction de pondération introduite et la pondération globale appliquée par le facteur de réduction. La pondération log-linéaire

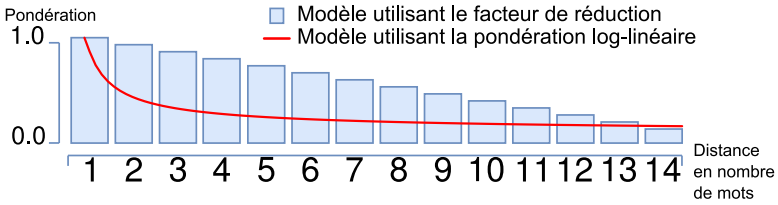


Figure *5.3* – *Comparaison des poids globaux accordés aux mots d'un contexte avec le facteur de réduction et avec la pondération locale par contextes continus.*

et le facteur de réduction ont tous les deux un effet de pondération sur l'importance des mots dans un contexte de tirage infini. La fonction de pondération log-linéaire a deux effets : elle pondère les contextes locaux (à chaque tirage) en retirant l'aléatoire et pondère log-linéairement les contextes de façon globale (pour tous les tirages cumulés). En comparaison, le facteur de réduction applique une pondération linéaire uniquement sur les contextes globaux.

Le choix d'une bonne fonction log-linéaire est déterminant. De plus, pour être cohérente avec l'hypothèse de distributivité, la fonction doit produire une pondération qui attribue un impact important aux mots proches du centre et un poids réduit à ceux qui sont éloignés. En variant les paramètres α, γ et β on peut adapter la fonction pour optimiser son efficacité pour un jeu de données ou pour une tâche en particulier.
La section suivante explicite l'intégration de la pondération dans les réseaux de neurones CBOW et SG.

5.3.1 Le CBOW avec pondération (LL-CBOW)

L'architecture du réseau LL-CBOW est présentée dans le Schéma 5.4. La pondération est appliquée sur le vecteur d'entrée avant la projection dans la matrice de poids globale. Dans ce cadre, la pondération attribue à chacun des mots de la fenêtre de contexte un score. Ce score est ensuite utilisé comme descripteur du mot dans le sac-de-mots en

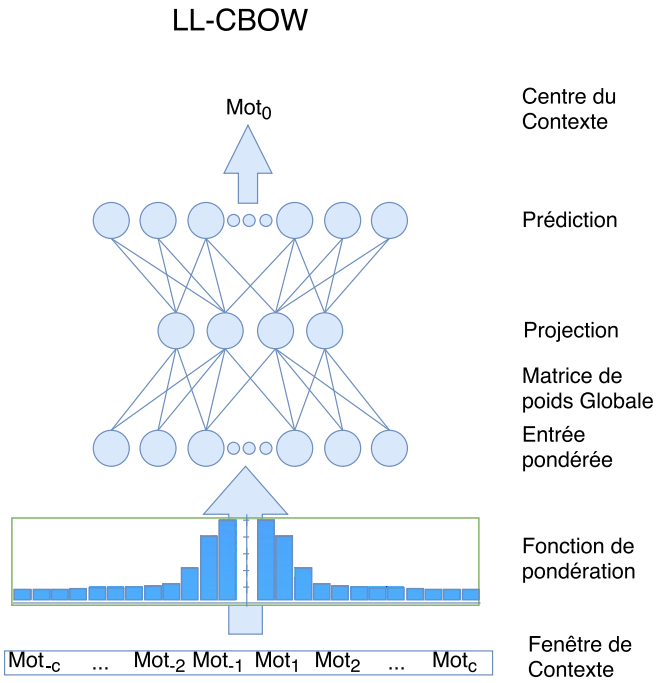


Figure 5.4 – *Le réseau LL-CBOW utilisant une pondération log-linéaire des contextes proposée.*

entrée du réseau de neurones. L'approche proposée diffère du réseau CBOW définie dans (Mikolov et al., 2013a) puisqu'elle remplace les caractéristiques binaires (1 si le mot $w_i \in c$, 0 sinon) par la pondération produite par la fonction $\delta(w)$. Ce modèle appelé LL-CBOW donne donc un poids différent à chaque mot w_i selon la position du mot dans la fenêtre de contexte c en fonction du mot courant w_α.
La pondération a pour effet de limiter ou amplifier l'influence de certains mots lors la création de la projection. Lors de la rétropropagation, le réseau adapte ses poids pour rehausser la vraisemblance des mots présents dans le contexte. Avec la pondération, la vraisemblance sera moins modifiée pour les mots ayant une pondération faible. Ainsi dans l'espace distribué, les mots partageant le même contexte avec un poids fort seront rapprochés. Les mots avec une pondération faible seront sensiblement moins impactés. Ce mécanisme est schématisé dans la Figure 5.5 qui montre un exemple d'évolution de la position des mots dans l'espace *Word2vec* avec ou sans pondération. En rouge sont présentés les mouvements sans pondération et en bleu les mouvements avec pondération. Dans les modèles CBOW et LL-CBOW, les positions dans l'espace de tous les mots du contexte sont modifiées. Le w_0 représente le centre du contexte. Il faut remarquer que les mots proches du centre du contexte (w_1 et w_2) sont plus faiblement impactés par la pondération. Alors que les w_3 et w_4 sont clairement moins déplacés.

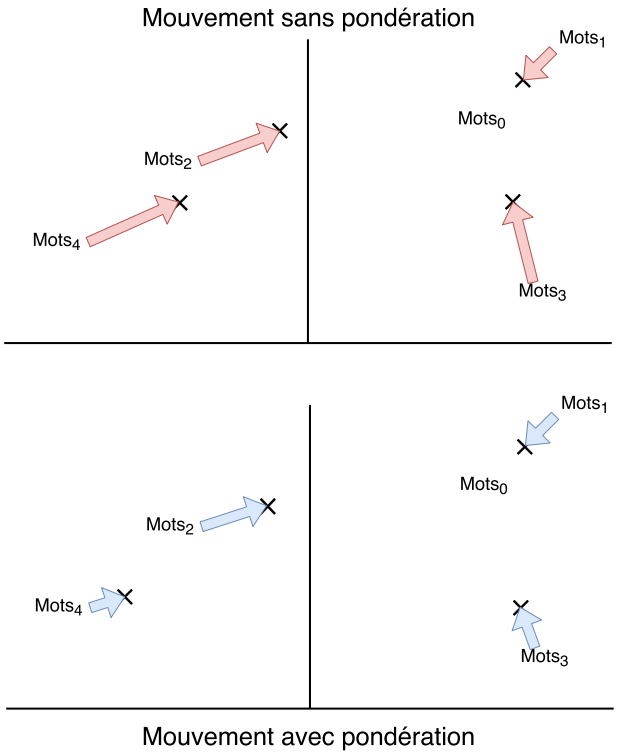


Figure *5.5* – *Mouvements dans l'espace Word2vec durant un apprentissage de réseau CBOW avec et sans pondération.*

5.3.2 Le Skip-gram avec pondération (LL-SG)

L'architecture SG a un fonctionnement différent du CBOW lors de l'apprentissage. La pondération log-linéaire ne peut pas être appliquée de façon similaire dans ce réseau. Lors de son apprentissage, le modèle SG ne prédit pas tous les mots du contexte en une seule fois. Au lieu de cela une prédiction et une rétropropagation indépendantes sont réalisées pour chaque mot contenu dans la fenêtre de contexte. Ce fonctionnement empêche d'appliquer la même pondération que pour le LL-CBOW. Pour le LL-SG, la pondération est introduite lors de la rétropropagation dans le réseau. La Figure 5.6 présente la nouvelle architecture du LL-SG.

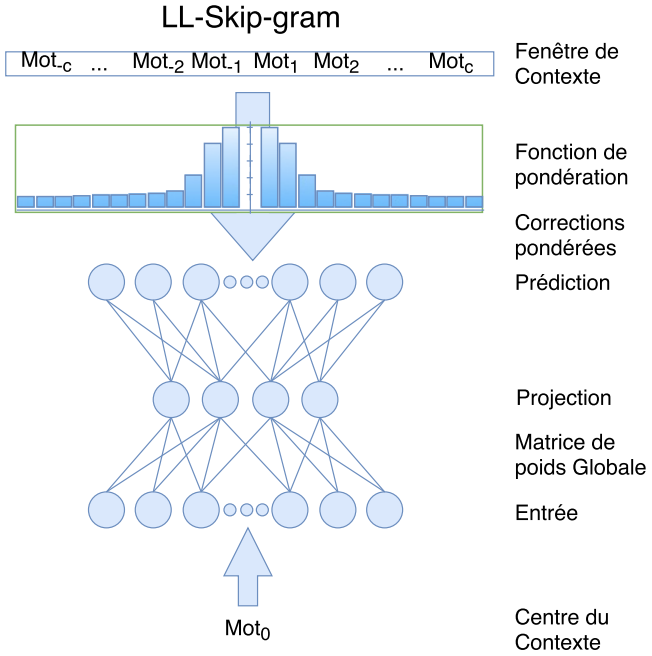


Figure *5.6* – *Le réseau LL-SG avec la pondération log-linéaire des contextes proposée.*

En pratique c'est l'erreur calculée pour un mot donné qui est pondérée en fonction de la position du mot dans le contexte. Ce fonctionnement permet de rapprocher l'impact final de celui de la pondération introduite dans le LL-CBOW. Elle permet donc de réduire les changements dans l'espace distribué impliqué par les mots aux extrémités du contexte. La Figure 5.7 présente un exemple des évolutions dans un espace créé par un réseau *Skip-gram* avec et sans pondération.

Dans cette figure, seul le mot au centre du réseau est affecté par l'apprentissage alors qu'avec les CBOW et LL-CBOW, ce sont les mots du contexte qui évoluent. Les vecteurs en rouge pleins montrent les modifications impliquées par un modèle sans pondération et les vecteurs pointillés montrent les changements réalisés par un modèle pondéré. On peut voir aussi que les mouvements impliqués par les mots proches (1 et 2) sont peu impactés alors que les mouvements des mots distants (3 et 4) sont plus faibles. La nouvelle position du w_0 est significativement différente dans les deux modèles.

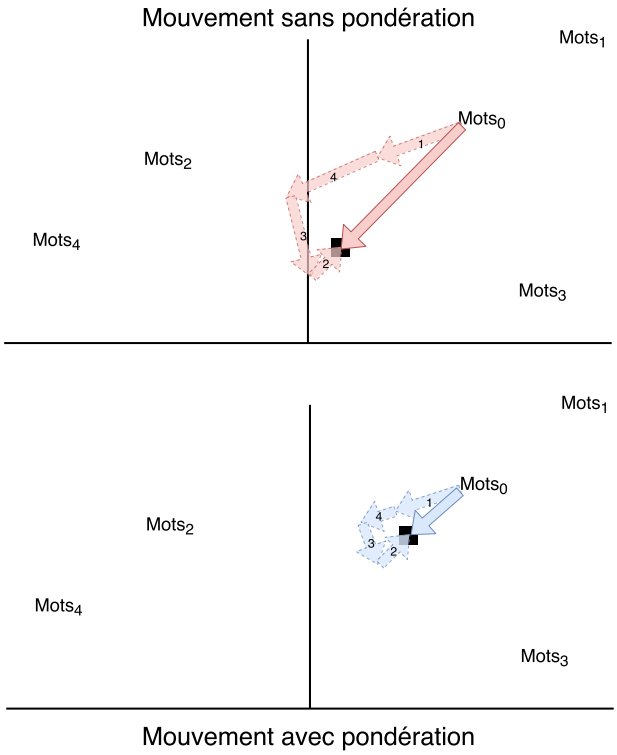


Figure 5.7 – *Mouvements dans l'espace Word2vec durant l'apprentissage de réseau Skip-gram avec et sans pondération.*

5.4 Analyses des modèles proposés et application à la compréhension de la parole

Dans cette section, deux expérimentations que nous avons réalisées sont présentées. D'abord, une évaluation des modèles *Word2vec* avec et sans fonction de pondération sur un test défini par (Mikolov et al., 2013a) est détaillée. Ensuite, l'efficacité de ces modèles est évaluée sur la tâche de compréhension de la parole TID présentée Chapitre 4. Les

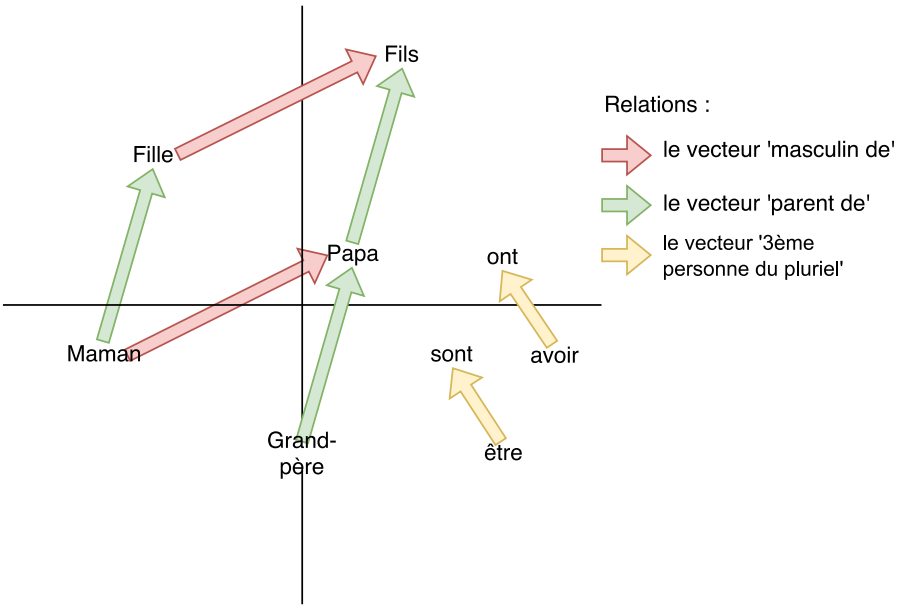


Figure *5.8* – *Exemple des relations apprises par les architectures Word2vec.*

espaces distribués des mots appris par les réseaux *Word2vec* sont capables de modéliser des relations sémantiques et/ou syntaxiques par de simples opérations algébriques (addition et soustraction). La Figure 5.8 montre des exemples classiques de relations sémantiques et grammaticales qui peuvent être capturées par ces modèles. Pour mesurer la capacité d'un espace distribué à correctement modéliser ces relations sémantiques ou syntaxiques, (Mikolov et al., 2013b) introduit un jeu de test appelé le *Semantic-Syntactic Word Relationship test* .
Cette tâche contient un ensemble de 19 000 questions en anglais. Chaque question est composée de deux paires de mots partageant une même relation.Des exemples de questions sont répertoriés dans le Tableau 5.1.

Le test d'une relation est réalisé en appliquant l'opération suivante :

$$x = w_1 - w_2 + w_4 \tag{5.4}$$

où x est une position dans l'espace résultant de l'opération. Les w_1 et w_2 sont deux mots partageant une relation (colonne 1 et 2 dans le Tableau 5.1). Le w_4 est supposé partager la même relation que les deux premiers mots avec w_3. Si dans l'espace distribué le mot le

Table 5.1 – *Exemples de questions présentes dans le « Semantic-Syntactic Word Relationship test» des mots définis dans (Mikolov et al., 2013a)*

Dakar	Sénégal	$\Longleftrightarrow$	Madrid	Espagne
il	elle	$\Longleftrightarrow$	oncle	tante
Japon	yen	$\Longleftrightarrow$	USA	dollar
Irlande	Irlandais	$\Longleftrightarrow$	Suède	Suédois
honnête	malhonnête	$\Longleftrightarrow$	rationnelle	irrationnelle
tard	tardivement	$\Longleftrightarrow$	bruyante	bruyamment
décroitre	décroissant	$\Longleftrightarrow$	écouter	écoutant

plus proche de la position x est bien w_3 alors on considère que la même relation sépare les mots 1 et 2 des mots 3 et 4 et cette question est considérée juste.

La qualité des espaces appris est mesurée en pourcentage de relations correctement modélisées.
Pour cette expérience, plusieurs réseaux SG, LL-Skip-gram, CBOW et LL-CBOW sont évalués dans différentes conditions de tailles de contextes et tailles de couches cachées. Le réseau de référence utilise 300 neurones dans sa couche cachée et une fenêtre de contexte de 10 mots. Deux tailles de couche cachée (120 et 300) et trois fenêtres de contexte (10, 15 et 100) sont évaluées. Les fenêtres de contexte de taille 100 utilisent les documents entiers comme contexte. En effet, moins de 1% des documents présents dans nos corpus dépasse les 100 mots.
Pour chaque configuration, un réseau avec pondération log-linéaire et un réseau sans pondération sont entrainés. La fonction de pondération déterminée expérimentalement pour ces expériences est définie comme suit :

$$\delta(w_i) = \frac{1 + \log(2)}{1 + \log(d(w_i))} \tag{5.5}$$

Où w_i est le mot à pondérer et d(w) et la distance entre le mot w_i et le mot au centre du contexte (w_0).

Le corpus anglais utilisé pour l'entrainement des réseaux de neurones est composé de :

Word Language Modeling Benchmark Un corpus créé pour les tâches de modélisation du langage contenant 31 millions de documents et 700 millions de mots.

Wikipedia Un extrait de Wikipédia anglais contenant 124 303 documents et environ 124 millions de mots.

Gigaword Le corpus Gigaword en anglais comprenant les versions de 1994 à 2011. Ce corpus contient 190 millions de documents et 3 771 milliards de mots.

The Brown corpus Un corpus d'anglais américain publié en 1961 contenant 500 documents et 1 million de mots.

Le corpus final contient environ 5 milliards de mots avec un vocabulaire d'un million de mots.

Le Tableau 5.2 et la Figure 5.9 présentent le voisinage de certains mots extraits avec les modèles CBOW et LL-CBOW entrainés avec ce corpus. On peut remarquer que le modèle intégrant la pondération log-linéaire tend à regrouper les mots thématiquement liés, ce qui n'est pas le cas du modèle standard. Par exemple, dans le voisinage de "Holidays", les termes qui ont une consonance religieuse sont regroupés dans l'espace LL-CBOW, alors qu'ils sont mélangés aux autres voisins dans l'espace CBOW.

Table *5.2* – *Exemple de mots voisins extraits de modèle entrainé sans et avec information contextuelle (CBOW et LL-CBOW). On peut voir dans chaque voisinage ressortir deux «catégories» de mots. Cette différence est représentée en gras.*

Holidays		Meat	
LL-CBOW	CBOW	LL-CBOW	CBOW
holiday	vacations	chicken	pork
vacation	**thanksgiving**	**beef**	**not-pasteurized**
festivities	vacation	pork	mutton
thanksgiving	**christmas**	**milk**	**eggs**
easter	celebration	**eggs**	cattle
christmas	**easter**	**seafood**	chicken

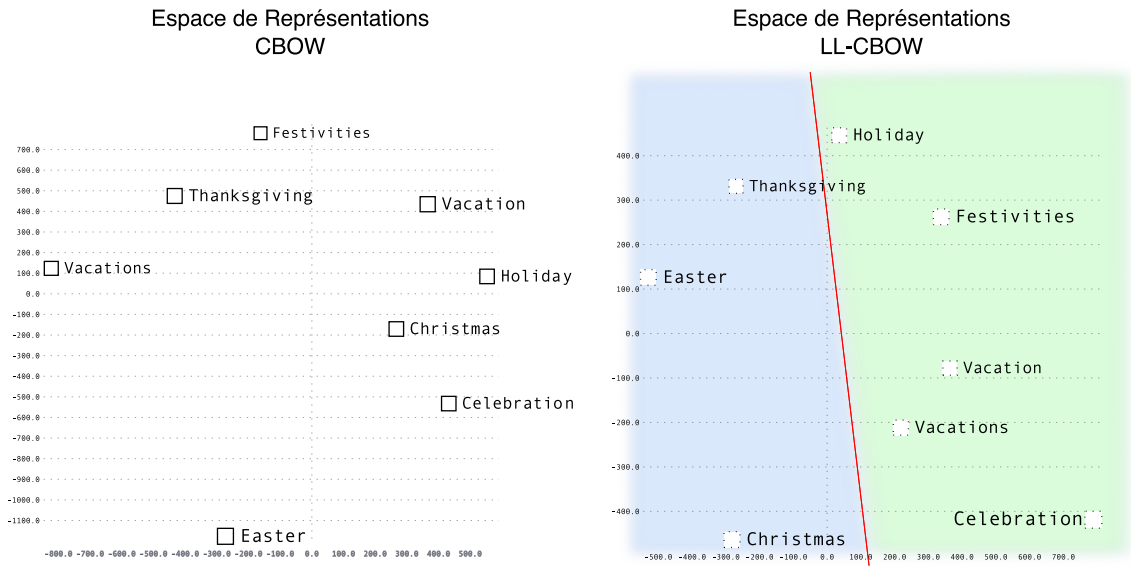


Figure *5.9* – *Voisinage du mot Holidays appris par un CBOW et un LL-CBOW projeté en 2D. On peut remarquer une séparation par topic sur l'axe des abscisses présent seulement chez le LL-CBOW*

Les Tableaux 5.3 et 5.4 montrent que les modèles appris en utilisant l'approche de pondération log-linaire proposée obtiennent globalement de meilleurs résultats que les modèles sans pondération. Les améliorations les plus importantes sont observées en utilisant les documents entiers comme contexte (100 mots). Dans ces conditions, la pondération apporte un gain de 7% pour le LL-CBOW et 7,7% pour le LL-SG. L'impact

négatif d'un mauvais choix pour la taille de la fenêtre de contexte est considérablement réduit et quasiment négligeable pour le LL-SG. Au contraire des performances atteintes par le CBOW et le SG qui chutent lorsque la fenêtre de contexte augmente.
On remarque dans le Tableau 5.4 que réduire la taille de la couche cachée implique un gain plus faible lors de l'utilisation de la pondération. Les neurones sont utilisés par le réseau comme mécanique de mémorisation. Une couche cachée réduite ne permet pas au réseau de retenir autant d'informations. Par conséquent, le réseau a plus de difficulté à mémoriser l'information additionnelle apportée par la pondération.

Table *5.3* – *Performance (%) sur la tâche Semantic-Syntactic Word Relationship test avec ou sans pondération en fonction de la taille du contexte utilisé (c) avec une couche cachée fixe de taille 300*

	Skip-gram			CBOW		
Fenêtre de contexte (nombre de mots)	10	15	100	10	15	100
Modèle standard	50,0	50,9	43,7	39	38,9	36,9
Modèle pondéré (LL)	**55,0**	53,7	51,4	39,9	39,6	**43,9**

Table *5.4* – *Performance (%) sur la tâche Semantic-Syntactic Word Relationship test avec ou sans pondération en fonction de la taille de la couche cachée utilisée avec une fenêtre de contexte fixe de taille 10.*

	Skip-gram		CBOW	
Taille de l'espace	120	300	120	300
Modèle standard	43,9	50,0	29,0	39,0
Modèle pondéré (LL)	45,1	**55,0**	30,3	**39,9**

La seconde expérience que nous avons réalisée vise à évaluer l'efficacité de l'intégration des caractéristiques *Word2vec* dans une tâche de SLU. Pour cette évaluation, la tâche de détection de thématique dans les conversations téléphoniques du projet DECODA (TID) sera utilisée comme présentée dans le Chapitre 4. Les documents du corpus sont en français. Pour réaliser la tâche, des modèles CBOW, LL-CBOW, SG et LL-SG en français sont entrainés en utilisant les données suivantes :

GigaWord La version française du corpus contenant 17 millions de documents et 500 millions de mots.

Wikipedia Un extrait de Wikipedia France composé de 16 millions de documents et 400 millions de mots.

Newspapers Des articles issus de divers journaux français comme *AFP*, *Le Monde* et *Le Soir*. Ce corpus contient 56 millions de documents et 737 millions de mots.

Documents collectés sur le web Documents collectés sur internet contenant 4 millions de documents et 108 millions de mots.

Transcriptions manuelles Les transcriptions manuelles provenant des récentes campagnes d'évaluation en français telles qu'ESTER, EPAC, ETAPE et REPERE, contenant 411 000 documents et 379 millions de mots.

Ces corpus contiennent un peu moins de 2 milliards de mots avec un vocabulaire de 3 millions de mots.

La génération des caractéristiques pour la classification est réalisée en trois étapes. D'abord, l'ensemble des mots qui composent les documents sont projetés dans la couche cachée du réseau de neurones en ignorant les mots vides (c.f. Chapitre 4). Ensuite, la moyenne des mots de chaque document est réalisée. Puis, les 100 mots les plus discriminants par classe (707 mots en tout, c.f. Chapitre 4) sont sélectionnés. Le score discriminant utilisé pour la sélection est le TF-IDF-GINI. Chacun de ces mots est projeté dans l'espace d'embeddings.
Ils sont utilisés comme repères dans l'espace distribué construit par le modèle *Word2vec*. Chaque document est ensuite représenté par la somme des représentations des mots qui le compose. Au vecteur des documents vient s'ajouter la distance cosinus entre le vecteur moyen du document et chacun des mots discriminants. Le vecteur composé de ces caractéristiques est utilisé pour représenter le document lors de l'étape de classification automatique.

Pour évaluer l'impact de la fonction de pondération de contexte continue pour une tâche de SLU, 4 modèles sont entrainés, un CBOW, un LL-CBOW, un SG et un LL-SG. Les deux réseaux LL-CBOW et LL-SG utilisent la fonction définie pour l'évaluation qualitative présentée dans l'Équation 5.5. Les réseaux CBOW et SG utilisent les modèles standards *Word2vec*.

Les caractéristiques générées seront utilisées par deux classifieurs différents. Le premier classifieur est un MLP identique à celui présenté dans le Chapitre 4 et dont les métaparamètres sont répertoriés dans le Tableau 4.2. Le second classifieur utilisé est un classifieur à base d'arbres de décision appelée *Gradient Tree Boosting* (GTB) (Pedregosa et al., 2011 ; Friedman, 2001). Des détails du GTB sont présentés en Annexe B. Le GTB est choisi pour son efficacité dans de nombreuses tâches lors de compétitions d'apprentissage automatique [1] et parce que la sélection des métaparamètres est simple. Chaque classifieur est entrainé indépendamment sur les 4 jeux de caractéristiques issues des 4 réseaux de neurones entrainés.

Le Tableau 5.5 présente les précisions observées pour la classification à la fois pour le corpus de développement et de test en utilisant les architectures SG/LL-SG et CBOW/LL-CBOW. On peut remarquer que parmi les deux classifieurs, ce sont les MLP qui réussissent le mieux à discriminer les différentes classes. Les résultats montrent aussi que toutes les configurations testées améliorent leurs résultats en utilisant l'approche pondérée. En effet, utiliser les projections créées par le LL-SG en lieu et place du SG permet un gain de 10 et 20 points respectivement avec le GTB et le MLP. Cette amélioration est aussi mesurée avec Le LL-CBOW qui apporte un gain de 33 et 34 points pour le GTB et le MLP.

1. https ://www.kaggle.com/

Avec le MLP, les performances en classification sont mesurées toutes les 10 itérations sur le corpus de développement et reportées dans la Figure 5.10. On peut voir que les MLP combinés aux modèles utilisant la pondération log-linéaire sont plus précis et convergent plus vite que les combinaisons sans pondération.

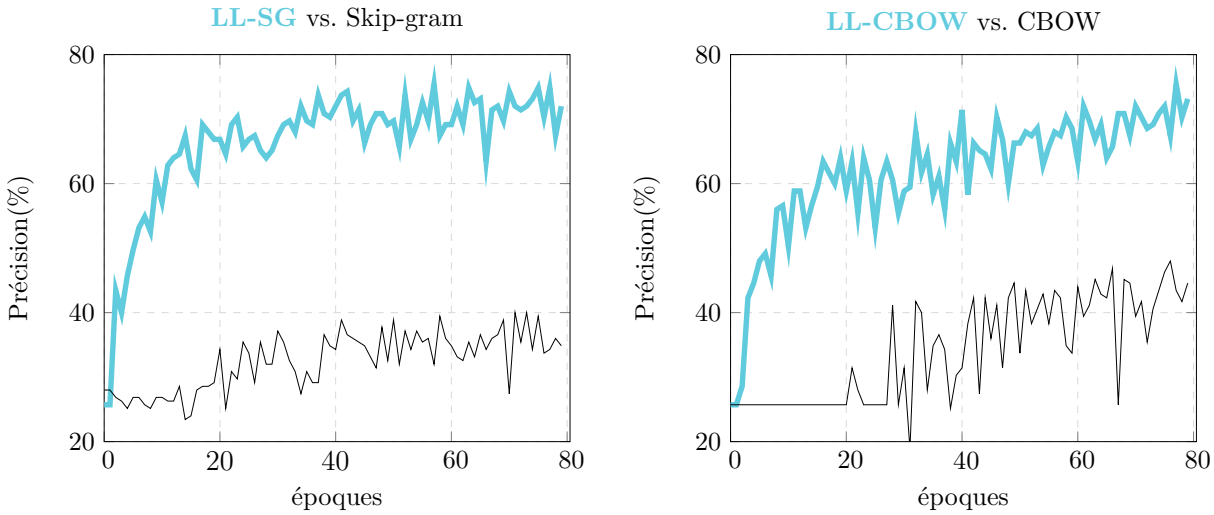


Figure *5.10* – *Précision de classification (en %) sur la tâche TID obtenue avec les modèles Skip-gram,* **LL-SG**, *CBOW,* **LL-CBOW** *en fonction du nombre d'époques d'entrainement.*

Table *5.5* – *Précision de classification (%) sur la tâche TID en utilisant deux classifieurs et les caractéristiques issues des modèles Skip-gram et CBOW avec et sans pondération log-linéaire.*

	Skip-gram		CBOW	
	Developpement	Test	Developpement	Test
GTB(standard)	39	42	28	27
GTB(pondéré)	56	52	66	60
MLP(standard)	50	50	41	37
MLP(pondéré)	75	**70**	74	**71**

Ces expériences montrent que les contextes continus produisent une pondération efficace qui permet d'introduire de l'information de distance dans les réseaux de neurones *Word2vec*. Cela leur permet de capturer plus d'information sur la distribution des mots dans les textes qui leur sont soumis. Plus la fenêtre de contexte utilisée est grande, plus l'information additionnelle est importante. Cette information additionnelle est mieux exploitée par les réseaux de neurones quand les couches cachées sont suffisamment grandes pour en capturer le maximum d'information pertinentes.

Les résultats au *Semantic-Syntactic Word Relationship test* mettent en évidence que cette information supplémentaire améliore les performances de modélisations des relations intermots. Les performances sur la tâche TID montrent que cette information supplémentaire permet d'améliorer de façon importante la qualité de la classification

thématique de documents parlés.

5.5 Conclusion

Les contextes dans la modélisation *Word2vec* sont représentés sous forme de sac-de-mots binaires. Cette représentation implique que tous les mots d'un contexte soient traités de la même manière. Nous avons introduit une information sur la position des mots dans le contexte par le moyen d'une pondération log-linéaire. Les expériences qualitatives et quantitatives montrent que l'apprentissage des représentations latentes des mots avec les poids améliore la qualité des modèles et les performances lors de leur application à une tâche de classification thématique.

La méthode de pondération proposée dans ces travaux renforce l'information contextuelle et préserve les relations distantes pour l'apprentissage de représentations distribuées de mots. Les expériences réalisées montrent un gain de performance de plus de 7% pour le *Semantic-Syntactic Word Relationship test* et plus de 20% absolu pour la tâche TID.

Globalement, on peut remarquer qu'une amélioration importante est introduite par la fonction de pondération en contexte continu sur la tâche de compréhension de la parole décrite Section 4.1. Malgré cette amélioration l'utilisation des caractéristiques *Word2vec* avec pondération parvient difficilement à obtenir 70% de précision et reste significativement en dessous du système de base (77%) introduit dans le Chapitre précédent.

Ces travaux ouvrent la porte à plusieurs voies d'amélioration. D'abord, la plus évidente serait d'adapter le système de classification pour tenir compte de la structure des documents. Il faudrait remplacer le MLP par un réseau de neurones récurrents comme les LSTM (Graves, 2012) ou les GRU (Chung et al., 2015) qui sont particulièrement efficaces avec ce genre de représentations. Cette méthode pourrait mettre au même niveau le système reposant sur les caractéristiques TF.IDF et celui basé sur les caractéristiques proposées dans ce chapitre. Cela permettrait aussi d'analyser l'impact de l'information de contextes continus dans ces conditions et notamment de créer un lien entre l'information de structure modélisée par les contextes continus et celle produite par la récurrence.

La seconde possibilité est d'analyser les pondérations apprises sans supervision par les méthodes d'attention (Ling et al., 2015) et leurs liens avec la pondération réalisée dans ces travaux. Dans une forme d'application comparable aux travaux présentés dans ce chapitre, l'attention permettrait de s'affranchir du choix de la fonction de pondération dans les contextes continus. L'analyse des poids appris par ces méthodes peut aussi apporter des informations précieuses sur le choix de meilleures fonctions de pondération par contexte continu à utiliser selon la tâche.

Une troisième piste à explorer est la généralisation des contextes continus, dans une autre représentation distribuée, telle que *Glove* (c.f. Chapitre 2). Cela nécessiterait d'avoir une formulation de la pondération hors du cadre des réseaux de neurones en

l'appliquant à des modèles basés sur les cooccurrences des mots.

La dernière piste d'amélioration vient du constat que l'ajout d'une information sémantique dans le processus de classification n'a pas permis d'améliorer les performances du système. On peut donc supposer que les résultats sont dégradés par un bruit, présent dans les données, certainement liés aux erreurs de reconnaissance automatique de la parole. C'est cette voie que nous allons suivre dans le prochain chapitre.

Chapitre 6

Utilisation d'autoencodeurs empilés pour le débruitage non supervisé de documents parlés

Sommaire

6.1 Introduction 85
6.2 Fonctionnement des autoencodeurs 87
6.3 Protocole expérimental 91
6.4 Résultats et discussion 93
6.5 Conclusion 95

6.1 Introduction

Pour résoudre les problématiques de SLU, de nombreuses solutions appliquent des méthodologies de recherche d'informations directement sur les hypothèses générées par un SRAP (Lee et al., 2015b). Ces méthodes sont conçues initialement pour traiter des sources textuelles très peu bruitées. Celles-ci fonctionnent bien dans le cadre de parole préparée comme des reportages radiophoniques et télédiffusés, mais elles ont plus de difficultés pour traiter la parole spontanée. En effet, la parole spontanée apporte deux différences fondamentales. La première vient des reprises, contradictions, répétitions et de tous les défauts de langage (disfluences) qui font la particularité de la parole spontanée. La seconde différence est liée au SRAP. Celui-ci est un système statistique limité qui peut produire une quantité importante d'erreurs lors de la génération des hypothèses de transcription. Ces erreurs sont dues à l'environnement sonore particulier du document, à la taille limitée du vocabulaire du système, et à la présence de plusieurs locuteurs qui parlent simultanément, etc.

Les travaux présentés dans ce chapitre reposent sur l'hypothèse que les erreurs introduites par le SRAP sont un bruit qu'un réseau de neurones artificiels peut modéliser

et gommer de la représentation. Nous proposons ainsi une méthode visant à réduire l'impact du processus de transcription automatique lors de la classification des segments transcrits. Cette méthode est appliquée entre le SRAP et la phase de classification comme montrée dans la Figure 4.2 du Chapitre 4. Les sorties de SRAP sont représentées par des sac-de-mots avec des descripteurs TF.IDF.GINI introduits aussi dans le Chapitre 4.

Les réseaux de neurones se sont récemment positionnés comme les méthodes les plus efficaces pour la réduction du bruit, notamment sur le traitement de la parole (Feng et al., 2014) et des tâches d'extractions d'informations depuis des tweets (Liu et Inkpen, 2015). Dans ces travaux, nous proposons deux approches différentes, qui exploitent les capacités à compresser l'information pour générer une représentation robuste.

La première approche consiste à utiliser des autoencodeurs empilés (Stacked autoencoder,SAE) (Vincent et al., 2010) pour produire de nouvelles représentations des documents. Utiliser un réseau plus profond permet de déduire et d'exploiter des concepts de plus haut niveau (plus abstraits) (Bengio, 2009). Dans le cadre d'un SAE, les couches cachées ont les mêmes caractéristiques qu'un autoencodeur débruitant. Cette spécificité permet aux couches profondes de produire des représentations plus abstraites, mais aussi plus robustes aux bruits. Les représentations latentes inférées par un SAE sont réalisées de manière à conserver toute l'information nécessaire pour pouvoir reconstruire le document d'origine. Elles sont donc : d'un plus haut niveau d'abstraction, plus robustes et aussi informatives que les données d'origine. Un classifieur entrainé sur celles-ci généralise mieux et obtient de meilleurs résultats (Hinton et Salakhutdinov, 2006 ; Bengio et al., 2007a).

La seconde considère les transcriptions générées par un SRAP comme des documents corrompus (ASR) et les transcriptions manuelles (TRS) comme la version propre des mêmes documents. Nous proposons d'utiliser un autoencodeur débruitant (DAE) dont le vecteur d'entrée n'est plus bruité artificiellement, mais utilise le bruit «naturel» introduit par le SRAP. Un DAE est entrainé pour supprimer le bruit des documents issus du SRAP. La variabilité introduite par les erreurs du SRAP est réduite en entrainant le réseau à reconstruire la même représentation, mais issue de transcriptions manuelles. C'est la principale différence avec la première méthode qui n'impose pas de supervision avec les transcriptions manuelles pour la reconstruction.

Les premières applications de réseaux de neurones capables de reconstruire des données à partir d'une représentation latente sont définies dans (Le Cun, 1987) et (Gallinari et al., 1987). Plus récemment (Hinton et Salakhutdinov, 2006) proposent des autoencodeurs profonds et une méthode d'apprentissage couche par couche. Cet apprentissage est réalisé à partir de poids appris sans supervision par des autoencodeurs "simples"(non profonds). Cette méthode permet de construire des représentations de tailles réduites contenant suffisamment d'informations pour reconstruire les documents d'origines. Des résultats intéressants sont obtenus quand l'apprentissage par couche est suivi d'un apprentissage global (dit fine tuning) (Erhan et al., 2010).

Dans le but d'améliorer l'efficacité et la stabilité des représentations obtenues avec un AE, des DAE sont proposés (Vincent et al., 2008a). Ils sont entrainés de manière à réduire la variabilité de la représentation de la couche k lorsque le bruit est introduit

dans la couche précédente $(k-1)$. Les représentations compressées, construites par des DAE, permettent aux systèmes automatiques de mieux généraliser et d'être résistants aux variabilités qui ne sont pas porteuses d'informations. Ces propriétés ont permis d'obtenir de bons résultats dans différents domaines comme la médecine (Xu et al., 2015b), la biologie (Camacho et al., 2015), le traitement de l'image (Maria et al., 2015), le traitement de la musique (Sarroff et Casey) et le traitement de la parole (Chao et al., 2014). Lors de l'estimation des poids optimaux de ces réseaux, les vecteurs d'entrée sont corrompus artificiellement par un bruit additif. Le réseau apprend ainsi à séparer le bruit de l'information importante. Dans la littérature, les DAE génèrent de meilleures représentations que les machines de Boltzmann restreintes, les SVM et les autoencodeurs sans bruit additif (Vincent et al., 2010 ; Tan et Eswaran, 2008).

Dans (Lu et al.), un DAE basé sur un autoencodeur empilé est proposé pour reconstruire des coefficients cepstraux de fréquence en échelle de Mel (MFCC) propres à partir de leurs versions bruitées. Dans (Feng et al., 2014) un DAE et un AE «antiréverbération» sont proposés pour compresser et supprimer la réverbération de signaux audios. Un DAE est aussi proposé pour débruiter un spectrogramme dans (Deng et al., 2010). Des résultats intéressants ont été obtenus avec des SAE pour l'extraction de caractéristiques acoustiques dans (Seide et al., 2011). Un SRAP dont les caractéristiques acoustiques sont adaptées par les couches cachées d'un réseau de neurones profonds est présenté dans (Yu et al., 2013 ; Sainath et al., 2012). Cette construction de descripteurs acoustiques permet d'obtenir de meilleurs résultats que des alternatives classiques (GMM, fMLLR et VTLN). Un SRAP robuste qui repose sur des caractéristiques articulatoires apprises par réseaux de neurones profonds combinés aux usuelles MFCC, est introduit dans (Vinyals et Ravuri, 2011).

L'approche proposée dans ce chapitre consiste à considérer les erreurs introduites par le SRAP comme un bruit. Ainsi, des autoencodeurs sont utilisés pour construire des représentations de l'information transcrites automatiquement qui sont robustes au bruit et donc facilitent l'extraction d'informations thématiques.

6.2 Fonctionnement des autoencodeurs

En premier lieu, le fonctionnement des autoencodeurs est détaillé. Ensuite des variantes particulières des autoencodeurs sont définies : le DAE et l'autoencodeur empilé (SAE).

6.2.1 Les concepts fondamentaux

Un autoencodeur est un réseau de neurones composé d'au moins trois couches. Il est dit *feedforward*, ce qui signifie que les connexions entre les neurones ne forment pas de cycle. Un autoencodeur se décompose en deux parties, l'encodeur et le décodeur comme présentés dans la Figure 6.1.

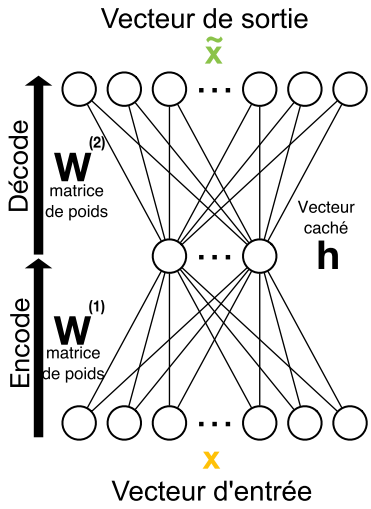


Figure *6.1* – *Schéma d'un autoencodeur. Les biais sont ignorés pour simplifier le schéma.*

Soit x le vecteur de taille n utilisé en entrée du réseau et h la couche cachée composée de m neurones. L'encodage de x dans h est réalisé de la façon suivante :

$$h = \sigma(W^{(1)}x + b^{(1)}) \ , \tag{6.1}$$

Où $W^{(1)}$ est une matrice de poids de taille $n \times m$ et $b^{(1)}$ est un vecteur de dimension m. $\sigma(.)$ est la fonction d'activation de l'encodeur. Le *framework* des autoencodeurs permet l'utilisation de plusieurs fonctions d'activation comme celle introduite dans le Chapitre 3. Dans ces travaux, la fonction d'activation utilisée est la tangente hyperbolique (*Tanh*) définie ainsi :

$$\sigma(y) = \frac{e^{y} - e^{-y}}{e^{y} + e^{-y}} \tag{6.2}$$

Le décodeur utilise la représentation cachée de h pour reconstruire le vecteur d'entrée x et génère le vecteur de sortie $\tilde{x}$:

$$\tilde{x} = \sigma(W^{(2)}h + b^{(2)}) \ , \tag{6.3}$$

Ici le vecteur reconstruit $\tilde{x}$ possède n dimensions, $W^{(2)}$ est une matrice de poids de dimension $m \times n$ et $b^{(2)}$ et un vecteur de biais aussi de taille n .

L'estimation des paramètres de l'autoencodeur $\theta = \{W^{(1)}, b^{(1)}, W^{(2)}, b^{(2)}\}$ est réalisée en minimisant l'erreur de reconstruction quadratique moyenne. L_{MSE} (Bengio, 2009) calculée par :

$$L_{\text{MSE}}(\theta) = \frac{1}{d}\sum_{x \in D} l_{\text{MSE}}(x, \tilde{x}) = \frac{1}{d}\sum_{x \in D} ||x - \tilde{x}||^2 \tag{6.4}$$

L'adaptation des paramètres θ est réalisée en utilisant un algorithme de descente de gradient présenté dans le Chapitre 3.

6.2.2 Autoencodeur débruitant (DAE)

Les autoencodeurs classiques compressent l'information présente dans les données d'entrée. Si ces données sont bruitées, celui-ci va affecter de la même manière les représentations compressées. Ce bruit peut dégrader les performances du système. De manière à rendre plus robustes les représentations apprises, le système retient la structure sous-jacente des données (Bengio, 2009). Dans ce but, (Vincent et al., 2010) proposent des réseaux de neurones dit DAE, dans lesquels, le vecteur d'entrée est corrompu artificiellement avant de l'encoder dans h. Ensuite, le vecteur h est utilisé pour décoder la version propre de l'entrée x. De cette manière, le DAE apprend la distribution des données et du bruit, pour reconstruire la représentation propre et ainsi devenir robuste aux bruits.

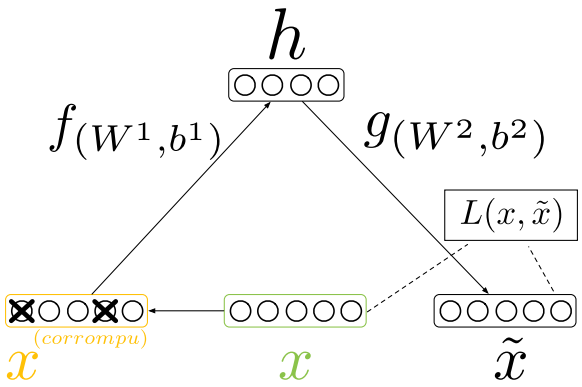


Figure *6.2* – *Modèle graphique d'un autoencodeur débruitant. Le vecteur d'entrée x est corrompu aléatoirement pour générer $x^{(corrompu)}$. Ensuite $x^{(corrompu)}$ est projeté dans l'espace latent h pour ensuite produire la reconstruction $\tilde{x}$. L'erreur de reconstruction est calculée avec L_{MSE}.*

La Figure 6.2 montre l'architecture d'un DAE. Dans ce modèle, le vecteur d'entrée x est considéré comme "propre". L'objectif de ce DAE est d'obtenir, à partir d'un vecteur d'entrée propre, un modèle de reconstruction robuste. C'est pourquoi x est artificiellement corrompu via une fonction qui peut-être :

- Un bruit Gaussien Isotropique Additif (additive gaussian noise, GN) $x^{(\text{corrompu})}|x \sim \mathcal{N}\left(x, \sigma^2 I\right)$.
- Un *bruit par masque* (masking noise, MN), qui force une fraction des éléments de x à 0.
- Un *bruit sel-et-poivre* (salt-and-pepper noise, SN), qui force la valeur de certains éléments de x à leur valeur maximum ou minimum selon un tirage aléatoire (Vincent et al., 2010).

La projection de $x^{(\text{corrompu})}$ dans la couche cachée se calcule comme ci-dessous :

$$h = f_{(W^{(1)},b^{(1)})}(x^{(\text{corrompu})}) = \sigma(W^{(1)}x^{(\text{corrompu})} + b^{(1)}) \tag{6.5}$$

et le vecteur reconstruit $\tilde{x}$ s'obtient de manière similaire :

$$\tilde{x} = g_{(W^{(2)},b^{(2)})}(h) = \sigma(W^{(2)}h + b^{(2)}) \tag{6.6}$$

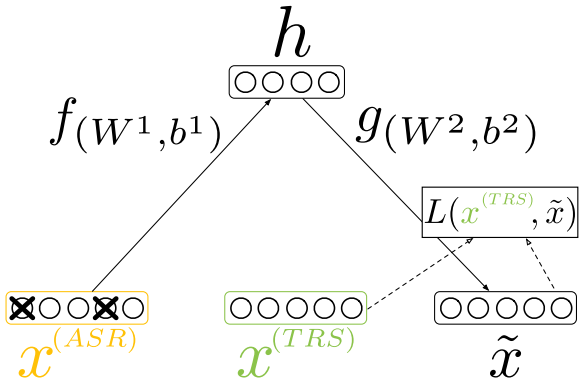


Figure *6.3* – *Modèle graphique d'un autoencodeur débruitant hétérogène. Le vecteur d'entrée $x^{(ASR)}$ est corrompu par le SRAP. La sortie $x^{(TRS)}$ provient de transcriptions manuelles.*

Le processus d'apprentissage du réseau détermine les paramètres $\theta = \{W^{(1)}, b^{(1)}, W^{(2)}, b^{(2)}\}$ qui minimisent l'erreur de reconstruction $L_{\text{MSE}}(x, \tilde{x})$.

Dans le cadre de nos travaux, le problème peut être formulé différemment. Les caractéristiques d'entrée utilisées sont issues d'un SRAP. Les transcriptions automatiques contiennent en particulier des erreurs de transcription et des disfluences qui peuvent gêner la compréhension d'un segment audio. On peut donc considérer les transcriptions comme bruitées. Ce bruit produit est complexe et imprévisible. Ce constat a motivé une approche originale qui utilise un autoencodeur débruitant dont les données corrompues d'entrée et les données propres de sortie ne sont pas issues de la même source (hétérogènes). Celui-ci exploite à la fois la version du SRAP (ASR) et la version transcrite manuellement (TRS) de chaque document. Il ignore le bruit naturellement présent dans les documents transcrits automatiquement pour apprendre une représentation robuste. Le modèle graphique de ce réseau est présenté dans la Figure 6.3. On peut remarquer dans cette Figure que les données corrompues ASR sont encodées dans la couche cachée h, utilisée à son tour pour construire une version du document propre $\tilde{x}$. Cette vision est comparée au document transcrit manuellement qui est considéré comme propre.

6.2.3 Autoencodeur empilé (SAE)

Les réseaux de neurones profonds sont capables de construire des représentations de plus en plus abstraites à l'aide de leurs multiples couches cachées. C'est une des raisons principales à leurs excellentes performances dans de nombreuses tâches (Bengio et al., 2007b ; Hinton et al., 2006). Dans l'optique d'utiliser cette capacité d'abstraction, des SAE sont utilisés. Ils sont composés d'un nombre variable de couches cachées k. Les représentations latentes de la i-ème couche cachée $h^{(i)}$, pour une entrée x donnée sont calculées à partir du vecteur $h^{(i-1)}$ comme suit :

$$h^{(i)} = \sigma(W^{(i)} h^{(i-1)} + b^{(i)}) \ \forall i \in \{1, \ldots, k\} \tag{6.7}$$

et

$$h^{(0)} = x \tag{6.8}$$

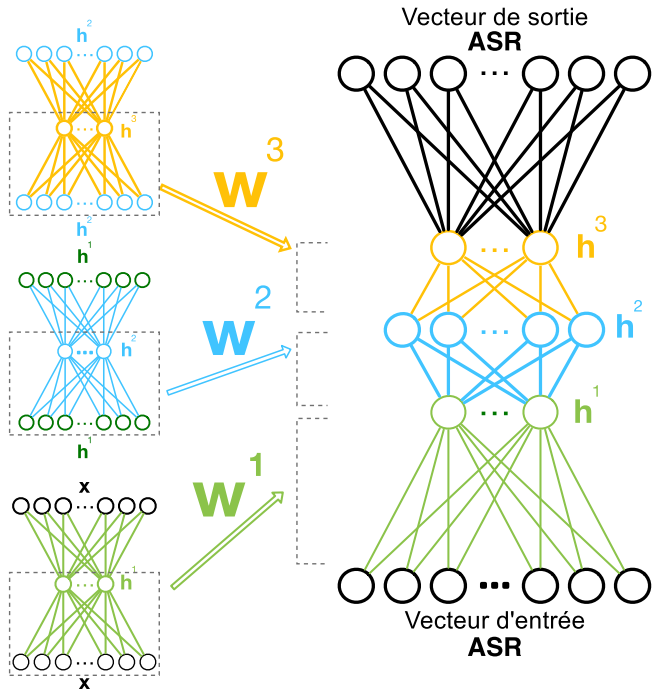


Figure *6.4* – *Architecture d'un autoencodeur empilé (SAE). Les autoencodeurs simples pour le préentrainement sont à gauche. Leurs poids sont utilisés pour initialiser le réseau complet (à droite).*

La méthode d'apprentissage du SAE se déroule en deux étapes. D'abord, les poids de chaque couche sont préentrainés indépendamment par un autoencodeur simple qui utilise la projection de la couche précédente. La représentation calculée de chacun des documents projetés dans la couche $h^{(i)}$ est conservée et servira pour calculer $h^{(i+1)}$ comme montré dans la Figure 6.4 (à gauche). Chaque couche cachée supplémentaire apprend ainsi une représentation plus robuste que la précédente. Ensuite, un réapprentissage global (*fine tuning*) est effectué sur l'ensemble des couches empilées entier (cf. Figure 6.4 à droite). Cette étape affine les poids du réseau en entrainant la pile entière à reconstruire le vecteur x d'origine.

6.3 Protocole expérimental

Cette section présente d'abord les différents autoencodeurs implémentés et les méta paramètres de chacun d'eux dans ces expériences. Les représentations latentes produites par ces réseaux sont évaluées sur la tâche TID présentée dans le Chapitre 4.

En premier lieu, trois autoencodeurs débruitants simples (non profonds) sont entrainés pour évaluer les différentes compositions de sources de données et pour valider l'apport des modèles profonds.

1. **AE_{ASR} :** utilise en entrée les données $x^{(\text{ASR})}$ issues du SRAP avec une corruption artificielle supplémentaire. Il reconstruit le vecteur $x^{(\text{ASR})}$.
2. **AE_{TRS} :** est un miroir de AE_{ASR}, mais en utilisant les données transcrites manuellement $x^{(\text{TRS})}$.
3. **DAE :** repose sur le principe que les documents produits par le SRAP sont une version bruitée des documents transcrits manuellement. Il utilise en caractéristiques d'entrée le vecteur $x^{(\text{ASR})}$ et tente de reconstruire le vecteur $x^{(\text{TRS})}$ comme présenté dans la Figure 6.3.

Les couches cachées de ces autoencodeurs respectivement $h^{(\text{ASR})}$, $h^{(\text{TRS})}$ et $h^{(\text{DAE})}$ sont composées de 50 neurones chacune. Cette taille a été déterminée en fonction des performances des autoencodeurs sur les données de développement de la tâche. Ensuite, trois autoencodeurs profonds supplémentaires sont entrainés pour ces expérimentations. Ces autoencodeurs devraient avoir les meilleures capacités de modélisation, en construisant des représentations plus abstraites dans les couches plus profondes du réseau.

1. **DAE profond (DDAE) :** Une évolution plus profonde du DAE. Il possède plusieurs couches cachées, $x^{(\text{ASR})}$ est utilisé en entrée pour reconstruire $x^{(\text{TRS})}$. Ce réseau est présenté dans la Figure 6.5.
2. **SAE_{ASR} :** est un autoencodeur empilé qui corrompt artificiellement les vecteurs d'entrée $x^{(\text{ASR})}$ par masquage. Il reconstruit le vecteur $x^{(\text{ASR})}$.
3. **SAE_{TRS} :** est similaire à SAE_{ASR} mais utilise les transcriptions manuelles $x^{(\text{TRS})}$.

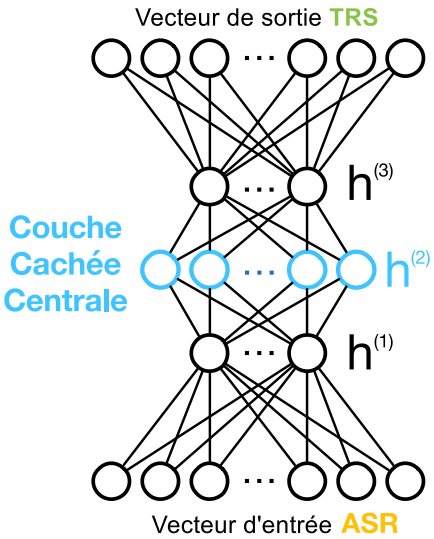


Figure *6.5* – *Architecture d'autoencodeurs débruitants profonds hétérogènes. Ses vecteurs d'entrée sont naturellement bruités par un SRAP et ses vecteurs de sortie propres sont les transcriptions manuelles.*

Les couches cachées de ces réseaux profonds sont de taille 50, 300 et 50. Tous ces réseaux de neurones reposent sur la même procédure d'apprentissage. Ils utilisent une descente de gradient *Adam*, un taux d'apprentissage initial de 10^{-3}, du *early stopping*

AE	DAE	**DDAE**	**SAE**	MLP
10 (min.)	10 (min.)	25 (min.)	50 (min.)	8-15 (min)

Table *6.1* – *Temps d'apprentissage pour les différents réseaux de neurones réalisés avec une carte graphique (GPU) Nvidia Geforce TITAN X.*

(où arrêt prématuré) pour arrêter l'apprentissage à convergence avec une patience de 20 itérations.

Les représentations produites par les différents autoencodeurs sont évaluées en classification sur la tâche TID détaillée dans le Chapitre 4. Le classifieur utilisé est le MLP présenté dans le Chapitre 4.3.

Tous ces réseaux de neurones sont développés avec les bibliothèques Keras (Chollet) et Theano (Bergstra et al., 2010b) pour la manipulation des tenseurs et de l'accélération GPU. Les calculs matriciels sont réalisés par une carte graphique[1] avec 3500 coeurs, 12 GB de mémoire. Les temps nécessaires pour l'apprentissage de ces différents réseaux sont reportés dans le Tableau 6.1 et seront commentés dans la Section 6.5.

6.4 Résultats et discussion

Les résultats expérimentaux sur la tâche TID avec les représentations apprises par les autoencodeurs débruitants simples sont détaillés et commentés dans la section 6.4.1. Ensuite, la section 6.4.2 présente les résultats obtenus avec les autoencodeurs profonds.

6.4.1 Analyse des autoencodeurs simples

Le Tableau 6.2 présente les précisions obtenues par le MLP en utilisant les caractéristiques produites par les différents modèles non profonds.

Parmi ces résultats, les meilleures précisions sont obtenues avec les représentations latentes issues des AE qui utilisent les mêmes données en entrée et en sortie. Ils apportent une amélioration de $3,9\%$ par rapport aux représentations TF.IDF avec les données ASR et $0,7\%$ avec les données TRS.
La forte amélioration obtenue avec AE_{ASR} montre que lorsqu'une information non pertinente est présente, l'autoencodeur arrive à l'ignorer. L'amélioration en précision plus réduite observée par le AE_{TRS} met en évidence que moins d'informations non pertinentes sont présentes avec les données TRS. Cela confirme l'hypothèse que les erreurs introduites par le SRAP peuvent s'apparenter à du bruit pour un DAE et qu'il est possible de le supprimer au moins en partie.
La précision obtenue avec les représentations du DAE hétérogène (ASR $\rightarrow$ TRS) est de $74,3\%$ soit une précision d'environ 3% inférieure aux représentations TF.IDF.GINI

1. Nvidia Geforce TITAN X

de départ. Cette dégradation des résultats montre que malgré les bons résultats des autoencodeurs débruitants, ceux-ci ne sont pas capables de débruiter les erreurs introduites par le SRAP. Les représentations $x^{(ASR)}$ et $x^{(TRS)}$ sont probablement si différents que le système de débruitage ne parvient pas à extraire une structure commune.
La section suivante aborde les résultats des réseaux de neurones profonds, dont les capacités de modélisation supplémentaire doivent faciliter le débruitage et améliorer les représentations latentes.

Méthodes	**Données d'entrée**	**Données de sortie**	**Précisions du système**		
			TF.IDF	couche cachée h	Reconstuction $\tilde{x}$
AE_{ASR}	ASR	ASR	77,1	**81**	79
AE_{TRS}	TRS	TRS	83,4	84,1	83,7
DAE	ASR	TRS	77,1	74,3	70,3

Table *6.2* – *Précision(%) sur la tâche TID en utilisant les représentations produites par les différentes couches cachées des autoencodeurs simples débruitants.*

6.4.2 Analyse des autoencodeurs profonds

Le Tableau 6.3 compare les performances en classification des caractéristiques obtenues avec les autoencodeurs profonds. Les résultats utilisant le vecteur d'entrée x ne sont pas répétés dans ce tableau puisqu'ils sont identiques à la colonne TF.IDF présentée dans le Tableau 6.2.

Autoencodeur employé	**Type d'entrée**	**Type de sortie**	**Couche utilisée**	**Précision obtenue**
SAE_{ASR}	ASR	ASR	$h^{(1)}$	81,7
	ASR	ASR	$h^{(2)}$	**82,0**
	ASR	ASR	$h^{(3)}$	80,1
	ASR	ASR	$h^{(4)}$	81,0
SAE_{TRS}	TRS	TRS	$h^{(1)}$	84,1
	TRS	TRS	$h^{(2)}$	84,7
	TRS	TRS	$h^{(3)}$	85,3
	TRS	TRS	$h^{(4)}$	84,4
DDAE	ASR	TRS	$h^{(1)}$	72,5
	ASR	TRS	$h^{(2)}$	70
	ASR	TRS	$h^{(3)}$	69,4
	ASR	TRS	$\tilde{x}$	69,7

Table *6.3* – *Précision (%) de classification thématique en utilisant les caractéristiques produites par les différentes couches des autoencodeurs profonds.*

Ce tableau met en évidence que les meilleurs résultats sont obtenus par les réseaux homogènes. En effet le SAE_{ASR} obtient 82% soit $4,9$ point d'amélioration absolue et surpasse toutes les autres configurations utilisant les données bruitées issues du SRAP.

Le SAE_{TRS} permet de gagner $2,2\%$ comparé aux données d'origine. Lorsque les données d'entrée et de sortie sont issues d'un environnement similaire, un autoencodeur profond arrive à découvrir une forme de structure dans les données, pour construire une représentation moins sensible aux bruits, avec un effet positif sur les capacités de généralisation du modèle. Au contraire du DDAE, qui voit l'ensemble de ses couches proposer des représentations latentes de piètre qualité. Il n'obtient que $72,5\%$ de précision dans le meilleur des cas, soit plus de 4 points en dessous du système de référence. Les capacités d'abstraction supplémentaires apportées par les couches cachées ne changent pas l'incapacité du réseau à débruiter les représentations issues du SRAP. Le bruit introduit par la transcription automatique est trop imprévisible. Il apporte une variabilité trop importante pour modéliser une structure commune aux représentations issues des sources hétérogènes (manuelle et automatique).

Les autoencodeurs empilés améliorent de façon notable la qualité des représentations pour la classification. Par contre, on remarque qu'augmenter le niveau d'abstraction de la représentation des données n'est pas forcément le facteur qui permet aux réseaux de neurones d'améliorer la qualité des représentations. En effet, les dernières couches (h^4) produisent des représentations de moins bonnes qualités que les précédentes. Avoir plus d'une couche cachée est bénéfique pour le système global, mais la robustesse aux bruits joue aussi un rôle décisif dans les gains apportés pour cette tâche. Les performances en classification des documents parlés se voient améliorées par l'utilisation des SAE, à la fois pour les transcriptions manuelles et les transcriptions automatiques. L'écart entre les performances obtenues avec les documents issus du SRAP et les documents transcrits manuellement, lui se réduit. En effet, dans leurs versions sans encodage, le bruit introduit une erreur de classification de $6,4\%$. Après encodage par les SAE, la différence n'est plus que de $3,3\%$. On peut en déduire qu'une partie de la variabilité inutile introduite par le SRAP est effectivement modélisée et supprimée de façon non supervisée.

6.5 Conclusion

Ce chapitre présente plusieurs représentations latentes de transcriptions de documents parlés construites à partir de réseaux de neurones. Parmi les réseaux de neurones utilisés, les autoencodeurs (AE) qui utilisent des caractéristiques homogènes permettent une amélioration intéressante de la qualité des représentations. En effet, ces réseaux modélisent la structure des données employées et génèrent une représentation robuste aux variabilités non pertinentes. Les meilleures performances sont obtenues par les SAE qui permettent, en projetant les documents dans plusieurs couches cachées, de créer une modélisation des données plus abstraite. Celle-ci réduit l'effet relatif des erreurs du SRAP et facilite la classification des documents, ce qui permet au classifieur de généraliser plus facilement à de nouvelles données.

En revanche, les autoencodeurs hétérogènes n'arrivent pas à modéliser de relations utiles entre les données bruitées et les données propres. Les erreurs contenues dans les documents ASR produisent des représentations qui sont trop différentes des données TRS. Les (D)DAE ne parviennent pas à modéliser une variabilité utile de ces différences.

Les temps d'apprentissage reportés dans le Tableau 6.1 montrent que seule l'architecture joue un rôle sur le temps d'apprentissage. Les AE et le DAE nécessitent le même temps d'apprentissage alors qu'ils ne reposent pas sur des données identiques et pourraient ne pas converger vers une solution à la même vitesse. On remarque aussi que le SAE nécessite cinq fois plus de temps d'entrainement que l'AE_{ASR} à cause des préentrainements, alors qu'il n'apporte qu'un gain de 1%. Si le temps d'apprentissage est un critère important, se contenter d'un AE simple peut être une solution viable.
Ces travaux sont ouverts à plusieurs axes d'amélioration pour continuer à réduire l'impact du bruit introduit par le SRAP et combler la différence de performance observée avec les transcriptions manuelles.
Un premier axe serait de trouver une méthode plus robuste que le DDAE, capable d'exploiter l'information des transcriptions manuelles, dans le but d'améliorer la transcription automatique. Cet axe sera abordé dans les Chapitres 8 et 9.
Le second axe serait d'appliquer le processus présenté dans ce chapitre, sur d'autres caractéristiques de description textuelle (c.f. Chapitre 2), issues de documents transcrits automatiquement.

Troisième partie

Contribution : Exploiter plusieurs visions d'un document : des réseaux de neurones multivues

Chapitre 7

Représentations multivues par compression pour la classification thématique de documents

Sommaire

7.1 Introduction **99**
7.2 Modèle *Author-Topic* (AT) **100**
7.3 Systèmes multivues : compresser et combiner l'information . . **103**
7.4 Classification à l'aide de la distance de Mahalanobis **106**
7.5 Protocole expérimental **107**
7.6 Résultats **109**
7.7 Conclusion **116**

7.1 Introduction

Dans le Chapitre 6, des autoencodeurs débruitants sont utilisés pour réduire l'impact des erreurs introduites inévitablement par un système de reconnaissance automatique de la parole (SRAP) sur des documents représentés par des vecteurs TF.IDF (c.f. Chapitre 2). Ces réseaux de neurones appliquent une sélection et une compression de l'information contenue dans le document d'origine pour en extraire la variabilité pertinente et en ignorer le bruit. Un autre moyen possible permettant d'obtenir des documents robustes proposé dans la littérature est d'utiliser des représentations issues de modèles thématiques probabilistes. En effet, ces représentations se sont montrées assez résistantes aux erreurs de transcription automatique (Morchid et al., 2014a). Elles seraient particulièrement stables lorsque des erreurs d'omissions sont commises par le SRAP (Su et al., 2016). Dans ce but, nous proposons dans ce chapitre d'employer une représentation basée sur des modèles dits *Author-Topic* (AT) (Rosen-Zvi et al., 2004). Les modèles AT ont réduit la sensibilité aux bruits des systèmes automatiques (Morchid et al.) comparés

aux alternatives classiques l'allocation latente de Dirichlet(LDA) et LDA supervisées (sLDA) (Mcauliffe et Blei, 2008) présentées dans le Chapitre 2. Lors de l'optimisation de ces modèles, le choix de bons paramètres est un élément complexe qui impacte les représentations finales. Pour les modèles comme LDA et AT, le nombre de topics utilisés pour la modélisation des documents peut avoir un impact sur les performances et la robustesse d'un système automatique (Morchid et al. ; Wei et Croft, 2006b).

Une solution à cette problématique a été proposée (Morchid et al., 2015) : il s'agit de combiner plusieurs modèles thématiques de granularités différentes. Dans ce but, un processus complexe extrait d'un ensemble de modèles une représentation commune de faible dimension appelée c-vecteur. Cette représentation obtient de meilleurs résultats que le modèle AT. Ce résultat montre que les espaces thématiques latents appris avec différents niveaux de granularités (variation du nombre de *topics*) ne modélisent pas la même information et que ces informations sont complémentaires. Dans ce chapitre, nous proposons deux alternatives à la méthode c-vecteur : l'une repose sur un espace réduit appris par un autoencodeur débruitant et la seconde sur un sous-espace thématique latent (LTS) construit par décomposition en valeur propre (eigen value decomposition, EVD). Les représentations construites par ces différentes méthodes sont évaluées sur deux tâches de classification thématique : une tâche de classification thématique de documents bruités avec le corpus du projet DECODA et une tâche de classification de documents «propres» avec le corpus «20-Newsgroups». Pour cette expérience, la méthode de classification du plus proche centroïde reposant sur la distance de Mahalanobis est employée. Cette méthode a déjà obtenu de bonnes performances dans ce même contexte (Morchid et al., 2014, 2015). Les détails de cette méthode sont introduits dans la Section 7.4.

Le reste du chapitre est organisé de la manière suivante : D'abord, la Section 7.2 présente le modèle AT. Ensuite, la méthode de compression par c-vecteur est présentée dans la Section 7.3.1. Après, la compression construite par projection dans la couche cachée d'un autoencodeur et la compression pour la méthode LTS sont présentées dans les Sections 7.3.2 et 7.3.3 respectivement. Puis la Section 7.4 introduit la méthode de classification basée sur la distance de Mahalanobis. Après, les expériences pour évaluer ces compressions sont présentées Section 7.5 et les résultats des ces expériences sont analysés dans la Section 7.6. Enfin, les conclusions et perspectives de ces travaux sont abordées dans la Section 7.7.

7.2 Modèle *Author-Topic* (AT)

Le modèle *Author-Topic* (AT) (Rosen-Zvi et al., 2004) est un modèle génératif probabiliste qui modélise à la fois les auteurs et le contenu des documents par un mélange de thématiques. Pour décrire ce modèle, nous utiliserons la notation suivante : un document d est un vecteur de N_d mots w_d. Chaque mot w_{id} est choisi parmi un vocabulaire de taille V et un vecteur a_d d'auteurs choisi parmi un ensemble de A auteurs. Chaque auteur est associé à une distribution sur les topics θ qui suit une loi de Dirichlet symétrique α. Pour chaque mot w_{id} du document, un auteur x est tiré selon une loi uniforme. Ensuite un topic z est choisi selon la distribution des topics de cet auteur θ. Le mot w est généré

selon la distribution des mots dans le topic ϕ qui suit une loi de Dirichlet symétrique (β).

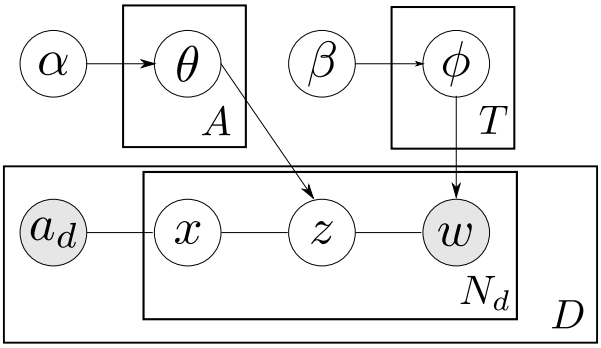


Figure 7.1 – *Le modèle Author-Topic sous forme de Plate notation.*

Le modèle graphique de la Figure 7.1 présente les relations entre les variables observées et les variables latentes du modèle AT. L'estimation des paramètres ϕ et θ permet de déterminer une représentation des documents en termes de topics ainsi que les topics abordés habituellement par les auteurs. Cette estimation peut être réalisée par échantillonnage de Gibbs de manière similaire à la LDA (Rosen-Zvi et al., 2004 ; Griffiths et Steyvers, 2004). Dans le cadre d'une classification supervisée, les auteurs du modèle thématique peuvent être assimilés aux labels de la tâche. Ainsi le modèle AT apprend la distribution des topics et des mots en tenant compte de la supervision. Ce modèle permet ainsi d'encoder des dépendances statistiques entre le contenu d'un document (mots w_{id}) et un ou plusieurs auteurs x au travers de la distribution des topics latents z.

L'échantillonnage permet aussi de déterminer les paramètres ϕ et θ pour représenter un nouveau document dans l'espace de topic. Cette représentation est le vecteur $V_d = P(x|d)$ composé d'une valeur pour chaque auteur (label). Chacune de ces valeurs est définie par :

$$V_x = \sum_{z=1}^{T} \sum_{w=1}^{N_d} \phi_{w,z} \theta_{x,z} \tag{7.1}$$

où $\phi_{w,z} = P(w|z)$ est la probabilité du mot w_i d'être généré par le topic z et $\theta_{x,z} = P(x|z)$ la probabilité que le label x soit généré par le topic z. Ce processus de création de représentation est schématisé dans par la Figure 7.2.

Dans la tâche que nous voulons traiter, les auteurs représentent les labels de la classification thématique et sont donc clairement définis. Par contre, le choix du bon nombre de topics est problématique pour un modèle AT. La Section suivante présente des méthodes de combinaison multivues et de sélection de l'information pour dépasser cette limitation.

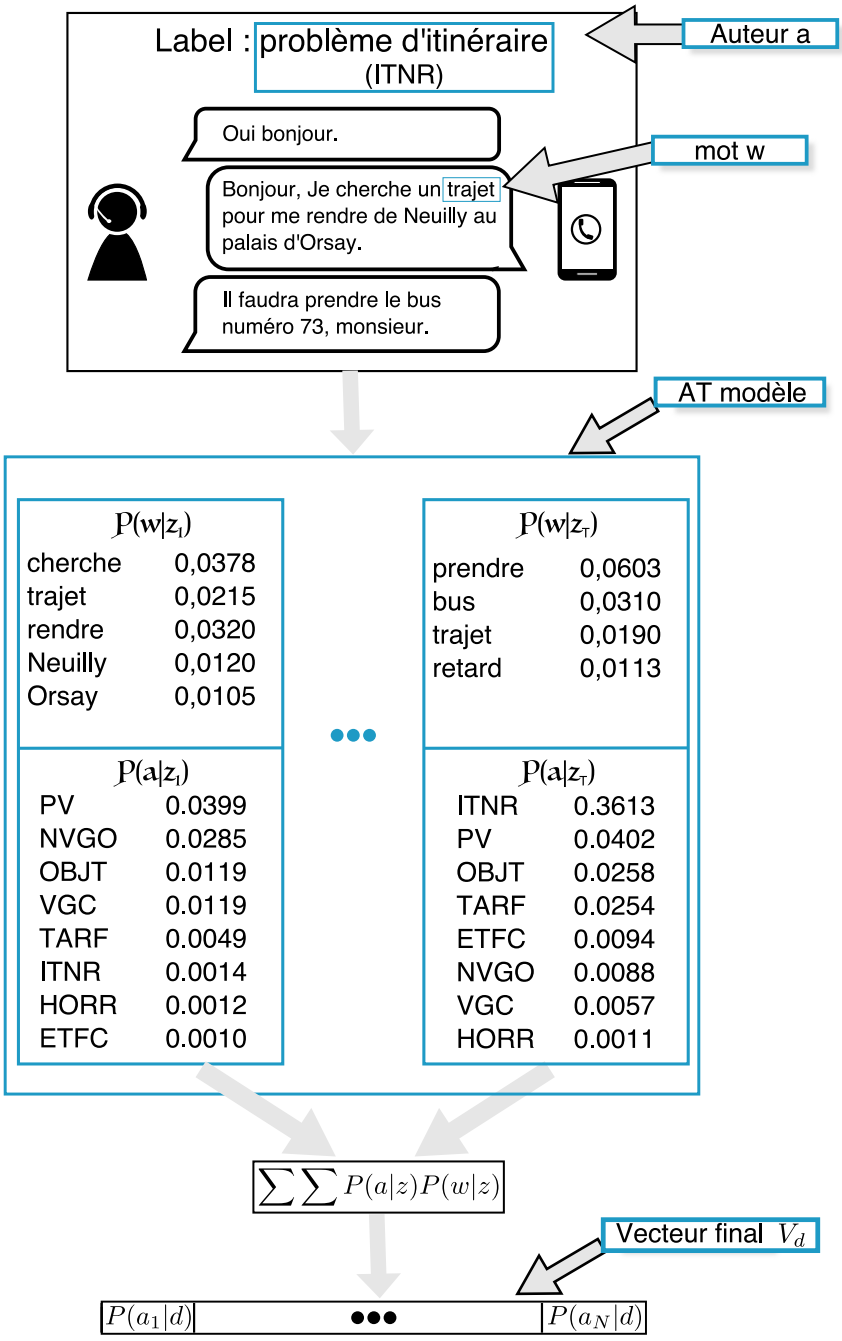


Figure 7.2 – *Exemple de conversation DECODA projeté dans un espace Author-Topic.*

7.3 Systèmes multivues : compresser et combiner l'information

Au lieu de choisir uniquement le meilleur modèle AT parmi un ensemble de variation, il est possible de construire une représentation utilisant l'ensemble des modèles. Les trois solutions présentées dans ce chapitre utilisent les projections d'un ensemble de modèles AT pour créer une représentation large unique appelée supervecteur. Le supervecteur est composé de la concaténation des projections dans chacun des modèles AT indépendamment. Les méthodes de compression vont ensuite extraire la variabilité la plus utile et ignorer celle qui est redondante dans ce lot d'informations. Elles peuvent aussi trouver les éléments complémentaires à travers les différents modèles en plus de les mettre en compétition. Ces propriétés sont importantes puisque la représentation supervecteur extraite des modèles AT peut contenir une quantité importante d'information redondante et de bruit. En effet, même si les différents niveaux de granularité appris par les différents modèles AT en variant le nombre de topics peuvent porter des informations complémentaires, plusieurs modèles vont aussi porter une information similaire.

La construction du supervecteur et les différentes méthodes de fusion des modèles sont schématisées dans la Figure 7.3

Les sous-sections suivantes présentent trois méthodes de traitement du supervecteur pour la compression des modèles AT. La méthode des *c*-vecteurs (Morchid et al., 2015) et deux propositions dont l'une repose sur un autoencodeur et l'autre sur des sous-espaces propres.

7.3.1 Compression de l'information en *c*-vecteurs

La méthode de construction des *c*-vecteurs (Morchid et al., 2015) est une adaptation pour le traitement du texte des *I*-vecteurs conçus à l'origine pour la reconnaissance du locuteur (Dehak et al., 2011) dans le but de compresser l'information. Le processus de constructions des *c*-vecteurs peut-être résumé comme présenté dans la Figure 7.3-a. Il est composé de trois étapes :

D'abord, la représentation large initiale est projetée dans un espace construit par un modèle de mélange de gaussiennes et de contexte universel (Gaussian Mixture Model-Universal Background Model, GMM-UBM) (Reynolds et Rose, 1995).
Ensuite, l'information est compressée dans un vecteur de taille réduite par une méthode d'analyse factorielle jointe (Dehak et al., 2011 ; Kenny, 2005). Enfin, une méthode de standardisation appelée transformation «eigen factor radial» (EFR) (Bousquet et al., 2011) est appliquée pour contrebalancer certaines faiblesses des représentations *I*-vecteur original.

Les vecteurs compressés et standardisés peuvent ensuite être utilisés pour la classification.

7.3.2 Compression avec un autoencodeur débruitant (DAE)

Dans le Chapitre 6, l'efficacité des DAE pour le débruitage et la compression de l'information a été démontrée. Le bruit présent dans les documents d'origine impacte différemment chacun des modèles AT employés. La quantité de bruit et d'information redondante dans les supervecteurs est donc potentiellement très importante. Un autoencodeur peut donc combiner et débruiter l'information contenue dans les différents modèles pour créer une représentation commune de taille réduite (Vincent et al., 2008b). En s'appuyant sur le même principe, des autoencodeurs ont été utilisés pour fusionner de l'information audiovisuelle (Kim et al., 2013 ; Ngiam et al.).

Fusionner les informations issues des 500 modèles AT avec un DAE permet de simplifier le processus de création de représentation introduite par les *c*-vecteurs tout en conservant de bonnes capacités de représentation. Le processus simplifié est présenté dans la Figure 7.3-b. Il ne contient plus qu'une unique étape intermédiaire non supervisée reposant sur le DAE.

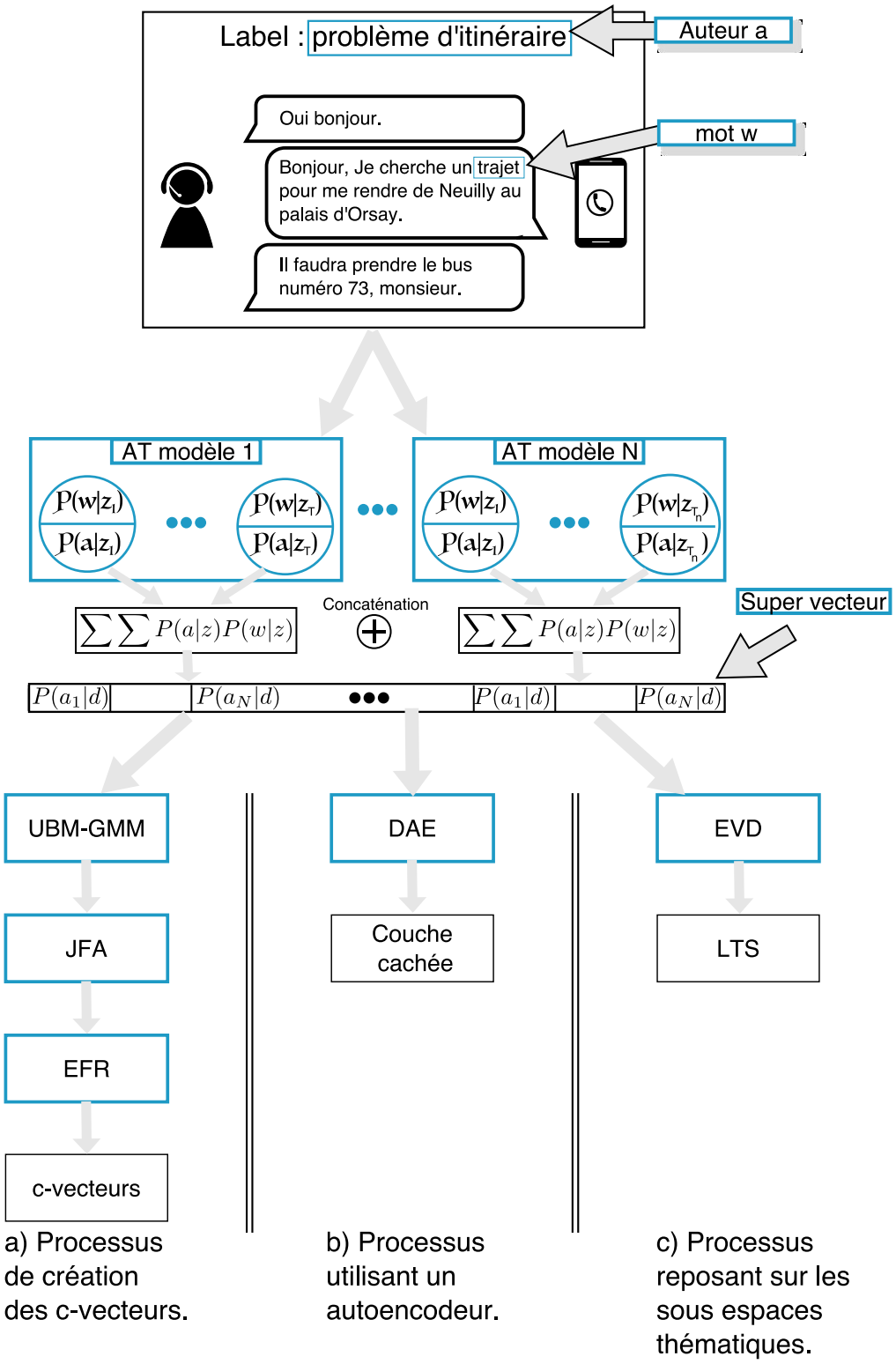


Figure 7.3 – *Trois processus de création des représentations compressées. (a) l'état de l'art c-vecteurs, (b) la proposition à base d'autoencodeur et (c) la proposition LTS*

7.3.3 Compression avec des sous-espaces thématiques latents (LTS)

La construction d'un c-vecteur passe par la projection des documents dans un modèle GMM-UBM qui génère une représentation avec un très grand nombre de dimensions (taille initiale multipliée par le nombre de gaussiennes utilisées par le modèle GMM-UBM). La méthode des sous-espaces de thématiques latentes (latent thematic subspaces,LTS) est-elle composée d'un ensemble de sous-espaces où chaque document est projeté. Ainsi, les différentes représentations d'un même document partagent une structure latente commune. Ces paramètres communs définissent un sous-espace thématique. Chaque supervecteur s_d, construit à partir d'un document d donné provenant d'un corpus de taille N, est partiellement associé avec un sous ensemble de variables latentes. Le supervecteur s_d est obtenu en concaténant les représentations issues des modèles AT $V_{d,r}^{a_k^r} \forall r \in t_1, \ldots, t_T$ où $t_1, \ldots, t_T$ est l'ensemble des espaces thématiques latents des modèles AT. On obtient ainsi une matrice § de supervecteur § $= [s_0, \ldots, s_d, \ldots, s_N]$ qui représente les documents dans le LTS. Elle est ensuite compressée par EVD (Abdi et Williams, 2010), (Golub et Reinsch, 1970) pour produire une représentation de faible dimension $\mathbf{h}_{d,e}$ dont la taille dépend des valeurs propres e définies dans :

$$\S = \mathbf{P}\Delta\mathbf{V}^T. \tag{7.2}$$

Où P est la matrice des vecteurs singuliers gauches , $\mathbf{V}$ est la matrice des vecteurs singuliers droits et $\mathbf{\Delta}$ la matrice diagonale des valeurs singulières. N est le rang de la matrice §.

La représentation compacte est donc définie par :

$$\mathbf{h}_{(d,e)} = (\mathbf{s}_d - \bar{s}) . \mathbf{V}_e^T. \tag{7.3}$$

Où $\mathbf{V}_e$ est la matrice de vecteurs propres et $\bar{s}$ et le centroïde de tous les supervecteurs contenus dans le corpus d'entrainement. Les processus de construction de ces représentations sont présentés par la Figure 7.3-c.

Les deux représentations détaillées ci-dessus proposent un processus de construction simplifié tout en conservant une capacité à fusionner l'information issue des différents modèles. La section suivante présente la méthode de classification utilisée dans ce chapitre qui repose sur la distance de Mahalanobis.

7.4 Classification à l'aide de la distance de Mahalanobis

La distance de Mahalanobis peut être utilisée pour attribuer à un document d la classe la plus probable $C_{k_{Bayes}}$. Pour un corpus d'entrainement donné, $\mathbf{W}$ représente la matrice de covariance inter documents définis par :

$$\mathbf{W} = \sum_{k=1}^{K} \frac{n_k}{n} \mathbf{W_k} = \frac{1}{n} \sum_{k=1}^{K} \sum_{i=0}^{n_t} (x - \mu_k) (x - \mu_k)^t \tag{7.4}$$

où $\mathbf{W_k}$ est la matrice de covariance de la thématique C_k et n_k correspond au nombre de documents appartenant à la classe C_k. n est le nombre total de documents et μ_k est le document moyen (centroïde) de la classe C_k. Il est donc possible d'écrire :

$$C_{k_{Bayes}} = \arg\max_k \left\{ P\left(x|C_k\right) \right\}. \tag{7.5}$$

La vraisemblance $P\left(x|C_k\right)$ suit une loi normale qui a pour fonction de densité $\mathcal{N}\left(x|\mu_k, \mathbf{W}\right)$. Pour la classe k donnée, elle est définie par :

$$P\left(x|C_k\right) = (2\pi)^{-\frac{p}{2}} \left|\mathbf{W}\right|^{-\frac{1}{2}} \exp\left\{ -\frac{1}{2}\left(x-\mu_k\right)^t \mathbf{W}^{-1}\left(x-\mu_k\right) \right\}$$

où μ_k est la moyenne de classe et p est la taille de ce vecteur alors :

$$\log P\left(x|C_k\right) = C - \frac{1}{2}\left(x-\mu_k\right)^t \mathbf{W}^{-1}\left(x-\mu_k\right)$$

$$\text{avec } C = -\frac{p}{2}\log\left(2\pi\right) - \frac{1}{2}\log\left|\mathbf{W}\right| \text{ et :}$$

$$\log P\left(x|C_k\right) = C - \frac{1}{2}x^t\mathbf{W}^{-1}x + x^t\mathbf{W}^{-1}\mu_k - \frac{1}{2}\mu_k^t\mathbf{W}^{-1}\mu_k. \tag{7.6}$$

Comme C et le terme quadratique $x^t\mathbf{W}^{-1}x$ sont indépendants de la classe, la règle de décision bayésienne devient :

$$C_{k_{Bayes}} = \arg\max_k \left\{ x^t\mathbf{W}\mu_k - \frac{1}{2}\mu_k^t\mathbf{W}\mu_k \right\}. \tag{7.7}$$

Dont tous les termes sont connus. Nous pouvons donc déduire une probabilité d'appartenance à chaque classe d'un document.

Nous avons défini le modèle AT pour construire des représentations des documents puis des méthodes multivues proposées pour combiner les projections dans plusieurs modèles AT d'un même document. La section suivante présente le protocole expérimental proposé pour évaluer les performances de ces systèmes de classification automatiques.

7.5 Protocole expérimental

L'efficacité des représentations proposées est évaluée en répétant la même expérience sur deux jeux de données différents. D'abord, sur la tâche de classification de documents

bruités en utilisant les données transcrites automatiquement du projet DECODA présentées dans le Chapitre 4 sous le nom de TID. Ensuite, sur la tâche de classification thématique (sans erreurs de transcription) en utilisant le corpus «20-Newsgroups».

Ce second corpus que nous présentons ici, est une collection de 20 000 documents extraits à partir de 20 groupes de discussions répartis en données d'entrainement et de test respectivement 60% et 40%. Environ 10% de ce corpus d'entrainement est utilisé comme données de développement dans ces expériences. Ce corpus a un vocabulaire de 130 107 mots uniques. La répartition des documents et des mots est présentée dans le Tableau 7.1 .

Table *7.1* – *La répartition des documents dans le 20-Newsgroups corpus.*

Label	*Nombre de documents*			*Nombre*
	entrainement	**développement**	**test**	*de mots*
alt.atheism	432	48	319	161 442
comp.graphics	526	58	389	123 428
comp.os.ms-windows.misc	515	57	394	95 238
comp.sys.ibm.pc.hardware	531	59	392	112 583
comp.sys.mac.hardware	521	57	385	97 182
comp.windows.x	534	59	392	175 024
misc.forsale	527	58	390	69 761
rec.autos	535	59	395	127 696
rec.motorcycles	539	59	398	115 430
rec.sport.baseball	538	59	397	119 407
rec.sport.hockey	540	60	399	153 949
sci.crypt	536	59	396	219 507
sci.electronics	532	59	393	115 503
sci.med	535	59	396	170 336
sci.space	534	59	394	168 481
soc.religion.christian	539	59	398	216 581
talk.politics.guns	491	54	364	191 723
talk.politics.mideast	508	56	376	272 488
talk.politics.misc	419	46	310	202 625
talk.religion.misc	340	37	251	129 611
Total	**10 172**	**1 121**	**7 528**	**3 037 995**

Pour ces expériences, les représentations construites par le modèle AT sont évaluées. Le nombre de topics utilisés dans un modèle AT permet de déterminer le niveau de granularité thématique modélisé par les différents topics. Un modèle possédant un petit nombre de topics va leur attribuer un sens plutôt générique alors qu'un nombre de topics plus élevés permet de faire des regroupements thématiques plus fins. Le choix du nombre de topics idéal pour une tâche détermine l'information portée par le modèle. Nous proposons d'employer pour chaque tâche 500 modèles AT. Chacun de ces modèles utilise un nombre croissant de topics compris entre 5 et 505. Pour chaque modèle l'ensemble des documents du corpus est projeté dans l'espace latent AT et se voit attribuer à une classe

par la distance de Mahalanobis présentée dans la Section 7.4. La capacité à construire des représentations robustes des modèles AT pris indépendamment est ainsi évaluée.

Dans un second temps, les 500 représentations construites ci-dessus sont concaténées pour former un unique supervecteur de très grande dimension. Les méthodes de fusion présentées vont compresser l'information accumulée dans les supervecteurs et seront ensuite évaluées en classification. Plusieurs configurations de *c*-vecteurs sont comparées en faisant varier le nombre de gaussiennes utilisées parmi 16, 32, 64 et 128 ainsi que la taille des *c*-vecteurs parmi 80, 120, 200 et 300. Pour la fusion par DAE plusieurs modèles sont entrainés en faisant varier la taille de la couche cachée entre 10 et 300 par pas de 10. Enfin, plusieurs projections par LTS sont aussi analysées en faisant varier le nombre de valeurs propres sélectionnées entre 40 et 300.

Les résultats en classification de ces différents modèles de compression sont présentés dans la Section 7.6 qui suit.

7.6 Résultats

Les résultats de ces expériences sont répartis en trois parties pour en faciliter l'analyse. La première Sous-Section 7.6.1 présente les résultats obtenus sur les deux jeux de données (TID et 20-Newsgroups) avec les approches unidimensionnelles. Ensuite, la seconde Sous-Section 7.6.2 présente les résultats obtenus par l'approche de compression multidimensionnelle sur les données de la tâche TID. Enfin, la troisième Sous-Section 7.6.3 présentera les résultats obtenus par les approches de compression multidimensionnelles sur les corpus 20-Newsgroups.

7.6.1 Résultats des représentations AT.

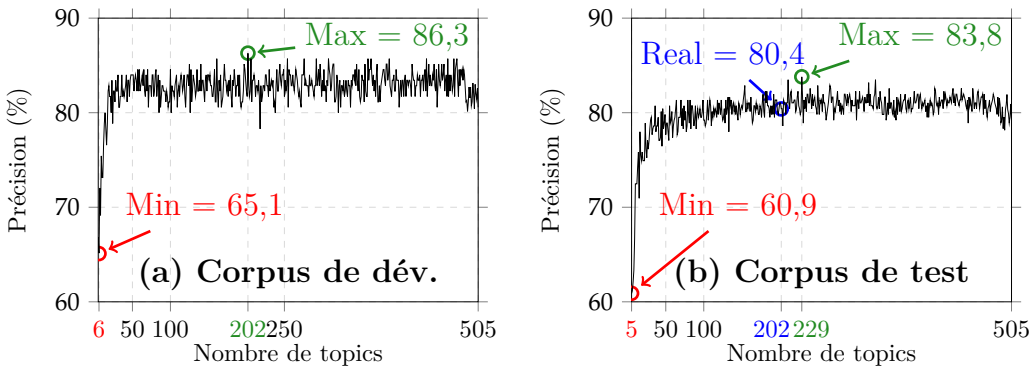


Figure 7.4 – *Performances (%) réalisées sur la tâche TID en utilisant des modèles Author-topic en variant le nombre de topics utilisé pour l'apprentissage.*

La Figure 7.4 montre les performances obtenues pour la tâche de classification des documents bruités. Elle présente les résultats obtenus sur le corpus de développement

(Figure 7.4-a) ainsi que sur le corpus test (Figure 7.4-b) selon le nombre de topics utilisés lors de l'entrainement. On peut remarquer dans ces graphiques que les performances en classification sont assez instables, ils varient entre 60,9% et 83,8%. Le modèle optimisé sur le corpus de développement obtient un résultat de 70,7% sur le même corpus et 80,4% sur les données de test, soit 3,4 points en dessous des meilleurs résultats possible. Ce score confirme l'instabilité des performances sur ces données.

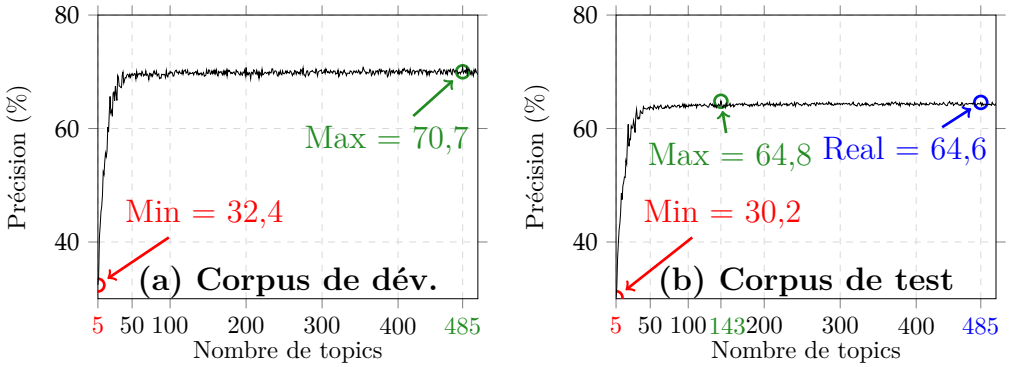


Figure 7.5 – *Précision réalisée sur le corpus* **20-Newsgroups** *en fonction du nombre de topic utilisé par les modèles Author-Topic.*

La Figure 7.5 détaille les performances réalisées sur le corpus 20-Newsgroups sur les données de développement (Figure 7.4-a) et de test (Figure 7.4-b) toujours en fonction du nombre de topics utilisé pour l'apprentissage. Le résultat final sélectionné sur les données de développement obtient 70,7% et 64,6% sur données de test en utilisant 485 topics pour l'apprentissage des modèles AT. Ce résultat est proche des 64,8% qui pourrait être obtenu par le meilleur modèle sur le corpus de test en fonction du nombre de topics. On peut noter aussi qu'avec la faible présence de bruit dans ce jeu de données, les représentations construites par les modèles AT sont robustes au changement du nombre de topics lors de l'apprentissage.

7.6.2 Approches multivues sur une tâche de classification de documents parlés

Cette section présente les résultats des approches multidimensionnelles réalisées sur le corpus de la tâche TID. Les supervecteurs construits sur ce jeu de données sont de taille $500 \times 8 = 4000$ (où 8 représente le nombre de classes). Les résultats sont présentés dans l'ordre suivant : d'abord les *c*-vecteur, puis en utilisant un DAE et enfin en utilisant la projection dans un LTS.

Compression en *c*-vecteur

Le Tableau 7.2 rapporte les précisions obtenues par les représentations en *c*-vecteur en compressant les 500 projections des différents modèles AT. Ces résultats varient en

Table 7.2 – *Précision (%) de classification thématique de dialogues transcrits automatiquement selon plusieurs configurations de c-vecteur.*

Taille du c-vecteur	*Dev.*			*Test*		
	Nombre de gaussiennes dans le modèle GMM-UBM					
	32	*64*	*128*	*32*	*64*	*128*
80	80,6	82,3	83,1	79,2	81,0	80,4
100	81,7	**84,6**	83,1	78,9	**82,3**	80,4
120	84,0	81,7	82,3	80,4	79,2	81,8

fonction de la taille des vecteurs compressés et le nombre de gaussiennes utilisées pour le modèle GMM-UBM. On peut remarquer que les résultats de cette représentation compacte dépassent les performances du modèle AT choisi sur le corpus de développement de 1,9 points sur les données de tests. Il est possible de remarquer aussi que cette approche généralise bien, puisque le modèle qui obtient les meilleures performances sur le corpus de développement (84,6%) et aussi le meilleur sur le corpus de test en obtenant 82,3%.

Compression par autoencodeur débruitant

La Figure 7.6 rapporte les résultats obtenus avec un DAE. Les précisions sont présentées en fonction de la taille de la couche cachée ($10 \leq |\mathbf{h}| \leq 300$) de l'autoencodeur. On peut remarquer que la précision réalisée par cette compression est de 89% sur le corpus de développement et 83% sur le corpus de test. Ce qui représente un gain de 2,6 points comparé à la représentation univue. On peut noter également que le résultat minimum est atteint trois fois pour les tailles 270, 275 et 300. Ce résultat montre que, dans ce cas, une couche cachée de trop grande taille dégrade les performances du modèle.

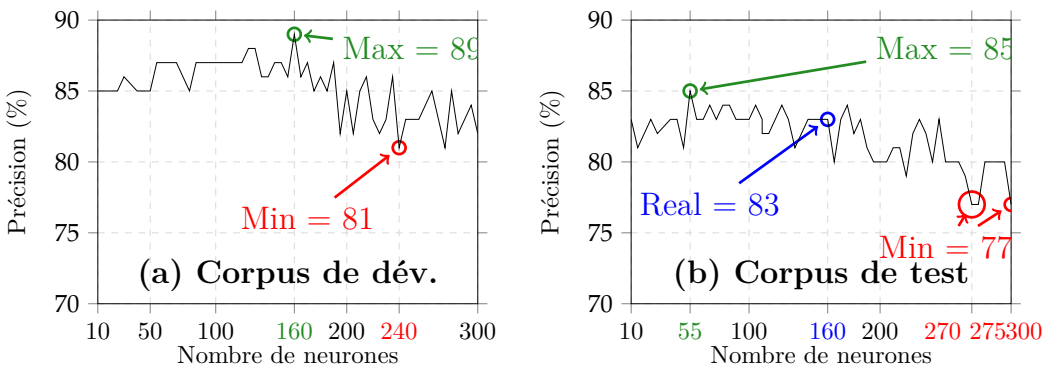


Figure 7.6 – *Performances (%) réalisées sur la tâche TID en utilisant la couche cachée d'un autoencodeur et en variant le nombre de neurones dans cette couche entre 10 et 300 neurones.*

Compression par sous-espace latent thématique

Les performances obtenues par la compression LTS sont rapportées dans la Figure 7.7. On peut voir que le modèle, obtenant les meilleures performances sur le corpus de développement, obtient une précision de 85,0% sur le corpus de test et est très proche du meilleur résultat possible de 85,3%. On peut noter aussi que les performances sont relativement stables selon le nombre de vecteurs propres choisis avec un minimum à 83,2%.

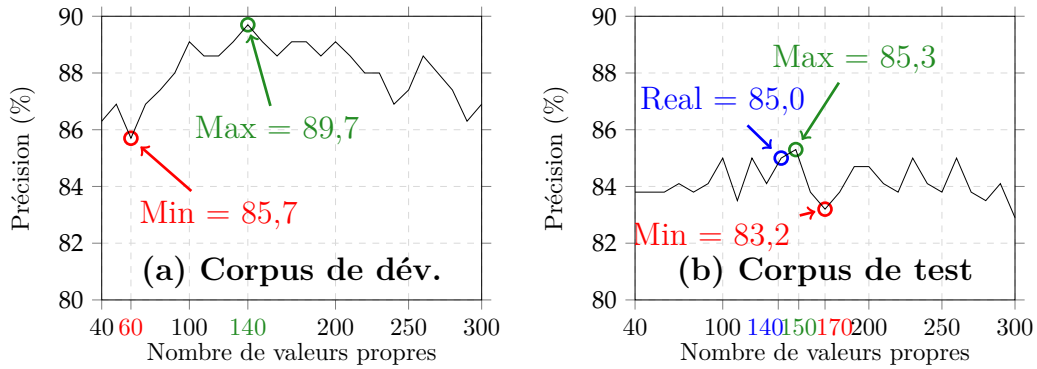


Figure 7.7 – *Précision (%) obtenue sur la tâche d'identification de thématique TID, en utilisant les projections des documents dans les sous-espaces LTS en variant le nombre de valeurs propres entre 40 et 300.*

Bilan et analyse

Table 7.3 – *Précision en classification thématique (%) sur la tâche TID.*

représentation	Dev.		Test	
	Taille	Max	Max	Résultat
LDA (Morchid et al.)	160	82.3	78.3	73.1
sLDA (Morchid et al.)	100	80.0	76.8	74.6
Auteur Topic Model (AT) (référence)	202	86.3	83.8	80.4
AT + *c*-vecteur	100	84.6	82.3	82.3
AT + DAE	160	89.0	85.0	83.0
AT + LTS	140	89.7	85.3	**85.0**

Le Tableau 7.3 rappelle les meilleurs résultats obtenus pour chacune des configurations ainsi que les performances d'autres approches reposant sur des modèles thématiques présentent dans la littérature. La première information visible dans ce tableau est que les modèles *Author-Topic* offrent une modélisation plus efficace de l'information de supervision (*labels* $\leftrightarrow$ *auteurs*) que les modèles de sLDA. Les *c*-vecteurs apportent une amélioration de 1,9 points comparée aux résultats obtenus avec le modèle AT. Le DAE apporte un gain réel de 2,6% absolu par rapport aux représentations unidimensionnelles soit environ 0,7 point de plus que les *c*-vecteurs pour un processus beaucoup plus

simple. Dans le meilleur des cas, un résultat de 85% peut être obtenu et est supérieur au meilleur modèle AT seul. Il offre ainsi une performance équivalente aux LTS. En effet, les modèles LTS permettent d'obtenir les meilleures performances de cette expérience avec un résultat de 85% tout en offrant aussi un processus simplifié comparativement aux *c*-vecteurs.

Ces résultats montrent que les différents modèles AT portent une information complémentaire que les modèles de fusion sont capables d'extraire. Ils montrent également que la création de *c*-vecteur est non seulement plus complexe et couteuse en calcul, mais aussi qu'elle n'apporte aucun avantage dans ces conditions comparé à des méthodes plus directes.

7.6.3 Approche multivues sur une tâche de classification de documents écrits

Dans cette section les résultats des expériences réalisées sur le corpus 20-Newsgroups sont présentés. Les supervecteurs construits avec ces modèles sont de taille $500 \times 20 = 10000$ (où 20 est le nombre de classes).
D'abord, les précisions obtenues par compression en *c*-vecteurs sont détaillées. Puis les compressions par DAE et par LTS sont présentées. Pour finir, une comparaison globale de ces différentes représentations est réalisée.

Compression *c*-vecteur

Table 7.4 – *Précision en classification (%) sur les données 20-Newsgroups utilisant des c-vecteurs en variant leur taille et le nombre de gaussiennes utilisées lors de l'apprentissage.*

Taille des *c*-vecteurs	*Dev.*				*Test*			
	Nombre de gaussiennes (GMM-UBM)							
	16	*32*	*64*	*128*	*16*	*32*	*64*	*128*
80	69,3	67,9	67,3	66,1	62,3	62,5	62,1	60,9
120	69,0	68,7	68,3	66,7	63,7	62,5	62,2	61,5
200	69,4	69,3	67,9	68,9	64,0	62,4	61,9	62,6
300	**70,4**	67,4	67,7	64,8	**64,9**	61,1	61,0	59,7

Les précisions obtenues en compressant les 500 modèles AT dans des *c*-vecteurs sont répertoriées dans le Tableau 7.4. On peut remarquer que cette méthode n'apporte que peu d'information supplémentaire. Dans la meilleure situation, un gain de $0,3$ point est notable comparativement aux représentations AT seul. Ces résultats sont obtenus avec un faible nombre de gaussiennes mais des *c*-vecteurs larges. L'utilisation d'un espace GMM-UBM est pénalisante dans ces conditions.

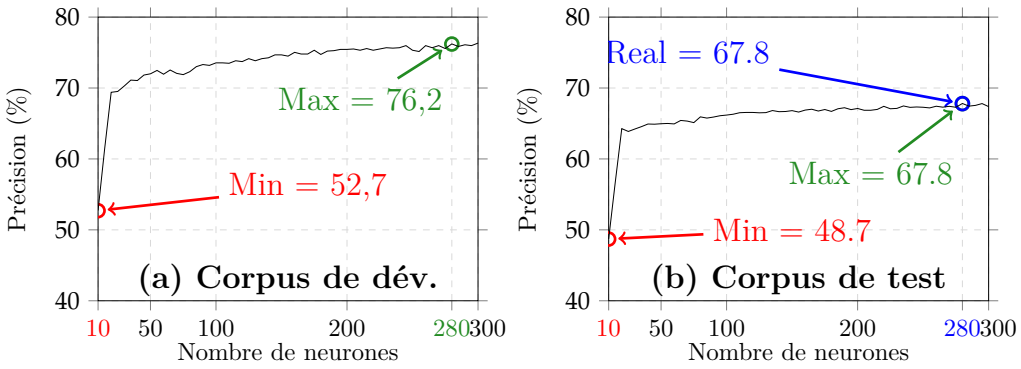


Figure 7.8 – *Précision (%) obtenue sur la tâche de classification de documents textes* **20-Newsgroups** *en utilisant les couches cachées d'un autoencodeur variant le nombre de neurones de 10 et 300.*

Compression par autoencodeur débruitant

Les courbes de la Figure 7.8 montrent les précisions obtenues à partir de représentations compressées construites par un DAE en fonction de la taille de sa couche cachée. Elle varie de 10 et 300. Cette représentation obtient une précision de $76,2\%$ sur le corpus de développement soit un gain de 6 points comparé aux méthodes précédentes. Cette amélioration ne se retrouve pas sur le corpus de test avec un score de $67,8\%$ apportant un gain absolu d'environ 3%. Cette différence souligne que le DAE ne généralise pas assez bien quand une variance importante est introduite dans le modèle.

Compression par sous-espace latent thématique

Le taille des sous-espaces latents thématiques (LTS) pour cette expérience varie de 40 à 300. La Figure 7.9 présente les précisions réalisées en classification sur les corpus de développement et de test lorsque les supervecteurs sont projetés dans les LTS. Cette méthode permet d'obtenir $75,9\%$ et $67,3\%$ sur les données de développement et de test respectivement. Les résultats réalisés sont un peu plus faibles que les résultats avec compression par DAE de $0,3$ et $0,5$ point sur ces mêmes données.

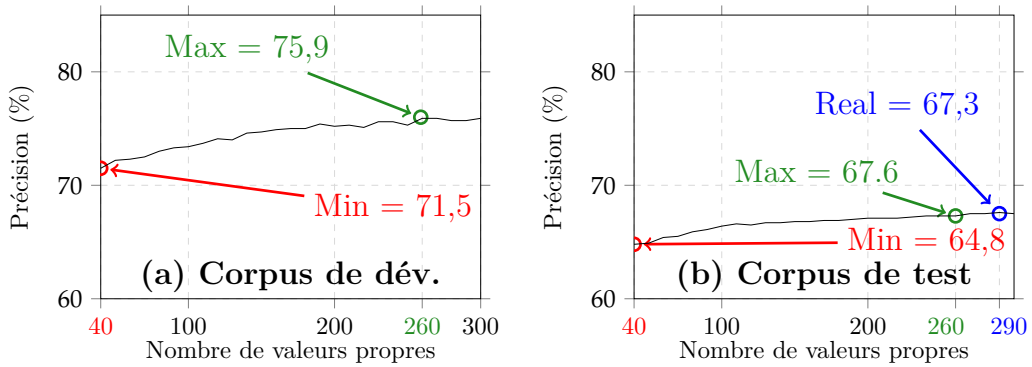


Figure 7.9 – *Précision (%) obtenue sur la tâche de classification de documents textes* **20-Newsgroups** *en utilisant les projections des documents dans les sous espaces* **LTS** *en variant le nombre de valeurs propres de 40 et 300.*

Bilan et analyse

Table 7.5 – *Précisions (%) obtenues par les meilleures configurations de chacun des systèmes présentés sur le Corpus 20-Newsgroups.*

Type de représentation	**Dev.** **taille**	Max	**Test** Résultat
Modèle *Author-Topic* (AT) (référence)	485	70,7	64,6
AT + *c*-vecteur	300	70 ,4	64,9
AT + DAE	280	67,8	**67,8**
AT + LTS	260	75,9	67,3

Le Tableau 7.5 compile les meilleures performances réalisées sur le corpus 20-Newsgroups en utilisant les représentations classiques et multivues. On peut remarquer que dans ce jeu de données, la compression c-vecteur n'apporte qu'un faible gain ($+0,3$) comparé à la représentation AT simple. Ce résultat peut s'expliquer par l'invariance au nombre de topics dans cette tâche. Si tous les modèles sont très proches, il est probable qu'ils modélisent la même information et donc que la fusion n'apporte que peu d'intérêt. La méthode LTS permet un gain absolu de $2,6\%$. En plus de fusionner l'information, le choix du nombre de valeurs propres permet d'ignorer la variabilité portée par les valeurs propres les plus faibles qui n'importent pas pour la classification. Enfin, la compression par DAE permet une amélioration de $3,2$ points. Le fait que les meilleurs c-vecteurs soient obtenus avec le minimum de gaussiennes et que le DAE surpasse les autres représentations montrent que, sur ces données, la capacité du modèle à filtrer l'information inutile est un facteur important.

7.7 Conclusion

Dans ce chapitre, nous avons proposé deux représentations multidimensionnelles capables de fusionner un ensemble important de modèles thématiques (i.e. *Author-Topic*) appris avec différents paramètres.

La première méthode proposée utilise un autoencodeur débruitant pour créer une représentation latente de taille réduite tout en réduisant l'impact du bruit. La seconde repose sur une décomposition en valeurs propres pour créer un sous-espace thématique latent commun qui ne conserve que la variabilité la plus importante.

Ces deux méthodes ont été évaluées pour la classification sur deux corpus de données : un ensemble de données bruitées issues de transcriptions automatiques et un jeu de documents textuels issus de groupes de discussion et comparés aux c-vecteurs.

Dans les deux situations, ces méthodes apportent un gain important comparé aux représentations construites à partir des modèles AT pris indépendamment ou à partir d'une méthode de fusion inspirée des I-vecteurs. Sur les documents bruités, le LTS et le DAE apportent un gain absolu $2,7\%$ et $0,7\%$ respectivement comparé aux c-vecteurs. Sur les données issues des groupes de discussion, ils permettent une amélioration absolue de $2,4\%$ et $2,9\%$. Nous pouvons remarquer que lorsque le corpus est de taille limitée (TID), la méthode à base de LTS tend à être plus robuste alors que le DAE semble capable de construire de meilleures représentations quand le corpus est suffisamment grand (20-Newsgroups).

Pour ces travaux nous voyons deux pistes d'évolution intéressantes. La première consiste à utiliser des méthodes de compression à base de réseaux de neurones génératifs tels que les Machines de Botlzmann profondes, des autoencodeurs variationnels ou des réseaux génératifs adverses qui sont présentés dans le Chapite 3. Ces méthodologies permettraient de réaliser des représentations de taille réduite en conservant les facultés génératives des modèles thématiques d'origine. Ces expériences doivent être conduites à plusieurs niveaux. En effet, ces réseaux de neurones peuvent se substituer aux modèles thématiques (AT/LDA) ainsi qu'aux méthodes de compression (DAE/LTS). Si la quantité de données d'apprentissage est suffisante, il est aussi possible de réaliser les deux opérations simultanément dans un modèle neuronal bout en bout (end to end), ce qui simplifierait encore le processus. Il serait également utile d'introduire une comparaison détaillée avec des modèles profonds déterministes plus classiques comme les réseaux de convolution ou les LSTM.

La seconde évolution que nous proposons consiste à exploiter les capacités des autoencodeurs débruitants à compresser efficacement l'information provenant de différents modèles (démontré dans cette expérience) pour élaborer un réseau de neurones artificiels capable de combiner à la fois l'information bruitée et l'information propre, afin d'apprendre à construire une représentation qui tiendrait compte du bruit pour être plus robuste. Dans le cadre de classification de documents parlés, nous pourrions créer une représentation multidimensionnelle qui emploierait, lors de l'apprentissage, la transcription automatique ainsi que la transcription manuelle. Ce réseau permettrait de modéliser

plus efficacement le bruit et de créer une représentation robuste et efficace. C'est cet axe d'amélioration que nous avons choisi d'approfondir dans les Chapitres 8 et 9.

Chapitre 8

Supervision de l'apprentissage des autoencodeurs pour améliorer la compréhension de la parole.

Sommaire

8.1 Introduction 119
8.2 Autoencodeur profond supervisé (SDAE) 122
8.3 Expériences 124
8.4 Résultats 125
8.5 Discussion 128
8.6 Conclusion 129

8.1 Introduction

Les chapitres précédents ont présenté et adapté des méthodes de classification thématique de documents parlés. Une des principales difficultés rencontrées par un système qui repose sur des transcriptions automatiques est de gérer les taux d'erreurs importants introduits par les SRAP en amont. Le Chapitre 6 a montré que des DAE peuvent être utilisés pour créer des représentations latentes robustes des documents transcrits automatiquement. Ces représentations sont construites de manière à être plus réduites, plus denses, moins sensibles aux variabilités. Elles mettent en évidence les différences interdocuments. et permettent ainsi d'améliorer les performances globales du système de SLU. Toujours dans le Chapitre 6, nous avons introduit des autoencodeurs débruitants profonds (DDAE) qui apprennent à reconstruire les transcriptions manuelles à l'aide de transcriptions automatiques. Cependant, les résultats obtenus avec les représentations latentes construites par ces DDAE se sont révélés décevants. L'hypothèse qu'un système, qui connait les représentations idéales (oracle) et corrompues d'un même élément, puisse en déduire les patrons que suit le processus de corruption de l'information parait

pourtant réaliste. Nous en avons déduit que les erreurs introduites par le SRAP sont trop nombreuses et trop codépendantes pour qu'un DDAE puisse les modéliser aussi directement.

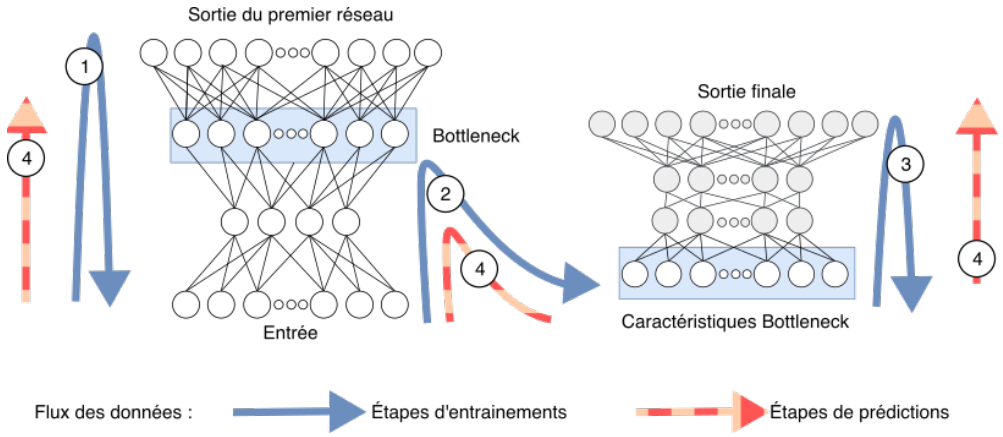


Figure *8.1* – *Processus d'utilisation de caractéristiques Bottleneck.*

Des méthodes d'apprentissage de réseaux de neurones reposent sur un apprentissage en deux processus distincts. Un premier réseau est entrainé pour l'extraction de caractéristiques propres aux données d'entrainement. Alors qu'un second réseau utilise la projection produite par le premier pour réaliser la tâche traitée. Les vecteurs produits lors de la projection sont appelés caractéristiques *bottleneck*. Le terme anglais *bottleneck* se traduit par goulot d'étranglement ou entonnoir en français. La Figure 8.1 présente cette idée. Dans ce schéma, l'étape 1 correspond à l'apprentissage d'un réseau pour la génération de représentation latente. La fonction objectif utilisée par celui-ci est variable, elle peut être identique (Grézl et al., 2007) ou différente (Tan et Mak) de l'objectif de la tâche finale. Durant l'étape 2 , les documents d'entrainements sont projetés dans la couche bottleneck (représentée par l'encadré bleu). Lors de l'étape 3, le second réseau est entrainé pour résoudre la tâche en utilisant comme vecteurs d'entrée les projections construites durant l'étape 2. L'étape 4 montre les étapes successives nécessaires pour réaliser une prédiction pour un nouveau document en utilisant des caractéristiques bottleneck.

La Figure 8.2 présente la génération des caractéristiques *bottleneck* correspondant à l'étape 2 du précédent schéma. Le plus souvent, ce terme fait référence à la plus petite couche cachée positionnée au centre du réseau. Ce n'est cependant pas une obligation (Doddipatla et al., 2014), (Carcenac et Redif, 2015), (Zhang et al., 2014). Chaque couche cachée d'un réseau de neurones profonds génère des représentations avec un niveau de compression. La couche cachée optimale à utiliser peut être sélectionnée en fonction d'une interprétation de sa capacité de représentation (Grézl et al., 2007) ou en fonction de ses résultats sur un corpus de développement.

La formulation des autoencodeurs dans le Chapitre 6 peut être vue comme un cas particulier de cette méthode dont le premier réseau de neurones artificiels est composé d'un encodeur et d'un décodeur. C'est le cas aussi des *Word2vec* présentés dans le Chapitre 5.

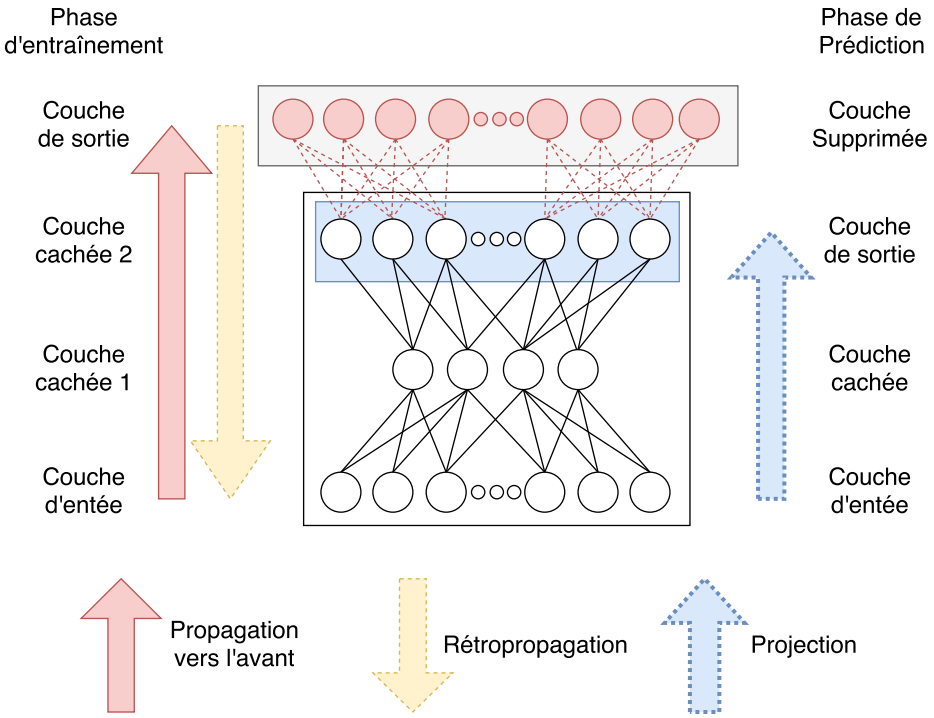


Figure *8.2* – *Génération de caractéristiques Bottleneck.*

Cette méthode permet d'éviter la disparition du gradient. Durant l'étape de rétropropagation, le gradient devient de plus en plus faible au fur et à mesure qu'il est propagé dans les couches les plus profondes. Si un réseau profond est séparé en deux parties moins profondes, avec chacune leur propre descente de gradient, le phénomène de disparition est alors moins important. Lorsque le premier réseau est entrainé avec un objectif propre, par exemple avec l'emploi d'un autoencodeur, elle apporte des avantages supplémentaires. D'abord, elle permet de s'assurer que le second réseau conserve de bonnes capacités de généralisation. Enfin, elle réduit le risque de surapprentissage et converge vers un minimum local éloigné de l'optimum recherché. Ce second effet est notamment utile quand la quantité de données disponibles est un facteur limitant pour l'apprentissage d'un réseau de neurones profonds (Zhang et al., 2014).
En contrepartie, le nombre de métaparamètres à déterminer et le temps nécessaire à l'apprentissage augmentent considérablement.

L'utilisation de paramètres issus d'une couche *bottleneck* obtient de bons résultats dans des domaines de recherche variés. Par exemple, dans le traitement de la parole, (Tan et Mak) utilisent des enchainements de *bottleneck* produits par des réseaux débruitants. Cet enchainement de réseaux construit des modèles de locuteurs robustes pour une reconnaissance efficace appliquée à une tâche où les données disponibles ne permettent pas d'entrainer des réseaux très profonds. (Tian et al., 2015) exploitent le même principe pour proposer une alternative aux i-vecteurs et à l'analyse linéaire discriminante probabiliste (PLDA). En reconnaissance de la parole, cette méthode a permis de diminuer considérablement les "taux d'erreurs mots". Elle est employée aussi pour mélanger

des modèles génératifs et discriminants dans un même modèle (Hinton et al., 2012). Cette combinaison permet d'obtenir des modèles de SLU avec de meilleures capacités de généralisation.

Dans la littérature, cette forme d'apprentissage est aussi employée pour lier des domaines de représentations très différents alors que les relations entre les espaces ne sont pas évidentes. (Chuangsuwanich et al., 2016) et (Fér et al.) proposent des réseaux de neurones qui génèrent des caractéristiques multilingues exploitées par des systèmes de reconnaissance de la parole. (Wu et al.) proposent un outil de synthèse de la parole qui s'appuie sur des *bottleneck* pour créer un espace de représentation unique qui contient des informations à la fois sur les graphèmes d'entrée et les hypothèses de phonèmes à produire. Elle est exploitée aussi sur des tâches de traitement de l'image, notamment dans (Carcenac et Redif, 2015), où les auteurs proposent de découper un réseau de neurones en plusieurs "modules". Chaque module traite un niveau de résolution différent. Les modules de bas niveau sont combinés comme entrée pour les modules de haut niveau afin de former une chaine de traitement capable d'appliquer des transformations complexes sur l'image d'origine.

Dans ce chapitre, un autoencodeur original utilisant un apprentissage en deux temps, de manière à exploiter conjointement les transcriptions manuelles et les transcriptions automatiques, est proposé. Dans un premier temps, deux DAE sont entrainés pour débruiter les représentations propres et corrompues des données. Ensuite, un troisième réseau est entrainé pour créer un espace intermédiaire liant les deux représentations préalablement apprises. La section 8.2 introduit cet autoencodeur et les particularités de son apprentissage. La Section 8.3 présente les expériences réalisées pour évaluer les capacités de débruitage de cet autoencodeur pour la compréhension de la parole. La Section 8.4 détaille les résultats de ces expériences qui sont analysées et commentées Section 8.5.

8.2 Autoencodeur profond supervisé (SDAE)

8.2.1 Intérêt du SDAE

La problématique considérée est toujours de construire une représentation robuste à partir de documents bruités ou corrompus issus du SRAP. Nous proposons un autoencodeur profond supervisé (Supervised Deep Autoencoder, SDAE) pour extraire cette représentation. Cette variation originale des autoencodeurs est capable d'utiliser la supervision offerte par les documents transcrits manuellement (*TRS*) pour le débruitage des documents transcrits automatiquement (*ASR*). Les expériences du Chapitre 6 ont montré qu'une approche classique qui emploie des AE profonds pour apprendre à reconstruire les documents propres échoue. En effet, ils sont incapables de produire des représentations améliorant la classification. Cette faiblesse s'explique par l'écart entre les deux représentations, trop important pour créer une représentation intermédiaire utile. Les AE débruitants homogènes (entrée et sortie de la même source) sont capables de créer des représentations latentes denses et efficaces. De la même façon que le SAE

empile sur ces AE débruitants, il est important que le réseau proposé capitalise cette force. Le SDAE utilise un apprentissage en plusieurs temps pour créer un lien entre les deux espaces propres et bruités.

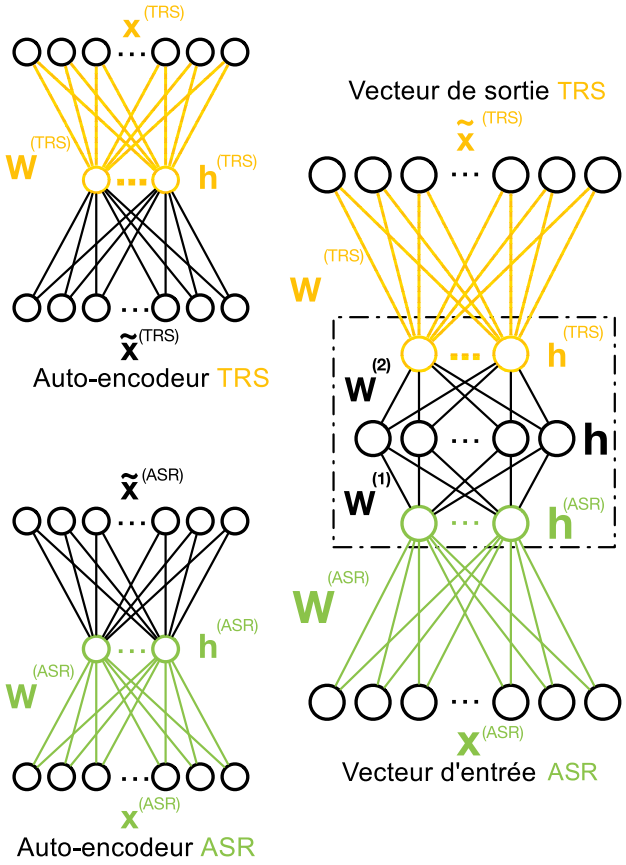


Figure *8.3* – *Architecture de l'autoencodeur profond supervisé. À gauche, les deux réseaux préentrainés et à droite le réseau final.*

L'architecture de ce réseau est présentée dans la Figure 8.3. On peut voir à gauche les deux autoencodeurs simples entrainés indépendamment sur les données issues du SRAP et sur les données manuelles. Les poids des couches cachées $W^{(ASR)}$ et $W^{(TRS)}$ (en vert et jaune respectivement) sont transmises au SDAE complet présenté à droite du schéma. Le cadre en pointillé montre les seules matrices de poids utilisées pour son apprentissage. La méthodologie utilisée pour entrainer le SDAE est détaillée ci-dessous.

8.2.2 Particularités de l'entrainement du SDAE

L'apprentissage du SDAE se déroule en deux phases.

D'abord, deux DAE homogènes classiques sont entrainés. Le premier prend en vecteur d'entrée et de sortie les données bruitées. Le second quant à lui prend des données propres. Ces dernières sont considérées comme la supervision naturelle du SDAE. Ils sont

nommés $AE_{corrompu}$ et AE_{propre} respectivement. Ils vont déterminer une représentation robuste de leurs documents respectifs qui conserve un maximum de variabilités utiles et ignore l'information non pertinente.

Ensuite, le SDAE, composé de trois couches cachées appelées $h_{corrompu}$, h et h_{propre}, est entrainé. Les poids de ses matrices extérieures $W_{corrompu}$ et W_{propre} du SDAE sont initialisés à partir des poids appris par $AE_{corrompu}$ et AE_{propre} et sont fixés lors de l'apprentissage. Seules les matrices à l'intérieur de l'encadré pointillé ($W^{(1)}$ et $W^{(2)}$) sont affectées par la descente de gradient. Ainsi les capacités de débruitage apprises par $AE_{corrompu}$ et AE_{propre} ne sont pas détériorées. La couche *bottleneck* h qui lie l'espace bruité à l'espace propre a besoin d'être particulièrement large. En effet, la mise en place d'une telle architecture n'a de réel intérêt que dans le cas où le lien entre les deux espaces peut être qualifié de complexe. Dans ce contexte, un autoencodeur ne risque pas d'approximer la fonction identité (Baldi, 2012) et la compression de l'information ne fait pas partie des objectifs. Il n'est donc pas gênant d'avoir une couche intermédiaire plus grande que celle d'entrée et de sortie. La taille idéale de cette couche peut être déterminée, comme d'autres métaparamètres, à l'aide d'un corpus de développement.

8.3 Expériences

8.3.1 Protocole expérimental

Cette expérience est réalisée sur la tâche TID présentée dans le Chapitre 4. Les représentations TF.IDF.GINI présentées dans ce même chapitre sont compressées et débruitées par un autoencodeur. Ensuite un MLP classifie les documents débruités parmi 8 catégories.

Dans cette expérience, plusieurs autoencodeurs sont comparés sur la tâche TID. Ils utilisent tous une stratégie d'apprentissage différente pour mettre en évidence chacune des particularités du SDAE. Pour faciliter la compréhension, les deux autoencodeurs simples du SDAE sont entrainés séparément et ajoutés à la comparaison. Ils servent de préentrainement à plusieurs des AE profonds.

L'ensemble des réseaux utilisent les mêmes paramètres que dans le Chapitre 6. C'est-à-dire qu'ils utilisent : des fonctions d'activation de type «tanh», un bruit de masquage qui affecte aléatoirement 50% des neurones de la couche d'entrée pour produire de meilleures représentations intermédiaires, mais aussi pour réduire le surapprentissage. Les poids sont adaptés par descente de gradient «Adam». Le nombre d'itérations est déterminé par arrêt anticipé lorsque la fonction de cout (L_{MSE}) ne réduit plus.

8.3.2 les DAE simples

L'$AE_{corrompu}$ utilise les documents issus du SRAP x_{ASR}. Il est appelé AE_{ASR} dans ce contexte. l'AE_{TRS} est entrainé pour débruiter les documents transcrits manuellement

x_{TRS}. Il est donc nommé AE_{TRS}. Ces deux AE ont la même architecture que celle définie dans le Chapitre 6.
Ces deux AE consomment peu de ressources de calcul et sont indépendants, ils peuvent aisément être entrainés en parallèle.

8.3.3 Le SDAE

Le SDAE est composé de trois couches cachées. Les couches h_{ASR} et h_{TRS} sont initialisées avec les poids d'encodage des réseaux simples (AE_{ASR}, AE_{TRS}). Elles sont donc de taille 50 aussi. La couche h centrale qui contient la projection finale est de taille 300, comparable au SAE présenté dans le Chapitre 6. Ce réseau utilise en vecteur d'entrée les documents x_{ASR} et en sortie les documents x_{TRS}.

8.3.4 Le FSDAE

L'autoencodeur supervisé profond réadapté (Finetuned Supervised Deep Autoencodeur, FSDAE) est un réseau de neurones profonds dont l'architecture est identique au SDAE. L'unique particularité du FSDAE est que lors de l'apprentissage final, les matrices de poids W_{ASR} et W_{TRS} sont adaptées pour la reconstruction des documents TRS. Nous faisons l'hypothèse ici que l'adaptation des poids dégrade les performances du système. En effet, ce mécanisme peut permettre au classifieur de trouver une meilleure solution grâce à l'information apprise par les AE simples. Mais il prend le risque que la répercussion des erreurs de reconstruction des documents propres soit toujours trop complexe pour le réseau, et le fasse retomber dans une situation similaire au DDAE en impactant la matrice W_{ASR} qui permet déjà d'extraire une représentation robuste des transcriptions automatiques.

8.4 Résultats

La Figure 8.4 montre les précisions en classification obtenues par le SDAE proposé, FSDAE et les autoencodeurs profonds présentés dans le Chapitre 6. La qualité des réseaux est évaluée sur la tâche TID en évaluant toutes les représentations intermédiaires produites par chacun des autoencodeurs. Les poids des couches cachées $h^{(\mathrm{ASR})}$ et $h^{(\mathrm{TRS})}$ du SDAE sont préentrainés puis fixés lors de l'apprentissage. Elles sont donc identiques aux couches intermédiaires de AE_{ASR} et AE_{TRS} respectivement, seule la couche h du SDAE est informative.

Le SDAE obtient une précision de $83,2\%$. C'est le meilleur système utilisant les documents ASR. Ces résultats sont inférieurs de 1% absolu comparativement aux résultats obtenus par AE_{TRS} ce qui réduit encore l'écart de qualité d'information contenue entre les représentations bruitées et les représentations propres. Les performances se rapprochent fortement de celles obtenues en utilisant les documents TRS ($84,1\%$ et $85,3\%$) qui repré-

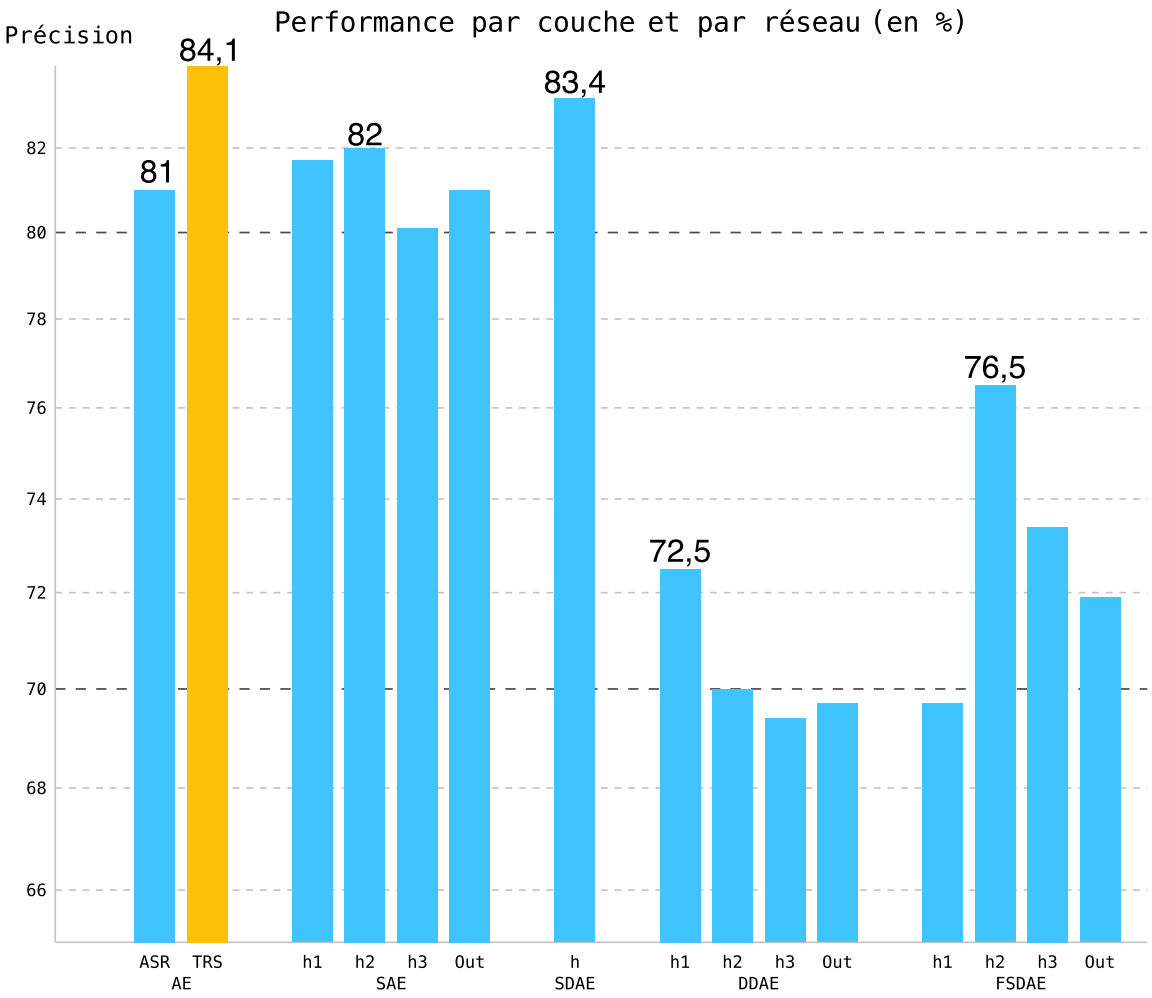


Figure ***8.4*** – *Performances par couche cachée et par réseau sur la tâche TID. Les caractéristiques AE_{TRS} (en jaune) sont les seules apprises et évaluées uniquement avec les documents transcrits manuellement.*

sentent un cadre sans erreur de transcription. Ils représentent donc l'objectif à réaliser pour les systèmes qui exploitent les transcriptions automatiques.

Comme attendu, les résultats du FSDAE montrent que ne pas fixer les matrices $W^{(\text{ASR})}$ et $W^{(\text{TRS})}$ lors de l'apprentissage final dégrade la capacité de modélisation du réseau de neurones. En effet, le FSDAE ne réalise qu'un score de $76,5\%$ en utilisant la projection produite par la couche cachée centrale h_2. Les couches cachées h_1 et h_3 qui utilisent le préentrainement obtiennent malgré cela une précision de $69,7\%$ et $73,4\%$. Les représentations de ces deux couches sont largement dégradées lors du réapprentissage.

Pour rappel, les performances du DDAE qui n'utilisent pas de méthode de préentrainement oscillent entre $69,4$ et $72,5\%$. Ces deux derniers systèmes sont en dessous des performances de départ ($77,1\%$).

Le SAE avec ses $82,0\%$ de précision est le deuxième meilleur système, seulement $1,2$

point inférieur au SDAE.

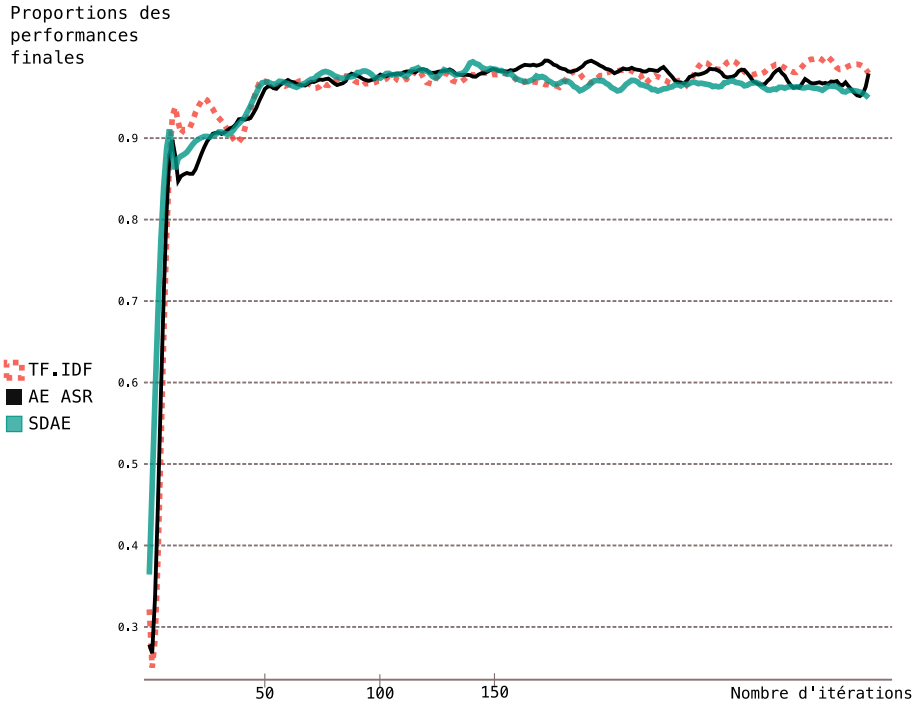


Figure *8.5* – *Proportions* $\frac{Performance_i}{Performance_{Max}}$ *en fonction des itérations pour les MLP utilisant les caractéristiques des Autoencodeurs sans early stopping*

.

La Figure 8.5 montre le nombre d'itérations nécessaires pour obtenir les résultats optimaux avec les caractéristiques de base (TF.IDF) et celles générées par certains autoencodeurs. Les trois caractéristiques réagissent de façon similaire. La vitesse de convergence est forte jusqu'à environ 50 itérations. Ensuite les MLP apprennent peu et les performances stagnent ou se dégradent.

L'histogramme présenté Figure 8.6 représente les temps nécessaires pour réaliser l'apprentissage de chacun des réseaux de neurones sur le matériel présenté Chapitre 6. Le SAE a toujours le temps de calcul le plus long, 50 minutes. En effet, il nécessite trois apprentissages intermédiaires puis un apprentissage global. Ces apprentissages doivent être réalisés séquentiellement puisque chaque couche cachée dépend de la précédente. Le SDAE est second : il apparait deux fois dans le tableau, une fois en entrainant les réseaux intermédiaires en séquentiel (35 minutes) et une fois en parallèle. En entrainant les réseaux en parallèle, le SDAE converge en 25 minutes, soit deux fois moins que le SAE et l'équivalent d'un réseau débruitant standard (DDAE). Les autoencodeurs simples sont évidemment les plus rapides avec un seul entrainement de 10 minutes. Similairement au Chapitre 6 quand les temps d'apprentissages sont un critère important, les AE simples sont envisageables, car leur apprentissage est rapide pour des performances acceptables.

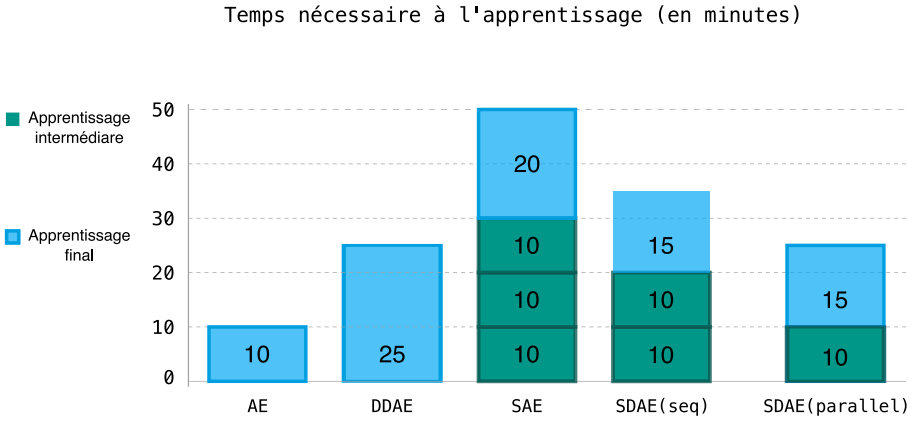


Figure *8.6* – *Temps de calcul nécessaire aux apprentissages des différents Autoencodeurs profonds.*

8.5 Discussion

Données d'entrée ASR

Méthode employée	**Vecteur d'entrée**	**Précision obtenue**
DDAE	$h^{(1)}$	72,5
DAE	h	74,3
FSDAE	h	76,5
TF.IDF	–	77,1
AE_{ASR}	h	81
SAE	$h^{(2)}$	82,0
SDAE	h	**83,2**

Données d'entrée TRS

Méthode employée	**Vecteur d'entrée**	**Précision obtenue**
AE (TRS)	h	84,1
SAE (TRS)	$h^{(3)}$	85,3

Table *8.1* – *Récapitulatif des meilleures performances (%) observées pour chaque réseau de neurones artificiels sur la tâche TID.*

Le Tableau 8.1 compare les meilleures précisions réalisées sur la tâche TID pour chacune des architectures proposées. Il est intéressant de souligner que le AE_{ASR} et le SAE ont des performances globalement bonnes. C'est d'autant plus le cas que le premier est particulièrement simple et tous deux sont entrainés de façon entièrement non supervisée. Ces résultats sont obtenus grâce aux capacités de filtrage et de sélection de l'information des autoencodeurs débruitants qui ont permis de nettoyer une bonne partie du bruit présent dans les documents transcrits automatiquement. Néanmoins, aucune de ces méthodes n'est capable d'obtenir des résultats semblables à l'utilisation des transcriptions manuelles. Les résultats des SAE introduits dans le Chapitre 6 apportent un gain relatif de 2,7% sur les documents TRS et de 6,4% sur les documents transcrits automatiquement. Ces différences indiquent qu'une partie du bruit introduit par la transcription automatique est capturée, et avec celui-ci un bruit intrinsèque aux documents qui est également présent dans les transcriptions manuelles est retiré.

La précision obtenue par le SDAE atteint 83,2%, seulement 0,9% en dessous de la représentation propre utilisée comme objectif (AE_{TRS} avec 84,1%). L'introduction de la supervision dans le processus de débruitage permet au réseau de modéliser et capturer une forme de bruit non traitée par les méthodes non supervisées et améliore ainsi la qualité de la représentation produite. Cette amélioration permet au système réel d'être compétitif avec les résultats obtenus à partir de la transcription oracle.

Enfin, les résultats faibles des DDAE et FSDAE sont eux aussi reportés dans le Tableau 8.1. Ils montrent que les bruits cumulés présents dans les documents transcrits automatiquement sont difficiles à supprimer même pour un réseau profond qui traite l'information sur différents niveaux d'abstraction.

La force du SDAE vient des couches extérieures qui capitalisent sur les capacités de débruitage de AE_{ASR} et AE_{TRS}. En effet, ces deux réseaux produisent deux représentations latentes d'un même document dont une partie du bruit de plus bas niveau est supprimée. Même si les représentations visibles sont éloignées, les représentations latentes ne doivent contenir que l'essence du document. Elles doivent donc avoir plus de points communs. Cela permet aux SDAE de créer un lien entre les deux représentations et de supprimer une partie supplémentaire du bruit introduit par le SRAP. La couche cachée centrale h est le lien entre ces deux représentations. Elle est plus abstraite et moins bruitée que h_{ASR}.

8.6 Conclusion

Ce chapitre propose une nouvelle représentation latente pour la compréhension de documents parlés. Elle utilise les caractéristiques issues d'une couche *Bottleneck* d'un autoencodeur profond. Ce réseau de neurones est initialisé par deux DAE appris sur des données d'origine différentes. Il exploite l'information provenant de documents corrompus de facto et de documents propres.

Cette représentation a pour effet de réduire l'impact des erreurs de transcription automatique sur la compréhension de documents parlés. L'impact des erreurs du SRAP est réduit en entrainant une projection intermédiaire non linéaire qui lie les documents produits par le SRAP à ceux transcrits manuellement.

Des représentations latentes des deux types de documents (respectivement ASR et TRS) sont apprises préalablement et ne sont plus modifiées lors de l'entrainement de l'espace intermédiaire. Cette décision est appuyée par nos expériences qui montrent un gain de 6% comparé au système de référence et jusqu'à 10% comparé aux méthodes qui adaptent l'ensemble des paramètres du réseau lors de l'apprentissage final. Ces résultats montrent que la représentation d'un même document apprise par les AE_{ASR} et AE_{TRS} contiennent des informations latentes d'ordre sémantique qui sont plus similaires que les représentations apparentes.

Dans la situation où les informations de supervision ne sont pas disponibles et que les temps de calcul ne sont pas une contrainte, alors l'utilisation d'un SAE est envisageable.

En effet, il est capable de produire de bonnes représentations latentes, avec un gain de plus de 4% par rapport au système de référence. Le SAE est seulement $1,2\%$ en dessous du SDAE, sans supervision, au prix d'un temps d'apprentissage deux fois plus important.

Nous envisageons trois possibilités intéressantes d'évolution pour ces travaux.

La première serait d'évaluer ce même système en passant les données à l'échelle. En effet, la taille du corpus de la tâche TID est limitée. Sur un corpus plus gros, l'écart entre les méthodes supervisées et non supervisées devrait se creuser. Un corpus annoté thématiquement, transcrit manuellement et de taille suffisamment importante pour faire une différence est rare.

La seconde ouverture que nous proposons concerne la représentation d'origine TF.IDF. Par exemple, utiliser des modèles graphiques probabilistes (comme la LDA (Blei et al., 2003), le modèle "Author-Topic" (Rosen-Zvi et al., 2004)), ou des réseaux de neurones capables de modéliser la structure des documents comme les réseaux LSTM (Graves, 2012) ou GRU (Chung et al., 2015) serait intéressant. En effet, la séquentialité des mots est un indice important pour caractériser les erreurs introduites par un SRAP.
Utiliser d'autres représentations textuelles permettrait aussi d'étudier le lien entre : les performances d'un réseau de neurones, le niveau d'abstraction (et de complexité) des données d'entrée et la quantité de données d'apprentissage disponibles. Même si ce n'est pas directement lié à la compréhension de la parole, c'est une piste d'étude intéressante.

La troisième possibilité serait d'introduire la supervision plus tôt dans la création des représentations des documents. Cette méthode est approfondie dans le Chapitre 9.

Chapitre 9

Utilisation de caractéristiques supervisées dans un autoencodeur hétérogène.

Sommaire

9.1 Introduction 131
9.2 Autoencodeur débruitant avec caractéristiques spécifiques à la tâche (TDAE) 133
9.3 Expérimentations 135
9.4 Résultats expérimentaux 137
9.5 Conclusion 138

9.1 Introduction

Dans le Chapitre 6, nous avons vu qu'un DAE profond entrainé en considérant directement les données issues d'un SRAP comme bruitées et les données transcrites manuellement comme références ne parvient pas à construire une représentation efficace. Dans le chapitre précédent, nous avons proposé une représentation robuste aux erreurs de transcription automatique en exploitant la supervision apportée par les documents transcrits manuellement. Pour cela, un autoencodeur débruitant particulier a été introduit, le SDAE. Celui-ci crée un espace de projection intermédiaire en apprenant à reconstruire un document transcrit manuellement à partir des transcriptions automatiques. L'entrainement du SDAE apprend à lier deux représentations latentes apprises de manière non supervisée. La première étant obtenue à l'aide de transcriptions automatiques et la seconde grâce à celles manuelles.

Les résultats expérimentaux présentés dans la Section 8.4 ont montré qu'il est plus facile de retrouver l'information latente d'un document propre que de le reconstruire sous sa forme «visible». La construction des représentations latentes de manière non

supervisée permet de construire un vecteur qui contient toute l'information nécessaire pour reconstruire le document d'origine. Cette particularité favorise la généralisation du système. En contrepartie, l'excès de généralisation sur l'information favorise le risque que le système conserve plus d'informations que nécessaire pour la tâche et donc plus de bruits aussi. Cela rend la représentation plus difficile à nettoyer.

Nous proposons dans ce chapitre d'introduire de la supervision plus tôt dans le processus de débruitage des documents transcrits automatiquement. Celle-ci est employée pour l'extraction de caractéristiques *bottleneck* comme présentée dans la Figure 9.1.

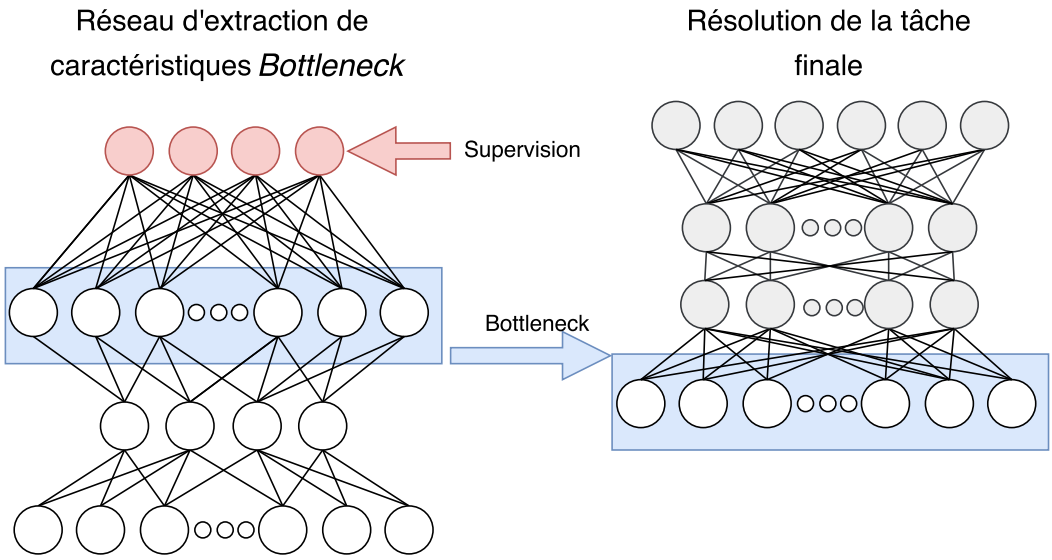


Figure *9.1* – *Extraction de caractéristiques bottleneck dans un réseau supervisé.*

Cette information de supervision permet d'obtenir des représentations propres à la tâche, ne modélisant que l'information utile à la classification. Ces représentations intermédiaires sont plus proches, ce qui met en évidence la présence de bruit. Celui-ci peut donc plus facilement être modélisé par un autoencodeur débruitant hétérogène. La couche cachée de cet autoencodeur apprend une représentation intermédiaire débruitée plus facile à classifier et plus robuste au bruit.

Dans le Chapitre 8 nous avons présenté et utilisé des paramètres produits par une couche *bottleneck* d'un réseau de neurones artificiels entrainés dans un contexte **non supervisé**. Les caractéristiques issues d'une couche *bottleneck* d'un réseau **supervisé** ont aussi été employées intensivement, en particulier pour le traitement de la parole. Elles ont été exploitées, par exemple, pour créer des descripteurs acoustiques robustes pour la transcription automatique (Grézl et al., 2007). On les retrouve aussi dans (Doddipatla et al., 2014), où des caractéristiques *bottleneck* ont été utilisées pour réaliser une adaptation locuteur du SRAP. De la même manière, un empilement de couches *bottleneck* a permis de produire un ensemble de caractéristiques efficace pour la reconnaissance du langage (Matejka et al., 2014). D'autres travaux (Ren et al., 2016) introduisent une comparaison des descripteurs obtenus par un perceptron multicouche (MLP) et par DAE ainsi que leurs différentes combinaisons. Ces analyses ont montré que, prises indépen-

damment, les caractéristiques provenant du MLP,dont l'information décorrélée de la tâche est supprimée, sont les plus efficaces des deux pour la déréverberation de la parole distante.

Dans un contexte multimodal, des descripteurs bimodaux ont été introduits (Takashima et al., 2016). Ils combinent des couches *bottlenecks* issues de deux réseaux supervisés pour fusionner l'information acoustique et l'information visuelle afin d'améliorer les performances de transcription automatique. En synthèse vocale, des combinaisons de couches *bottleneck* construites avec des MLP ont amélioré les performances de systèmes automatiques (Wu et al.). D'autres travaux (Parviainen, 2010) ont montré que le MLP est un moyen efficace pour réduire la dimension des données d'entrée afin d'améliorer des modèles de régression.

Pour introduire l'information de supervision dans le débruitage de documents transcrits automatiquement nous proposons une application originale des autoencodeurs, inspirée du SDAE présenté dans le Chapitre 8. Celle-ci, appelée autoencodeur débruitant avec des caractéristiques spécifiques à la tâche (Task specific denoising autoencoder, TDAE), exploite les capacités de modélisation de deux MLP : un premier pour créer un espace latent de documents propres, et un second pour créer espace latent de documents bruités. Ensuite, un troisième réseau de neurones est entrainé pour lier les deux espaces latents et produire une représentation débruitée des documents ASR.

La Section 9.2 introduit cette architecture et son processus d'apprentissage. La Section 9.3 détaille les expériences réalisées pour évaluer les capacités de débruitage de cet autoencodeur pour la classification thématique de documents bruités. La Section 9.4 présente les résultats expérimentaux et leurs analyses. Enfin, la Section 9.5 conclut ce chapitre en faisant le bilan de ces travaux.

9.2 Autoencodeur débruitant avec caractéristiques spécifiques à la tâche (TDAE)

La problématique considérée est de construire une représentation robuste à partir de documents bruités ou corrompus produits par un SRAP. Nous proposons, dans ce chapitre, une architecture d'autoencodeur particulière (TDAE) pour apprendre cette représentation. D'abord, nous présenterons les spécificités de cette architecture, ensuite nous aborderons les particularités liées à son apprentissage.

9.2.1 Spécificités du TDAE

Cette architecture est capable de combiner lors de l'apprentissage deux formes de supervision. La première supervision est introduite lors de l'apprentissage de représentation latente homogène par un MLP. Cette information permet la sélection des informations conservées dans l'espace latent pour optimiser la classification. La seconde supervision est apportée par les documents transcrits manuellement (*TRS*) pour le débruitage des

documents transcrits automatiquement (*ASR*). Les expériences présentées dans le Chapitre 8 ont montré que les caractéristiques latentes d'un même document appris dans des environnements différents sont plus proches et donc plus faciles à débruiter que les formes apparentes de surface. Le TDAE exploite des représentations latentes apprises par des MLP spécifiquement entrainés pour résoudre cette tâche de classification et propose un apprentissage en deux temps où chaque étape a un objectif différent. Contrairement au SDAE où trois autoencodeurs sont entrainés dans l'objectif de débruiter les représentations, nous proposons maintenant d'avoir une étape de sélection de la variabilité utile à la classification et ensuite un débruitage restreint à l'information pertinente. L'autoencodeur profond final capitalise ainsi sur les capacités de modélisation supervisées du MLP.

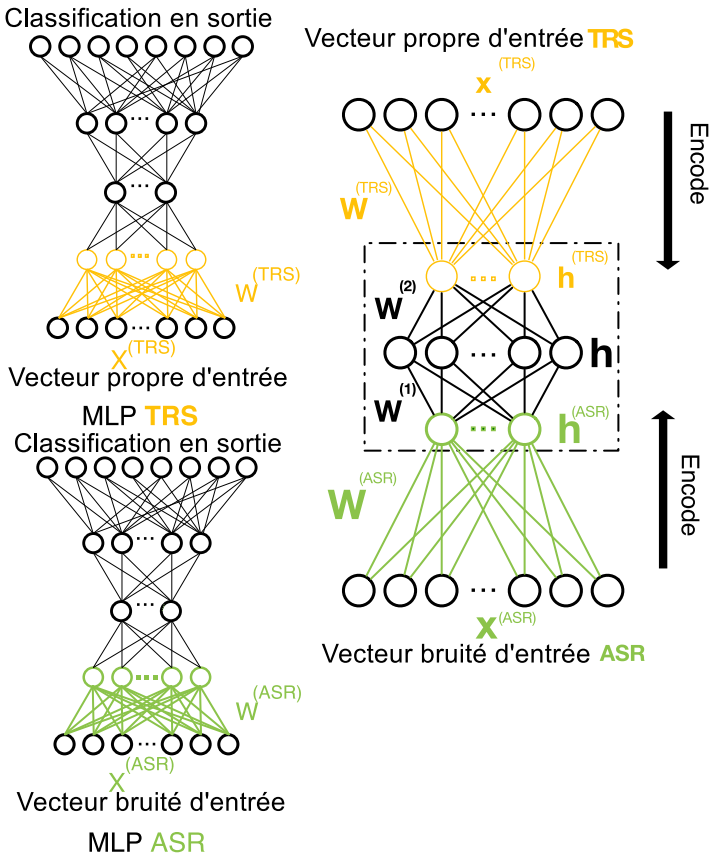


Figure *9.2* – *Architecture de l'autoencodeur débruitant avec des caractéristiques spécifiques à la tâche. À gauche, les deux réseaux préentrainés et à droite le réseau final.*

L'architecture de ce réseau est présentée dans la Figure 9.2. On peut voir à gauche les deux MLP simples entrainés indépendamment sur les données issues du SRAP et sur les données manuelles pour extraire l'information utile à la classification. Les poids des couches cachées $W^{(ASR)}$ et $W^{(TRS)}$ (en vert et jaune respectivement) sont transmises au TDAE complet présenté à droite du schéma. Le cadre en pointillé montre les deux matrices de poids $W^{(1)}$ et $W^{(2)}$ mises à jour lors de son apprentissage. La méthode utilisée pour entrainer le TDAE est détaillée ci-dessous.

9.2.2 Méthode d'entrainement du TDAE

L'apprentissage du TDAE est réalisé en deux étapes.
D'abord, deux MLP homogènes sont entrainés sur la tâche finale de classification. Le premier prend comme vecteur d'entrée les données bruitées, le second quant à lui prend des données propres. Ils sont nommés MLP_{ASR} et MLP_{TRS} respectivement. De même que pour le SDAE, les documents propres sont considérés comme la supervision du débruitage. Ils vont déterminer une représentation réduite de leurs documents respectifs en conservant l'information utile pour résoudre la tâche dans chacun des environnements. Ensuite, le TDAE, composé d'au moins trois couches cachées, appelées h^i_{ASR}, h et h^i_{TRS}, est entrainé.
De la même manière que pour la construction d'un autoencodeur, pour augmenter la capacité de modélisation du réseau, l'optimisation du nombre de couches cachées avant et après la couche *bottleneck* h peut être réalisée sur un corpus de développement. Les poids des matrices extérieures $W^{(ASR)}$ et $W^{(TRS)}$ du TDAE sont initialisés à partir des poids appris par MLP_{ASR} et MLP_{TRS}. Ces matrices de poids seront fixées lors de l'apprentissage. Seules les matrices à l'intérieur de l'encadré pointillé ($W^{(1)}$ et $W^{(2)}$) sont affectées par la descente de gradient. Cette configuration permet de séparer les fonctions dans les différentes couches du réseau. Les couches extérieures sont utilisées pour construire des représentations latentes qui contiennent l'information utile du document. Les couches centrales ont pour rôle de débruiter ces représentations qui sont potentiellement plus similaires que les formes de surfaces initiales. L'objectif de la couche *bottleneck* h n'est pas de compresser l'information, mais d'apprendre une représentation robuste liant les deux espaces provenant d'environnements différents.

9.3 Expérimentations

De manière à être comparable aux résultats des chapitres précédents, le TDAE est évalué sur la tâche TID présentée dans le Chapitre 4. Sur cette tâche plusieurs systèmes seront comparés : Le TDAE avec les deux MLP pour le préentrainement, une version du TDAE qui n'utilise pas les documents propres (MLP-AE), le SDAE et une analyse en composantes principales (Principale Component Analysis, PCA). En effet, dans le Chapitre 7 la méthode LTS qui utilise une analyse en composantes principales obtient, dans certains cas, de meilleures performances que l'autoencodeur pour la fusion et la sélection d'informations.

Les couches cachées des réseaux de neurones utilisent une fonction d'activation «tanh». Leurs poids sont adaptés par descente de gradient «Adam». Le nombre d'itérations pour l'apprentissage est déterminé par arrêt anticipé. Les autoencodeurs utilisent une fonction de cout L_{MSE} telle que définie dans le Chapitre 4.

Les sections suivantes détaillent les paramètres et choix d'implémentations des différents réseaux entrainés. D'abord les MLP puis le TDAE et enfin le MLP-AE sont présentés.

9.3.1 Perceptrons multicouches homogènes

Les MLP homogènes (MLP_{ASR}, MLP_{TRS}) sont composés de trois couches cachées de taille 256, 128 et 256. Les tailles des différentes couches cachées sont déterminées en optimisant les résultats du TDAE sur le corpus de développement. Une régularisation à base de *Dropout* (c.f. Chapitre 3) est employée. Elle permet de limiter le surapprentissage des MLP et d'introduire un bruit additif comparable à celui des AE. Ces deux MLP sont indépendants et peuvent être entrainés en parallèle. Les couches cachées produites par MLP_{ASR} sont utilisées comme *bottleneck* et évaluées aussi à des fins d'analyse.

9.3.2 TDAE

Les couches h^i_{ASR} et h^i_{TRS} du TDAE sont initialisées avec les poids des MLP homogènes. Les poids de ces couches sont fixés à l'apprentissage. La couche h centrale qui contient la projection finale est de taille 300 comparable aux autres architectures. Ce réseau utilise en vecteur d'entrée les documents x_{ASR} et en sortie les documents x_{TRS}.

Une seconde version du TDAE est évaluée aussi dans ces expériences. Cette variante propose de remplacer les fonctions d'activation des couches centrales du TDAE par une fonction relu. Les réseaux de neurones qui utilisent cette fonction d'activation tendent à avoir de meilleures performances (c.f. Chapite 3). La fonction tanh sélectionnée dans les expériences préliminaires a été choisie, car l'utilisation de la fonction relu résulte en une portion importante de neurones morts (dont l'état est à 0 systématiquement). Avoir des neurones morts dans un réseau implique une diminution drastique des capacités de modélisation de celui-ci. Dans la nouvelle configuration avec un apprentissage en deux temps, le phénomène de neurones morts est réduit. Nous proposons donc d'évaluer aussi un TDAE dont les couches cachées centrales utilisent cette fonction d'activation. Pour différencier les deux réseaux, nous appellerons ces deux réseaux TDAE(tanh) et TDAE(relu).

9.3.3 MLP-AE

De manière à évaluer l'apport des transcriptions manuelles dans le processus de débruitage du TDAE, une version de ce même réseau est entrainée sans cette supervision. Le processus d'apprentissage est identique au TDAE à l'exception près que l'erreur de ce réseau est calculée sur les données bruitées au lieu des données propres. Cette configuration (MLP-AE) revient à mettre en cascade deux MLP et un autoencodeur en n'utilisant que des vues bruitées des documents.

9.3.4 Analyse en composantes principales (PCA)

Le nombre de composants principaux pour projeter les documents est sélectionné à postériori en fonction des performances en classification sur le corpus de développement.

Les résultats de ces systèmes en classification sur le corpus de la tâche TID sont aussi comparés aux différents modèles présents dans le Chapitre 8.

9.4 Résultats expérimentaux

Système	Vecteur d'entrée	Vecteur de sortie	Couche utilisée	Précision (%)
SDAE	x_{ASR}/W_{ASR}	x_{TRS}/W_{TRS}	h	83,2
PCA	ASR	-	-	82,5
MLP_{ASR}	ASR	Y'	h^1	70,4
	ASR	Y'	h^2	71,3
	ASR	Y'	h^3	71,3
MLP-AE	ASR	ASR	h^1	80.4
	ASR	ASR	h^2	81.3
	ASR	ASR	h^3	80.4
TDAE (tanh)	x_{ASR}/W_{ASR}	x_{TRS}/W_{TRS}	h^i_{ASR}	82,3
	x_{ASR}/W_{ASR}	x_{TRS}/W_{TRS}	h	84,1
	x_{ASR}/W_{ASR}	x_{TRS}/W_{TRS}	h^i_{TRS}	83,2
TDAE (relu)	x_{ASR}/W_{ASR}	x_{TRS}/W_{TRS}	h^i_{ASR}	83,2
	x_{ASR}/W_{ASR}	x_{TRS}/W_{TRS}	h	**85,2**
	x_{ASR}/W_{ASR}	x_{TRS}/W_{TRS}	h^i_{TRS}	83,4

Table *9.1* – *Précisions des différents systèmes présentés sur la tâche de classification thématique de documents parlés TID. Dans ce tableau Y' correspond à l'hypothèse de classification.*

Les résultats de ces expérimentations sont présentés dans le Tableau 9.1. On peut remarquer dans ce tableau que les meilleures performances sont obtenues par les deux configurations du TDAE. La variation avec relu surpasse la version originale avec $85,2\%$ et $84,1\%$ respectivement.

Avec les deux variations du TDAE, les meilleures performances sont obtenues en utilisant les projections produites dans la couche centrale (h^2) des MLP_{ASR} et MLP_{TRS}.

Les résultats du SDAE sont les troisième meilleurs avec $83,2\%$, soit une différence relative de presque $1,1\%$ par rapport au TDAE qui utilise des paramètres comparables (tanh).

Le MLP-AE obtient une précision de $81,3\%$ ce qui correspond à une perte absolue de $2,8\%$ comparativement au TDAE. Les documents transcrits manuellement sont donc utiles aux processus de débruitage.

Les systèmes utilisant les couches intermédiaires de MLP_{ASR} comme caractéristiques obtient le plus mauvais score de ce jeu d'expérience avec aux mieux $71,3\%$ de réussite.

C'est une perte de 6,7 points par rapport à un système qui réalise la classification directement avec MLP_{ASR} et obtient le score de 78%.

Enfin, les descripteurs projetés dans les composantes principales d'une PCA obtiennent le score de 82,5%. Ce score est obtenu en utilisant les 26 premières composantes principales. Il s'agit du meilleur résultat obtenu sans supervision parmi les systèmes présentés dans le tableau.

La Figure 9.3 compare le temps d'apprentissage nécessaire pour le TDAE avec ceux du SDAE. On peut voir dans ce schéma que le TDAE peut être entrainé de deux manières, soit en parallèle, ce qui signifie que les deux MLP (MLP_{ASR} et MLP_{TRS}) sont entrainés en simultané, soit en séquentiel en entrainant les réseaux les uns après les autres. Dans les deux conditions, les temps d'apprentissage sont relativement similaires. Les gains apportés par le TDAE ne s'obtiennent pas au prix d'une augmentation du temps d'apprentissage, mais consomment plus de ressources mémoires.

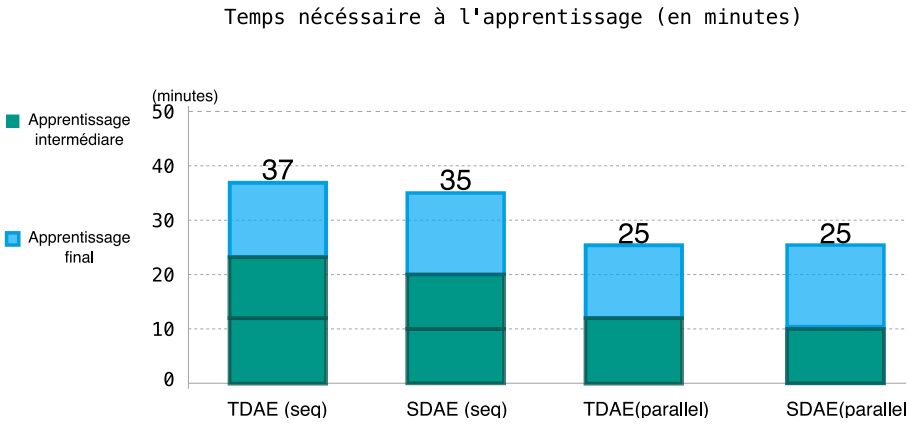


Figure *9.3* – *Temps de calcul nécessaire aux apprentissages du TDAE et du SDAE.*

9.5 Conclusion

Dans ce chapitre nous avons proposé d'introduire de la supervision via des références de classification combinée à la supervision de débruitage par l'intermédiaire des documents transcripts manuellement lors de la construction de représentations robustes. Ces représentations sont exploitées ensuite pour la classification thématique de documents parlés. Elles sont construites par un autoencodeur profond dont les poids des couches extérieures sont préentrainés et figés lors de l'apprentissage final. Les préentrainements sont réalisés à l'aide des couches *bottleneck* de deux MLP, le premier est entrainé sur les documents propres issus de transcriptions manuelles et le second sur les documents bruités produit par le SRAP. Ces réseaux de neurones ont pour objectif de sélectionner l'information pertinente pour la classification. Ensuite le TDAE complet apprend à reconstruire une version propre des documents à partir de leurs versions bruitées. La couche cachée du TDAE apprend une représentation intermédiaire qui lie l'espace bruité à l'espace propre.

Les représentations construites par le TDAE sont plus efficaces pour la classification de documents parlés. Un gain relatif atteignant $10,5\%$ est mesurable en utilisant ces représentations par rapport à celles des documents initiaux. L'utilisation de représentations intermédiaires, lorsqu'elles sont construites avec la supervision de la tâche par les couches *bottleneck* d'un MLP, rend le débruitage plus simple et permet une meilleure classification qu'un apprentissage non supervisé d'autoencodeur. Avec ces descripteurs, l'exploitation des documents bruités ne provoque plus qu'une perte relative de $0,11\%$ comparés à des représentations abstraites propres nettoyées. Les erreurs induites par le SRAP sont en majorité gommées par notre système d'extraction de caractéristiques débruitées.

Nous voyons plusieurs perspectives intéressantes à ces travaux, dont deux sont communes au SDAE.

La première ouverture, commune avec le SDAE, que nous proposons concerne la représentation des données d'origine. Cette méthode ne pose pas de conditions à priori sur le type de représentations utilisé. De plus, nous avons montré dans le Chapitre 7 que l'utilisation d'autoencodeurs sur des représentations issues de modèles *Author-Topic* a un intérêt. L'utilisation de ces méthodes de débruitage sur des représentations de plus haut niveau pourrait avoir un effet positif sur les résultats globaux de la tâche.

La seconde ouverture repose sur des représentations qui modélisent la structure des documents construites par des LSTM (Graves, 2012) ou des GRU (Chung et al., 2015). Appliquées aux sorties de SRAP, elles pourraient modéliser aussi les erreurs de transcriptions et de reporter les erreurs du SRAP sur la tâche de compréhension du langage. L'intégration du débruitage lors de l'entrainement de ces structures permettrait de détecter le bruit, de prévenir ses effets plus facilement, et par conséquent de construire des représentations plus robustes et plus informatives.

La troisième ouverture, commune avec le SDAE, serait de travailler avec un corpus de données plus volumineux. Un corpus sensiblement plus important justifierait l'emploi de méthodes plus profondes. De plus l'utilisation des couches *bottleneck* est moins efficace lorsque beaucoup de données d'apprentissage sont disponibles. Valider (ou non) le comportement mis en évidence sur la tâche TID serait utile. Mais un corpus annoté thématiquement et transcrit manuellement suffisamment volumineux est difficile à trouver ou collecter.

Nous avons utilisé deux types de préentrainement dans nos travaux. L'application des méthodes d'apprentissage joint (Caruana, 1998) (Collobert et Weston, 2008) est une piste intéressante qui permettrait d'améliorer les systèmes de débruitage que nous avons présenté et d'en simplifier le processus, puisqu'il n'y aurait plus qu'un seul apprentissage, tout en conservant ses propriétés. Cette méthode est à étudier en particulier dans les cas où les données d'apprentissage sont suffisamment conséquentes pour réduire le gain observé des couches *bottleneck* comparées à un réseau plus profond.

Enfin les architectures récentes GAN et VAE proposent des outils puissants pour apprendre des modèles génératifs. Ces modèles pourraient être une piste d'étude intéressante pour fournir des alternatives aux autoencodeurs débruitants. Ces méthodes

pourraient reconstruire des documents lisibles, au lieu de représentations latentes, ce qui peut faciliter l'interprétation des choix du système automatique et ouvrir de nouvelles applications.

Nous avons présenté une nouvelle architecture qui utilise deux visions des documents pour apprendre à créer des représentations aussi efficaces que celles construites à partir de documents propres. Nous avons aussi proposé plusieurs pistes d'évolution à ces travaux. Le chapitre suivant va conclure sur l'ensemble des travaux présentés dans ce manuscrit, des perspectives plus globales seront ensuite introduites et discutées.

Chapitre 10

Bilan et significativité

Dans ce chapitre, nous proposons un bilan plus global des résultats obtenus. En effet, pour pouvoir interpréter les résultats avec précision sur la tâche TID, il est important de les analyser en tenant compte des résultats sur la même tâche de l'ensemble des expériences. Dans un premier temps, nous comparons la matrice de confusion du meilleur système homogène avec la matrice du meilleur système hétérogène et nous déterminons la significativité des différents résultats obtenus. Dans un second temps, les meilleurs scores réalisés par chacun des systèmes introduits dans les chapitres 6,et 8 et 9 sont résumés et analysés via la Figure 10.3.

10.1 Confusion et Significativité

Les Figures 10.2 et 10.1 nous présentent les confusions réalisées par les classifieurs automatiques. On peut remarquer logiquement que la proportion de confusions diminue, mais que certaines catégories d'erreurs n'évoluent que très peu. On peut voir notamment que les catégorie des *Horaires* et des *Offres spéciales* sont plus difficiles à détecter correctement alors que *Carte de transport* et *État du trafic* ont tendance à être choisis par erreur. Ces confusions peuvent s'expliquer par la proximité sémantique potentielle de ces catégories. Par exemple, si une offre spéciale concerne une carte de transport, le vocabulaire commun entre les deux catégories peut être important. La possibilité qu'une conversation aborde plusieurs sujets alors que nous recherchons seulement le sujet principal peut aussi expliquer la proximité sémantique des classes. Il est par exemple commun de demander des informations d'horaires supplémentaires lorsque l'on aborde des sujets comme un Itinéraire ou l'État du trafic. Si l'on compare les deux projections, on peut remarquer que l'ensemble des catégories d'erreur diminue ou stagne avec l'emploi du TDAE. Les méthodes employées, en encodant l'information, réduisent les sources de confusion dans le système, mais en introduisent peu de nouvelles. On peut donc accepter que l'encodage dans ces espaces réduits supprime de l'information qui porte à confusion ou met en avant l'information utile. C'est l'effet qu'on attend d'un outil de débruitage. Cela confirme que les autoencodeurs remplissent bien leur rôle de débruitage.

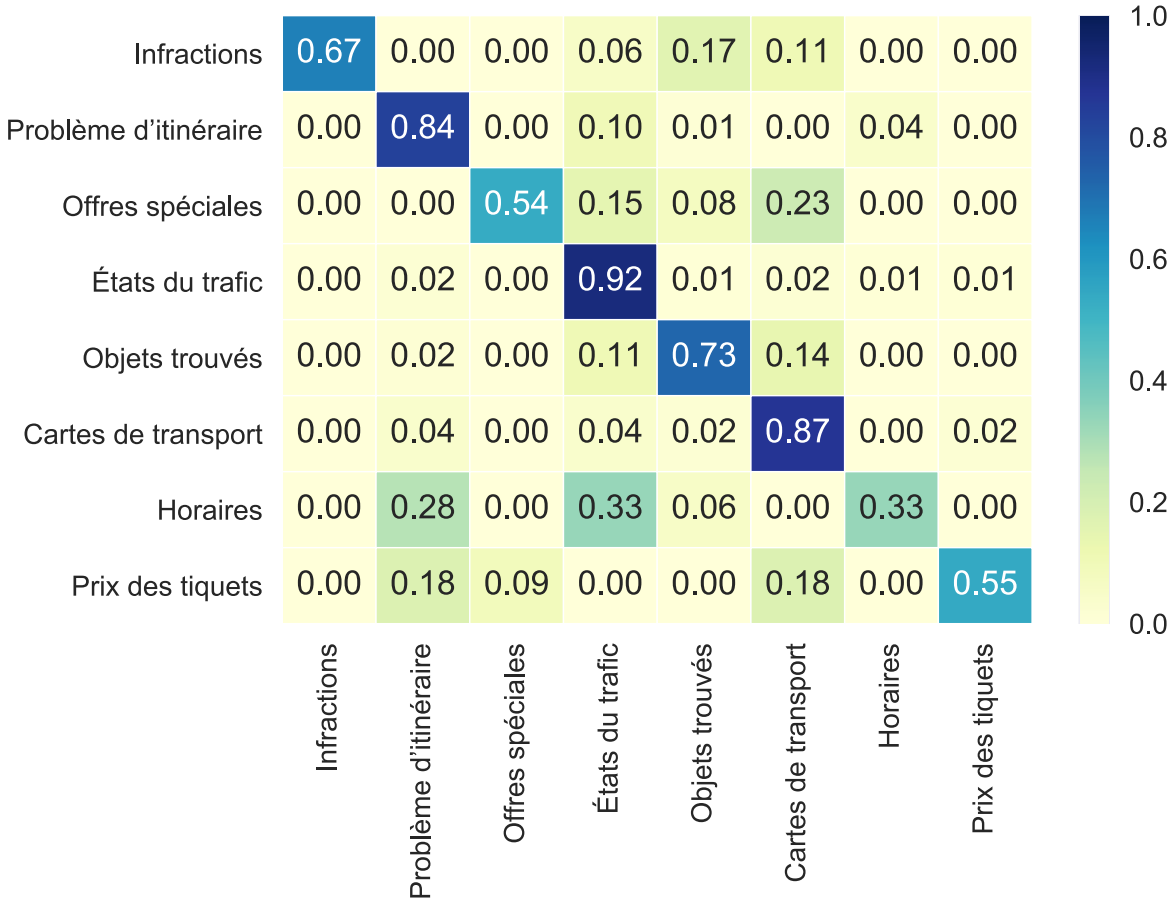


Figure *10.1* – *Projection en 2D des confusions réalisées lors de la classification thématique avec des représentations construites par un autoencodeur simple.*

Pour analyser la significativité des résultats présentés dans le Chapitre 9 nous avons utilisé les résultats de catégorisation réalisées par les dix entrainements de 3 réseaux de neurones artificiels : les réseaux homogènes simples (AE_{ASR} et AE_{TRS}) et le $TDAE_{ASR}$. Ces trois variables aléatoires ont un coefficient d'aplatissement inférieur à 1 (0,49 ;-0,74 et -0,66 respectivement) et un coefficient de dissymétrie lui aussi inférieur à 1 (-0,75 ; -0,31 et -0,39 respectivement). Nous considèrerons donc que ces variables aléatoires suivent des lois normales et donc que nous pouvons tester la significativité des différences de performances avec un test de Student pour des échantillons indépendants (t-test). Trois tests ont été réalisés :

— Le premier test compare AE_{ASR} et AE_{TRS} avec pour hypothèse nulle que les échantillons de performances réalisées par les deux systèmes d'apprentissage ont la même moyenne. On obtient pour ce test une p-value de 6×10^{-3} qui est inférieure à 5%. On peut donc rejeter l'hypothèse nulle.

— Le second compare AE_{TRS} et $TDAE_{ASR}$. Avec une hypothèse nulle identique, on obtient une p-value de $0,27$ largement supérieure à 5%. On ne peut donc pas rejeter l'hypothèse nulle. La différence entre les performances qui utilisent les transcrip-

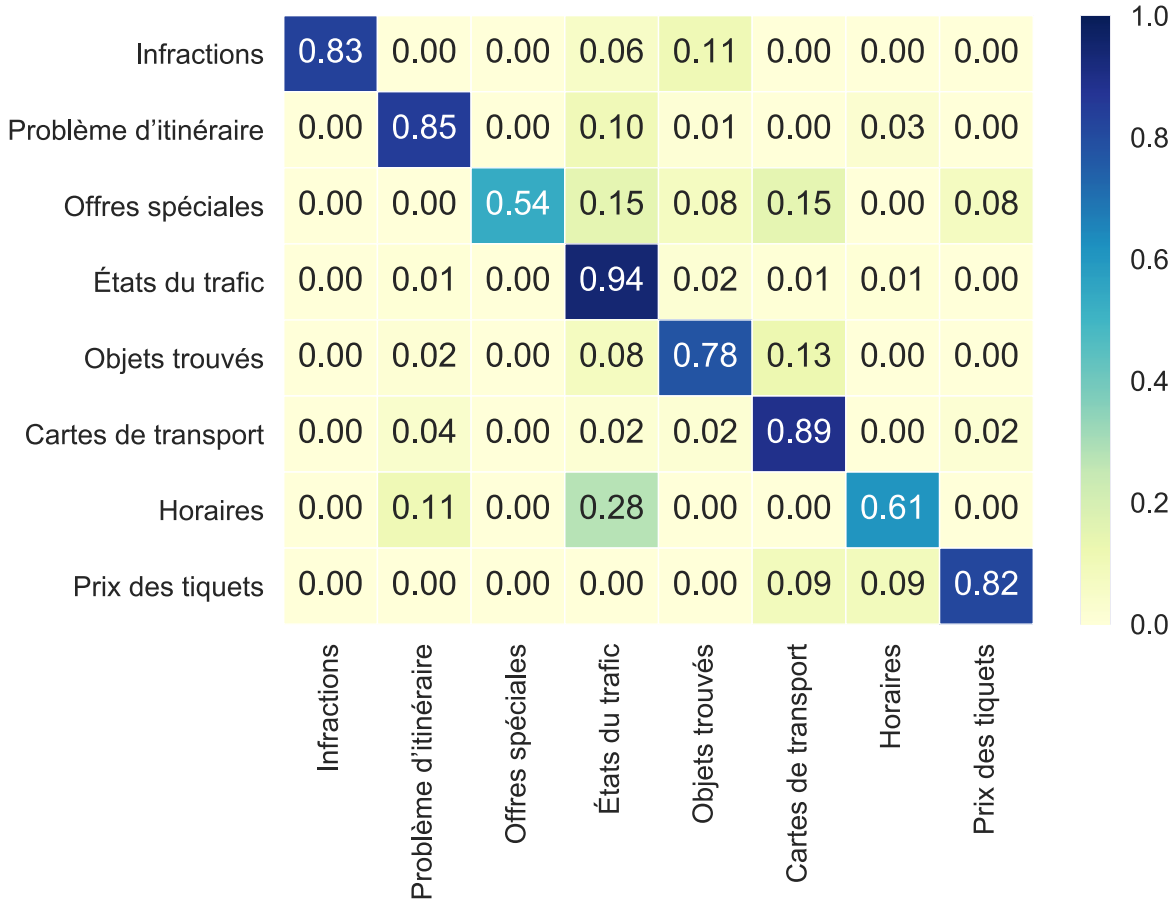


Figure *10.2* – *Projection en 2D des confusions réalisées lors de la classification thématique avec des représentations construites par un TDAE.*

tions manuelles ne sont pas significativement différentes des performances obtenues avec les transcriptions débruitées par le $TDAE_{ASR}$.

— Le dernier compare $TDAE_{ASR}$ et AE_{ASR} avec toujours pour hypothèse nulle que les échantillons de performances réalisées par les deux systèmes d'apprentissage ont la même moyenne la p-value devient 2×10^{-4}. La différence de performance est donc significative.

Ces différents tests confirment des hypothèses émises au vu des résultats des chapitres précédents. On peut en noter deux en particulier. La première est la confirmation que l'utilisation des transcriptions manuelles produit des résultats significativement différents. Les erreurs introduites par l'utilisation d'un SRAP sont donc significatives dans ces conditions expérimentales. Et la seconde est que les performances obtenues en utilisant des représentations débruitées artificiellement par un réseau adapté sont comparables à celles produites par un système qui repose sur des transcriptions manuelles.

10.2 Comparaison globale

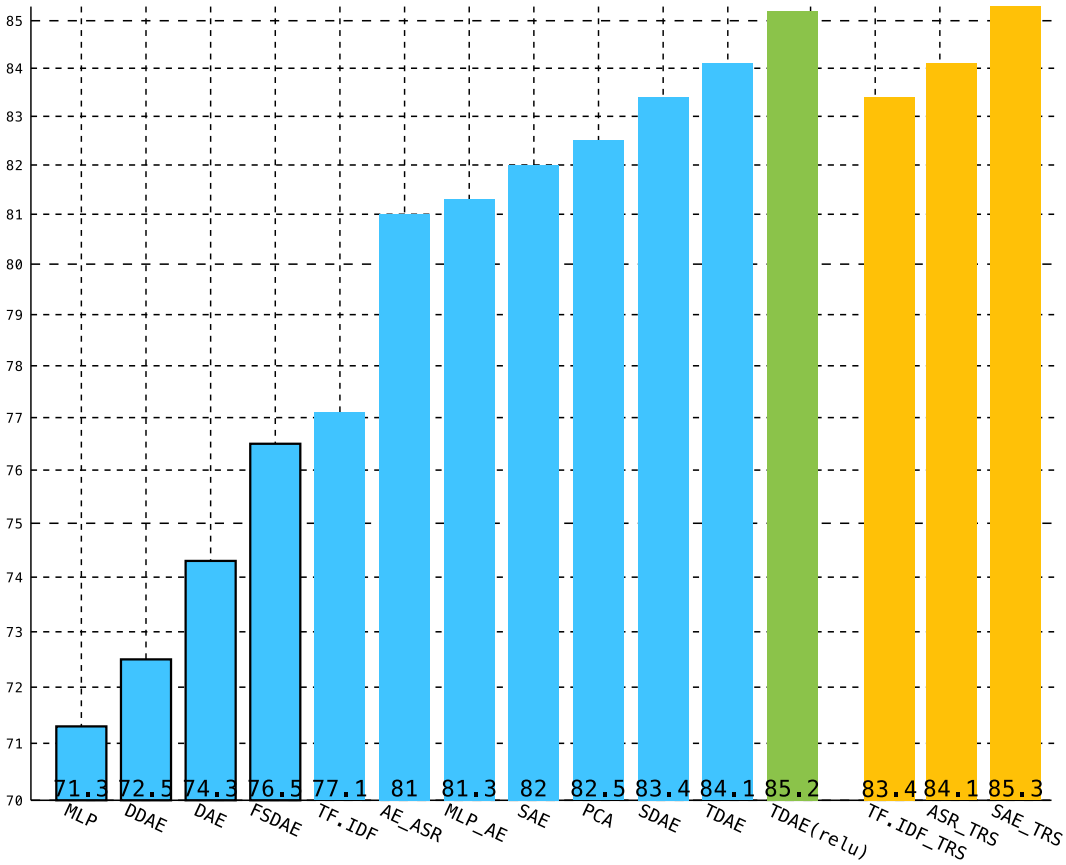


Figure *10.3* – *Récapitulatif des performances obtenues avec des autoencodeurs pour la classification thématique de documents parlés sur la tâche TID.*

Trois catégories de résultats sont présentées dans cet histogramme. Les scores en jaune (à droite) sont obtenus à partir des documents transcrits manuellement. Les scores en bleu et vert sont obtenus à partir des documents transcrits automatiquement et les scores bordés de noir représentent les systèmes qui dégradent les performances initiales. Il faut noter que le TDAE(relu) en vert utilise une configuration expérimentale légèrement différente des autres modèles, la comparaison avec ceux-ci est à nuancer. À titre de rappel, les scores réalisés par $TF.IDF_{TRS}$, ASR_{TRS} et SAE_{TRS} sont obtenus en utilisant des documents propres transcrits manuellement. Ils représentent les résultats optimaux des systèmes proposés et reflètent une situation sans bruit de transcription automatique. Dans cette situation, l'utilisation d'autoencodeur débruitant améliore la qualité des représentations en supprimant une variabilité qui est liée au langage, à la forme, à la sémantique employée et non au SRAP. Cette variabilité est donc présente aussi dans les documents transcrits automatiquement, mais certainement altérée par les erreurs de transcription automatique. L'utilisation d'autoencodeur débruitant sur les documents propres permet d'obtenir jusqu'à $85,3\%$ de précision, soit une augmentation relative d'environ $2,3\%$.

Le score de référence utilisé dans ces travaux est celui obtenu en utilisant des descripteurs TF.IDF, pour la classification, sans traitement supplémentaire. Ceux-ci permettent au classifieur d'atteindre un score de $77,1\%$. Le bruit introduit par le SRAP provoque donc une perte absolue de plus de 6%. L'utilisation d'autoencodeurs débruitants classiques ou empilés permet de produire une représentation plus robuste des documents bruités d'origine en réalisant 81% et 82% respectivement, soit une augmentation relative de 5% et $6,3\%$. La différence d'amélioration entre les deux environnements, l'un propre et l'autre bruité montre qu'une partie du bruit retiré par les autoencodeurs débruitants est bien le produit du SRAP.
Ces performances sont comparables avec les scores obtenus en projetant les données bruitées dans un espace appris par une analyse en composantes principales. En effet, utiliser une PCA permet d'obtenir une précision de $82,5\%$. Cette faible différence n'est pas surprenante au vu de la proximité théorique entre les autoencodeurs linéaires et la PCA (Baldi, 2012). L'utilisation d'une PCA permet un calcul plus simple et plus rapide que l'entrainement d'un autoencodeur ainsi qu'une interprétation plus évidente. En contrepartie, une PCA est sensible au passage à l'échelle. De plus, l'intégration de l'information de supervision de débruitage (document TRS) et l'apprentissage de relations non linéaire ne sont pas réalisables avec une PCA.
Les résultats des DAE, DDAE et FSDAE rappellent qu'un réseau de neurones, même profond, initialisé avec des poids issus d'un autoencodeur débruitant, ne parvient pas à construire une représentation intermédiaire pertinente en exploitant l'information des documents transcrits manuellement. Ils montrent aussi que les représentations visibles d'un même document sont trop différentes pour arriver à créer un espace intermédiaire directement. Les représentations construites par ces systèmes obtiennent des résultats inférieurs aux représentations initiales.

Les descripteurs extraits du MLP obtiennent les plus mauvais scores du panel. Ce résultat montre que l'utilisation d'une couche *bottleneck* n'a pas d'effet positif en soi sur les performances globales du système. Si elle est utilisée sans un objectif précis, elle dégrade les performances comparées à une classification directe. Ce comportement est logique puisque cette utilisation d'un MLP s'approche du simple fait de rajouter des couches dans un unique réseau. L'ajout de couches cachées augmente le nombre de paramètres dans le réseau et rend donc son apprentissage plus difficile. L'excès de paramètres a un effet particulièrement négatif puisque le corpus est relativement petit. Dans cette expérience, elle provoque au mieux une chute relative de $7,5\%$.

La différence absolue de $3,9\%$ entre le MLP-AE et le TDAE montre que la supervision introduite par les documents propres joue un rôle important pour le débruitage des documents. Elle permet au réseau de détecté la variabilité introduite par le SRAP et donc de créer une représentation qui ne contient pas les éléments récurrents qui induisent le système en erreur.

Les performances du SDAE et TDAE, $83,4\%$ et $84,1\%$ respectivement, apportent un gain relatif de $8,2\%$ et $9,1\%$ comparés aux résultats du système de référence. Ces gains confirment que l'information des documents propres peut être utilisée pour créer une représentation plus robuste des documents bruités. La différence de performance avec les résultats des DDAE et FSDAE montre que les contraintes structurelles imposées

aux SDAE et TDAE ont un effet positif. Ceux-ci sont contraints de se concentrer sur les relations entre les informations utiles des deux espaces de représentation, alors que les homologues sans contraintes lient des variabilités certainement statistiquement plus proches, mais qui s'apparentent à du bruit. On peut déduire aussi de ces résultats qu'il est plus efficace de travailler sur des représentations latentes compressées des documents. En effet, ces représentations modélisent une information forte liée à l'organisation sous-jacente des documents. Il est plus simple pour un système automatique de différencier ces modélisations que les documents d'origines.
Le gain relatif de $0,8\%$ du TDAE comparé au SDAE montre que l'information réduite issue du MLP est comparable à un autoencodeur dans le sens où une partie de la variabilité inutile est supprimée. En revanche, l'adaptation de l'information pour la tâche la rend plus pertinente pour la classification.

En fin de compte, l'information construite de cette manière permet d'obtenir des résultats similaires à l'information employée comme référence pour le débruitage (AE_{TRS}). Le TDAE permet donc de compenser le bruit introduit par le SRAP.

Enfin le TDAE avec relu (en vert) obtient le score de $85,2\%$. Ce score n'est pas comparable au SDAE qui pourrait employer une fonction d'activation équivalente pour l'apprentissage de sa couche centrale et potentiellement obtenir la même progression. Par contre, ce résultat montre qu'une représentation creuse est bénéfique pour la classification finale. Il montre aussi que le TDAE atteint une performance équivalente au SAE_{TRS}. Dans cette version, l'étape de débruitage supervisé produit une représentation plus robuste, mais aussi d'un plus haut niveau d'abstraction que les documents d'entrée et de sortie. L'utilisation de méthodes de débruitage ne permet pas de construire un document lisible par un humain et sans erreurs de transcription. En revanche, il est possible de gommer très fortement l'impact négatif du SRAP, à l'aide de données d'un système de débruitage adapté, pour rendre les performances d'un système de classification thématique statistiquement comparable à un système utilisant des transcriptions manuelles.

En résumé, le TDAE obtient les meilleurs résultats en conditions bruitées, mais il n'est pas toujours applicable. Ce réseau repose sur la présence d'informations de supervision de deux types. L'étape de débruitage utilise une forme de supervision en ayant recours aux transcriptions manuelles, alors que l'étape de préentrainement utilise la supervision de classification (la tâche).
Selon la situation, les différentes solutions présentées dans ces travaux peuvent être intéressantes.
Dans la situation où les transcriptions manuelles sont disponibles :

— Si la supervision de classification est utilisable lors de la création des descripteurs alors le TDAE est idéal.

— Sinon le SDAE est une solution.

Dans le situation où les transcriptions manuelles ne sont pas utilisable lors de l'apprentissage, des solutions relativement performantes sont envisageables :

— Si le volume de données disponible n'est pas très important, alors l'utilisation d'une PCA est recommandée.

— Si les données nécessitent une solution capable de passer à l'échelle sans contrainte de temps d'apprentissage alors un SAE est une bonne solution.

— Si le volume de données est important et que l'apprentissage doit être rapide alors le DAE est le seul réseau, parmi les présentés, envisageable.

Chapitre 11

Conclusion et perspectives

La compréhension automatique de la parole est une problématique difficile, en particulier parce qu'indépendamment de l'approche choisie, un système automatique de compréhension de la parole (SLU) fait face à l'accumulation de plusieurs facteurs adverses. Dans le cadre d'une tâche d'identification thématique de documents parlés issus de systèmes de reconnaissance automatique de la parole, le système doit surmonter plusieurs sources d'erreurs. Parmi les plus importantes, on retrouve la présence de fortes variabilités acoustiques qui vont impliquer des erreurs de transcription et des disfluences fréquentes, qui introduisent des incertitudes fortes dans l'analyse de la parole. À celles-ci s'ajoutent des problématiques communes avec traitement du langage écrit telles que la polysémie ou les anaphores.
Face à ces freins, deux approches sont possibles : réduire ces sources d'erreurs, ou permettre au système d'être aussi peu affecté que possible par celles-ci lorsqu'elles se présentent. L'introduction de processus adaptés à la SLU pour introduire de la robustesse dans le système automatique est donc un facteur clé pour le bon fonctionnement du système, dans un cadre réel où les données sont faiblement contrôlées. Les travaux de ce manuscrit proposent d'utiliser des réseaux de neurones artificiels pour construire des espaces de représentation qui modélisent l'information latente des données, afin de limiter l'impact des erreurs de SRAP. Dans ce but, nous avons d'abord proposé différentes formes de représentation des documents parlés. Des représentations en sac-de-mots, par *embeddings*, débruitées et thématiques ont été proposées avec un succès variable. Ensuite, nous avons analysé la composition, via des autoencodeurs, de plusieurs vues de mêmes documents, elles-mêmes produites par des modèles thématiques de granularités différentes. Enfin, nous avons proposé des méthodes capables d'exploiter une vue propre des documents, disponible seulement sur les de documents de référence, en plus de la vue bruitée, pour faciliter la compréhension de documents uniquement disponibles sous la vue dégradée.

11.1 Bilan des contributions

Dans le Chapitre 5, des représentations construites à partir des réseaux de neurones *Word2Vec* sont utilisées pour projeter les documents dans un espace modélisant les relations sémantiques et syntaxiques entre les mots. Ces méthodes considèrent que des mots partageant le même voisinage portent la même information. Avec ce prédicat, des réseaux de neurones sont entrainés à prédire un mot, en fonction de son contexte proche. L'espace latent construit par la projection des mots dans la couche cachée de ces réseaux de neurones, modélise des relations sémantiques et syntaxiques intéressantes. Lors de cet apprentissage, ces modèles n'utilisent pas l'information d'ordonnancement des mots. Nous avons proposé une variante de cette méthode qui pondère l'importance des mots en fonction de leur position dans le contexte, de manière à ce que le système puisse tenir compte de cette information de positionnement durant l'apprentissage. Une analyse qualitative des modèles entrainés avec et sans pondération a montré que l'utilisation de l'information d'ordonnancement révèle des segmentations dans l'espace sémantique qui n'existe pas avec la représentation classique. Cette propriété a été confirmée par les expérimentations réalisées dans ce chapitre. La première expérience utilise une tâche d'analogie sémantique et syntaxique, qui montre que les représentations construites avec une pondération adaptée, modélisent mieux les proximités sémantiques et syntaxiques entre les mots. Enfin, la seconde expérience montre que cette information d'ordonnancement peut être également exploitée par un classifieur, pour améliorer de manière importante les résultats en identification de thématique de documents parlés. Cependant, cette même expérience montre également que malgré les gains apportés par l'évolution du modèle *Word2vec*, une composition des documents dans l'espace sémantique reposant uniquement sur les propriétés linéaires de celui-ci, produit une vue des documents qui n'est pas compétitive avec l'état-de-l'art dans ce domaine.

Le Chapitre 6 aborde directement le problème des erreurs introduites par le SRAP. Dans ce chapitre, des représentations construites à partir de documents transcrits automatiquement sont comparées à des représentations transcrites manuellement. Les erreurs de transcription automatique provoquent une chute directe des performances du système. Dans ces travaux, nous proposons d'utiliser des autoencodeurs qui vont apprendre à reconstruire le document d'origine à partir d'une version artificiellement corrompue des documents transcrits automatiquement. Ainsi, ce réseau de neurones apprend sans supervision à construire une vue de taille réduite et débruitée des documents transcrits. Les expérimentations réalisées dans ce chapitre montrent que l'utilisation d'un autoencodeur débruitant empilé permet, via un apprentissage couche par couche, de construire une représentation de plus haut niveau d'abstraction plus robuste, qui réduit nettement l'écart entre l'utilisation de documents transcrits manuellement et les documents transcrits automatiquement.

Le Chapitre 7 propose d'exploiter la robustesse à l'incertitude inhérente aux modèles génératifs thématiques latents. Nous utilisons dans ce cadre des modèles dérivés de l'allocation de Dirichlet latente appelés *Author-Topic* capables de tenir compte de l'information de classification pour apprendre un modèle génératif robuste. Une des faiblesses de ces modèles réside dans le choix des paramètres initiaux, en particulier le nombre de

thématiques, qui peuvent être utilisées par l'AT pour modéliser les données d'apprentissage. Chaque modèle AT peut être considéré comme une vue différente des données pour une quantité de thématiques d'apprentissage données. Le choix de la bonne vue a un impact significatif sur les résultats pour la tâche de compréhension de la parole. Pour pallier cette faiblesse, une méthode de fusion de toutes les vues construites par les modèles AT à l'aide d'un autoencodeur débruitant est proposée. En fusionnant les différentes vues des documents, cette méthode permet de construire une représentation de meilleure qualité que celle obtenue en utilisant chaque modèle AT indépendamment. Les descripteurs ainsi construits sont plus robustes, plus efficaces et plus simples à mettre en oeuvre que la solution présente dans la littérature.

Le Chapitre 8 s'inspire des propositions précédentes pour proposer des approches multivues qui combinent une représentation (vue) construite avec les données transcrites automatiquement par le SRAP et une vue construite à partir des transcriptions manuelles. Dans le Chapitre 6, les autoencodeurs débruitants apprennent à reconstruire un document propre, à partir d'un document bruité artificiellement. Nous pouvons faire un parallèle avec les documents parlés, où la transcription automatique est la version bruitée et la transcription manuelle la version propre. Un autoencodeur débruitant pourrait utiliser l'information de supervision apportée par les documents propres pour apprendre à les reconstruire à partir des documents bruités. Cette solution aurait l'avantage de construire une représentation intermédiaire liant les deux vues, qui serait de meilleure qualité que les documents bruités, mais qui ne nécessiterait les documents propres que lors de l'apprentissage. Les expérimentations de ce chapitre montrent que cette approche n'est pas viable. Les formes apparentes des documents propres et bruités sont trop différentes. Les autoencodeurs ne parviennent pas à apprendre une fonction permettant de passer de l'espace bruité à l'espace propre. Nous proposons une architecture alternative d'autoencodeur capable d'exploiter la vue propre des documents combinés avec la vue bruitée. Elle lie des espaces portant une information sémantique latente, plus proche que les espaces portant les formes de surface des documents. Les représentations des documents construites de cette manière sont démontrées expérimentalement plus robustes et plus efficaces pour la classification, que les représentations standards et débruitées par autoencodeur empilé.

Enfin le Chapitre 9, se place comme une évolution des travaux présentés dans le Chapitre 8. En effet, la méthode précédente utilise des autoencodeurs pour construire des vues latentes des documents propres et des documents bruités qui sont ensuite encodés dans un espace commun. Nous proposons dans une nouvelle architecture d'introduire l'information de supervision de la tâche au plus tôt dans le processus de débruitage. Cette information supplémentaire permet, lors de la construction des vues latentes propres et bruitées, de filtrer les documents pour conserver seulement l'information pertinente pour la tâche. Ces vues, d'un plus haut niveau d'abstraction, sont donc moins bruitées et plus faciles à combiner dans une représentation intermédiaire débruitée robuste. Les résultats expérimentaux confirment que la représentation créée de cette manière est plus robuste que les représentations alternatives. Ils montrent aussi que cette architecture de débruitage est capable de combler, presque intégralement, la perte induite par l'utilisation d'un SRAP, en obtenant des résultats comparables à l'utilisation de représentations propres

d'un haut niveau d'abstraction.

11.2 Perspectives de ces travaux

Au fil des différents chapitres, nous avons proposé plusieurs évolutions possibles pour les différents travaux présentés. Certaines ont été étudiées tandis que d'autres sont toujours en suspens. Parmi les problématiques non abordées, l'utilisation des classifieurs capables d'exploiter la structure des documents est un point important. En effet, la structure des documents parlés est particulièrement porteuse d'informations. L'utilisation de réseaux de neurones récurrents tels que des LSTM (Graves, 2012) ou des GRU (Chung et al., 2015) pourrait apporter un gain significatif pour le traitement des documents bruités. De plus, une des limitations importantes des méthodes proposées dans ces travaux est que les représentations débruitées construites ne sont exploitables que par des machines. L'interprétation de ces représentations par un humain n'est pas possible puisque les relations aux mots d'origine et la structure des documents sont perdues lors du processus de création de documents multivues. Permettre au système automatique de produire un texte complet interprétable par un humain est un enjeu important qui ouvrirait des champs applicatifs intéressants. Cette problématique peut être approchée de plusieurs manières. Une première piste consisterait à utiliser des méthodes d'extraction d'information de façon similaire aux méthodes de résumé automatique de texte. Une seconde approche consisterait à combiner des réseaux de neurones récurrents avec des réseaux génératifs puissants comme ceux présentés dans le chapitre, 3 pour construire de nouveaux documents propres.

Une autre piste d'évolution est celle des méthodes d'apprentissage jointes. Avec ces méthodes, des réseaux de neurones sont employés pour résoudre plusieurs tâches par un apprentissage unique. Une adaptation des réseaux de neurones qui lient les vues latentes des documents propres et bruités pourrait être proposée pour apprendre à construire les vues latentes et créer une représentation robuste, en une seule étape d'apprentissage, au lieu des deux utilisées actuellement. Ces méthodes sont capables de créer de meilleures représentations lorsque les corpus de données sont suffisamment grands. Dans ce framework, il serait aussi possible à l'instar du Chapitre 7 de combiner les vues construites par des *embeddings* de mots et par des modèles thématiques, afin de créer une représentation multivues, dont les vues varieraient en fonction de la source (propre/bruitée) et du type de modèle (AT,TF.IDF, *Word2vec*) pour permettre la prise de décision en fonction des formes latentes et des formes apparentes.

Enfin, il serait pertinent d'ouvrir ces travaux à d'autres tâches et d'analyser l'effet positif de ces méthodes sur d'autres problématiques de compréhension de la parole. Par exemple, l'utilisation de plusieurs formes de supervisions à l'instar du TDAE peut permettre notamment de réduire l'impact négatif du SRAP dans le cadre de conversations homme/machine. Elles pourraient aussi avoir un effet positif dans de nombreux domaines nécessitant de travailler avec des données dégradées, comme des données sensorielles ou des segments d'images.

Liste des illustrations

2.1 La décomposition matricielle de la méthode LSA 26
2.2 Le modèle pLSA sous forme de *Plate notation*. 27
2.3 Le modèle pLSA d'un point de vue d'analyse factorielle 28
2.4 Le modèle LDA sous forme de *Plate notation*. 29

3.1 Chronologie des évènements les plus marquants de l'apprentissage profond. Une table de correspondance des publications est disponible en annexe A. 37
3.2 Schémas d'un neurone artificiel (a) et des fonctions d'activations historiques (b) 38
3.3 Architecture d'un Perceptron (a) et exemple classification avec un Perceptron (b et c) 38
3.4 Présentation des machines de Boltzmann 40
3.5 Architecture des machines de Boltzmann restreintes. 40
3.6 Architecture des réseaux de croyance profonds et des machines de Boltzmann profondes. 41
3.7 Présentation d'un Perceptron multicouche. 42
3.8 Schéma d'un réseau de convolutions. 45
3.9 Architecture des réseaux de neurones récurrents Elman et Jordan et un exemple de réseau déroulé. 46
3.10 Principe de fonctionnement d'un réseau récurrent bidirectionnel. 47
3.11 Schéma d'une cellule LSTM. 48
3.12 Schéma d'un réseau de neurones récurrent avec attention. 49
3.13 Fonction d'activation relu et softplus (Glorot et al., 2011). 51
3.14 Schéma de deux réseaux de neurones avec et sans dropout. 52

4.1 Exemple de dialogue, transcrit manuellement, extrait de la tâche TID attribué à la catégorie "problème d'itinéraire" par un expert, mais qui contient aussi le thème "État du trafic". 59
4.2 Processus simplifié de labellisation de conversation pour la tâche TID. . . 61

5.1 Les architectures Word2vec. 70
5.2 Exemple du facteur de réduction appliqué à des fenêtres de contexte dans un réseau Skip-gram. 71

5.3 Comparaison des poids globaux accordés aux mots d'un contexte avec le facteur de réduction et avec la pondération locale par contextes continus. 72
5.4 Le réseau LL-CBOW utilisant une pondération log-linéaire des contextes proposée. . . . 73
5.5 Mouvements dans l'espace Word2vec durant un apprentissage de réseau CBOW avec et sans pondération. . . . 74
5.6 Le réseau LL-SG avec la pondération log-linéaire des contextes proposée. . 75
5.7 Mouvements dans l'espace Word2vec durant l'apprentissage de réseau Skip-gram avec et sans pondération. . . . 76
5.8 Exemple des relations apprises par les architectures Word2vec. . . . 77
5.9 Voisinage du mot *Holidays* appris par un CBOW et un LL-CBOW projeté en 2D. On peut remarquer une séparation par topic sur l'axe des abscisses présent seulement chez le LL-CBOW . . . 79
5.10 Précision de classification (en %) sur la tâche TID obtenue avec les modèles Skip-gram, **LL-SG**, CBOW, **LL-CBOW** en fonction du nombre d'époques d'entrainement. . . . 82

6.1 Schéma d'un autoencodeur. Les biais sont ignorés pour simplifier le schéma. 88
6.2 Modèle graphique d'un autoencodeur débruitant. Le vecteur d'entrée x est corrompu aléatoirement pour générer $x^{(\text{corrompu})}$. Ensuite $x^{(\text{corrompu})}$ est projeté dans l'espace latent h pour ensuite produire la reconstruction $\tilde{x}$. L'erreur de reconstruction est calculée avec L_{MSE}. . . . 89
6.3 Modèle graphique d'un autoencodeur débruitant hétérogène. Le vecteur d'entrée $x^{(ASR)}$ est corrompu par le SRAP. La sortie $x^{(TRS)}$ provient de transcriptions manuelles. . . . 90
6.4 Architecture d'un autoencodeur empilé (SAE). Les autoencodeurs simples pour le préentrainement sont à gauche. Leurs poids sont utilisés pour initialiser le réseau complet (à droite). . . . 91
6.5 Architecture d'autoencodeurs débruitants profonds hétérogènes. Ses vecteurs d'entrée sont naturellement bruités par un SRAP et ses vecteurs de sortie propres sont les transcriptions manuelles. . . . 92

7.1 Le modèle *Author-Topic* sous forme de *Plate notation*. . . . 101
7.2 Exemple de conversation DECODA projeté dans un espace *Author-Topic*. 102
7.3 Trois processus de création des représentations compressées. (a) l'état de l'art c-vecteurs, (b) la proposition à base d'autoencodeur et (c) la proposition LTS . . . 105
7.4 Performances (%) réalisées sur la tâche TID en utilisant des modèles Author-topic en variant le nombre de topics utilisé pour l'apprentissage. . 109
7.5 Précision réalisée sur le corpus **20-Newsgroups** en fonction du nombre de topic utilisé par les modèles Author-Topic. . . . 110
7.6 Performances (%) réalisées sur la tâche TID en utilisant la couche cachée d'un *autoencodeur* et en variant le nombre de neurones dans cette couche entre 10 et 300 neurones. . . . 111

7.7 Précision (%) obtenue sur la tâche d'identification de thématique TID, en utilisant les projections des documents dans les sous-espaces *LTS* en variant le nombre de valeurs propres entre 40 et 300. 112
7.8 Précision (%) obtenue sur la tâche de classification de documents textes **20-Newsgroups** en utilisant les couches cachées d'un *autoencodeur* variant le nombre de neurones de 10 et 300. 114
7.9 Précision (%) obtenue sur la tâche de classification de documents textes **20-Newsgroups** en utilisant les projections des documents dans les sous espaces **LTS** en variant le nombre de valeurs propres de 40 et 300. . . . 115

8.1 Processus d'utilisation de caractéristiques Bottleneck. 120
8.2 Génération de caractéristiques Bottleneck. 121
8.3 Architecture de l'autoencodeur profond supervisé. À gauche, les deux réseaux préentrainés et à droite le réseau final. 123
8.4 Performances par couche cachée et par réseau sur la tâche TID. Les caractéristiques AE_{TRS} (en jaune) sont les seules apprises et évaluées uniquement avec les documents transcrits manuellement. 126
8.5 Proportions $\frac{Performance_i}{Performance_{Max}}$ en fonction des itérations pour les MLP utilisant les caractéristiques des Autoencodeurs sans *early stopping* 127
8.6 Temps de calcul nécessaire aux apprentissages des différents Autoencodeurs profonds. 128

9.1 Extraction de caractéristiques bottleneck dans un réseau supervisé. 132
9.2 Architecture de l'autoencodeur débruitant avec des caractéristiques spécifiques à la tâche. À gauche, les deux réseaux préentrainés et à droite le réseau final. 134
9.3 Temps de calcul nécessaire aux apprentissages du TDAE et du SDAE. . . 138

10.1 Projection en 2D des confusions réalisées lors de la classification thématique avec des représentations construites par un autoencodeur simple. . . 142
10.2 Projection en 2D des confusions réalisées lors de la classification thématique avec des représentations construites par un TDAE. 143
10.3 Récapitulatif des performances obtenues avec des autoencodeurs pour la classification thématique de documents parlés sur la tâche TID. 144

B.1 Schéma d'un arbre de décisions. 190

Liste des tableaux

2.1 Liste non exhaustive de fonctions d'importance pour le filtrage des mots où t_k est le terme du vocabulaire à évaluer en fonction de la classe c_i, $DF_{c_i}(t_k)$ est le nombre de documents associés à la classe c_i contenant le terme t_k et d est un facteur d'amortissement. 25

4.1 Description des classes dans la tâche TID. 60
4.2 Métaparamètres sélectionnés pour le MLP appliqué sur TID. 64
4.3 Précisions des méthodes simples sur TID et les résultats MLP TRS dans le cas où des transcriptions idéales (manuelles) remplacent le SRAP. . . . 64

5.1 Exemples de questions présentes dans le « *Semantic-Syntactic Word Relationship test*» des mots définis dans (Mikolov et al., 2013a) 78
5.2 Exemple de mots voisins extraits de modèle entrainé sans et avec information contextuelle (CBOW et LL-CBOW). On peut voir dans chaque voisinage ressortir deux «catégories» de mots. Cette différence est représentée en gras. 79
5.3 Performance (%) sur la tâche *Semantic-Syntactic Word Relationship test* avec ou sans pondération en fonction de la taille du contexte utilisé (c) avec une couche cachée fixe de taille 300 80
5.4 Performance (%) sur la tâche *Semantic-Syntactic Word Relationship test* avec ou sans pondération en fonction de la taille de la couche cachée utilisée avec une fenêtre de contexte fixe de taille 10. 80
5.5 Précision de classification (%) sur la tâche TID en utilisant deux classifieurs et les caractéristiques issues des modèles Skip-gram et CBOW avec et sans pondération log-linéaire. 82

6.1 Temps d'apprentissage pour les différents réseaux de neurones réalisés avec une carte graphique (GPU) Nvidia Geforce TITAN X. 93
6.2 Précision(%) sur la tâche TID en utilisant les représentations produites par les différentes couches cachées des autoencodeurs simples débruitants. 94
6.3 Précision (%) de classification thématique en utilisant les caractéristiques produites par les différentes couches des autoencodeurs profonds. 94

7.1 La répartition des documents dans le 20-Newsgroups corpus. 108

7.2 Précision (%) de classification thématique de dialogues transcrits automatiquement selon plusieurs configurations de c-vecteur. 111
7.3 Précision en classification thématique (%) sur la tâche TID. 112
7.4 Précision en classification (%) sur les données 20-Newsgroups utilisant des c-vecteurs en variant leur taille et le nombre de gaussiennes utilisées lors de l'apprentissage. 113
7.5 Précisions (%) obtenues par les meilleures configurations de chacun des systèmes présentés sur le Corpus 20-Newsgroups. 115

8.1 Récapitulatif des meilleures performances (%) observées pour chaque réseau de neurones artificiels sur la tâche TID. 128

9.1 Précisions des différents systèmes présentés sur la tâche de classification thématique de documents parlés TID. Dans ce tableau Y' correspond à l'hypothèse de classification. 137

Bibliographie

(Abdi et Williams, 2010) H. Abdi et L. J. Williams, 2010. Principal component analysis. *Wiley Interdisciplinary Reviews : Computational Statistics* *2*(4), 433–459.

(Ackley et al., 1985) D. H. Ackley, G. E. Hinton, et T. J. Sejnowski, 1985. A learning algorithm for boltzmann machines. *Cognitive Science 9*, 147–169.

(Amodei et al., 2016) D. Amodei, S. Ananthanarayanan, R. Anubhai, J. Bai, E. Battenberg, C. Case, J. Casper, B. Catanzaro, Q. Cheng, G. Chen, J. Chen, J. Chen, Z. Chen, M. Chrzanowski, A. Coates, G. Diamos, K. Ding, N. Du, E. Elsen, J. Engel, W. Fang, L. Fan, C. Fougner, L. Gao, C. Gong, A. Hannun, T. Han, L. V. Johannes, B. Jiang, C. Ju, B. Jun, P. LeGresley, L. Lin, J. Liu, Y. Liu, W. Li, X. Li, D. Ma, S. Narang, A. Ng, S. Ozair, Y. Peng, R. Prenger, S. Qian, Z. Quan, J. Raiman, V. Rao, S. Satheesh, D. Seetapun, S. Sengupta, K. Srinet, A. Sriram, H. Tang, L. Tang, C. Wang, J. Wang, K. Wang, Y. Wang, Z. Wang, Z. Wang, S. Wu, L. Wei, B. Xiao, W. Xie, Y. Xie, D. Yogatama, B. Yuan, J. Zhan, et Z. Zhu, 2016. Deep speech 2 : End-to-end speech recognition in english and mandarin. Dans les actes de *ICML*, ICML'16, 173–182. JMLR.org.

(Arora et Ravindran, 2008) R. Arora et B. Ravindran, 2008. Latent dirichlet allocation based multi-document summarization. Dans les actes de *AND*.

(Asuncion et al., 2009) A. Asuncion, M. Welling, P. Smyth, et Y. W. Teh, 2009. On smoothing and inference for topic models. Dans les actes de *25th UAI*, 27–34. AUAI Press.

(Baker et McCallum, 1998) L. D. Baker et A. K. McCallum, 1998. Distributional clustering of words for text classification. Dans les actes de *21st ACM SIGIR*, 96–103. ACM.

(Baldi, 2012) P. Baldi, 2012. Autoencoders, unsupervised learning, and deep architectures. Dans les actes de *Proceedings of ICML Workshop on Unsupervised and Transfer Learning*, Volume 27, 37–49.

(Baroni et al., 2014) M. Baroni, G. Dinu, et G. Kruszewski, 2014. Don't count, predict ! a systematic comparison of context-counting vs. context-predicting semantic vectors. Dans les actes de *ACL (1)*, 238–247.

(Béchet et Charton, 2010) F. Béchet et E. Charton, 2010. Unsupervised knowledge acquisition for extracting named entities from speech. Dans les actes de *Acoustics Speech and Signal Processing (ICASSP), 2010 IEEE International Conference on*, 5338–5341. IEEE.

(Bechet et al., 2012) F. Bechet, B. Maza, N. Bigouroux, T. Bazillon, M. El-Beze, R. De Mori, et E. Arbillot, 2012. Decoda : a call-centre human-human spoken conversation corpus. Dans les actes de *LREC*, 1343–1347. LREC'12.

(Bellman, 1961) R. E. Bellman, 1961. *Adaptive Control Processes : A Guided Tour*. Princeton university press.

(Bengio et Heigold, 2014) S. Bengio et G. Heigold, 2014. Word embeddings for speech recognition. Dans les actes de *INTERSPEECH*, 1053–1057. ISCA.

(Bengio, 2009) Y. Bengio, 2009. Learning deep architectures for AI. *Foundations and Trends in Machine Learning* *2*(1), 1–127.

(Bengio et al.,) Y. Bengio, O. Delalleau, et N. Le Roux. The curse of dimensionality for local kernel machines. *Techn. Rep 1258*.

(Bengio et al., 2007a) Y. Bengio, P. Lamblin, D. Popovici, H. Larochelle, et al., 2007a. Greedy layer-wise training of deep networks. *NIPS 19*, 153.

(Bengio et al., 2007b) Y. Bengio, Y. LeCun, et al., 2007b. Scaling learning algorithms towards ai. *Large-scale kernel machines* *34*(5).

(Bergstra et al., 2010a) J. Bergstra, O. Breuleux, F. Bastien, P. Lamblin, R. Pascanu, G. Desjardins, J. Turian, D. Warde-Farley, et Y. Bengio, 2010a. Theano : a CPU and GPU math expression compiler. Dans les actes de *SciPy*. Oral Presentation.

(Bergstra et al., 2010b) J. Bergstra, O. Breuleux, F. Bastien, P. Lamblin, R. Pascanu, G. Desjardins, J. Turian, D. Warde-Farley, et Y. Bengio, 2010b. Theano : a CPU and GPU math expression compiler. Dans les actes de *SciPy*. Oral Presentation.

(Bernard et al., 2009) G. Bernard, S. Rosset, O. Galibert, E. Bilinski, et G. Adda, 2009. Limsi participation in the qast 2009 track. Dans les actes de *CLEF*.

(Berry et al., 1995) M. Berry, S. Dumais, et G. O'Brien, 1995. Using linear algebra for intelligent information retrieval. *SIAM review* *37*(4), 573–595.

(Bigot et al., 2013) B. Bigot, G. Senay, G. Linares, C. Fredouille, et R. Dufour, 2013. Combining acoustic name spotting and continuous context models to improve spoken person name recognition in speech. *Interspeech*, 2539–2543.

(Bishop, 1995) C. M. Bishop, 1995. *Neural networks for pattern recognition*. Oxford university press.

(Blei et Lafferty, 2009) D. Blei et J. Lafferty, 2009. Text mining : Classification, clustering, and applications. *chapter Topic Models, Chapman & Hall/CRC*.

(Blei, 2012) D. M. Blei, 2012. Probabilistic topic models. *Communications of the ACM 55*(4), 77–84.

(Blei et Lafferty, 2005) D. M. Blei et J. D. Lafferty, 2005. Correlated topic models. Dans les actes de *NIPS*.

(Blei et al., 2003) D. M. Blei, A. Y. Ng, et M. I. Jordan, 2003. Latent dirichlet allocation. *JMLR 3*, 993–1022.

(Bosch et al., 2006) A. Bosch, A. Zisserman, et X. Muñoz, 2006. Scene classification via plsa. *Computer Vision–ECCV 2006*, 517–530.

(Bost et al., 2013) X. Bost, I. Brunetti, L. A. Cabrera-Diego, J.-V. Cossu, A. Linhares, M. Morchid, J.-M. Torres-Moreno, et D. R. El-Bèze, Marc, 2013. Systèmes du lia à deft'13. Dans les actes de *ASRU*, 40.

(Bousquet et al., 2011) P.-M. Bousquet, D. Matrouf, et J.-F. Bonastre, 2011. Intersession compensation and scoring methods in the i-vectors space for speaker recognition. Dans les actes de *Interspeech*, 485–488.

(Bowman et al., 2016) S. R. Bowman, L. Vilnis, O. Vinyals, A. M. Dai, R. Józefowicz, et S. Bengio, 2016. Generating sentences from a continuous space. Dans les actes de *CoNLL*, 10–21. ACL.

(Camacho et al., 2015) F. Camacho, R. Torres, et R. Ramos-Pollán, 2015. Feature learning using stacked autoencoders to predict the activity of antimicrobial peptides. Dans les actes de *CMSB*, 121–132.

(Carcenac et Redif, 2015) M. Carcenac et S. Redif, 2015. A highly scalable modular bottleneck neural network for image dimensionality reduction and image transformation. *Appl. Intell. 44*.

(Caropreso et al., 2001) M. F. Caropreso, S. Matwin, et F. Sebastiani, 2001. A learner-independent evaluation of the usefulness of statistical phrases for automated text categorization. *Text databases and document management : Theory and practice 5478*, 78–102.

(Carreira-Perpinan et Hinton, 2005) M. A. Carreira-Perpinan et G. E. Hinton, 2005. On contrastive divergence learning. Dans les actes de *AISTATS*, Volume 10, 33–40. Citeseer.

(Caruana, 1998) R. Caruana, 1998. Multitask learning. Dans les actes de *Learning to learn*, 95–133. Springer.

(Chan et al., 2016) W. Chan, N. Jaitly, Q. Le, et O. Vinyals, 2016. Listen, attend and spell : A neural network for large vocabulary conversational speech recognition. Dans les actes de *Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on*, 4960–4964.

(Chao et al., 2011) J. Chao, F. Shen, et J. Zhao, 2011. Forecasting exchange rate with deep belief networks. Dans les actes de *IJCNN*, 1259–1266. IEEE.

(Chao et al., 2014) L. Chao, J. Tao, M. Yang, et Y. Li, 2014. Improving generation performance of speech emotion recognition by denoising autoencoders. Dans les actes de *ISCSLP*, 341–344.

(Chellapilla et al., 2006) K. Chellapilla, S. Puri, et P. Simard, 2006. High performance convolutional neural networks for document processing. Dans les actes de *Tenth International Workshop on Frontiers in Handwriting Recognition.*

(Chen, 2009) B. Chen, 2009. Latent topic modelling of word co-occurence information for spoken document retrieval. *2009 IEEE International Conference on Acoustics, Speech and Signal Processing*, 3961–3964.

(Chen et al., 2015) T. Chen, M. Li, Y. Li, M. Lin, N. Wang, M. Wang, T. Xiao, B. Xu, C. Zhang, et Z. Zhang, 2015. Mxnet : A flexible and efficient machine learning library for heterogeneous distributed systems. *CoRR abs/1512.01274.*

(Chieu et Ng, 2002) H. L. Chieu et H. T. Ng, 2002. Named entity recognition : a maximum entropy approach using global information. Dans les actes de *19th COLING*, 1–7.

(Chihi, 1998) I. Chihi, 1998. Understanding kohonen networks. *ENSI, National School for Computer Sciences*, 1–10.

(Chollet,) F. Chollet. Keras : Theano-based deep learning library. *Code : https ://github.com/fchollet Documentation : http ://keras.io.*

(Chopra et al., 2016) S. Chopra, M. Auli, A. M. Rush, et S. Harvard, 2016. Abstractive sentence summarization with attentive recurrent neural networks. *Proceedings of NAACL-HLT16*, 93–98.

(Chorowski et al.,) J. K. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, et Y. Bengio. Attention-based models for speech recognition. Dans les actes de *NIPS*, 577–585.

(Choudhary et al., 2008) A. Choudhary, M. Pal, S. Banerjee, et S. Chaudhury, 2008. Unusual activity analysis using video epitomes and plsa. Dans les actes de *ICVGIP*, 390–397. IEEE.

(Chuangsuwanich et al., 2016) E. Chuangsuwanich, Y. Zhang, et J. R. Glass, 2016. Multilingual data selection for training stacked bottleneck features. Dans les actes de *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP).*

(Chung et al., 2015) J. Chung, C. Gulcehre, K. Cho, et Y. Bengio, 2015. Gated feedback recurrent neural networks. Dans F. Bach et D. Blei (Eds.), *ICML*, Volume 37 de *Proceedings of Machine Learning Research*, Lille, France, 2067–2075. PMLR.

(Chung et al., 2016) Y.-A. Chung, C.-C. Wu, C.-H. Shen, H.-Y. Lee, et L.-S. Lee, 2016. Audio word2vec : Unsupervised learning of audio segment representations using sequence-to-sequence autoencoder. Dans les actes de *Interspeech*, 765–769.

(Cire**c**san et al., 2013) D. C. Cire**c**san, A. Giusti, L. M. Gambardella, et J. Schmidhuber, 2013. Mitosis detection in breast cancer histology images with deep neural networks. Dans les actes de *MICCAI*, 411–418. Springer.

(Clevert et al., 2015) D. Clevert, T. Unterthiner, et S. Hochreiter, 2015. Fast and accurate deep network learning by exponential linear units (elus). *ICLR*.

(Clinchant et Perronnin, 2013) S. Clinchant et F. Perronnin, 2013. Aggregating continuous word embeddings for information retrieval. Dans les actes de *CVSC*.

(Collobert et al.,) R. Collobert, K. Kavukcuoglu, et C. Farabet. Torch7 : A matlab-like environment for machine learning. Dans les actes de *BigLearn, NIPS Workshop*.

(Collobert et Weston, 2008) R. Collobert et J. Weston, 2008. A unified architecture for natural language processing : Deep neural networks with multitask learning. Dans les actes de *ICML*, 160–167. ACM.

(Collobert et al., 2011) R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, et P. Kuksa, 2011. Natural language processing (almost) from scratch. *JMLR 12*, 2493–2537.

(Comas et al., 2010) P. Comas, J. Turmo, et L. M. i Villodre, 2010. Using dependency parsing and machine learning for factoid question answering on spoken documents. Dans les actes de *INTERSPEECH*.

(Cooijmans et al., 2017) T. Cooijmans, N. Ballas, C. Laurent, **c**C. Gül**c**cehre, et A. Courville, 2017. Recurrent batch normalization. *ICLR*.

(Cybenko, 1989) G. Cybenko, 1989. Approximation by superpositions of a sigmoidal function. *Mathematics of control, signals and systems 2*(4), 303–314.

(Dagan et al., 1997) I. Dagan, Y. Karov, et D. Roth, 1997. Mistake-driven learning in text categorization. Dans les actes de *EMNLP*, 55–63.

(Dahl et al.,) G. Dahl, H. Larochelle, et R. P. Adams. Training restricted boltzmann machines on word observations. Dans les actes de *ICML*, 679–686.

(Dai et Le, 2015) A. M. Dai et Q. V. Le, 2015. Semi-supervised sequence learning. Dans les actes de *NIPS*, 3079–3087.

(Dauphin et al., 2014) Y. N. Dauphin, R. Pascanu, C. Gulcehre, K. Cho, S. Ganguli, et Y. Bengio, 2014. Identifying and attacking the saddle point problem in high-dimensional non-convex optimization. Dans les actes de *NIPS*, 2933–2941.

(De Mori et al., 2007) R. De Mori, F. Bechet, D. Hakkani-Tur, M. McTear, G. Riccardi, et G. Tur, 2007. Spoken language understanding : a survey. Dans les actes de *ASRU*.

(De Mori et al., 2008) R. De Mori, F. Bechet, D. Hakkani-Tur, M. McTear, G. Riccardi, et G. Tur, 2008. Spoken language understanding. *IEEE Signal Processing Magazine 25*(3).

(Deerwester et al., 1990) S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, et R. Harshman, 1990. Indexing by latent semantic analysis. *Journal of the American society for information science* *41*(6), 391–407.

(Dehak et al., 2011) N. Dehak, P. J. Kenny, R. Dehak, P. Dumouchel, et P. Ouellet, 2011. Front-end factor analysis for speaker verification. *IEEE Transactions on Audio, Speech, and Language Processing* *19*(4), 788–798.

(Deng et al., 2010) L. Deng, M. L. Seltzer, D. Yu, A. Acero, A.-r. Mohamed, et G. E. Hinton, 2010. Binary coding of speech spectrograms using a deep auto-encoder. Dans les actes de *Interspeech*, 1692–1695. Citeseer.

(Deveaud et al., 2013) R. Deveaud, E. SanJuan, et P. Bellot, 2013. Estimating topical context by diverging from external resources. Dans les actes de *SIGIR*.

(Doddipatla et al., 2014) R. Doddipatla, M. Hasan, et T. Hain, 2014. Speaker dependent bottleneck layer training for speaker adaptation in automatic speech recognition. Dans les actes de *Interspeech*, 2199–2203.

(Dong et al.,) T. Dong, W. Shang, et H. Zhu. An improved algorithm of bayesian text categorization. *Journal of Software* *6*(9), 1837–1843.

(Doulaty et al., 2015) M. Doulaty, O. Saz-Torralba, et T. Hain, 2015. Unsupervised domain discovery using latent dirichlet allocation for acoustic modelling in speech recognition. Dans les actes de *INTERSPEECH*.

(Dowding et al., 1994) J. Dowding, R. Moore, F. Andry, et D. Moran, 1994. Interleaving syntax and semantics in an efficient bottom-up parser. Dans les actes de *ACL*, 110–116. Association for Computational Linguistics.

(Duchi et al., 2011) J. Duchi, E. Hazan, et Y. Singer, 2011. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research* *12*(Jul), 2121–2159.

(Dumais, 1994) S. Dumais, 1994. Latent semantic indexing (lsi) and trec-2. *NIST*, 105–105.

(Dumais et Chen, 2000) S. Dumais et H. Chen, 2000. Hierarchical classification of web content. Dans les actes de *ACM SIGIR*, 256–263. ACM.

(Dumais et al., 1998) S. Dumais, J. Platt, D. Heckerman, et M. Sahami, 1998. Inductive learning algorithms and representations for text categorization. Dans les actes de *7th CIKM*, 148–155. ACM.

(Eisenstein et Barzilay, 2008) J. Eisenstein et R. Barzilay, 2008. Bayesian unsupervised topic segmentation. Dans les actes de *EMNLP*, 334–343. ACL.

(Elman, 1990) J. L. Elman, 1990. Finding structure in time. *Cognitive Science 14*, 179–211.

(Erhan et al., 2010) D. Erhan, Y. Bengio, A. Courville, P.-A. Manzagol, P. Vincent, et S. Bengio, 2010. Why does unsupervised pre-training help deep learning ? *JMLR 11*, 625–660.

(Esteve et al., 2015) Y. Esteve, M. Bouallegue, C. Lailler, M. Morchid, R. Dufour, G. Linares, D. Matrouf, et R. De Mori, 2015. Integration of word and semantic features for theme identification in telephone conversations. Dans les actes de *Natural Language Dialog Systems and Intelligent Assistants*, 223–231.

(Fautsch et Savoy, 2009) C. Fautsch et J. Savoy, 2009. Algorithmic stemmers or morphological analysis an evaluation. *JASIST 60*, 1616–1624.

(Favre et al., 2005) B. Favre, F. Béchet, et P. Nocéra, 2005. Robust named entity extraction from large spoken archives. Dans les actes de *EMNLP*, 491–498. Association for Computational Linguistics.

(Fei-Fei et Perona, 2005) L. Fei-Fei et P. Perona, 2005. A bayesian hierarchical model for learning natural scene categories. *CVPR 2*, 524–531 vol. 2.

(Feng et al., 2014) X. Feng, Y. Zhang, et J. Glass, 2014. Speech feature denoising and dereverberation via deep autoencoders for noisy reverberant speech recognition. Dans les actes de *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 1759–1763. IEEE.

(Fér et al.,) R. Fér, P. Matejka, F. Grézl, O. Plchot, et J. Cernocký. Multilingual bottleneck features for language recognition. Dans les actes de *INTERSPEECH*.

(Ferreira et al., 2015) E. Ferreira, B. Jabaian, et F. Lefèvre, 2015. Online adaptative zero-shot learning spoken language understanding using word-embedding. Dans les actes de *ICASSP*.

(Fiscus et al., 1999) J. G. Fiscus, G. R. Doddington, J. S. Garofolo, et A. F. Martin, 1999. Nist's 1998 topic detection and tracking evaluation (tdt2). Dans les actes de *EUROSPEECH*.

(Foltz et Dumais, 1992) P. Foltz et S. Dumais, 1992. Personalized information delivery : An analysis of information filtering methods. *Communications of the ACM 35*(12), 51–60.

(Fox et Roberts, 2012) C. W. Fox et S. J. Roberts, 2012. A tutorial on variational bayesian inference. *Artificial intelligence review*, 1–11.

(Friedlander et al., 2012) A. Friedlander, M. Frean, M. Johnston-Hollitt, et C. Hollitt, 2012. Latent dirichlet allocation for image segmentation and source finding in radio astronomy images. Dans les actes de *IVCNZ*.

(Friedman, 2001) J. H. Friedman, 2001. Greedy function approximation : a gradient boosting machine. *Annals of statistics*, 1189–1232.

(Gal et Ghahramani, 2015) Y. Gal et Z. Ghahramani, 2015. On modern deep learning and variational inference. Dans les actes de *Advances in Approximate Bayesian Inference workshop, NIPS*.

(Gallinari et al., 1987) P. Gallinari, Y. LeCun, S. Thiria, et F. Fogelman-Soulie, 1987. Memoires associatives distribuees. *Proceedings of COGNITIVA 87*, 93.

(Ganguly et al., 2015) D. Ganguly, D. Roy, M. Mitra, et G. J. F. Jones, 2015. Word embedding based generalized language model for information retrieval. Dans les actes de *SIGIR*.

(Garnier-Rizet et al.,) M. Garnier-Rizet, G. Adda, F. Cailliau, J. Gauvain, S. Guillemin-Lanne, L. Lamel, S. Vanni, et C. Waast-Richard. Callsurf-automatic transcription, indexing and structuration of call center conversational speech for knowledge extraction and query by content. Dans les actes de *LREC*.

(Garofolo et al., 2000) J. S. Garofolo, C. G. Auzanne, et E. M. Voorhees, 2000. The trec spoken document retrieval track : A success story. Dans les actes de *Content-Based Multimedia Information Access-Volume 1*, 1–20. LE CENTRE DE HAUTES ETUDES INTERNATIONALES D'INFORMATIQUE DOCUMENTAIRE.

(Gemello et al., 2011) R. Gemello, F. Mana, et P. D. Batzu, 2011. Topic identification from audio recordings using rich recognition results and neural network based classifiers. Dans les actes de *INTERSPEECH*.

(Glavitsch et Schäuble, 1992) U. Glavitsch et P. Schäuble, 1992. A system for retrieving speech documents. Dans les actes de *15th ACM SIGIR*, 168–176. ACM.

(Glorot et al., 2011) X. Glorot, A. Bordes, et Y. Bengio, 2011. Deep sparse rectifier neural networks. Dans les actes de *Aistats*, Volume 15, 275.

(Golub et Reinsch, 1970) G. H. Golub et C. Reinsch, 1970. Singular value decomposition and least squares solutions. *Numerische mathematik 14*(5), 403–420.

(Goodfellow et al., 2014) I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, et Y. Bengio, 2014. Generative adversarial nets. Dans les actes de *NIPS*, 2672–2680.

(Gorin et al., 1997) A. L. Gorin, G. Riccardi, et J. H. Wright, 1997. How may i help you ? *Speech communication 23*(1), 113–127.

(Gouws et al., 2015) S. Gouws, Y. Bengio, et G. S. Corrado, 2015. Bilbowa : Fast bilingual distributed representations without word alignments. Dans les actes de *ICML*.

(Graves, 2012) A. Graves, 2012. Supervised sequence labelling. Dans les actes de *Supervised Sequence Labelling with Recurrent Neural Networks*, 5–13. Springer.

(Graves et al., 2006) A. Graves, S. Fernández, F. Gomez, et J. Schmidhuber, 2006. Connectionist temporal classification : labelling unsegmented sequence data with recurrent neural networks. Dans les actes de *ICML*, 369–376. ACM.

(Graves et Jaitly,) A. Graves et N. Jaitly. Towards end-to-end speech recognition with recurrent neural networks. Dans les actes de *ICML*, Volume 14, 1764–1772.

(Grefenstette et Tapanainen, 1994) G. Grefenstette et P. Tapanainen, 1994. What is a word, what is a sentence ?: problems of tokenisation.

(Gregor et al., 2015) K. Gregor, I. Danihelka, A. Graves, D. Rezende, et D. Wierstra, 2015. Draw : A recurrent neural network for image generation. Dans F. Bach et D. Blei (Eds.), *ICML*, Volume 37 de *Proceedings of Machine Learning Research*, Lille, France, 1462–1471. PMLR.

(Grézl et al., 2007) F. Grézl, M. Karafiát, S. Kontár, et J. Cernocky, 2007. Probabilistic and bottle-neck features for lvcsr of meetings. Dans les actes de *2007 IEEE International Conference on Acoustics, Speech and Signal Processing-ICASSP'07*, Volume 4, IV–757. IEEE.

(Griffiths et Steyvers, 2002) T. Griffiths et M. Steyvers, 2002. A probabilistic approach to semantic representation. Dans les actes de *24th annual conference of the cognitive science society*, 381–386. Citeseer.

(Griffiths et Steyvers, 2004) T. L. Griffiths et M. Steyvers, 2004. Finding scientific topics. *National academy of Sciences of the United States of America 101*(Suppl 1), 5228–5235.

(Guggilla et al.,) C. Guggilla, T. Miller, et I. Gurevych. CNN- and lstm-based claim classification in online user comments. Dans les actes de *26th COLING*, 2740–2751.

(Harashima et Kurohashi, 2010) J. Harashima et S. Kurohashi, 2010. Summarizing search results using plsi. Dans les actes de *Proc. 2nd Workshop on NLPIX*, 12–20.

(Harris, 1954) Z. S. Harris, 1954. Distributional structure. *Word 10*(2-3), 146–162.

(Hazen, 2011) T. Hazen, 2011. Topic identification. *Spoken Language Understanding : Systems for Extracting Semantic Information from Speech*, 319–356.

(Hazen,) T. J. Hazen. Direct and latent modeling techniques for computing spoken document similarity. Dans les actes de *SLT*.

(Hazen et al., 2007) T. J. Hazen, F. Richardson, et A. Margolis, 2007. Topic identification from audio recordings using word and phone recognition lattices. Dans les actes de *ASRU*.

(He et al., 2015) K. He, X. Zhang, S. Ren, et J. Sun, 2015. Delving deep into rectifiers : Surpassing human-level performance on imagenet classification. Dans les actes de *ICCV*, 1026–1034.

(He et al., 2016) K. He, X. Zhang, S. Ren, et J. Sun, 2016. Deep residual learning for image recognition. Dans les actes de *CVPR*, 770–778.

(He et al., 2013) S. He, S. Wang, W. Lan, H. Fu, et Q. Ji, 2013. Facial expression recognition using deep boltzmann machine from thermal infrared images. Dans les actes de *ACII*, 239–244. IEEE.

(Heinrich,) G. Heinrich. Parameter estimation for text analysis. *Web : http ://www. arbylon. net/publications/text-est. pdf*.

(Hennig et Labor, 2009) L. Hennig et D. Labor, 2009. Topic-based multi-document summarization with probabilistic latent semantic analysis. Dans les actes de *RANLP*, 144–149.

(Hinton et al., 2012) G. Hinton, L. Deng, D. Yu, G. E. Dahl, A. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, et B. Kingsbury, 2012. Deep neural networks for acoustic modeling in speech recognition : The shared views of four research groups. *IEEE Signal Process. Mag. 29*.

(Hinton, 2002) G. E. Hinton, 2002. Training products of experts by minimizing contrastive divergence. *Neural computation 14*(8), 1771–1800.

(Hinton et al., 2006) G. E. Hinton, S. Osindero, et Y.-W. Teh, 2006. A fast learning algorithm for deep belief nets. *Neural computation 18*(7), 1527–1554.

(Hinton et Salakhutdinov, 2006) G. E. Hinton et R. R. Salakhutdinov, 2006. Reducing the dimensionality of data with neural networks. *Science 313*(5786), 504–507.

(Hochreiter et Schmidhuber, 1997) S. Hochreiter et J. Schmidhuber, 1997. Long short-term memory. *Neural computation 9*(8), 1735–1780.

(Hofmann, 1999a) T. Hofmann, 1999a. Probabilistic latent semantic analysis. Dans les actes de *Proceedings of UAI*, 21.

(Hofmann, 1999b) T. Hofmann, 1999b. Probabilistic latent semantic analysis. Dans les actes de *UAI*, 21.

(Hofmann, 1999c) T. Hofmann, 1999c. Probabilistic latent semantic indexing. Dans les actes de *22nd annual international ACM SIGIR*, 50–57.

(Hopfield, 1982) J. J. Hopfield, 1982. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the national academy of sciences 79*(8), 2554–2558.

(Hu et al., 2014) P. Hu, W. Liu, W. Jiang, et Z. Yang, 2014. Latent topic model for audio retrieval. *Pattern Recognition 47*, 1138–1143.

(Hu et al., 2009) X. Hu, R. Isotani, et S. Nakamura, 2009. Spoken document retrieval using topic models. Dans les actes de *IUCS*.

(Hull et al.,) D. A. Hull et al. Stemming algorithms : A case study for detailed evaluation. *JASIS 47*(1), 70–84.

(Ioffe et Szegedy, 2015) S. Ioffe et C. Szegedy, 2015. Batch normalization : Accelerating deep network training by reducing internal covariate shift. Dans les actes de *ICML*, 448–456.

(Ittner et al., 1995) D. J. Ittner, D. D. Lewis, et D. D. Ahn, 1995. Text categorization of low quality images. Dans les actes de *SDAIR*, 301–315. Citeseer.

(John et al.,) G. H. John, R. Kohavi, K. Pfleger, et al. Irrelevant features and the subset selection problem. Dans les actes de *Machine learning : proceedings of the 11th international conference*, 121–129.

(Jones et al., 1996) G. J. Jones, J. T. Foote, K. S. Jones, et S. J. Young, 1996. Retrieving spoken documents by combining multiple index sources. Dans les actes de *19th ACM SIGIR*, 30–38. ACM.

(Jordan, 1986) M. Jordan, 1986. Serial order : A parallel. *Distributed Processing Approach, Advances in psychology 121*(8604), 471.

(Juang et al., 2004) B. H. Juang, L. R. Rabiner, et A. Graham, 2004. Automatic speech recognition – a brief history of the technology development.

(Karpathy et al.,) A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, et L. Fei-Fei. Large-scale video classification with convolutional neural networks. Dans les actes de *CVPR*, 1725–1732.

(Kenny, 2005) P. Kenny, 2005. Joint factor analysis of speaker and session variability : Theory and algorithms. *CRIM, Montreal,(Report) CRIM-06/08-13*.

(Khoo et al., 2006) A. Khoo, Y. Marom, et D. Albrecht, 2006. Experiments with sentence classification. Dans les actes de *Australasian language technology workshop*, 18–25.

(Kim et al., 2017) S. Kim, L. F. D'Haro, R. E. Banchs, J. D. Williams, et M. Henderson, 2017. The fourth dialog state tracking challenge. Dans les actes de *Dialogues with Social Robots*, 435–449.

(Kim et al., 2010) S. Kim, P. Georgiou, et S. Narayanan, 2010. Supervised acoustic topic model for unstructured audio information retrieval.

(Kim, 2014) Y. Kim, 2014. Convolutional neural networks for sentence classification.

(Kim et al., 2013) Y. Kim, H. Lee, et E. M. Provost, 2013. Deep learning for robust feature generation in audiovisual emotion recognition. Dans les actes de *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, 3687–3691. IEEE.

(Kingma et Ba,) D. Kingma et J. Ba. Adam : A method for stochastic optimization. *ICLR*.

(Kingma et Ba, 2015) D. P. Kingma et J. Ba, 2015. Adam : A method for stochastic optimization. *ICLR*.

(Kingma et Welling, 2014) D. P. Kingma et M. Welling, 2014. Auto-encoding variational bayes. *ICLR 2014*.

(Koesdwiady et al., 2016) A. Koesdwiady, R. Soua, F. Karray, et M. S. Kamel, 2016. Recent trends in driver safety monitoring systems : State of the art and challenges. *IEEE Transactions on Vehicular Technology*.

(Kohonen, 1982) T. Kohonen, 1982. Self-organized formation of topologically correct feature maps. *Biological cybernetics 43*(1), 59–69.

(Kottur et al., 2016) S. Kottur, R. Vedantam, J. M. Moura, et D. Parikh, 2016. Visual word2vec (vis-w2v) : Learning visually grounded word embeddings using abstract scenes. Dans les actes de *CVPR*, 4985–4994.

(Krizhevsky et al.,) A. Krizhevsky, I. Sutskever, et G. E. Hinton. Imagenet classification with deep convolutional neural networks. Dans les actes de *NIPS*, 1097–1105.

(Kulkarni et al.,) T. D. Kulkarni, W. F. Whitney, P. Kohli, et J. Tenenbaum. Deep convolutional inverse graphics network. Dans les actes de *NIPS*, 2539–2547.

(Lagus et Kuusisto, 2002) K. Lagus et J. Kuusisto, 2002. Topic identification in natural language dialogues using neural networks. Dans les actes de *SIGdial Workshop on Discourse and Dialogue*, Philadelphia, Pennsylvania, USA, 95–102. Association for Computational Linguistics.

(Landauer et Dumais, 1997) T. Landauer et S. Dumais, 1997. A solution to plato's problem : The latent semantic analysis theory of acquisition, induction, and representation of knowledge. *Psychological review 104*(2), 211.

(Landauer et al., 1997) T. Landauer, D. Laham, B. Rehder, et M. Schreiner, 1997. How well can passage meaning be derived without using word order ? a comparison of latent semantic analysis and humans. Dans les actes de *19th of the Cognitive Science Society*, Volume 10, 412.

(Lane et al., 2007) I. R. Lane, T. Kawahara, T. Matsui, et S. Nakamura, 2007. Out-of-domain utterance detection using classification confidences of multiple topics. *IEEE Transactions on Audio, Speech, and Language Processing 15*, 150–161.

(Larochelle et Bengio, 2008) H. Larochelle et Y. Bengio, 2008. Classification using discriminative restricted boltzmann machines. Dans les actes de *ICML*, 536–543. ACM.

(Le Cun, 1987) Y. Le Cun, 1987. *Modèles connexionnistes de l'apprentissage*. Thèse de Doctorat.

(Lecouteux et al., 2009) B. Lecouteux, G. Linares, et B. Favre, 2009. Combined low level and high level features for out-of-vocabulary word detection. Dans les actes de *Tenth Annual Conference of the International Speech Communication Association*.

(LeCun et al.,) Y. LeCun, B. Boser, J. Denker, D. Henderson, R. Howard, W. Hubbard, et L. Jackel. Handwritten digit recognition with a back-propagation network, 1989. Dans les actes de *Neural Information Processing Systems (NIPS)*.

(LeCun et al., 1998) Y. LeCun, L. Bottou, Y. Bengio, et P. Haffner, 1998. Gradient-based learning applied to document recognition. *Proceedings of the IEEE 86*(11), 2278–2324.

(Lee et Osindero, 2016) C.-Y. Lee et S. Osindero, 2016. Recursive recurrent nets with attention modeling for ocr in the wild. Dans les actes de *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

(Lee et al., 2009) H. Lee, R. Grosse, R. Ranganath, et A. Y. Ng, 2009. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. Dans les actes de *ICML*, 609–616. ACM.

(Lee et al., 2015a) L.-S. Lee, J. R. Glass, H. yi Lee, et C. an Chan, 2015a. Spoken content retrieval - beyond cascading speech recognition with text retrieval. *IEEE-ACM Trans. Audio, Speech and Language Processing 23*, 1389–1420.

(Lee et al., 2015b) L.-S. Lee, J. R. Glass, H. yi Lee, et C. an Chan, 2015b. Spoken content retrieval - beyond cascading speech recognition with text retrieval. *IEEE-ACM Trans. Audio, Speech and Language Processing 23*, 1389–1420.

(Lei et al.,) Y. Lei, L. Ferrer, A. Lawson, M. McLaren, et N. Scheffer. Application of convolutional neural networks to language identification in noisy conditions. *Proc. Odyssey-14, Joensuu, Finland*.

(Levy et Goldberg, 2014a) O. Levy et Y. Goldberg, 2014a. Dependency-based word embeddings. *ACL*, 302–308.

(Levy et Goldberg, 2014b) O. Levy et Y. Goldberg, 2014b. Neural word embedding as implicit matrix factorization. Dans les actes de *NIPS*, 2177–2185.

(Levy et al., 2015) O. Levy, Y. Goldberg, et I. Dagan, 2015. Improving distributional similarity with lessons learned from word embeddings. *Transactions of the ACL 3*, 211–225.

(Lewis, 1992) D. D. Lewis, 1992. An evaluation of phrasal and clustered representations on a text categorization task. Dans les actes de *15th ACM SIGIR*, 37–50.

(Lewis et Gale, 1994) D. D. Lewis et W. A. Gale, 1994. A sequential algorithm for training text classifiers. Dans les actes de *17th ACM SIGIR*, 3–12.

(Li et al., 2010) F. Li, M. Huang, et X. Zhu, 2010. Sentiment analysis with global topics and local dependency. Dans les actes de *AAAI*.

(Li et McCallum,) W. Li et A. McCallum. Pachinko allocation : Dag-structured mixture models of topic correlations. Dans les actes de *ICML*.

(Liao et al., 2014) X. Liao, Q. Jiang, W. Zhang, et K. Zhang, 2014. Bimodal latent dirichlet allocation for text and image. *2014 4th IEEE International Conference on Information Science and Technology*, 736–739.

(Lienhart et al., 2009) R. Lienhart, S. Romberg, et E. Hörster, 2009. Multilayer plsa for multimodal image retrieval. Dans les actes de *ACM CIVR*, 9. ACM.

(Linarès et al., 2007) G. Linarès, P. Nocéra, D. Massonie, et D. Matrouf, 2007. The lia speech recognition system : from 10xrt to 1xrt. Dans les actes de *Text, Speech and Dialogue*, 302–308. Springer.

(Ling et al., 2015) W. Ling, Y. Tsvetkov, S. Amir, R. Fermandez, C. Dyer, A. W. Black, I. Trancoso, et C. Lin, 2015. Not all contexts are created equal : Better word representations with variable attention. Dans les actes de *EMNLP*, 1367–1372.

(Liu et Inkpen, 2015) J. Liu et D. Inkpen, 2015. Estimating user location in social media with stacked denoising auto-encoders. Dans les actes de *Proceedings of NAACL-HLT*, 201–210.

(Lu et al.,) X. Lu, Y. Tsao, S. Matsuda, et C. Hori. Speech enhancement based on deep denoising autoencoder. Dans les actes de *Interspeech*, 436–440.

(Lukins et al., 2008) S. K. Lukins, N. A. Kraft, et L. H. Etzkorn, 2008. Source code retrieval for bug localization using latent dirichlet allocation. Dans les actes de *WCRE*.

(Luong et al., 2015) T. Luong, H. Pham, et C. D. Manning, 2015. Effective approaches to attention-based neural machine translation. Dans les actes de *EMNLP*.

(Maas et al., 2013) A. L. Maas, A. Y. Hannun, et A. Y. Ng, 2013. Rectifier nonlinearities improve neural network acoustic models. Dans les actes de *Proc. ICML*, Volume 30.

(Maria et al., 2015) J. Maria, J. Amaro, G. Falcao, et L. A. Alexandre, 2015. Stacked autoencoders using low-power accelerated architectures for object recognition in autonomous systems. *Neural Processing Letters*, 1–14.

(Matejka et al., 2014) P. Matejka, L. Zhang, T. Ng, H. S. Mallidi, O. Glembek, J. Ma, et B. Zhang, 2014. Neural network bottleneck features for language identification. *Proc. IEEE Odyssey*, 299–304.

(Mcauliffe et Blei, 2008) J. D. Mcauliffe et D. M. Blei, 2008. Supervised topic models. Dans J. C. Platt, D. Koller, Y. Singer, et S. T. Roweis (Eds.), *NIPS*, 121–128. Curran Associates, Inc.

(McCulloch et Pitts, 1943) W. S. McCulloch et W. Pitts, 1943. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics* *5*(4), 115–133.

(Mcnamee et Mayfield, 2004) P. Mcnamee et J. Mayfield, 2004. Character n-gram tokenization for european language text retrieval. *Information retrieval* *7*(1), 73–97.

(McTear, 1998) M. F. McTear, 1998. Modelling spoken dialogues with state transition diagrams : experiences with the cslu toolkit. *development* *5*(7).

(Melamed et Gilbert, 2011) I. Melamed et M. Gilbert, 2011. Speech analytics. *Spoken Language Understanding : Systems for Extracting Semantic Information from Speech*, 397–416.

(Mesnil et al., 2015) G. Mesnil, Y. Dauphin, K. Yao, Y. Bengio, L. Deng, D. Hakkani-Tur, X. He, L. Heck, G. Tur, D. Yu, et al., 2015. Using recurrent neural networks for slot filling in spoken language understanding. *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)* *23*(3), 530–539.

(Mikolov et al., 2013a) T. Mikolov, K. Chen, G. S. Corrado, et J. Dean, 2013a. Efficient estimation of word representations in vector space. *CoRR abs/1301.3781*.

(Mikolov et al., 2013b) T. Mikolov, G. Corrado, K. Chen, et J. Dean, 2013b. Efficient estimation of word representations in vector space. *ICLR*, 1–12.

(Mikolov et al., 2011) T. Mikolov, A. Deoras, D. Povey, L. Burget, et J. **v**Cernocký, 2011. Strategies for training large scale neural network language models. *2011 IEEE Workshop on Automatic Speech Recognition and Understanding, ASRU 2011, Proceedings*, 196–201.

(Mikolov et al., 2010) T. Mikolov, M. Karafiát, L. Burget, J. Cernocký, et S. Khudanpur, 2010. Recurrent neural network based language model. Dans les actes de *INTERSPEECH*.

(Mikolov et al., 2013a) T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, et J. Dean, 2013a. Distributed representations of words and phrases and their compositionality. Dans les actes de *NIPS*, 3111–3119.

(Mikolov et al., 2013b) T. Mikolov, W.-t. Yih, et G. Zweig, 2013b. Linguistic regularities in continuous space word representations. Dans les actes de *HLT-NAACL*, 746–751.

(Mohamed et al., 2009) A. Mohamed, G. Dahl, et G. Hinton, 2009. Deep belief networks for phone recognition,[in :] nips workshop on deep learning for speech recognition and related applications. *1*(9), 39.

(Morchid et al., 2014) M. Morchid, M. Bouallegue, R. Dufour, G. Linarès, D. Matrouf, et R. De Mori, 2014. I-vector based approach to compact multi-granularity topic spaces representation of textual documents. Dans les actes de *the Conference of Empirical Methods on Natural Lnguage Processing (EMNLP) 2014*. SIGDAT.

(Morchid et al., 2015) M. Morchid, M. Bouallegue, R. Dufour, G. Linarès, D. Matrouf, et R. De Mori, 2015. Compact multiview representation of documents based on the total variability space. *Audio, Speech, and Language Processing, IEEE/ACM Transactions on* *23*(8), 1295–1308.

(Morchid et al., 2014b) M. Morchid, R. Dufour, P.-M. Bousquet, M. Bouallegue, G. Linarès, et R. De Mori, 2014b. Improving dialogue classification using a topic space representation and a gaussian classifier based on the decision rule. Dans les actes de *ICASSP*.

(Morchid et al., 2014c) M. Morchid, R. Dufour, P.-M. Bousquet, M. Bouallegue, G. Linares, et R. De Mori, 2014c. Improving dialogue classification using a topic space representation and a gaussian classifier based on the decision rule. Dans les actes de *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 126–130. IEEE.

(Morchid et al., 2014a) M. Morchid, R. Dufour, et G. Linarès, 2014a. A lda-based topic classification approach from highly imperfect automatic transcriptions. Dans les actes de *LREC*.

(Morchid et al.,) M. Morchid, R. Dufour, G. Linarès, et Y. Hamadi. Latent topic model based representations for a robust theme identification of highly imperfect automatic transcriptions. Dans les actes de *International Conference on Intelligent Text Processing and Computational Linguistics (CICLing) 2015*.

(Moschitti et al., 2014) Moschitti, A., B. Pang, et W. Daelemans (Eds.), 2014. *EMNLP, A meeting of SIGDAT, a Special Interest Group of the ACL*. ACL.

(Mrva et Woodland, 2004) D. Mrva et P. C. Woodland, 2004. A plsa-based language model for conversational telephone speech. Dans les actes de *INTERSPEECH*. Citeseer.

(Muneeb et al., 2015) T. Muneeb, S. K. Sahu, et A. Anand, 2015. Evaluating distributed word representations for capturing semantics of biomedical concepts. *Proceedings of ACL-IJCNLP*, 158.

(Nair et Hinton, 2010) V. Nair et G. E. Hinton, 2010. Rectified linear units improve restricted boltzmann machines. Dans les actes de *ICML*, 807–814.

(Nakano et al., 2014) T. Nakano, K. Yoshii, et M. Goto, 2014. Vocal timbre analysis using latent dirichlet allocation and cross-gender vocal timbre similarity. *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 5202–5206.

(Nasrabadi et Feng, 1988) N. M. Nasrabadi et Y. Feng, 1988. Vector quantization of images based upon the kohonen self-organizing feature maps. Dans les actes de *IEEE 1988 International Conference on Neural Networks*.

(Natarajan et al., 2002) P. Natarajan, R. Prasad, R. M. Schwartz, et J. Makhoul, 2002. A scalable architecture for directory assistance automation. Dans les actes de *Acoustics, Speech, and Signal Processing (ICASSP), 2002 IEEE International Conference on*, Volume 1, I–21. IEEE.

(Neal, 2001) R. M. Neal, 2001. Annealed importance sampling. *Statistics and computing 11*(2), 125–139.

(Newman et al., 2007) D. Newman, A. U. Asuncion, P. Smyth, et M. Welling, 2007. Distributed inference for latent dirichlet allocation. Dans les actes de *NIPS*, Volume 20, 1081–1088.

(Ngiam et al.,) J. Ngiam, A. Khosla, M. Kim, J. Nam, H. Lee, et A. Y. Ng. Multimodal deep learning. Dans les actes de *ICML*, 689–696.

(Parviainen, 2010) E. Parviainen, 2010. Dimension reduction for regression with bottleneck neural networks. Dans les actes de *International Conference on Intelligent Data Engineering and Automated Learning*, 37–44.

(Pedregosa et al., 2011) F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, et E. Duchesnay, 2011. Scikit-learn : Machine learning in Python. *Journal of Machine Learning Research 12*, 2825–2830.

(Pennington et al., 2014) J. Pennington, R. Socher, et C. D. Manning, 2014. Glove : Global vectors for word representation. Dans les actes de *Empirical Methods in Natural Language Processing, EMNLP*, 1532–1543.

(Pham et al., 2007) T.-T. Pham, N. E. Maillot, J.-H. Lim, et J.-P. Chevallet, 2007. Latent semantic fusion model for image retrieval and annotation. Dans les actes de *16th ACM CIKM*, 439–444. ACM.

(Popescu et al., 2011) A.-M. Popescu, M. Pennacchiotti, et D. Paranjpe, 2011. Extracting events and event descriptions from twitter. Dans les actes de *20th international conference companion on World wide web*, 105–106. ACM.

(Porteous et al., 2008) I. Porteous, D. Newman, A. Ihler, A. Asuncion, P. Smyth, et M. Welling, 2008. Fast collapsed gibbs sampling for latent dirichlet allocation. Dans les actes de *14th ACM SIGKDD*, 569–577.

(Porter, 1980) M. F. Porter, 1980. An algorithm for suffix stripping. *Program 14*(3), 130–137.

(Price, 1990) P. Price, 1990. Evaluation of spoken language systems : The atis domain. Dans les actes de *3rd DARPA Speech and Natural Language Workshop*, 91–95. Morgan Kaufmann.

(Purver, 2011) M. Purver, 2011. Topic segmentation. *Spoken Language Understanding : Systems for Extracting Semantic Information from Speech*, 291–317.

(Rama et al., 2008) G. M. Rama, S. Sarkar, et K. Heafield, 2008. Mining business topics in source code using latent dirichlet allocation. Dans les actes de *ISEC*.

(Ramage et al., 2009) D. Ramage, D. L. W. Hall, R. Nallapati, et C. D. Manning, 2009. Labeled lda : A supervised topic model for credit attribution in multi-labeled corpora. Dans les actes de *EMNLP*.

(Rasiwasia et Vasconcelos, 2013) N. Rasiwasia et N. Vasconcelos, 2013. Latent dirichlet allocation models for image classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence 35*, 2665–2679.

(Reed et al., 2016) S. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, et H. Lee, 2016. Generative adversarial text to image synthesis. Dans les actes de *ICML*, Volume 3.

(Ren et al., 2016) B. Ren, L. Wang, L. Lu, Y. Ueda, et A. Kai, 2016. Combination of bottleneck feature extraction and dereverberation for distant-talking speech recognition. *Multimedia Tools and Applications 75*(9), 5093–5108.

(Reynolds et Rose, 1995) D. A. Reynolds et R. C. Rose, 1995. Robust text-independent speaker identification using gaussian mixture speaker models. *IEEE Transactions on Speech and Audio Processing 3*(1), 72–83.

(Ritter et al., 2011) A. Ritter, S. Clark, Mausam, et O. Etzioni, 2011. Named entity recognition in tweets : An experimental study. Dans les actes de *EMNLP*.

(Robinson et Fallside, 1987) A. Robinson et F. Fallside, 1987. *The utility driven dynamic error propagation network*. University of Cambridge Department of Engineering.

(Rosen-Zvi et al., 2004) M. Rosen-Zvi, T. Griffiths, M. Steyvers, et P. Smyth, 2004. The author-topic model for authors and documents. Dans les actes de *20th UAI*, 487–494.

(Rosenblatt, 1958) F. Rosenblatt, 1958. The perceptron : A probabilistic model for information storage and organization in the brain. *Psychological review 65*(6), 386.

(Roth et al., 2015) H. R. Roth, J. Yao, L. Lu, J. Stieger, J. E. Burns, et R. M. Summers, 2015. Detection of sclerotic spine metastases via random aggregation of deep convolutional neural network classifications. Dans les actes de *Recent Advances in Computational Methods and Clinical Applications for Spine Imaging*, 3–12.

(Rubino, 2011) R. Rubino, 2011. *Traduction automatique statistique et adaptation à un domaine spécialisé*. Thèse de Doctorat, Université d'Avignon et des Pays de Vaucluse.

(Rumelhart et al., 1985) D. E. Rumelhart, G. E. Hinton, et R. J. Williams, 1985. Learning internal representations by error propagation. Rapport technique.

(Sainath et al., 2012) T. N. Sainath, B. Kingsbury, et B. Ramabhadran, 2012. Auto-encoder bottleneck features using deep belief networks. Dans les actes de *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 4153–4156.

(Sainath et al., 2011) T. N. Sainath, B. Kingsbury, B. Ramabhadran, P. Fousek, P. Novak, et A.-r. Mohamed, 2011. Making deep belief networks effective for large vocabulary continuous speech recognition. Dans les actes de *Automatic Speech Recognition and Understanding (ASRU), 2011 IEEE Workshop on*, 30–35. IEEE.

(Salakhutdinov et Hinton, 2009) R. Salakhutdinov et G. E. Hinton, 2009. Deep boltzmann machines. Dans les actes de *AISTATS*, Volume 1, 3.

(Salakhutdinov et al., 2007) R. Salakhutdinov, A. Mnih, et G. Hinton, 2007. Restricted boltzmann machines for collaborative filtering. Dans les actes de *ICML*, 791–798. ACM.

(Salman et al., 2015) A. G. Salman, B. Kanigoro, et Y. Heryadi, 2015. Weather forecasting using deep learning techniques. Dans les actes de *2015 International Conference on Advanced Computer Science and Information Systems (ICACSIS)*.

(Salton, 1989) G. Salton, 1989. Automatic text processing : the transformation. *Analysis and Retrieval of Information by Computer*.

(Salton et Buckley, 1988) G. Salton et C. Buckley, 1988. Term-weighting approaches in automatic text retrieval* 1. *Information processing & management* *24*(5), 513–523.

(Sarikaya et al., 2014) R. Sarikaya, G. E. Hinton, et A. Deoras, 2014. Application of deep belief networks for natural language understanding. *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)* *22*(4), 778–784.

(Sarroff et Casey,) A. M. Sarroff et M. Casey. Musical audio synthesis using autoencoding neural nets.

(Schofield et Mimno, 2016) A. Schofield et D. M. Mimno, 2016. Comparing apples to apple : The effects of stemmers on topic models. *TACL 4*.

(Schuster et Paliwal, 1997) M. Schuster et K. K. Paliwal, 1997. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing* *45*(11), 2673–2681.

(Schwartz et al., 1997) R. Schwartz, S. Miller, D. Stallard, et J. Makhoul, 1997. Hidden understanding models for statistical sentence understanding. Dans les actes de *Acoustics, Speech, and Signal Processing, 1997. ICASSP-97., 1997 IEEE International Conference on*, Volume 2, 1479–1482. IEEE.

(Seide et al., 2011) F. Seide, G. Li, et D. Yu, 2011. Conversational speech transcription using context-dependent deep neural networks. Dans les actes de *Interspeech*, 437–440.

(Seneff, 1992) S. Seneff, 1992. Robust parsing for spoken language systems. Dans les actes de *Acoustics, Speech, and Signal Processing, 1992. ICASSP-92., 1992 IEEE International Conference on*, Volume 1, 189–192. IEEE.

(Sheikh et al., 2015) I. Sheikh, I. Illina, et D. Fohr, 2015. Study of Entity-Topic Models for OOV Proper Name Retrieval. Dans les actes de *Interspeech 2015*, Dresden, Germany.

(Sheikh et al., 2016) I. Sheikh, I. Ulina, D. Fohr, et G. Linares, 2016. Document level semantic context for retrieving oov proper names. Dans les actes de *Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on*, 6050–6054. IEEE.

(Simonyan et Zisserman, 2016) K. Simonyan et A. Zisserman, 2016. Very deep convolutional networks for large-scale image recognition. *ICLR*.

(Singhal et al., 1996) A. Singhal, G. Salton, M. Mitra, et C. Buckley, 1996. Document length normalization. *Information Processing & Management 32*(5), 619–633.

(Smaragdis et al., 2006) P. Smaragdis, B. Raj, et M. Shashanka, 2006. A probabilistic latent variable model for acoustic modeling. *Advances in models for acoustic processing, NIPS 148*, 8–1.

(Smolensky,) P. Smolensky. Parallel distributed processing : Explorations in the microstructure of cognition, vol. 1. chapter information processing in dynamical systems : Foundations of harmony theory. *MIT Press, Cambridge, MA, USA 15*, 18.

(Song et Xiao, 2016) S. Song et J. Xiao, 2016. Deep sliding shapes for amodal 3d object detection in rgb-d images. Dans les actes de *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

(Soutner et Müller, 2015) D. Soutner et L. Müller, 2015. On continuous space word representations as input of lstm language model. Dans les actes de *International Conference on Statistical Language and Speech Processing*, 267–274. Springer.

(Sparck Jones, 1972) K. Sparck Jones, 1972. A statistical interpretation of term specificity and its application in retrieval. *Journal of documentation 28*(1), 11–21.

(Srivastava et al., 2014) N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, et R. Salakhutdinov, 2014. Dropout : a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research 15*(1), 1929–1958.

(Srivastava et al., 2013) N. Srivastava, R. Salakhutdinov, et G. Hinton, 2013. Modeling documents with a deep boltzmann machine. Dans les actes de *29th UAI*, UAI'13, Arlington, Virginia, United States, 616–624. AUAI Press.

(Srivastava et Salakhutdinov, 2012) N. Srivastava et R. R. Salakhutdinov, 2012. Multimodal learning with deep boltzmann machines. Dans les actes de *NIPS*, 2222–2230.

(Story, 1996) R. Story, 1996. An explanation of the effectiveness of latent semantic indexing by means of a bayesian regression model. *Information Processing & Management 32*(3), 329–344.

(Su et al., 2016) J. Su, O. Boydell, D. Greene, et G. Lynch, 2016. Topic stability over noisy sources. Dans les actes de *Workshop on Noisy User-generated Text (W-NUT)*.

(Sutskever et al., 2014) I. Sutskever, O. Vinyals, et Q. V. Le, 2014. Sequence to sequence learning with neural networks. Dans les actes de *NIPS*, 3104–3112.

(Szegedy et al., 2015) C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, et A. Rabinovich, 2015. Going deeper with convolutions. Dans les actes de *IEEE CVPR*, 1–9.

(Tai et al., 2015) K. S. Tai, R. Socher, et C. D. Manning, 2015. Improved semantic representations from tree-structured long short-term memory networks. Dans les actes

de *Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL*, 1556–1566.

(Takashima et al., 2016) Y. Takashima, R. Aihara, T. Takiguchi, Y. Ariki, N. Mitani, K. Omori, et K. Nakazono, 2016. Audio-visual speech recognition using bimodal-trained bottleneck features for a person with severe hearing loss. *Interspeech 2016*, 277–281.

(Tan et Eswaran, 2008) C. C. Tan et C. Eswaran, 2008. Performance comparison of three types of autoencoder neural networks. Dans les actes de *Modeling & Simulation, 2008. AICMS 08. Second Asia International Conference on*, 213–218.

(Tan et Mak,) Z. Tan et M.-W. Mak. Bottleneck features from snr-adaptive denoising deep classifier for speaker identification. Dans les actes de *2015 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA)*.

(Teh et al., 2003) Y. W. Teh, M. I. Jordan, M. J. Beal, D. M. Blei, et Q. Fu, 2003. Hierarchical dirichlet processes.

(Teh et al., 2006) Y. W. Teh, D. Newman, et M. Welling, 2006. A collapsed variational bayesian inference algorithm for latent dirichlet allocation. Dans les actes de *NIPS*, Volume 6, 1378–1385.

(Tian et al., 2015) Y. Tian, L. He, et J. Liu, 2015. Stacked bottleneck features for speaker verification. Dans les actes de *2015 IEEE China Summit and International Conference on Signal and Information Processing (ChinaSIP)*.

(Tieleman et Hinton, 2012) T. Tieleman et G. Hinton, 2012. Lecture 6.5-rmsprop : Divide the gradient by a running average of its recent magnitude. *COURSERA : Neural networks for machine learning* *4*(2).

(Tür et al., 2008) G. Tür, A. Stolcke, L. L. Voss, J. Dowding, B. Favre, R. Fernández, M. Frampton, M. W. Frandsen, C. Frederickson, M. Graciarena, D. Z. Hakkani-Tür, D. Kintzing, K. Leveque, S. Mason, J. Niekrasz, S. Peters, M. Purver, K. Riedhammer, E. Shriberg, J. Tien, D. Vergyri, et F. Yang, 2008. The calo meeting speech recognition and understanding system. Dans les actes de *SLT*.

(Turmo et al., 2008) J. Turmo, P. R. Comas, S. Rosset, L. Lamel, N. Moreau, et D. Mostefa, 2008. Overview of qast 2008. Dans les actes de *Workshop of the Cross-Language Evaluation Forum for European Languages*, 314–324. Springer.

(Vincent et al., 2008a) P. Vincent, H. Larochelle, Y. Bengio, et P.-A. Manzagol, 2008a. Extracting and composing robust features with denoising autoencoders. Dans les actes de *ICML*, 1096–1103. ACM.

(Vincent et al., 2008b) P. Vincent, H. Larochelle, Y. Bengio, et P.-A. Manzagol, 2008b. Extracting and composing robust features with denoising autoencoders. Dans les actes de *ICML*, 1096–1103.

(Vincent et al., 2010) P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, et P.-A. Manzagol, 2010. Stacked denoising autoencoders : Learning useful representations in a deep network with a local denoising criterion. *JMLR 11*, 3371–3408.

(Vinyals et Ravuri, 2011) O. Vinyals et S. V. Ravuri, 2011. Comparing multilayer perceptron to deep belief network tandem features for robust asr. Dans les actes de *2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 4596–4599. IEEE.

(Wang et al., 2016) J. Wang, L.-C. Yu, K. R. Lai, et X. jie Zhang, 2016. Dimensional sentiment analysis using a regional cnn-lstm model. Dans les actes de *ACL*.

(Wang et al., 2016) P. Wang, B. Xu, J. Xu, G. Tian, C.-L. Liu, et H. Hao, 2016. Semantic expansion using word embedding clustering and convolutional neural network for improving short text classification. *Neurocomputing 174*, 806–814.

(Wang et McCallum, 2006) X. Wang et A. McCallum, 2006. Topics over time : a non-markov continuous-time model of topical trends. Dans les actes de *KDD*.

(Wang et Acero, 2006) Y.-Y. Wang et A. Acero, 2006. Discriminative models for spoken language understanding. Dans les actes de *Interspeech*.

(Wang et al., 2002) Y.-Y. Wang, A. Acero, C. Chelba, B. J. Frey, et L. Wong, 2002. Combination of statistical and rule-based approaches for spoken language understanding. Dans les actes de *INTERSPEECH*.

(Wang et al., 2008) Y.-Y. Wang, D. Yu, Y.-C. Ju, et A. Acero, 2008. An introduction to voice search. *IEEE Signal Processing Magazine 25*(3).

(Ward et Issar, 1994) W. Ward et S. Issar, 1994. Recent improvements in the cmu spoken language understanding system. Dans les actes de *ACL : HLT*, 213–216. Association for Computational Linguistics.

(Wei et Croft, 2006a) X. Wei et W. B. Croft, 2006a. Lda-based document models for ad-hoc retrieval. Dans les actes de *SIGIR*.

(Wei et Croft, 2006b) X. Wei et W. B. Croft, 2006b. Lda-based document models for ad-hoc retrieval. Dans les actes de *29th ACM SIGIR*, 178–185. ACM.

(Werbos, 1974) P. Werbos, 1974. Beyond regression : New tools for predictions and analysis in the behavioral science. cambridge, ma, itd.

(Werbos, 1988) P. J. Werbos, 1988. Generalization of backpropagation with application to a recurrent gas market model. *Neural networks 1*(4), 339–356.

(WhyeTeh et Hinton, 2001) V. WhyeTeh et G. E. Hinton, 2001. Rate-coded restricted boltzmann machines for face recognition.

(Widrow et al., 1960) B. Widrow, M. E. Hoff, et al., 1960. Adaptive switching circuits. Dans les actes de *IRE WESCON convention record*, Volume 4, 96–104. New York.

(Willett, 2006) P. Willett, 2006. The porter stemming algorithm : then and now. *Program 40*(3), 219–223.

(Williams, 2008) J. D. Williams, 2008. The best of both worlds : unifying conventional dialog systems and pomdps. Dans les actes de *INTERSPEECH*, 1173–1176.

(Wintrode et Kulp, 2009) J. Wintrode et S. Kulp, 2009. Confidence-based techniques for rapid and robust topic identification of conversational telephone speech. Dans les actes de *Proc. Interspeech*.

(Wolf et al., 2014) L. Wolf, Y. Hanani, K. Bar, et N. Dershowitz, 2014. Joint word2vec networks for bilingual semantic representations. *International Journal of Computational Linguistics and Applications 5*(1), 27–44.

(Wu et al., 2015) Q. Wu, H. Zhang, S. Liu, et X. Cao, 2015. Multimedia analysis with deep learning. Dans les actes de *2015 IEEE International Conference on Multimedia Big Data*.

(Wu et al.,) Z. Wu, C. Valentini-Botinhao, O. Watts, et S. King. Deep neural networks employing multi-task learning and stacked bottleneck features for speech synthesis. Dans les actes de *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*.

(Xiong et al., 2017a) W. Xiong, J. Droppo, X. Huang, F. Seide, M. L. Seltzer, A. Stolcke, D. Yu, et G. Zweig, 2017a. Toward human parity in conversational speech recognition. *IEEE/ACM Transactions on Audio, Speech, and Language Processing 25*(12), 2410–2423.

(Xiong et al., 2017b) W. Xiong, J. Droppo, X. Huang, F. Seide, M. L. Seltzer, A. Stolcke, D. Yu, et G. Zweig, 2017b. Toward human parity in conversational speech recognition. *IEEE/ACM Transactions on Audio, Speech, and Language Processing 25*(12), 2410–2423.

(Xu et al., 2015b) J. Xu, L. Xiang, Q. Liu, H. Gilmore, J. Wu, J. Tang, et A. Madabhushi, 2015b. Stacked sparse autoencoder (ssae) for nuclei detection on breast cancer histopathology images.

(Xu et al., 2015a) K. Xu, J. Ba, R. Kiros, K. Cho, A. C. Courville, R. Salakhutdinov, R. S. Zemel, et Y. Bengio, 2015a. Show, attend and tell : Neural image caption generation with visual attention. Dans les actes de *ICML*, Volume 14, 77–81.

(Yan et al., 2009) F. Yan, N. Xu, et Y. Qi, 2009. Parallel inference for latent dirichlet allocation on graphics processing units. Dans les actes de *NIPS*, 2134–2142.

(Yang et al., 2016) W. Yang, W. Ouyang, H. Li, et X. Wang, 2016. End-to-end learning of deformable mixture of parts and deep convolutional neural networks for human pose estimation. Dans les actes de *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

(Yang et Pedersen,) Y. Yang et J. O. Pedersen. A comparative study on feature selection in text categorization. Dans les actes de *Icml*, Volume 97, 412–420.

(Yeh et al., 2005) J.-Y. Yeh, H.-R. Ke, W.-P. Yang, et I.-H. Meng, 2005. Text summarization using a trainable summarizer and latent semantic analysis. *Information processing & management* *41*(1), 75–95.

(Yu et al., 2010) D. Yu, L. Deng, et G. Dahl, 2010. Roles of pre-training and fine-tuning in context-dependent dbn-hmms for real-world speech recognition. Dans les actes de *Proc. NIPS Workshop on Deep Learning and Unsupervised Feature Learning*.

(Yu et al., 2013) D. Yu, M. L. Seltzer, J. Li, J.-T. Huang, et F. Seide, 2013. Feature learning in deep neural networks-studies on speech recognition tasks. *ICLR*.

(Zeiler et al., 2013) M. D. Zeiler, M. Ranzato, R. Monga, M. Mao, K. Yang, Q. V. Le, P. Nguyen, A. Senior, V. Vanhoucke, J. Dean, et al., 2013. On rectified linear units for speech processing. Dans les actes de *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, 3517–3521. IEEE.

(Zhang et al., 2014) Y. Zhang, E. Chuangsuwanich, et J. Glass, 2014. Language id-based training of multilingual stacked bottleneck features. Dans les actes de *Proc. Interspeech*, 1–5.

(Zhang et al., 2012) Y. Zhang, R. Salakhutdinov, H.-A. Chang, et J. Glass, 2012. Resource configurable spoken query detection using deep boltzmann machines. Dans les actes de *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on*, 5161–5164. IEEE.

(Zhi-Hua Zhou, 2017) J. F. Zhi-Hua Zhou, 2017. Deep forest : Towards an alternative to deep neural networks. 3553–3559.

(Zhu et al., 2016) J.-Y. Zhu, P. Krähenbühl, E. Shechtman, et A. A. Efros, 2016. Generative visual manipulation on the natural image manifold. Dans les actes de *European Conference on Computer Vision*, 597–613. Springer.

(Zibran, 2016) M. F. Zibran, 2016. On the effectiveness of labeled latent dirichlet allocation in automatic bug-report categorization. *2016 IEEE/ACM 38th International Conference on Software Engineering Companion (ICSE-C)*, 713–715.

Bibliographie personnelle

Journal international

1. K. Janod, M. Morchid, R. Dufour, G. Linares, et R. De Mori, 2017. Denoised bottleneck features from deep autoencoders for telephone conversation analysis. *IEEE/ACM Transactions on Audio, Speech and Language Processing*, 25(9), 1505-1516.

Conférences internationales

1. J.V. Cossu, K. Janod, E. Ferreira, J. Gaillard, et M. El Beze, 2014. Lia@replab 2014 : 10 methods for 3 tasks. *Dans les actes de CLEF 2014 Evaluation Labs and Workshop-Working Notes Papers.*
2. J.V. Cossu, K. Janod, E. Ferreira, J. Gaillard, et M. El Beze, 2015. Nlp-based classi ers to generalize experts assessments in e-reputation. *Dans les actes de Experimental IR meets Multilinguality Multimodality and Interaction.*
3. K. Janod, M. Morchid, R. Dufour, et G. Linares, 2016a. A log-linear weighting approach in the word2vec space for spoken language understanding. *Dans les actes de International Spoken Language Technology Workshop (SLT) 2016.*
4. K. Janod, M. Morchid, R. Dufour, G. Linares, et R. De Mori, 2016c. Deep stacked autoencoders for spoken language understanding. *Dans ISCA (Ed.), ISCA INTERSPEECH 206.*
5. M. Morchid, M. Bouaziz, W. Ben Kheder, K. Janod, P.M. Bousquet, R. Dufour, et G. Linares, 2016a. Spoken language understanding in a latent topic-based subspace. *Dans ISCA (Ed.), ISCA INTERSPEECH 2016.*

Conférences nationales

1. K. Janod, M. Morchid, R. Dufour, et G. Linares, 2015. Apport de l'information temporelle des contextes pour la représentation vectorielle continue des mots. *Dans les actes de TALN.*

2. K. Janod, M. Morchid, R. Dufour, G. Linares, et R. De Mori, 2016. Auto-encodeurs pour la compréhension de documents parlés. *Dans les actes de JEP 2016.*
3. M. Bouaziz, M. Morchid, P.M Bousquet, R. Dufour, K. Janod, et W. Ben Kehder, 2016. Un sous-espace thématique latent pour la compréhension du langage parlé. *Dans les actes de JEP 2016.*

Annexes

Annexes A

Les Références de la frise chronologique

La liste ci-dessous répertorie l'ensemble des références présentées chronologiquement dans la Figure 3.1.

1. (Rosenblatt, 1958)
2. (Werbos, 1974)
3. (Kohonen, 1982)
4. (Hopfield, 1982)
5. (Rumelhart et al., 1985)
6. (Ackley et al., 1985)
7. (Smolensky)
8. (Jordan, 1986)
9. (Elman, 1990)
10. (Hochreiter et Schmidhuber, 1997)
11. (Schuster et Paliwal, 1997)
12. (Caruana, 1998)
13. (LeCun et al., 1998)
14. (Graves et al., 2006)
15. (Hinton et al., 2006)
16. (Bengio et al., 2007a)
17. (Chellapilla et al., 2006)
18. (Salakhutdinov et Hinton, 2009)
19. (Mohamed et al., 2009)
20. (Glorot et al., 2011)
21. (Duchi et al., 2011)

22. (Krizhevsky et al.)
23. (Mikolov et al., 2013b)
24. (Kingma et Welling, 2014)
25. (Srivastava et al., 2014)
26. (Bahdanau et al., 2014) D. Bahdanau, K. Cho, et Y. Bengio, 2014. Neural machine translation by jointly learning to align and translate. preprint arXiv :1409.0473.
27. (Goodfellow et al., 2014)
28. (Sutskever et al., 2014)
29. (He et al., 2016)
30. (Ioffe et Szegedy, 2015)
31. (Xiong et al., 2017b)

Annexes B

Apprentissage par Gradient Tree Boosting

Le «Gradient Tree Boosting» (GTB) employé dans le Chapitre 5 est un algorithme d'apprentissage automatique utilisé pour la classification et la régression. Il est introduit par (Friedman, 2001). Cet algorithme repose sur des d'arbres de décisions simples, réunis dans un ensemble par boosting, pour permettre des décisions plus complexes. La création de cet ensemble est réalisée par une méthode additive similaire à une descente de gradient classique.

B.1 Les arbres de décisions

Un arbre de décisions (Classification And Regression Tree, CART) est un modèle statistique discriminant, représentant un ensemble de décisions appliquées à des données vectorisées, sous la forme d'un arbre présenté dans la Figure B.1. Dans ce schéma les noeuds (en blanc) représentent les critères de décisions (ou test) appliqués sur les données d'entrée. Les feuilles de l'arbre (en bleu) sont les valeurs prédites par l'arbre en fonction du chemin parcouru. Les arbres de décisions peuvent être utilisés pour réaliser des régressions multivariées ou de la classification. Ce sont des classifieurs dits faibles. Ils optimisent la fonction de cout B.1.

$$C(\Theta) = L(\theta) + \Omega(\Theta) \tag{B.1}$$

Où $L(\theta)$ est une fonction d'erreur qui indique l'efficacité de l'arbre à prédire les valeurs attendues. Parmi les fonctions d'erreur utilisables, on trouve la fonction L_{MSE} (Bengio, 2009) comme définie dans le Chapitre 6. Le second terme $\Omega(\Theta)$ est la régularisation du modèle qui permet de limiter la complexité de l'arbre et de réduire le surapprentissage.

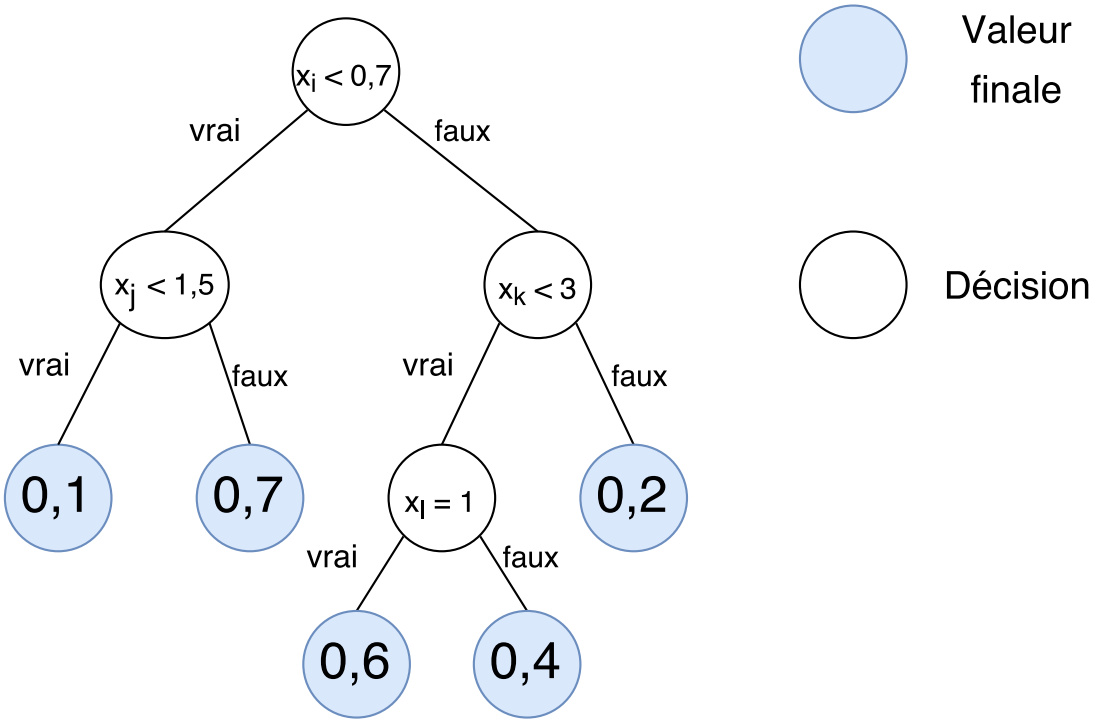


Figure *B.1* – *Schéma d'un arbre de décisions.*

B.2 Méthode de boosting

Un classifieur faible n'est pas capable d'apprendre l'ensemble des particularités des données. C'est pourquoi la méthode de boosting propose de combiner plusieurs classifieurs faibles. La fonction objective devient :

$$C(\theta) = \sum_{i}^{A} L(\Theta) + \sum_{i}^{A} \Omega(a_i) \tag{B.2}$$

où A représente l'ensemble des arbres construits. Ainsi à chaque étape d'entrainement, l'arbre qui minimise la fonction de cout et qui est choisi parmi toutes les configurations possibles, est ajouté à l'ensemble. La descente de gradient classique optimise des paramètres. Dans le cadre du GTB, la descente de gradient est réalisée dans l'espace des fonctions (arbres). L'apprentissage peut être considéré terminé selon deux critères :

— le nombre d'arbres qui a rejoint l'ensemble

— la valeur de la fonction de cout

.

Il est possible de construire ainsi un modèle prédictif très peu sensible au surapprentissage, capable de répondre à des tâches de classification ou de régression. Ils ont été utilisés avec succès dans différentes tâches comme dans citeyamagishi2008phone où le GTB dépasse les performances d'un MLP pour estimer la durée des voyelles en chinois

et en japonais, dans (Popescu et al., 2011) il est utilisé pour détecter des évènements dans des tweets, etc.