Institut National des Langues et Civilisations Orientales

Département Textes, Informatique, Multilinguisme

Construction de modèles *grapheme-to-phoneme* dans le système de reconnaissance de la parole à base de ressources linguistiques *open source*

MASTER

TRAITEMENT AUTOMATIQUE DES LANGUES

Parcours :

Technologie de la Traduction et Traitement des Données Multilingues

par

Hayoung SEO

Directeurs de mémoire :

Ilaine Wang, Patrick Paroubek

Encadrant :

Imed Laaridh

Année universitaire 2021/2022

Remerciements

Je voudrais tout d'abord remercier mes encadrants de ce mémoire, Ilaine Wang et Patrick Paroubek, pour m'avoir encadrée pour ce mémoire, pour leurs conseils permanents et précieux.

Je souhaiterais remercier mon tuteur de stage, Imed Laaridh, de m'avoir confié cette mission passionnante. Un grand merci pour sa disponibilité constante et ses conseils avisés tout au long de mon stage.

Je tiens à remercier Alya Yacoubi, *Head of Zaion Lab*, de m'avoir donné l'opportunité de faire partie de l'équipe tellement passionnante qui m'offre aujourd'hui de nouvelles perspectives professionnelles.

Je veux également remercier tous les membres d'équipe R&D de Zaion qui m'ont merveilleusement intégré, plus particulièrement DATA team pour leur humour et leur accueil qui m'ont permis de passer mon stage dans la joie.

Bien entendu, je n'oublie pas de remercier tous mes camarades de la formation pour leur solidarité qui m'a permis d'avancer.

Je souhaiterais remercier Jooeun Lee et Chaewon Lee, qui m'ont accompagnée toutes ces années passées en France, pour leur soutien morale et leur confiance.

Je veux remercier Mustapha Ouaadi pour son soutien permanence et pour sa confiance en moi, même dans les moments les plus difficiles.

Enfin, mes parents de leur soutien indéfectible tout au long de mes études. Je ne leur en remercierai jamais assez.

TABLE DES MATIÈRES

Liste des figures

Liste des tableaux

RÉSUMÉ

Le système de reconnaissance automatique de la parole d'aujourd'hui a atteint un niveau significatif avec les avancées technologiques. Cependant, les ressources linguistiques à grande échelle constituant les modèles linguistiques tels que modèle de prononciation pour améliorer la qualité de ce système ne peuvent être appréciées que par les langues les plus économiquement réalisables. Répondre à ces problèmes dépendants des ressources linguistiques est l'un des défis majeurs dans le domaine de la reconnaissance vocale.

Ce mémoire a donc pour l'objectif de proposer une méthode de développement rapide et pertinent de modèle de prononciation à l'aide de Phonetisaurus, un convertisseur *Grapheme-to-phoneme* (G2P), s'appuyant sur l'ensemble des dictionnaires accessibles sur le Web afin de capitaliser le dictionnaire de prononciation de qualité. Nous utiliserons Wiktionnaire, un dictionnaire multilingue accessible sur le Web, et Lexique4linguists, une base de données lexicale du français. Ensuite, nous comparons ces dictionnaires avec un autre dictionnaire de prononciation, qui est le résultat de la prédiction d'un modèle de phonétisation existant, et corrigeons le résultat de la prédiction avec une connaissance phonologique et un effort humain minimal. Enfin, en répartissant l'ensemble des trois dictionnaires, nous entraînons trois modèles de phonétisation et les évaluons avec notre méthode d'évaluation WPER (*Weighted Phoneme Error Rate*) basée sur l'algorithme de *Weighted Levenshtein*. Nous avons finalement obtenu un modèle *grapheme-to-phoneme* avec une performance de 97,88 % d'accuracy.

Mots clés : *dictionnaire de prononciation, Weighted-Levenshtein, reconnaissance automatique de la parole, Phonetisaurus, G2P, PER*

INTRODUCTION

Présentation générale

Les humains communiquent avec les autres en écoutant avec leurs oreilles et en parlant avec leur bouche. Le désir de créer un être qui puisse parler comme un humain été imaginé, écrit et étudié par de nombreuses personnes depuis longtemps. De nos jours, ce souhait s'est concrétisé avec l'avènement de l'ère de l'Intelligence Artificielle (IA) conversationnelle qui écoute les voix, comprend les informations et transmet les informations en créant des voix. La reconnaissance automatique de la parole (*Automatic Speech Recognition* en anglais ; ASR) accélère la révolution IoT (*Internet Of Things*) en s'intégrant en permanence dans les produits que nous utilisons au quotidien, tels que l'assistant vocal comme Google Assistant, Siri d'Apple, l'enceinte intelligente comme Alexa d'Amazon, Cortana de Microsoft. En particulier, la pandémie mondiale provoquée par le Covid-19 ces dernières années a accru la demande de services à distance et le champ d'application des solutions d'ASR s'est étendu même aux services publics et médicaux en agitant comme un pont entre les humains et les machines. Cependant, apprendre à une machine à comprendre la parole n'est pas une tâche facile. En effet, il y a tellement d'informations cachées dans une seule phrase vocale qu'un humain peut faire en quelques secondes. De plus, il existe un phénomène de polarisation selon les ressources linguistiques.

Objectif et problématique

Le système de reconnaissance automatique de la parole traditionnel a un modèle de langue, un modèle acoustique et un modèle de prononciation. Parmi eux, nous nous intéressons pour ce mémoire au modèle de prononciation, aussi appelé dictionnaire de prononciation qui relie les séquences de phonèmes prédites par le modèle acoustique aux graphèmes. Le problème est que les dictionnaires de prononciation, qui sont vérifiés soigneusement par des experts linguistiques, sont coûteux et chronophages, et ne sont souvent pas ouverts au grand public. Les grands dictionnaires annotés et vérifiés manuellement sont très rares. De plus, ces dictionnaires ne sont réservés qu'à quelques-unes des langues les plus étudiées et les plus économiques parmi les quelque sept mille langues du monde. De plus, beaucoup de néologismes font leur apparition chaque jour. Il est donc difficile d'avoir toujours le dictionnaire de prononciation complètement à jour même pour les langues riches en ressources pour construire et/ou améliorer la performance de système de reconnaissance automatique de la parole.

Notre objectif est de corriger les erreurs de la prédiction du modèle de *Grapheme-to-phoneme* par comparaison de manière précise avec des dictionnaires de prononciation facilement accessibles sur le Web en introduisant les connaissances linguistiques minimales. Dans ce mémoire, nous essayons d'abord d'expliquer quelle information est contenue dans une seule phrase prononcée et écoutée par un humain, comment

elle est utilisée pour construire le système de reconnaissance de la parole. Nous nous attarderons également sur les obstacles potentiels qu'elle peut poser à ce système. Nous introduirons une méthode d'évaluation du taux d'erreur de phonèmes de manière pondérée basée sur l'algorithme *Weighted Levenshtein* pour évaluer plus délicatement les dictionnaires de prononciation.

Première partie

Contexte général

CHAPITRE 1

ÉTAT DE L'ART

Sommaire

1.1 Introduction

Afin de savoir comment une machine peut comprendre la parole humaine, il est nécessaire d'appréhender comment les être humains comprennent la parole. Cela nécessite des connaissances linguistiques de base. Dans l'état de l'art sont décrits le contexte phonologique de la façon dont les humains peuvent communiquer, ensuite la reconnaissance automatique de la parole, enfin l'éventuel obstacle dans le système ASR d'aujourd'hui.

1.2 Quelques notions linguistiques

1.2.1 Comment l'être humain comprend une parole ?

La capacité d'abstraction, appelé aussi la pensée abstraite, désigne la capacité d'abstraire des objets par le raisonnement observée chez les humains [Dehaene et al., 1998]. Par exemple, bien que chaque chat ait une apparence différente, nous faisons abstraction et nous percevons de divers types de chats en tant que « chats ». Selon la définition linguistique de [De Saussure, 1989], même si le sens (signifié) du signifiant que chaque individu pense de « chat » est différent, les humains sont capables à communiquer entre eux sans problème à travers le signifiant. Par conséquent, cette abstraction est également appliquée à la reconnaissance de la parole chez les humains. Les humains communiquent généralement par la voix. Cependant, les humains ne reconnaissent pas les ondes sonores qui se sont échappées de la bouche de l'autre personne telles qu'elles sont. Les éléments insignifiants sont naturellement exclus de l'onde sonore, et les humains ne sélectionnent que les ondes

sonores qui jouent un rôle distinctif dans leur langue et les comprennent comme des phonèmes. Lorsque les phonèmes ainsi compris forment un morphème combiné, l'information du morphème est convertie en sens. Selon [Levelt et al., 1999], les humains reconnaissent la parole en unités de mots. La reconnaissance vocale chez l'être humain peut être divisée selon les trois caractéristiques suivantes :

1. **Fréquence** (*Frequency* en anglais) : l'être humain reconnaît rapidement les mots à haute fréquence [Broadbent, 1967]. Les mots qui apparaissent fréquemment peuvent être reconnus avec plus de précision que d'autres mots, même dans un environnement bruyant ou même avec des sons très bas.
2. **Parallélisme** (*Parallelism* en anglais) : Plusieurs mots peuvent être reconnus à la fois chez les humains même si plusieurs locuteurs parlent simultanément [Marslen-Wilson, 1987].
3. **Traitement basé sur les indices** (*Cue-based processing* en anglais) : [Jur, 2021] soutient que la reconnaissance de la parole chez les humains est basée sur des « indices » (*Cue* en anglais) tels que des indices acoustiques, des indices lexicaux et des indices visuels.
   — Indice acoustique : cela inclut le formant, délai d'établissement du voisement (DES ; *Voice Onset Time* en anglais ; VOT). Le formant fait référence au pic où l'énergie acoustique est concentrée dans le spectre décrit dans la section 1.3.2. L'être humain perçoit les sons de la parole différemment selon la bande de fréquences dans laquelle le formant (Notion définie dans la section 1.3.2) est constitué.
   — Indice lexical : selon l'effet de restauration phonémique (*Phonemic restoration effect* en anglais) présenté par [Warren, 1970], lorsqu'un (ou plusieurs) des phonèmes constituant un mot est remplacé ou supprimé intentionnellement par un son sans signification tel qu'un son de toux, si le lexique correspondant est déjà connu, il est reconnu comme si le phonème correspondant avait été entendu.
   — Indice visuel : un repère visuel représentatif est l'effet Mcgurk. Ce concept, introduit pour la première fois par [Calvert et al., 1997], fait référence à un phénomène dans lequel un son est perçu comme différent du son réel en raison de l'influence de la forme de la bouche ou d'autres informations sensorielles.

1.2.2 Phonème et traits distinctifs

Les sons produits par les humains à l'aide des organes d'articulation sont physiquement différents pour chaque individu, et même les sons de la parole d'une même personne sont réalisés différemment à chaque fois qu'ils sont prononcés. Cependant, les humains perçoivent certains sons comme des sons identiques. Par exemple, en français, la prononciation du son [r] du mot *père* peut se varier différemment [pɛr], [pɛʀ] ou [pɛʁ][1]. Bien qu'ils aient des valeurs phonétiques différentes, les francophones du français standard les considèrent comme une unité phonologique /r/. La plus petite unité qui est reconnue comme un son dans une langue donnée et distingue le sens d'un mot s'appelle un **phonème** (/r/ dans l'exemple), et l'unité réellement prononcé est un **son** ([r], [ʀ] et [ʁ] dans l'exemple). Le phonème varie sous l'influence de l'environnement vocal, et certains d'entre eux ne sont pas reconnus par le locuteur.

1. https ://fr.wikipedia.org/wiki/Allophone_(phonologie)

Le phonème est en fait subdivisé en fonction des caractéristiques phonétiques du phonème. Proposé par [Jakobson et al., 1951], ce concept est l'argument pour définir un phonème comme les qualités qui le composent. Il a été reconstruit par [Chomsky and Halle, 1968] sur la base de la phonétique articulatoire. Cette distinction, étant un ensemble de **traits distinctifs**, fonctionne de manière binaire [Davenport and Hannahs, 2013]. Ces traits distinctifs, appelés aussi traits pertinents servent à différencier les phonèmes individuels les uns des autres. Cette propriété unique des phonèmes peut être spécifiquement décrite comme une **paire minimale**. Lorsque deux phonèmes ne s'opposent que par un seul trait pertinent, les deux phonèmes sont des paires minimales. Par exemple, en français, les phonèmes /t/ et /d/ partagent les traits « + oral », « + occlusif » et « + dental », mais le trait « voisement » les distinguent (/t/ : -voisé, /d/ : +voisé). Cela explique pourquoi « toit » (/twa/) et « doigt » (/dwa/) ne sont pas le même mot en français du point de vue phonologie.

1.3 Reconnaissance automatique de la parole

La reconnaissance automatique de la parole (ou *Automatic Speech Recognition*, ci-après ASR) est un des domaines du traitement de la parole. Il s'agit d'une domaine très actif de nos jours grâce à la prolifération d'interfaces ou de dispositifs informatiques pouvant supporter de nombreuses applications et le traitement de la parole. Le plus grand avantage d'ASR est qu'il permet aux humains d'interagir avec la machine par la voix, qui est la méthode de communication la plus naturelle chez les humains. Un système d'ASR est une technologie appliquée ayant pour objectif de transcrire un signal vocal en un message orthographique grâce à un algorithme mis en œuvre dans un programme informatique. La transcription de reconnaissance automatique de la parole par n'importe quel locuteur dans n'importe quel environnement est encore loin d'être résolue [Jur, 2021]. Cependant, la technologie d'ASR de nos jours a atteint une maturité telle qu'elle est viable pour de nombreuses tâches pratiques. La manière d'intégrer efficacement la parole dans les applications dépend de la nature de l'interface d'utilisateur et de l'application [Huang and Deng, 2010]. L'ASR est également utile pour la transcription générale dans des domaines où la dictée joue un rôle important. Par exemple, Youtube, un site d'hébergement de vidéos, utilise l'ASR de Google [2] pour générer automatiquement des sous-titres de leurs vidéos.

1.3.1 Évolution d'ASR

La technologie d'ASR s'est développée au cours d'un demi-siècle, surmontant plusieurs limitations. Trois scientifiques du Laboratoire Bell, K.H. Davis, Rulon Biddulph et Stephen Balashekd ont publié en 1952 **AUDREY** (*AUtomatic Digit REcognition*) capable de reconnaître les chiffres parlés. Audrey a été le premier outil de reconnaissance vocale connu et documenté [Pieraccini, 2012]. Son nom « *Reconnaissance numérique automatique* » faisait allusion à sa capacité à reconnaître tous les chiffres existants, mais la technologie de l'époque limitait considérablement la capacité du système. AUDREY ne pouvait distinguer que 10 chiffres, de 0 à 9. De plus, son fonctionnement se limitait à la reconnaissance des numéros correctement prononcés par le locuteur désigné.

Dans les années 1960, IBM a développé la **Shoebox** , un système capable de reconnaître des nombres et des commandes arithmétiques telles que « additionner » et

2. https ://googleblog.blogspot.com/2009/11/automatic-captions-in-youtube.html

« somme ». De plus, Shoebox peut transmettre un problème mathématique à une machine d'addition pour calculer et imprimer la réponse. L'Université Carnegie Mellon a publié **Harpy** dans les années 1970, qui pouvait reconnaître la parole avec une précision raisonnable en utilisant un vocabulaire de 1 011 mots [Lowerre, 1976]. Le système Harpy est le premier système à utiliser des réseaux à états finis pour réduire les calculs et déterminer efficacement la chaîne correspondante la plus proche [Juang and Rabiner, 2005b]. Cependant, la technologie ASR à la fin des années 1970 était loin de l'ASR que nous connaissons aujourd'hui.

La vulgarisation de l'architecture basé sur le modèle de Markov caché (*Hidden Markov Model* en anglais, ci-après HMM) [Rabiner and Juang, 1986] au milieu des années 1980 a apporté une grande amélioration dans le domaine de l'ASR. Cette approche a représenté un changement significatif, de simples méthodes de reconnaissance de formes à une méthode statistique de traitement de la parole [Rabiner, 1989], qui contribue à augmenter la précision. Les progrès de la technologie informatique ainsi que ces approches statistiques ont été une grande force dans l'amélioration des performances d'ASR.

À partir des années 1990, la technologie ASR a commencé à être appliquée au niveau général des consommateurs. Avec l'introduction d'ordinateurs dotés de processeurs plus rapides dans les années 90, les logiciels de reconnaissance vocale étaient enfin accessibles au grand public. **Dragon Dictate**, le premier produit de reconnaissance vocale grand public lancé par Dragon en 1990, était lent à reconnaître la parole (30 à 40 mots par minute) à environ un quart de la vitesse de la parole humaine, et était au prix de 9 000 $ qui était élevé pour de nombreux consommateurs.

Après 2000, des machines basées sur un ASR ciblant un grand nombre de consommateurs ont commencé à apparaître. Le système **Google Voice Search** est un système de reconnaissance vocale révolutionnaire publié sous la forme d'une application basée sur des données massives. À partir de là, les systèmes ASR tels que **Siri** d'Apple, **Alexa** d'Amazon et **Cortana** de Microsoft peuvent désormais être facilement trouvés dans la vie quotidienne des consommateurs.

1.3.2 Fonctionnement d'ASR

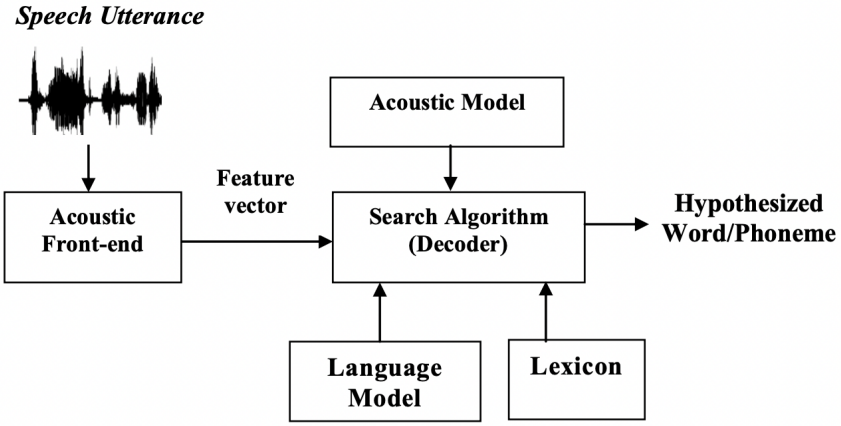

FIGURE 1.1 – Architecture typique de l'ASR traditionnel

Les figures 1.1 [Juang and Rabiner, 2005a] et 1.2 illustrent le fonctionnement d'un modèle traditionnel d'ASR, donc modèle de type HMM-GMM.

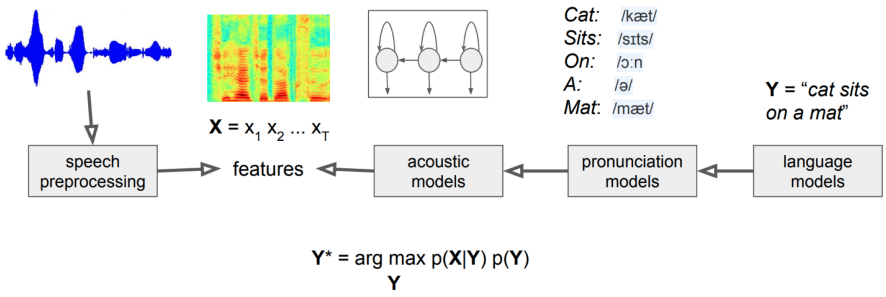

FIGURE 1.2 – Architecture détaillée de l'ASR traditionnel [3]

1. Extraction des caractéristiques (*Feature extraction*)
2. Modèle acoustique (*Acoustic Model* ; AM)
3. Modèle de langue (*Language Model* ; LM)
4. Modèle de prononciation (*Lexicon*)
5. Décodeur (*Decoder*)

Extraction des caractéristiques

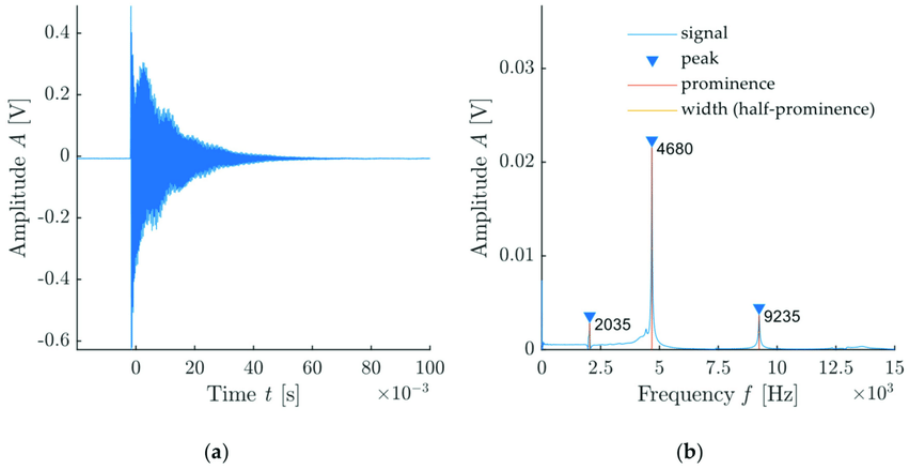

FIGURE 1.3 – Exemples d'un signal audio et d'un spectre [4]

La première chose à faire pour la reconnaissance vocale est d'extraire les caractéristiques des sons à partir d'informations inutiles telles que le bruit et les bruits de fond du signal audio. Un signal audio est une répartition de la pression acoustique dans le domaine temporel (voir la figure 1.3 a). En appliquant l'algorithme de transformation de Fourier rapide (*Fast Fourier Transform* ; FTT) à ce signal, la pression acoustique peut être exprimée dans le domaine fréquentiel. Cette représentation d'un signal audio en termes de fréquence est un **spectre** (voir la figure 1.3 b). Le pic observable dans le spectre, appelé aussi **formant** indiquent la partie dominante du signal audio. Le système auditif humain est plus sensible aux basses fréquences qu'aux hautes fréquences. Le **spectre Mel** est obtenu en appliquant une

4. https ://www.researchgate.net/publication/348858767/figure/fig4/AS :988795549777921@1612758638613/Example-of-a-signal-and-its-frequency-spectrum-created-by-FFT-a-Signal-from-the-sensor.png

banque de filtres basée sur l'échelle de Mel (*Mel scale*) au spectre. L'échelle de Mel fait référence à la formulation de la relation entre les fréquences physiques et les fréquences perçues par les humains en fonction de leurs connaissances linguistiques par des experts en phonétique et phonologie. Le **coefficient cepstral de fréquence Mel** (*Mel-frequency Cepstral Coefficient* ; MFCC) faisant référence à un algorithme qui convertit le signal audio en un « vecteur de caractéristiques » est extrait en effectuant une analyse cepstrale sur le spectre Mel obtenu en reflétant les caractéristiques de l'organe auditif humain [Davis and Mermelstein, 1980]. Il s'agit d'une méthode d'extraction basée sur la connaissance linguistique et les caractéristiques sont déterminées lorsqu'une entrée vocale est donnée. Le MFCC est largement utilisé, de l'ASR traditionnel aux modèles modernes de bout en bout (*End-to-End* ; E2E) [Kurenkov, 2020].

Récemment, l'extraction de caractéristiques par les réseaux de neurones (*Neural Network* en anglais) a également attiré l'attention. Contrairement à la méthode MFCC dont le processus est déterministe, l'extraction par le réseau neuronal basées sur *Deep Learning* (Apprentissage profonde) est probabiliste. Elle peut changer pendant le processus d'apprentissage même si le signal d'entrée est la même et ne nécessite pas beaucoup de connaissances linguistiques. Des exemples représentatifs sont Wav2Vec [Schneider et al., 2019] et SincNet [Ravanelli and Bengio, 2018].

Sources linguistiques : AM, LM et PM

Dans le système ASR traditionnel, la reconnaissance vocale est effectuée à l'aide de trois sources de connaissances linguistiques : un modèle de langue, un modèle acoustique et un dictionnaire de prononciation. La propriété de reconnaissance vocale humaine consistant à reconnaître les sons de la parole dans les unités de mots décrites dans la section 1.2.1 est également introduite dans la reconnaissance automatique de la parole [Jur, 2021].

— **Modèle acoustique (AM)**

Le AM est utilisé dans l'ASR pour représenter la relation entre les signaux vocaux et les phonèmes ou autres unités linguistiques constituant la parole. Le AM basé sur **HMM-GMM** (*Hidden Markov Model-Gaussian Mixture Model*) est un composant qui a joué un rôle important avant l'apparition du modèle E2E basé sur Deep learning. Ce système modélise la probabilité que l'observation se produise compte tenu de l'état, c'est-à-dire la fonction de probabilité d'émission. Pour apprendre le modèle acoustique, des données vocales et la transcription des données vocales sont nécessaires. Cependant, les données enregistrées en donnant et en lisant la transcription pour la collecte de données vocales ne peuvent pas être considérées comme reflétant l'environnement réel de l'utilisateur. Par conséquent, un prototype est créé à l'aide de ces données, puis un modèle acoustique qui reflète l'environnement réel de l'utilisateur est formé à l'aide de l'entrée des données du journal pendant le service.

— **Modèle de langue (LM)**

Le LM est la partie responsable de la modélisation des mots et des séquences de mots dans une langue. Le problème fondamental est qu'il existe

4. https ://jonathan-hui.medium.com/speech-recognition-gmm-hmm-8bb5eff8b196

d'innombrables combinaisons de mots dans une langue donnée. Même si toutes les phrases du monde sont collectées en tant que données, une phrase avec une nouvelle combinaison de mots apparaîtra toujours. Toutefois, selon [Jurafsky, 2000], presque tous les problèmes de traitement de la parole et du langage peuvent être reconstruits avec la **probabilité**, c'est-à-dire « choisir le plus probable étant donné N choix pour une entrée ambiguë ». La fonction clé de LM est d'indiquer la plausibilité d'une phrase donnée avec probabilité.
Un groupe de N mots consécutifs est appelé un **N-gram**. Étant donné que le nombre d'éléments à calculer augmente à mesure que N augmente, un modèle statistique est généralement créé en mélangeant 1-gram, 2 gram et 3 gram. Cela ressemble à une méthode simple, mais lorsque beaucoup de données sont accumulées, cela devient un très bon modèle [Jurafsky, 2000]. Si un nouveau mot, c'est-à-dire un mot hors vocabulaire (*Out-Of-Vocabulary* ; OOV) tels que nom propre et emprunt par exemple, est inséré dans le modèle N-gram déjà formé, la probabilité du nouveau mot est calculée comme 0 car le modèle N-gram est basé sur des statistiques. Pour éviter cela, une technique appelée *Smoothing* est utilisée. Les modèles de langue d'intelligence artificielle (IA) textuels tels que BERT, RoBERTA et GPT-3 ont fait de grands progrès ces dernières années. Ils peuvent générer un texte très réaliste sur n'importe quel sujet en fournissant des mots écrits en entrée et n'utiliser que quelques étiquettes ou exemples pour couvrir une variété de processus difficiles de traitement du langage naturel, y compris l'analyse des sentiments, la traduction, la recherche d'informations, le raisonnement et le résumé.

— **Modèle de prononciation (PM)**

Le modèle de prononciation, également appelé *lexicon* (lexique), associe les mots aux séquences de phonèmes. Il s'agit d'un composant qui relie le modèle acoustique et le modèle de langue dans un système d'ASR traditionnel. Autrement dit, le AM est utilisé pour prédire la séquence de phonèmes correspondant au signal audio, et le PM relie cette séquence de phonèmes prédite au mot correspondant. Le PM le plus basique utilisé dans le système ASR traditionnel est un ensemble de mots dans lequel le mot et sa prononciation sont marqués. Ce dictionnaire statique est généralement produit par les experts linguistiques, ou créé à l'aide d'une approche basée sur les données (Data-Driven) [Rutherford et al., 2014]. Étant donné que la qualité du dictionnaire de prononciation affecte le processus de décodage, il doit toujours être révisé et corrigé. Les expériences menés par [Jouvet et al., 2012] ont montré que l'identification et la correction des variations phonétiques des mots les plus fréquemment utilisés entraînaient des gains de performances notables. Cependant, ces dictionnaires statiques, quelle que soit leur taille, ne peuvent pas contenir tous les mots en raison de l'apparition d'OOV inévitable dû à la nature dynamique du langage lui-même comme expliqué dans la section 1.3.2. Par conséquent, il est nécessaire de les compléter avec l'outil dynamique comme G2P (*Grapheme-to-phoneme*) qui génère la prononciation correspondante lorsqu'un mot est donné. G2P est une partie essentielle du système ASR traditionnel car il est appelé chaque fois qu'il n'y a pas de mot dans le dictionnaire statique. Récemment, de nombreuses études ont été menées sur des systèmes ASR basés sur des réseaux de neurones pour réduire l'importance du PM ou pour l'éliminer complètement. [Maas et al., 2015] et [Liu et al., 2022]

ont proposé une approche sans dictionnaire de prononciation basée sur un réseau de neurones. Cependant, bien que les réseaux de neurones aient tendance à être plus précis que les méthodes « traditionnelles » et puissent contenir plus de données, ils sont plus lents à former par rapport aux modèles traditionnels, ce qui a un coût en termes de performance et d'efficacité.

Décodeur

Le décodeur est le processus de recherche du chemin le plus optimal dans l'espace de recherche composé du AM et du LM, et estime quelle séquence de mots la voix représente. L'ASR traditionnel utilise principalement le décodage basé sur l'arbre lexical et le décodage basé sur WFST (*Weighted Finite State Transducer*). Le premier implémente le modèle HMM, les informations de contexte, la séquence de prononciation et les informations de grammaire séparément, puis crée un réseau pour la reconnaissance et les recherches. L'avantage de l'arbre lexical est qu'il peut réduire l'espace de recherche en partageant des nœuds avec les mêmes propriétés acoustiques, mais le taux de reconnaissance peut être réduit car le LM ne peut pas être appliqué avant d'atteindre la feuille du nœud. D'autre part, WFST a l'avantage de combiner et d'optimiser les réseaux utilisés pour le modèle HMM, les informations de contexte et les informations de grammaire dans un cadre unifié. L'ASR à grand vocabulaire récemment annoncé utilise la technologie de décodeur basée sur WFST. La technologie de décodeur basée sur WFST est rapide et précise car il s'agit d'un réseau combiné avant la recherche des séquences. Toutefois, si le PM ou le LM change même légèrement, le réseau doit être reconfiguré. De plus, au fur et à mesure que le vocabulaire et le LM se développent, beaucoup de mémoire est nécessaire pour établir le réseau.

1.3.3 Obstacles pour un système ASR

L'un des aspects les plus difficiles dans un système ASR est la parole elle-même. Tant d'informations sont cachées dans une phrase qu'un humain peut prononcer en quelques secondes. [Marslen-Wilson, 1973] montre que les humains sont capables à traiter l'ensemble du processus de reconnaissance de la parole comme la segmentation des mots, l'analyse et l'interprétation dans un temps très court (250 ms) depuis la réception de la parole. Autrement dit, les humains reconnaissent la parole presque en temps réel. De plus, le signal vocal est fortement influencé par l'état physique et émotionnel du locuteur et par l'environnement qui l'entoure. Par exemple, le locuteur exprime le phonème différemment selon l'environnement qui l'entoure. Pour communiquer dans un endroit bruyant comme un marché, le volume de la voix augmente sans s'en rendre compte. Étant donné que l'environnement dans lequel les données vocales sont généralement collectées est un environnement insonorisé avec un minimum de bruit, il peut être très différent des données vocales collectées et analysées dans l'environnement réel. Compte tenu de ces points, afin d'augmenter la précision du système ASR, il est nécessaire d'analyser et de comprendre les différents modèles linguistiques utilisés dans la pratique pour apprendre les données. Cependant cela pose un autre problème : *comment et où se procurer autant de ressources linguistiques ?*

Un autre problème est le manque de ressources linguistiques tels que les corpus audio, ses transcriptions et les dictionnaires de prononciation, ne peuvent être créés

5. https ://commoncrawl.github.io/cc-crawl-statistics/plots/languages

crawl (%)	CC-MAIN-2022-27	CC-MAIN-2022-33	CC-MAIN-2022-40
anglais	46.5384	46.7774	45.8786
russe	5.8779	5.5156	5.9692
allemand	5.4824	5.2400	5.8811
japonais	4.8135	4.3489	4.7884
chinois	4.6777	5.0383	4.8747
français	4.4002	4.3112	4.7254
espagnol	4.3178	4.2915	4.4690

TABLE 1.1 – Distribution des données sur Web par langues [5]

que dans la langue la plus économiquement viable. Sur environ sept mille langues dans le monde, seules quelques langues dominantes peuvent avoir un système ASR performant. Même pour ces langues, la création d'une ressource linguistique authentique peut être coûteuse et prendre du temps [Schlippe et al., 2010], car les performances du modèle de langage dépendent non seulement de la collecte de nombreuses ressources linguistiques, mais également du processus de correction de ces ressources. Selon « Common Crawl » [6], une plate-forme ouverte qui maintient et gère des référentiels publics permettant à quiconque d'accéder et d'analyser facilement les données explorées sur le Web, près de la moitié des ressources linguistiques sur Web sont écrites en anglais (Table 1.1). Cependant, contrairement à l'anglais, la plupart des langues du monde ne bénéficient pas de cette technologie en raison du manque d'ensembles de données étendus. Récemment, diverses tentatives ont été faites pour se libérer de la dépendance aux ressources linguistiques. Meta AI a publié en open source le « *Generative Spoken Language Model* » (GSLM) [Lakhotia et al., 2021], le premier modèle innovant de traitement du langage naturel à hautes performances pour résoudre la dépendance aux ressources linguistiques. GSLM peut fonctionner directement sur des signaux audio bruts sans données textuelles. En d'autres termes, il s'agit d'une méthode de génération d'un modèle de langue pour potentiellement toutes les langues du monde, même pour une langue qui ne dispose pas d'ensemble de données textuelles. Néanmoins, bien que l'importance des données textuelles puisse diminuer, le besoin de données audio suggère toujours qu'il existe des problèmes de ressources linguistiques pour les langues minoritaires.

1.4 Conclusion

Nous avons pu voir dans ce chapitre tout d'abord comment les humains comprennent la parole d'un point de vue linguistique avant d'aborder la façon dont les machines comprennent la parole humaine.

Ensuite, nous avons étudié ce qu'est l'ASR à travers son évolution dans le temps et sa structure. Aujourd'hui, l'ASR a fait des progrès technologiques significatifs et, comme dans la plupart des domaines basés sur la haute technologie, nous avons vu que les tentatives de création de systèmes sans assistance humaine sont courantes. À titre d'exemple représentatif, les grands dictionnaires de prononciation qui étaient créés avec du temps par des experts en langues dans le passé sont maintenant créés rapidement et avec moins d'intervention humaine grâce à G2P. Cependant, afin de former le modèle G2P, des ressources linguistiques sont également nécessaires.

6. https ://commoncrawl.github.io

En fin de compte, la concurrence fondamentale parmi les systèmes ASR d'aujourd'hui dépend de la façon dont nous pouvons disposer de ressources linguistiques plus sophistiquées et plus diversifiées avec moins d'intervention humaine.

CHAPITRE 2

MÉTHODES

Sommaire

2.1 Introduction

Dans ce chapitre, nous présentons tout d'abord les symboles que nous avons employés pour comparer nos dictionnaires de prononciation afin d'entraîner les modèles de prononciation. Ensuite, nous expliquons l'outil G2P de notre choix, *Phonetisaurus*. Enfin nous présentons les métriques classiques utilisées pour l'évaluation des ASR, *Word Error Rate* et *Phoneme Error Rate*, avant de proposer une métrique adaptée à notre recherche et basée sur l'algorithme *Weighted-Levenshtein*.

2.2 Symboles pour dictionnaire de prononciation

L'ensemble des symboles utilisés pour représenter la prononciation d'un mot est déterminé par les caractéristiques de la langue et le modèle acoustique du système comme nous avons vu dans la section 1.2.2. Par conséquent, il existe plusieurs convention de symboles phonétiques pour décrire les sons du langue. Notre objectif de recherche est de réviser et de corriger la prédiction de G2P par comparaison avec des dictionnaires de prononciation facilement accessibles sur le Web, et de préparer les jeux de données pour un nouveau PM du français à la suite de la correction. Pour ce faire, il est nécessaire de savoir les symboles déjà existants et utilisés pour d'autres

dictionnaires de prononciations. Dans cette sous-section, nous présentons les alphabets phonétiques les plus courants (API et SAMPA) puis deux autres conventions dont nous avons également utilisées.

2.2.1 API

L'idée de l'Alphabet Phonétique International (API, *International Phonetic Alpabet* en anglais ; IPA) [Association, 1999] était d'établir une méthode standard et indépendante pour représenter les sons de toutes les langues. L'Association Phonétique Internationale, fondée en 1886 par des professeurs de français et d'anglais sous la direction du linguiste français Paul Passy, a supervisé l'élaboration de la première version de API, qui a été publiée en 1888. Les symboles API utilisées dans chaque langue du monde sont cités dans les publications de l'Association Phonétique Internationale [1] sur lesquels nous trouvons les enregistrements sonores de chaque son correspondant. L'un des avantages de cet alphabet est qu'il nous permet de décrire non seulement des consonnes et des voyelles mais aussi les accents lexicaux et les tons des langues. La dernière édition de l'API [2] a été publié en 2020. L'API a l'avantage de pouvoir représenter presque tous les sons du monde selon le but pour lequel il a été créé. De plus, il a l'avantage d'être familier car il est utilisé dans les dictionnaires traditionnels que nous rencontrons habituellement, mais il a l'inconvénient de ne pas pouvoir être saisi sur un clavier. En raison de cette caractéristique de l'API, nous avons décidé de ne pas garder les transcriptions en API de Wiktionnaire mais de les convertir vers un autre système de transcription (Voir la section 3.2.3).

2.2.2 SAMPA

Développé par un groupe international de phonétique en 1989, le Speech Assessment Methods Phonetic Alphabet (SAMPA) [3] est un alphabet phonétique lisible par machine. SAMPA consiste à mettre en correspondance les symboles de l'Alphabet phonétique international (API) spécifiés dans la section précédente avec les codes ASCII dans les plages 33 à 127, qui sont des caractères ASCII imprimables sur 7 bits. SAMPA n'est pas l'initiative d'un seul auteur, mais plutôt le résultat d'une collaboration et d'une consultation entre des chercheurs spécialiste de la parole de différents pays. Les symboles SAMPA ont été développés en consultation avec des locuteurs natifs de toutes les langues dans lesquelles ils sont appliqués et sont normalisés au niveau international. Contrairement à l'API, tous les symboles du SAMPA sont conçus pour être saisis avec le clavier d'un ordinateur que nous utilisons normalement, ce qui est son principal avantage. Cela permet, par exemple, au phonème français /ʁ/ dans l'API d'être saisi par /R/ au lieu de /ʁ/. Cependant, les sons nasaux qui existent en français ne peuvent pas être écrits en une seule lettre à la manière SAMPA. Par exemple, une nasale /a/ (/ɑ̃/ en API) est exprimée sous la forme d'ajouter « ˜ » à côté de « a » dans SAMPA (a˜). Les codes SAMPA français se trouvent dans le tableau ci-dessous. [4]

1. https ://www.internationalphoneticassociation.org
2. https ://www.internationalphoneticassociation.org/IPAcharts/IPA_chart_trans/pdfs/IPA_Kiel_2020_full_fra.pdf
3. https ://www.phon.ucl.ac.uk/home/sampa/
4. https ://www.phon.ucl.ac.uk/home/sampa/french.htm

2.2.3 Lexique3

La convention des symboles phonétiques Lexique3 [5] est utilisée pour le dictionnaire de Lexique4Linguists (voir la section 3.2.2). Cette convention est très similaire à SAMPA, mais elle complète le problème de représentation des sons nasaux du français de SAMPA que nous avons vu précédemment. Dans SAMPA, les quatre sons nasaux /ɑ̃/, /ɛ̃/, /œ̃/ et /ɔ̃/ en français écrits avec deux lettres sont exprimés comme une seule lettre (@), (5), (1) et (§) respectivement dans la convention Lexique3.

2.2.4 Zaion

Les symboles utilisés pour le dictionnaire de prononciation de Zaion ne sont ni SAMPA, ni API, mais des symboles très proches de Lexique3. Nous avons établi des tableaux de comparaison des symboles de l'API, de Lexique3 et de Zaion dans les tableaux 3.1 et 3.2 du chapitre 3.3 décrivant les jeux de données.

2.3 G2P du choix : Phonetisaurus

Étant donné que nous devons créer un nouveau PM via G2P pour confirmer notre hypothèse, nous devions décider quel G2P utiliser. Phonetisaurus [6] [Novak et al., 2011] est un outil G2P open source basé sur WFST (*Weighted Finite State Transducer*) décrit dans la section précédente 1.3.2. Compte tenu des données d'entraînement et de test des paires mot-prononciation, un pipeline Phonetisaurus comprend les étapes suivantes :

1. Alignement des séquences
2. Génération de modèles N-gram par alignement des séquences
3. Compilation de modèles n-grammes en WFST
4. Décodage WFST avec OpenFst : le calcul du chemin le plus court (P) est effectué sur la synthèse pour trouver la meilleure prononciation.

Nous avons choisi Phonetisaurus car c'est un outil déjà utilisé par l'entreprise, il est donc facile à intégrer si le nouveau PM est jugé meilleur que l'actuel en fonction des résultats. En outre, il s'agit d'une boîte à outils simple basée sur HMM et qui permet un entraînement rapide du modèle et des prédictions précises. Selon les résultats de la comparaison des outils G2P testés par [Hahn et al., 2012], Phonetisaurus montre un très bon Phoneme Error Rate (PER) de 1,3 % sur environ 15 000 mots d'un jeu de test français.

2.4 Taux d'erreur de phonèmes pondéré

Dans cette section, nous décrivons le WER (*Word Error Rate*) et le PER (*Phoneme Error Rate*), les métriques traditionnellement utilisées pour l'évaluation des systèmes ASR, et comment nous avons mis en œuvre WPER (*Weighted Phoneme Error Rate* ; Taux d'erreur de phonèmes pondéré) en fonction de nos objectifs et de nos besoins.

5. http ://www.lexique.org/?page_id=286
6. https ://github.com/AdolfVonKleist/Phonetisaurus

2.4.1 Taux d'erreur de mots

$$WER = \frac{(\,\mathrm{S} + \mathrm{D} + \mathrm{I}\,)}{\mathrm{N}} \tag{2.1}$$

Dans la recherche ASR, une métrique couramment utilisée pour évaluer les performances d'ASR est le WER. La métrique WER, dérivée de l'algorithme de distances de Levenshtein, est utilisée pour évaluer la séquence de mots prédite par l'ASR. La formule WER présentée en 2.1 est composé des quatre composants suivants :

— S : nombre de substitution
— D : nombre de suppression
— I : nombre d'insertion
— N : nombre de mots dans la référence

Autrement dit, WER est la somme du nombre de mots incorrectement identifiés ($S + D + I$) lors de la reconnaissance, divisée par le nombre total de mots dans la référence (N). Les mots mal identifiés sont définis comme substitution, suppression et insertion. Par exemple, si la phrase « *Comment vas-tu aujourd'hui Jean* » est prédite par l'ASR comme « *Comment tu tiens aujourd'hui Jeanne* », la somme du nombre de mots incorrectement identifiés sera 3 :

— 1 substitution (*Jeanne* au lieu de Jean)
— 1 suppression (*vas*)
— 1 insertion (*tiens*)

La phrase de référence « *Comment vas tu aujourd'hui Jean* » est composée de 5 mots ($N = 5$). Par conséquent, nous obtenons 60 % de WER pour cette phrase. Une difficulté courante dans la métrique WER réside dans le fait que la séquence de mots reconnue peut avoir une longueur différente de la séquence de mots de référence. Dans le cas où de nombreuses suppressions et insertions se produisent, il peut y avoir une grande différence de longueur entre la phrase réelle et la phrase prédite. De plus, ce type de mesure ne fournit pas de détails sur la nature des erreurs et toutes les opérations de substitution, de suppression et d'insertion ont le même coût. Des travaux supplémentaires sont donc nécessaires pour identifier les principales causes d'erreurs.

2.4.2 Taux d'erreur de phonèmes

Dans notre cas, nous avons cependant besoin d'une méthode pour évaluer le taux d'erreurs de phonèmes dans un mot plutôt que d'une méthode pour évaluer le taux d'erreurs de mots dans une phrase. Le taux d'erreur de phonèmes (*Phoneme Error Rate* ; PER) est une métrique similaire au WER mais appliquée aux unités phonémiques. La formule 2.2 suivante est le calcul du taux d'erreur pour une certaine séquence de phonèmes lorsqu'elle est prédite.

$$PER = \frac{(\,\mathrm{S} + \mathrm{D} + \mathrm{I}\,)}{\mathrm{N}} \tag{2.2}$$

S, D et I spécifient respectivement le nombre minimum de substitutions, de suppressions et d'insertions requis pour transformer la séquence de phonèmes prédite en une séquence de phonème de référence. N désigne le nombre de phonèmes dans la séquence de référence. Plus la valeur PER est petite, meilleures sont les performances du système. Le PER « classique » est une mesure de la **distance de Levenstein** comme WER vu dans la sous-section 2.4.1 entre le réel et le prédit. Comme WER,

l'inconvénient de PER réside en ceci. Dans la métrique PER classique, les opérations de substitution, d'insertion et de suppression sont calculées au même coût, qui est en général *1*. Cependant, nous pensons que l'évaluation au niveau phonémique, contrairement à l'évaluation de séquences des mots, devrait être plus détaillée et précise. Étant donné que les phonèmes sont des unités plus petites que les mots et qu'il n'est pas possible de construire un modèle de langage qui couvre tous les mots du monde, comme nous avons discuté dans la section 1.3.2, les phonèmes sont déjà définis dans une langue donnée. Nous avons donc décidé de créer un nouveau taux d'erreurs de phonèmes à partir de cette idée.

2.4.3 Traits distinctifs des phonème et WPER

La première question que nous nous sommes posée pour créer le taux d'erreur de phonèmes de manière pondérée était de savoir quel poids distribuer à quels phonèmes. Comme nous allons voir dans la section 3.4.1, toutes les substitutions de phonèmes n'ont pas le même poids. Dans l'objectif de notre recherche, la substitution d'un phonème /v/ par un phonème /k/ ne doit pas avoir la même valeur que la substitution d'un phonème /ɑ/ par /a/. Comme nous avons évoqué dans la section 1.2.2, le phonème /v/ et le phonème /k/ ne partagent aucun traits distinctifs alors que la distinction du phonème /ɑ/ et du phonème /a/ est en train de disparaître en français standard moderne.

— /v/ : consonne fricative labiodentale voisée
— /k/ : consonne occlusive vélaire sourde

Nous devions également décider laquelle des trois opérations, substitution, insertion et suppression, devait être pondérée. Nous avons tout d'abord décidé de définir des poids différents uniquement pour les substitutions, qui sont généralement considérées comme une insertion et une suppression simultanées.

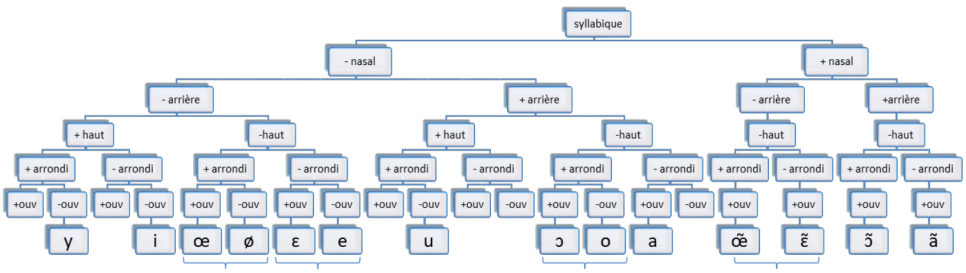

FIGURE 2.1 – Décomposition en traits distinctifs des voyelles française

Ensuite, concernant le poids des phonèmes, nous nous référons aux travaux de [Ghio et al., 2018] utilisant des arbres de décision (Figure 2.1) basés sur des traits distinctifs définis par [Chomsky and Halle, 1968] que nous avons vus dans la section 1.2.2. Nous avons modifié les matrices de coûts des consonnes et des voyelles basée sur la classification des phonèmes de Chomsky proposée par [Ghio et al., 2018] pour répondre à nos besoins (Tables 2.1 et 2.2). Nous avons enlevés le poids des archiphonèmes. Une archiphonème est une unité phonologique qui est l'ensemble des phonèmes qui exclut l'un l'autre dans certains positions tel que les pairs [e, ɛ].

	a	**i**	**u**	**o**	**e**	**y**	**2**	**E**	**O**	**9**	**@**	**5**	**N**	**1**
a	0	3	3	2	2	4	3	1	1	2	1	2	2	3
i	3	0	2	3	1	1	2	2	4	3	4	3	5	4
u	3	2	0	1	3	1	2	4	2	3	4	5	3	4
o	2	3	1	0	2	2	1	3	1	2	3	4	2	3
e	2	1	3	2	0	2	1	1	3	2	3	2	4	3
y	4	1	1	2	2	0	1	3	3	2	5	4	4	3
2	3	2	2	1	1	1	0	2	2	1	4	3	3	2
E	1	2	4	3	1	3	2	0	2	1	2	1	3	2
O	1	4	2	1	3	3	2	2	0	1	2	3	1	2
9	2	3	3	2	2	2	1	1	1	0	3	2	2	1
@	1	4	4	3	3	5	4	2	2	3	0	1	1	2
5	2	3	5	4	2	4	3	1	3	2	1	0	2	1
N	2	5	3	2	4	4	3	3	1	2	1	2	0	1
1	3	4	4	3	3	3	2	2	2	1	2	1	1	0

TABLE 2.1 – Matrice de coût des voyelles

	p	**t**	**k**	**b**	**d**	**g**	**f**	**s**	**S**	**v**	**z**	**Z**	**m**	**n**	**J**	**l**	**R**
p	0	1	2	1	2	3	1	2	3	2	3	4	3	4	5	3	4
t	1	0	1	2	1	2	2	1	2	3	2	3	4	3	4	2	3
k	2	1	0	3	2	1	3	2	1	4	3	2	5	4	3	3	4
b	1	2	3	0	1	2	2	3	4	1	2	3	2	3	4	2	3
d	2	1	2	1	0	1	3	2	3	2	1	2	3	2	3	1	2
g	3	2	1	2	1	0	4	3	2	3	2	1	4	3	2	2	3
f	1	2	3	2	3	4	0	1	2	1	2	3	4	5	6	4	3
s	2	1	2	3	2	3	1	0	1	2	1	2	5	4	5	3	2
S	3	2	1	4	3	2	2	1	0	3	2	1	6	5	4	4	3
v	2	3	4	1	2	3	1	2	3	0	1	2	3	4	5	3	2
z	3	2	3	2	1	2	2	1	2	1	0	1	4	3	4	2	1
Z	4	3	2	3	2	1	3	2	1	2	1	0	5	4	3	3	2
m	3	4	5	2	3	4	4	5	6	3	4	5	0	1	2	2	3
n	4	3	4	3	2	3	5	4	5	4	3	4	1	0	1	1	2
J	5	4	3	4	3	2	6	5	4	5	4	3	2	1	0	2	3
l	3	2	3	2	1	2	4	3	4	3	2	3	2	1	2	0	1
R	4	3	4	3	2	3	3	2	3	2	1	2	3	2	3	1	0

TABLE 2.2 – Matrice de coût des consonnes

2.4.4 Implémentation de WPER

Pour calculer la distance minimale d'édition entre réel et hypothèse, nous avons décidé d'utiliser l'algorithme de distance de Levenshtein pondéré (*Weighted Levenshtein*). Il est généralement utilisé pour les applications de reconnaissance optique de caractères (ROC ; *Optical character recognition* en anglais, ci-après OCR). Par exemple, du point de vue de reconnaissance optique de caractères, le coût de la substitution de [Q] et de [O] est inférieur au coût de la substitution de [Q] et [W] en raison de la similarité entre le graphème [Q] et le graphème [O]. Cet algorithme peut également être utilisé pour l'auto-correction de la frappe au clavier. Contrairement à

son usage dans le cas de l'OCR, la ressemblance de chaque graphème n'a pas d'importance. Cependant, le coût de substitution dépendra de la distance de chaque touche de clavier. Par exemple, le coût de la substitution de [Q] et de [W] est moins important sur un clavier AZERTY que la substitution de [Q] et de [O], car ces touches sont situées l'une à côté de l'autre. Par conséquent, la probabilité que l'utilisateur ait mal tapé ces touches est plus élevée. Nous pensons que cet algorithme est adapté pour calculer le taux d'erreur au niveau des phonèmes que nous voulons évaluer de manière pondérée car cet algorithme peut nous permettre d'attribuer un poids différent aux opérations comme la substitution, la suppression et l'insertion. Nous avons implémenté cet algorithme en utilisant le package Python `weighted-levenshtein`[7] dans notre script Python. Ce script se trouve en annexe A.2. L'avantage de cet implémentation est qu'elle prend en compte le trait distinctif des phonèmes du français, et donne plus de poids lorsque des phonèmes sont éloignés les uns des autres dans l'arbre de décision du trait distinctif. Le remplacement des phonèmes /k/ et /v/ illustrés dans la section 2.4.3 n'a plus le même poids que le remplacement de /a/ et /ɑ/ dans ce script. Enfin, nous définissons notre formule WPER comme 2.3. SP signifie la substitution pondérée.

$$WPER = \frac{(\ \mathrm{SP} + \mathrm{D} + \mathrm{I}\ )}{\mathrm{N}} \tag{2.3}$$

2.5 Conclusion

Dans ce chapitre, nous avons d'abord vu les conventions de prononciation représentatives qui permettent de représenter une prononciation dans les dictionnaires de prononciation. Les avantages et les inconvénients de l'API et du SAMPA, qui sont des conventions phonologiques représentatives, étaient clairs. Le premier a l'avantage de pouvoir transcrire tous les sons des langues du monde par des symboles uniques, mais n'est pas adapté à une utilisation sur le clavier d'ordinateur. En revanche, le seconde n'a aucun problème à être saisi sur le clavier, mais les sons nasaux du français s'expriment en deux lettres, ce qui peut avoir une incidence sur le processus de traitement automatique. Ensuite, nous avons vu comment fonctionne Phonetisaurus, l'outil G2P que nous utiliserons dans la section 4. Enfin, nous avons conclu que le PER n'était pas adaptée à notre étude, et nous avons trouvé un moyen de l'adapter de manière pondéré.

7. https ://github.com/infoscout/weighted-levenshtein

Deuxième partie

Expérimentations

CHAPITRE 3

JEUX DE DONNÉES

Sommaire

3.1 Introduction

Dans ce chapitre, nous allons présenter un dictionnaire utilisé par l'entreprise Zaion et deux dictionnaires en droit libre dont nous nous sommes servis pour préparer les jeux de données afin d'entraîner les modèles de phonétisation avec l'outil *Phonetisaurus*. Nous décrivons en détails les traitements appliqués aux dictionnaires de base et les obstacles rencontrés au cours de préparation des données. Enfin nous présentons les jeux de données finales prêt à employer pour l'expérimentation.

3.2 Trois dictionnaires de prononciations du français

Dans cette section, nous décrivons en détails les caractéristiques de trois dictionnaires de base (Zaion, Lexique4linguists et Wiktionnaire) que nous avons utilisés.

3.2.1 Zaion

Le dictionnaire de Zaion LV (Zaion Large Vocabulary) est constitué de 523 839 entrées au total et est le résultat prédiction du modèle de phonétisation interne de l'entreprise. Le format d'enregistrement de ce dictionnaire est *dict* qui est souvent utilisé pour le dictionnaire informatique. Chaque ligne du fichier contient un mot et sa prononciation, séparés par une tabulation. La prononciation elle-même utilise un espace comme séparateur pour chaque unité phonologique (Figure 3.1).

```
acérés  a s e R e
acésulfame  a s e z y l f a m
acétabulaire    a s e t a b y l E R
acétabulum  a s e t a b y l O m
acétal  a s e t a l
acétaldéhyde    a s e t a l d e i d
acétaminophène  a s e t a m i n o f E n
acétamipride    a s e t a m i p R i d
acétate a s e t a t
acétates    a s e t a t
acétazolamide   a s e t a z o l a m i d
```

FIGURE 3.1 – Extrait du dictionnaire Zaion

Le codage des unités phonologiques utilisés pour ce dictionnaire n'est ni SAMPA, ni API, mais un codage spécifique destiné à l'utilisation interne de l'entreprise pour des raisons techniques. Ce codage est très similaire avec le codage du dictionnaire Lexique4linguists comme nous le verrons en détails dans la section 3.3.

Ce modèle de phonétisation a été entraîné depuis le dictionnaire Zaion SM (Zaion Small) de la taille 125 460 entrées. Étant donnée la taille importante de Zaion LV, ce dictionnaire contient les erreurs de phonétisation issues du manque de données de Zaion SM utilisé pour l'entraînement de modèle G2P qui a prédit Zaion LV. Par exemple, le dictionnaire Zaion SM ne contient qu'un seul mot avec une terminaison *-ault* à savoir le mot « meursault » (/mœʁso/). Par conséquent, le modèle a prédit la phonétisation /-so/ pour tous les mots avec une terminaison *-ault* dans Zaion LV même si ces mots ne sont pas forcément finis par une terminaison *-sault*. Puisque nous ne sommes pas capable de vérifier le dictionnaire entier et de corriger à la main toutes les erreurs présentes par manque de temps, nous avons décidé de le corriger au maximum en comparant avec deux dictionnaires de qualité que nous pouvons facilement trouver sur le Web (voir les sous-sections 3.2.2 et 3.2.3). Nous détaillons ce processus de correction dans la section Pré-traitement de ce chapitre.

3.2.2 Lexique4linguists

Lexique4linguists [Schalchli, 2022] est une base de données lexicale du français en droit d'utilisation libre publiée le 4 avril 2022 sur le site d'ORTOLANG [1], une infrastructure de mutualisation de ressources linguistiques écrites et orales. Il s'agit du résultat d'une adaptation et d'un enrichissement de la 4ème mise à jour de la base de données Lexique [2]. Cette base de données est sous format XML-TEI permettant un affichage sous la forme d'une arborescence (Figure 3.2). Chaque entrée de cette base de données est constituée d'une forme orthographique d'un mot et sa représentation phonologique ainsi que les informations grammaticales comme par exemple sa partie du discours (*Part Of Speech* en anglais ; POS), mais aussi le genre et le nombre de ce mot, et sa fréquence dans les différents corpus. Dans cette base de données, les homographes ne sont pas regroupés dans une entrée mais dans plusieurs entrées car nous avons les informations grammaticales associées à chaque entrée d'homographe.

Dans la figure 3.2, nous pouvons apercevoir la représentation phonologique dans la balise « pron ». Comme nous avons vu dans la section 2.2, le codage phonologique utilisé ici suit la convention du Lexique3 qui est différent du codage SAMPA ou API. Le tableau de ce codage se trouve en annexe A.1. Néanmoins, ceci reste très proche de

1. https ://www.ortolang.fr/fr/accueil/
2. http ://www.lexique.org

SAMPA. Le nombre d'entités lexicales décrites est de 218 831 au total sachant qu'il existe des entités sans la représentation phonologique.

```
<entry xml:id="étudiant_ADJ_ms" type="mainEntry">
    <form>
        <orth>étudiant</orth>
        <pron>etydj@</pron>
    </form>
    <gramGrp>
        <gram type="pos">ADJ</gram>
        <gram type="gender">m</gram>
        <gram type="number">s</gram>
        <gram type="frequency">1493</gram>
    </gramGrp>
    <sense>
        <def>étudiant</def>
    </sense>
</entry>
```

FIGURE 3.2 – Structure de base de données Lexique4linguists

Les données du Lexique4linguists ont été extraits en laissant uniquement la forme orthographique et sa représentation phonologique via un script du Python. Certains mots de ce dictionnaire ont été exclus car la représentation phonétique n'existait pas, et nous avons obtenu un total de 159 218 mots.

3.2.3 Wiktionnaire

L'un des dictionnaires les plus courants sur le *World Wide Web* (WWW) est Wiktionary [3], un projet jumeau de Wikipedia [4]. Ce dictionnaire est un dictionnaire multilingue qui a été fondé en 2002 avec le slogan « Le dictionnaire libre et gratuit que chacun peut améliorer. ». *Wiktionnaire* [5] est la version française de Wiktionary, destiné aux utilisateurs de la francophonie comptant désormais 404 346 lemmes français. [6] Le contenu de Wiktionnaire est disponible au format Json [7]. Nous avons pu télécharger le dictionnaire de la taille 1 425 076 entrées. La figure 3.3 montre la structure du fichier Json de Wiktionnaire. Nous pouvons apercevoir la forme orthographique

{"accueil": ["a.kœj"], "lire": ["liʁ"], "encyclopédie": ["ɑ̃.si.klɔ.pe.di"],
"manga": ["mɑ̃.ga"], "ouvrage": ["u.vʁaʒ"], "siège": ["sjɛʒ"], "chaise":
["ʃɛz"], "fauteuil": ["fo.tœj"], "meuble": ["mœbl"], "mardi": ["maʁ.di"],
"lundi": ["lœ̃.di"], "semaine": ["sə.mɛn", "smɛn"], "militaire": ["mi.li.tɛʁ"],
"suis": ["sɥi"], "barbe à papa": ["baʁ.bɔa pa.pa"], "manchot": ["mɑ̃.ʃo"],
"pingouin": ["pɛ̃.gwɛ̃"], "mercredi": ["mɛʁ.kʁə.di"], "bande dessinée": ["bɑ̃d

FIGURE 3.3 – Structure de base de données Wiktionnaire

en clé et sa représentation phonologique en valeur entourée par les crochets. Plusieurs transcriptions phonologiques peuvent apparaître entre crochets si, la plupart

3. https ://www.wiktionary.org
4. https ://www.wikimedia.org
5. https ://fr.wiktionary.org
6. https ://fr.wiktionary.org/wiki/Wiktionnaire :Statistiques, consulté 26/10/2022
7. https ://dumps.wikimedia.org/backup-index.html

des cas, les homographes ne sont pas des homophones. Lorsque deux homographes ne sont pas des homophones, ils s'écrivent de la même manière et se prononcent différemment. Par exemple, nous pouvons trouver dans l'extrait de Wiktionnaire deux mots homographes qui ne sont pas en relation d'homophonie avec la même unité graphique ("couvent" : ["kuvɑ̃", "kuv"]).

En ce qui concerne les symboles phonétiques, Wiktionnaire emprunte l'API différemment des deux dictionnaires précédents. Nous pouvons constater que le symbole liaison est également utilisé dans les mots composés comme barbe à papa (/baʁ. b‿a pa.pa/). Nous pouvons aussi constater que les unités syllabiques se distinguent avec le point (.). Nous avons écrit un script Python pour extraire les données du Wiktionnaire. Au cours de l'extraction, les phrases longues comme les proverbes ont été exclues puis nous avons obtenu un total de 558 803 mots.

3.3 Pré-traitements

Dans la section précédente, nous avons décrit les caractéristiques de chaque dictionnaire dont nous avons la possession. Étant donnée la différende source de chaque dictionnaire, le pré-traitement est nécessaire. Nous allons présenter dans cette section les pré-traitements appliqués aux dictionnaires pour qu'ils puissent être comparés les uns aux autres. Nous avons cherché d'abord à remplacer les différents unités phonologiques utilisés dans le dictionnaire Lexique4linguists et Wiktionnaire par les symboles utilisés chez Zaion. Ensuite nous avons uniformisé le format de chaque dictionnaire au format « dict » afin de les comparer et finalement les employer dans Phonetisaurus.

3.3.1 L'uniformisation du codage

Afin de faciliter la comparaison pour que nous puissions corriger efficacement le dictionnaire de Zaion LV, nous avons d'abord commencé par remplacer les codages des unités phonologiques du dictionnaire Lexique4linguists et du dictionnaire Wiktionnaire, qui ont un codage autre que du dictionnaire Zaion. Notre choix s'est porté sur le codage de Zaion afin d'intégrer des futurs modèles en interne de l'entreprise dans l'avenir selon les résultats. La plupart des symboles sont identiques dans différents codages surtout lorsqu'ils peuvent être saisis au clavier comme des consonnes /p/, /b/, /t/, etc. Le tableau 3.1 illustre les consonnes qui ne sont pas identiques dans trois dictionnaires. Nous pouvons apercevoir que le dictionnaire de Zaion et le dictionnaire Lexique4linguists sont très similaires en matière du codage phonologique comparés à l'API que Wiktionnaire emploie. Les unités autre que dictionnaire Zaion sont remplacés sur le terminal à l'aide d'expressions régulières.

Nom de son	Zaion	Lexique4linguists	Wiktionnaire (API)
g occlusive	g	g	ɡ
emprunt anglais	G	G	ŋ
nasale palatale	J	N	ɲ
r fricative uvulaire	R	R	ʁ
ch fricative	S	S	ʃ
ge fricative	Z	Z	ʒ

TABLE 3.1 – Tableau comparatif de la représentation des consonnes

En ce qui concerne les symboles utilisés pour les voyelles (le tableau 3.2), nous nous sommes demandés comment traiter la voyelle a postérieure /ɑ/. La distinction de la voyelle /ɑ/ (a postérieur) et de /a/ (a antérieur) est en voie de disparition, surtout en français standard. C'est la raison pour laquelle le dictionnaire de Zaion et le dictionnaire de Lexique4linguists ne distinguent pas /ɑ/ et /a/ en termes du codage phonologique. Néanmoins, cette voyelle /ɑ/ est toujours présente dans les transcriptions de Wiktionnaire. Par conséquent, nous avons décidé de remplacer tous les /ɑ/ dans Wiktionnaire par /a/ car les deux autres dictionnaires ne font pas de distinction entre ces deux voyelles. Comme nous pouvons le voir sur le tableau 3.2, le format Zaion écrit en tout lettres pour certains symboles, par exemple /huit/ à la place de /ɥ/.

Nom de son	**Zaion**	**Lexique4linguists**	**Wiktionnaire(API)**
ui semi-voyelle	huit	8	ɥ
a postérieur	ABS	ABS	ɑ
e ouvert	E	E	ɛ
o ouvert	O	O	ɔ
schwa	^	°	ə
e ouvert	neuf	9	œ
e fermé	deux	2	ø
an nasale	@	@	ɑ̃
in nasale	cinq	5	ɛ̃
un nasale	un	1	œ̃
on nasale	N	§	ɔ̃

TABLE 3.2 – Tableau comparatif de la représentation des voyelles

3.3.2 L'uniformisation du format

Afin de comparer trois dictionnaires et de nous en servir pour l'entraînement de nos modèles via Phonetisaurus, nous avons converti le dictionnaire Lexique4Linguists (XML-TEI) et le dictionnaire Wiktionnaire (Json) au format « dict » à l'aide d'un script Python.

3.4 Préparation des données

Une fois l'uniformisation des dictionnaires de base terminée, nous pouvons comparer les dictionnaires progressivement dans l'objectif de corriger le dictionnaire Zaion LV. Nous avons considéré cette comparaison comme essentielle car le dictionnaire Lexique4linguists et le dictionnaire Wiktionnaire ne sont pas parfaitement corrects.

3.4.1 Comparaison et correction

Comme nous avons vu dans la section 3.2.1, le dictionnaire Zaion LV contient les erreurs de prédiction du modèle, il est donc nécessaire de le corriger avant de répartir les jeux de données finaux. Pour la correction de ce dictionnaire, nous avons d'abord dû nous demander à quel dictionnaire nous référer. Il est certain que les dictionnaires Lexique4linguists et Wiktionnaire sont des dictionnaires de bonne qualité, mais ils ne sont pas totalement exempts d'erreurs non plus en raison du grand nombre de

mots. Nous avons donc décidé de comparer les trois dictionnaires simultanément et de voir dans quelle mesure chaque dictionnaire s'accordait sur un mot. Notre hypothèse de base est que le dictionnaire Zaion LV est une prédiction du modèle et que le taux d'erreur est plus élevé que celui des deux autres dictionnaires car il n'a pas subi de correction manuelle. Ainsi, nous avons décidé de supposer qu'il est correct si les deux autres dictionnaires, à l'exception de Zaion LV, s'accordent sur un mot. Nous avons tout d'abord fait une liste de mots communs à chaque paire pour comparer les dictionnaires par paire pour savoir quelle paire de dictionnaires s'accorde le mieux. Nous avons décidé ici de les comparer de manière stricte, c'est-à-dire, même si un seul phonème était différent de toute la séquence de phonèmes d'un mot, celui-ci est traité comme faisant partie des mots en désaccord. Le dictionnaire Zaion LV et Wiktionnaire partagent 146 213 mots et seuls 109 648 mots sont en accord au niveau de la prononciation, soit 74,99 % de la totalité des entrées. Cependant, Wiktionnaire était plus en accord avec le dictionnaire Lexique4linguists avec le taux d'accord de 80,63 % sur 133 576 entrées communs. Le meilleur taux d'accord était observé dans la paire de dictionnaire de Zaion LV et Lexique4linguists, avec un taux d'accord de 91,15 % sur 132 502 mots communs. Toutefois, ce fait ne nous a pas surpris en raison du fait que le dictionnaire Zaion SM, qui a été utilisé comme données pour le modèle prédisant le dictionnaire Zaion LV, a été créé à partir de lexique3.83, le modèle prédécesseur de Lexique4linguists. A travers le pourcentage d'accord strict sur les

Zaion LV	Wiktionnaire	NB confusion	percent
o	O	22935	70,20 %
E	e	3739	11,44 %
e	E	1745	5,34 %
z	s	752	2,30 %

TABLE 3.3 – Liste de paires de phonèmes en désaccord : Wiktionnaire et Zaion LV

prononciations de chaque paire de dictionnaire, nous avons constaté quelles paires de dictionnaires étaient les plus en accord les unes avec les autres. Ensuite nous devions observer dans quelles paires de phonèmes les prononciations des mots en relation de désaccord apparaissaient le plus. Dans un premier temps, nous nous sommes intéressé à la paire Wiktionnaire-Zaion LV qui partageait le moins de représentations phonologiques. Le tableau 3.3 résume les paires de phonèmes en relation de désaccord entre Wiktionnaire et Zaion LV. Parmi les 37 490 prononciations, près de 85% des paires de phonèmes sont associées à des voyelles, notamment le trait distinctif « ouvert » des voyelles.

Zaion LV	Lexique4linguists	NB confusion	percent
o	O	5949	56,64 %
E	e	411	3,91 %
O	o	386	3,67 %
N	O	384	3,66 %
@	^	342	3,26 %
e	E	295	2,81 %

TABLE 3.4 – Liste de paires de phonèmes en désaccord entre les dictionnaires Lexique4linguists et Zaion LV

Ce phénomène a également été observé dans la comparaison entre Zaion LV et

Lexique4linguists (tableau 3.4). 60 % du total des erreurs était lié à l'ouverture de la voyelle. Contrairement au résultat précédent, des paires de phonème /@/ (/ɑ̃/ en API) et de phonème /^/ (/ə/ en API) ou bien des paires /N/ et /O/ sont apparues dans cette comparaison. À la suite de la vérification, nous avons vu que cela était dû aux formes orthographiques tels que « convergent » pouvant être à la fois verbe et adjectif ont été mal alignés dans le processus de comparaison. Concernant aux paires /N/ et /O/, il existait quelques confusion de ces paires dans le dictionnaire Lexique4linguists et ils ont été corrigés grâce à cette comparaison. Malheureusement, comme nous l'avons expliqué dans les chapitres précédents (section 1.2.2 et section 2.4.3), il est très difficile de corriger des phonèmes qui se distinguent par un seul trait distinctif, c'est-à-dire des paires minimales. Par conséquent, nous n'avons pu corriger qu'un petit pourcentage de paires de phonèmes anormales dans les paires de dictionnaires. A la fin du processus de correction, nous avons pu obtenir les dictionnaires finaux comme indiqué dans le tableau 3.5.

	Zaion	**Lexique4linguists**	**Wiktionnaire**
Format initial	dict	XML-TEI	Json
Codage	Zaion	Lexique3	API
Nombre d'entrée initial	523 839	218 831	1 425 076
Nombre d'entrée final		159 218	558 803
Entrées communes	90 015		
Codage final	Zaion		
Format final	dict		

TABLE 3.5 – Récapitulatif des données

3.4.2 Jeux de données finales

À partir des trois dictionnaires « propres », nous avons décidé de générer trois jeux de données pour entraîner les modèles. Dans l'objectif d'expérimenter notre recherche plus finement, nous avons décidé de répartir les dictionnaires de base que nous avons obtenus dans la section précédente selon nos besoins au lieu de les employer tels quels. Nous avons donc décidé de préparer trois jeux de données pour créer trois modèles au total. Autrement dit, nos hypothèses sont les suivantes :

1. Si les trois dictionnaires s'accordent le plus sur la prononciation d'un mot, le modèle sera-t-il plus performant même avec l'ensemble de données relativement petit ?
2. Le modèle fonctionnera-t-il mieux en utilisant un ensemble de données de qualité suffisamment nombreux, en utilisant un ensemble de cas où les paires de dictionnaires concordent sur la prononciation ?
3. Un grand ensemble de données constitue-t-il un bon modèle peu importe la qualité de dictionnaires ?

Reflétant les trois hypothèses ci-dessus, nous avons décidé de répartir nos données dans les catégories suivantes :

— **DICT1** : ce dictionnaire contient des mots et des phonétisations correspondantes lorsque les dictionnaires de base sont tous en accord au niveau de phonétisation d'un tel mot.

— **DICT2** : ce dictionnaire est le résultat de concaténation de DICT1 et toutes les intersections de chaque deux dictionnaires de base lorsqu'ils sont en accord sur la prononciation.
— **DICT3** : ce dictionnaire contient l'ensemble de dictionnaires de base peu importe leur accord sur la phonétisation. Les doublons sont éliminés.

Nom	Description	Nombre d'entrée
DICT1	intersection de trois dictionnaire de base	86 132
DICT2	DICT1 + intersection de chaque deux dictionnaires	165 909
DICT3	ensemble de tous les dictionnaires	991 413

TABLE 3.6 – Description de chaque jeu de données final

Le tableau 3.6 résume nos jeux de données. Comme expliqué dans nos hypothèses, DICT1 est le plus petit dictionnaire avec 86 132 entrées et DICT3 est le plus grand, avec 991 413 entrées, 10 fois plus d'entrées que DICT1.

Nous prenons 20% de chaque dictionnaire arrondi au supérieur pour tester chaque modèle de phonétisation. La proportion de chaque jeu se trouve dans le tableau 3.7.

Nom	Total	Train	Test
DICT1	86 132	68 905	17 227
DICT2	165 909	132 688	33 713
DICT3	991 413	750 692	187 673

TABLE 3.7 – Répartition de jeux de données

3.5 Conclusion

Nous avons expliqué dans ce chapitre comment nous avons collecté et pré-traité les données. Contrairement à nos attentes, la méthode de comparaison de trois dictionnaires n'était pas adaptée pour corriger de nombreux mots en raison de problèmes linguistiques, par exemple lorsqu'un seul unique trait distinctif (ouverture dans notre cas) pour la production des voyelles était présent dans les paires de phonèmes erronés, qui ne peuvent pas être corrigés. Cependant, nous avons montré qu'il était possible de corriger la prononciation erronée d'un certain nombre de mots avec cette méthode comme le cas de /@/ et /^/. Ensuite en nous basant sur nos hypothèses et avec trois dictionnaires calibrés, nous avons finalement réparti les dictionnaires pour l'entraînement et les tests pour l'outil Phonetisaurus, qui seront décrits dans le chapitre suivant.

CHAPITRE 4

RÉSULTATS

Sommaire

4.1 Introduction

Comme nous l'avons vu dans le chapitre précédent, nous avons préparé trois types de jeux de données pour nous adapter à nos trois hypothèses avec trois dictionnaires de prononciation corrigés par comparaison. L'ensemble des données a été arbitrairement séparé en jeux de *train* et de jeux de *test*. Nous avons pris au hasard 80 % de l'ensemble de données pour créer le jeu de train et défini 20 % des données non incluses dans le jeu de train pour le jeu de test.

Dans ce chapitre, nous allons créer et tester trois modèles de prononciation à l'aide de *Phonetisaurus*, un outil *Grapheme-to-Phoneme* basé sur *WSFT* en utilisant les données préparées, et discuter des résultats et de l'évaluation.

4.2 Entraînement des modèles et test

Le tableau 4.1 résume l'ensemble des processus d'entraînement et du test. Le temps d'entraînement était très rapide pour les modèles 1 et 2, respectivement 1 minute 30 secondes et 3 minutes 20 secondes. En revanche, le temps d'entraînement du modèle 3, qui possède le plus de données, environ 10 fois celui du modèle 1, a dépassé 20 minutes.

	Temps d'entraînement	**Train**	**Test**	**Accuracy (%)**	**NB Désaccord**
Model 1	1 min 30 s	68 905	17 227	97,88	365 (2 %)
Model 2	3 min 20 s	132 688	33 713	91,61	2 489 (8 %)
Model 3	20 min 10 s	750 692	187 673	87,58	23 292 (12 %)

TABLE 4.1 – Description d'entraînement sur Phonetisaurus

Les trois modèles ont montré globalement de bons résultats, et le modèle 1 en particulier a commis des erreurs uniquement pour 365 mots, ce qui ne représente

que 2 % du jeu de test. Ce fait montre que notre première hypothèse était la bonne. Cependant, nous n'avons pas arrêté d'expérimenter ici et avons décidé d'examiner plus en détail les résultats des tests de chaque modèle avec le script Python `wer.py` que nous avons écrit (voir la section 2.4.3). Le tableau 4.2 résume l'application de la méthode d'évaluation WPER aux résultats des tests des trois modèles.

	NB mots désaccord	**WPER (%)**
Model 1	365	24,68
Model 2	2 489	19,95
Model 3	26 146	28,56

TABLE 4.2 – Évaluation sur les prononciations en désaccord

Nous avons constaté que le modèle 1 présente un taux d'erreur inférieur sur le jeu de test entier, mais a fait plus d'erreurs que le modèle 2 en termes d'évaluation WPER. Cela suggère que le modèle 2 produit plus d'erreurs de phonétisation différents de ses références que le modèle 1, mais que les autres prononciations ne sont pas significativement différentes du point de vue linguistique que nous avons abordé précédemment dans la section 1.2.2 et 2.4.3. En d'autres termes, on peut supposer que l'erreur de prédiction du modèle 2 est liée à l'opposition des phonèmes avec moins de traits distinctifs que les modèles 1 ou 3. D'autre part, le modèle 3 présente des performances inférieures à celles des deux autres modèles à la fois dans l'ensemble de tests global et dans l'évaluation WPER. Nous avons confirmé que notre troisième hypothèse n'était pas adaptée à la construction d'un modèle de prononciation efficace.

4.3 Test sur un corpus commun

Nous nous sommes demandé quels seraient les résultats des trois modèles sur le même jeu de test. Ainsi, à partir de l'ensemble de données, nous avons construit une liste de mots qui n'ont pas été utilisés dans l'entraînement des trois modèles. Nous avons testé nos modèles sur cette liste d'un total de 689 mots sans transcription phonologique. Suite à la création de ce corpus commun, puisque ces mots n'étaient pas utilisés dans le processus d'entraînement d'aucun modèle, le problème était de savoir à quelle prononciation nous devions nous référer pour l'évaluation. Nous avons donc décidé d'utiliser le modèle 1, qui a montré le meilleur résultat sur son jeu de test, comme référence.

	NB mots	**Accuracy (%)**	**NB désaccord**	**WPER (%)**	**S**	**D**	**I**
Model 2	689	97,82	15 (2,18 %)	0,49	8	6	6
Model 3		95,79	29 (4,21 %)	0,86	18	9	7

TABLE 4.3 – Résultat d'évaluation WPER sur le résultat corpus commun

Même en considérant le petit nombre de mots dans le corpus de test, nous avons obtenu de bons résultats (Table 4.3). En particulier, nous pouvons voir que le taux de WPER sur la prédiction de prononciation erronée, a convergé vers presque 0 % dans les deux modèles. Comme dans la section précédente, nous supposons que ce faible WPER est dû au fait que les prononciations prédites ne sont pas linguistiquement éloignées des références. Comme les prononciations erronées n'était pas nombreuses dans les deux modèles, nous avons donc décidé d'observer chaque prédiction erronée.

Le tableau 4.4 illustre toutes les prononciations mal prédites par le modèle 2. Nous avons observé que un phonème liés à deux graphèmes, tels que « ravins » et « rabbins » (/ɛ̃/ - in), créent de l'ambiguïté pour le modèle. Ainsi, nous avons confirmé que des emprunts ou des mots étrangers tels que le mot « week-end » ou « container » peuvent également affecter les résultats. Nous avons aussi observé que les prononciations de mots tels que « concert », « brunet » et « circonflexe » étaient plutôt incorrectes en référence et prédites correctement par le modèle 2.

Mot	Ref	Hyp	WPER(%)
ravins	Rav5	Ravin	50,0
week-end	wikEnd	wik@	50,0
rabbins	Rab5	Rabin	50,0
container	kNtEne	kNtEn9R	33,33
photique	fotik	fOtik	22,0
bourre-pif	buRpif	buR^pif	16,67
concert	kNsERt	kNsER	16,67
brunet	bRynEt	bRynE	16,67
lèchefrite	lESfRit	lES^fRit	14,29
soubresaut	subREso	subR^so	14,29
coordonnant	koORdOn@	koORdon@	13,75
promenade	pROm^nad	pROmnad	12,5
circonflexe	siRkNflEk	siRkNflEks	11,11
footballistiques	futbOlistik	futbalistik	10,0
sous-secrétaire	suzs^kRetER	sus^kRetER	9,09

TABLE 4.4 – Model 2 : application WPER pour chauque erreur

Cependant, dans ce processus, nous avons pu trouver d'autres problèmes. Il a été observé que le script WPER que nous avons écrit ne s'alignait pas bien pour certains mots où la suppression et l'insertion se produisaient. En conséquence, l'évaluation PER au lieu de l'évaluation WPER a été automatiquement appliquée à certains mots. Nous pensons que le problème vient du package `weighted-levenshtein`.

4.4 Conclusion

Dans ce chapitre, nous avons construit et testé les modèles de phonétisation en utilisant l'outil G2P *Phonetisaurus* avec nos jeux de données. Les hypothèses que nous avons posées au début de ce chapitre pourraient être prouvées en testant sur des ensembles de jeux de tests et sur un corpus commun. Dans nos tests, le modèle 1, entraîné sur DICT1 qui est l'ensemble de données contenant les mots et ses transcriptions les plus « propres », a obtenu les meilleurs résultats. Cependant, une fois confirmé par notre métrique d'évaluation WPER plus en détails, il a été confirmé que le taux d'erreur dans l'ensemble de test était inférieur à celui du modèle 2, mais la distance des prononciations prédites était plus courte dans le résultat du modèle 2. Un test réalisé sur un corpus commun de 689 mots s'est bien comporté dans les deux modèles 2 et 3. Cependant, à la suite de tests sur ce petit corpus, nous avons constaté que notre script WPER avait un problème d'alignement dû à un problème avec le package `weighted-levenshtein` lui-même.

Conclusion générale

Le but de cette étude était de proposer une méthode pour créer un nouveau modèle de G2P afin d'améliorer le modèle de prononciation du système d'ASR en utilisant des dictionnaires de prononciation accessibles sur le Web.

Tout d'abord, il était important de savoir comment les humains comprennent la parole afin de savoir comment les machines peuvent comprendre la parole humaine. De plus, nous avions besoin de savoir de situer dans le système ASR le dictionnaire de prononciation que nous voulons étudier afin de comprendre comment la qualité ce dictionnaire de prononciation affecte ce système. Bien que nous pensions que les experts qui ne sont pas spécialistes de la langue et/ou de la parole devraient être capables de manipuler facilement les ressources linguistiques, nous pensons que comprendre et créer un dictionnaire phonétique nécessite des connaissances phonologiques minimales. En effet, l'usage et la constitution d'un dictionnaire de prononciation nécessite de comprendre sa nature, qui va au-delà d'une simple liste de paires de mots-prononciations : la notion de phonème ainsi que de traits distinctifs y est en réalité fondamentale.

Ensuite, nous avons examiné ce qu'est un système ASR, ainsi que le passé et le présent de ce système. Dans la structure de ce système, nous avons vu où se situe le dictionnaire de prononciation, comment il fonctionne et quels obstacles rendent difficile la construction du dictionnaire de prononciation. Des études récentes basées sur « Generative Spoken Language Model », comme [Lakhotia et al., 2021], ont montré des modèles qui ne nécessitent pas forcément de dictionnaire de prononciation. Néanmoins, la plupart des langues du monde, qui manquent de ressources linguistiques audio, n'ont d'autres choix que d'utiliser la méthode « traditionnelle » qui utilise le dictionnaire de prononciation, le modèle de langue et le modèle acoustique.

Pour comparer les dictionnaires de prononciation, des conventions phonologiques telles que API et SAMPA, qui permettent d'exprimer les sons en symboles, ont également été présentées, car les dictionnaires de prononciation peuvent être basés sur différentes conventions pour répondre à chaque étude ou besoin. Étant donné que nos trois dictionnaires, Zaion LV, Lexique4linguists et Wiktionnaire, utilisaient tous des conventions différentes, un processus de pré-traitement pour les unifier a été jugé essentiel. L'extension du fichier de chaque dictionnaire de prononciation était également différente pour les trois. De cela, nous avons appris que le processus de comparaison de plusieurs dictionnaires de prononciation accessibles sur le Web nécessite un pré-traitement.

Nous avons constaté que la plupart des cas dans lesquels les trois dictionnaires n'étaient pas d'accord sur la prononciation étaient les paires de voyelles ouvertes et fermées. En fait, il a été constaté que plus de 85 % des désaccords sur la pro-

nonciation des mots étaient dus à des degrés d'ouverture de la cavité buccale dans la paire de dictionnaire Zaion LV-Wiktionnaire. A travers cela, nous avons jugé que la méthode existante d'évaluation du taux d'erreur des phonèmes manquait de détails pour être appliquée à des dictionnaires de haute qualité tels que Wiktionnaire et Lexique4linguists. La métrique du taux d'erreur de phonèmes est basée sur l'algorithme de distance de Levenshtein, dans lequel la substitution, la suppression et l'insertion d'un phonème ont le même poids. Il a été déterminé que cette méthode n'était pas adaptée pour évaluer la substitution de phonèmes dans la relation de paires minimales telles que /toit/ et /doigt/.

Nous avons donc décidé de rechercher une méthode pour évaluer le taux d'erreur de phonèmes de manière pondérée. Nous n'avons pas pu trouver des travaux existants sur ce taux d'erreur de phonème pondéré, mais nous avons pu trouver une étude similaire. Nous nous inspirons notamment de la matrice de coût des voyelles et des consonnes, qui s'appuie sur l'arbre de décision des traits distinctifs de [Chomsky and Halle, 1968] proposé par [Ghio et al., 2018]. En réalité, nous avons modifié ces matrices en fonction de nos besoins et proposons une nouvelle métrique pour calculer le taux d'erreur de phonèmes via le module Python `weighted-levenshetein` en implémentant nos matrices du coût de substitution.

Enfin, les trois dictionnaires « propres » ont été divisés en trois jeux de données pour s'adapter à nos hypothèses. En effet, notre objectif était de créer de nouveaux modèles G2P à l'aide de Phonetisaurus, un outil G2P basé sur WFST, plutôt que de simplement corriger le résultat de prédiction. Parmi les trois modèles G2P entraînés, le meilleur modèle n'a généré que 2 % d'erreurs sur le jeu de test (modèle 1). Les deux autres modèles, 2 et 3, avaient des taux d'erreur de 8 % et 12 %, respectivement. Nous sommes particulièrement satisfaits des résultats des performances du modèle 1, qui compte le moins de mots dans l'ensemble de données.

Les modèles ont également été appliqués au jeu de test composé de 689 mots qui n'ont pas été utilisés dans l'entraînement des trois modèles. Nous avons donc utilisé les prédictions du modèle 1, qui a montré les meilleurs résultats dans la première expérience, comme référence. Dans cette expérience, les modèles 2 et 3 ont montré un taux d'erreur inférieur à 5 %. et bons résultats de 0,48 % et 0,86 % en WPER, respectivement.

Il reste à vérifier les ensembles de mots plus grands ou bien les noms propres, les sigles et les mots étrangers avec les modèles créés. La différence avec les modèles basés sur RNN qui ne nécessitent pas de modèle G2P ou de modèle de prononciation est que nous pouvons observer explicitement le processus de prédiction de la prononciation par nos modèles. Dans le futur, il sera intéressant d'appliquer le dictionnaire prédit par notre modèle G2P au système ASR et de comparer les résultats avec le système ASR basé sur le modèle RNN.

BIBLIOGRAPHIE

[Jur, 2021] (2021). Speech and language processing : an introduction to natural language processing, computational linguistics, and speech recognition. – Cité pages 14, 15 et 18.

[Association, 1999] Association, I. (1999). *Handbook of the International Phonetic Association: A Guide to the Use of the International Phonetic Alphabet*. Cambridge University Press. – Cité page 24.

[Broadbent, 1967] Broadbent, D. E. (1967). Word-frequency effect and response bias. *Psychological review*, 74(1):1. – Cité page 14.

[Calvert et al., 1997] Calvert, G. A., Bullmore, E. T., Brammer, M. J., Campbell, R., Williams, S. C., McGuire, P. K., Woodruff, P. W., Iversen, S. D., and David, A. S. (1997). Activation of auditory cortex during silent lipreading. *science*, 276(5312):593–596. – Cité page 14.

[Chomsky and Halle, 1968] Chomsky, N. and Halle, M. (1968). The sound pattern of english. – Cité pages 15, 27 et 46.

[Davenport and Hannahs, 2013] Davenport, M. and Hannahs, S. J. (2013). *Introducing phonetics and phonology*. Routledge. – Cité page 15.

[Davis and Mermelstein, 1980] Davis, S. and Mermelstein, P. (1980). Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. *IEEE transactions on acoustics, speech, and signal processing*, 28(4):357–366. – Cité page 18.

[De Saussure, 1989] De Saussure, F. (1989). *Cours de linguistique générale*, volume 1. Otto Harrassowitz Verlag. – Cité page 13.

[Dehaene et al., 1998] Dehaene, S., Dehaene-Lambertz, G., and Cohen, L. (1998). Abstract representations of numbers in the animal and human brain. *Trends in neurosciences*, 21(8):355–361. – Cité page 13.

[Ghio et al., 2018] Ghio, A., Lalain, M., Giusti, L., Pouchoulin, G., Robert, D., Rebourg, M., Fredouille, C., Laaridh, I., and Woisard, V. (2018). Une mesure d'intelligibilité par décodage acoustico-phonétique de pseudo-mots dans le cas de parole atypique. In *XXXIIe Journées d'Etudes sur la Parole*, pages 285–293. ISCA. – Cité pages 27 et 46.

[Hahn et al., 2012] Hahn, S., Vozila, P., and Bisani, M. (2012). Comparison of grapheme-to-phoneme methods on large pronunciation dictionaries and lvcsr tasks. In *Thirteenth Annual Conference of the International Speech Communication Association*. – Cité page 25.

[Huang and Deng, 2010] Huang, X. and Deng, L. (2010). *An Overview of Modern Speech Recognition*, pages 339–366. Chapman Hall/CRC, handbook of natural

language processing, second edition, chapter 15 (isbn: 1420085921) edition. – Cité page 15.

[Jakobson et al., 1951] Jakobson, R., Fant, C. G., and Halle, M. (1951). Preliminaries to speech analysis: The distinctive features and their correlates. – Cité page 15.

[Jouvet et al., 2012] Jouvet, D., Fohr, D., and Illina, I. (2012). Evaluating grapheme-to-phoneme converters in automatic speech recognition context. In *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4821–4824. IEEE. – Cité page 19.

[Juang and Rabiner, 2005a] Juang, B. and Rabiner, L. (2005a). Automatic speech recognition - a brief history of the technology development. – Cité page 16.

[Juang and Rabiner, 2005b] Juang, B.-H. and Rabiner, L. R. (2005b). Automatic speech recognition–a brief history of the technology development. *Georgia Institute of Technology. Atlanta Rutgers University and the University of California. Santa Barbara*, 1:67. – Cité page 16.

[Jurafsky, 2000] Jurafsky, D. (2000). *Speech and language processing*. Pearson Education India. – Cité page 19.

[Kurenkov, 2020] Kurenkov, A. (2020). A brief history of neural nets and deep learning. *Skynet Today*. – Cité page 18.

[Lakhotia et al., 2021] Lakhotia, K., Kharitonov, E., Hsu, W.-N., Adi, Y., Polyak, A., Bolte, B., Nguyen, T.-A., Copet, J., Baevski, A., Mohamed, A., et al. (2021). On generative spoken language modeling from raw audio. *Transactions of the Association for Computational Linguistics*, 9:1336–1354. – Cité pages 21 et 45.

[Levelt et al., 1999] Levelt, W. J., Roelofs, A., and Meyer, A. S. (1999). A theory of lexical access in speech production. *Behavioral and brain sciences*, 22(1):1–38. – Cité page 14.

[Liu et al., 2022] Liu, C., Ling, Z.-H., and Chen, L.-H. (2022). Pronunciation dictionary-free multilingual speech synthesis by combining unsupervised and supervised phonetic representations. *arXiv preprint arXiv:2206.00951*. – Cité page 19.

[Lowerre, 1976] Lowerre, B. T. (1976). *The harpy speech recognition system*. Carnegie Mellon University. – Cité page 16.

[Maas et al., 2015] Maas, A., Xie, Z., Jurafsky, D., and Ng, A. (2015). Lexicon-free conversational speech recognition with neural networks. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 345–354, Denver, Colorado. Association for Computational Linguistics. – Cité page 19.

[Marslen-Wilson, 1973] Marslen-Wilson, W. (1973). Linguistic structure and speech shadowing at very short latencies. *Nature*, 244(5417):522–523. – Cité page 20.

[Marslen-Wilson, 1987] Marslen-Wilson, W. D. (1987). Functional parallelism in spoken word-recognition. *Cognition*, 25(1-2):71–102. – Cité page 14.

[Novak et al., 2011] Novak, J., Yang, D., Minematsu, N., and Hirose, K. (2011). Initial and evaluations of an open source wfst-based phoneticizer. *The University of Tokyo, Tokyo Institute of Technology*. – Cité page 25.

[Pieraccini, 2012] Pieraccini, R. (2012). *The voice in the machine: building computers that understand speech*. MIT Press. – Cité page 15.

[Rabiner and Juang, 1986] Rabiner, L. and Juang, B. (1986). An introduction to hidden markov models. *ieee assp magazine*, 3(1):4–16. – Cité page 16.

[Rabiner, 1989] Rabiner, L. R. (1989). A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286. – Cité page 16.

[Ravanelli and Bengio, 2018] Ravanelli, M. and Bengio, Y. (2018). Speaker recognition from raw waveform with sincnet. In *2018 IEEE Spoken Language Technology Workshop (SLT)*, pages 1021–1028. IEEE. – Cité page 18.

[Rutherford et al., 2014] Rutherford, A., Peng, F., and Beaufays, F. (2014). Pronunciation learning for named-entities through crowd-sourcing. – Cité page 19.

[Schalchli, 2022] Schalchli, G. (2022). Lexique4linguists. ORTOLANG (Open Resources and TOols for LANGuage) –www.ortolang.fr. – Cité page 34.

[Schlippe et al., 2010] Schlippe, T., Ochs, S., and Schultz, T. (2010). Wiktionary as a source for automatic pronunciation extraction. In *Eleventh Annual Conference of the International Speech Communication Association*. – Cité page 21.

[Schneider et al., 2019] Schneider, S., Baevski, A., Collobert, R., and Auli, M. (2019). wav2vec: Unsupervised pre-training for speech recognition. *arXiv preprint arXiv:1904.05862*. – Cité page 18.

[Warren, 1970] Warren, R. M. (1970). Perceptual restoration of missing speech sounds. – Cité page 14.

ANNEXE

A.1 Convention des unités phonologiques Lexique3

Voyelles				Consonnes		
Codes Lexique	Exemples	Sons nommés		Codes Lexique	Exemples	Sons nommés
a	bat, plat	A		p	père, soupe	p (occlusive)
i	lit, émis	I		b	bon, robe	b (occlusive)
y	lu	U		t	terre, vite	t (occlusive)
u	roue	Ou		d	dans, aide	d (occlusive)
o	peau, mot	o (fermé)		k	carré, laque	k (occlusive)
O	éloge, fort	o (ouvert)		g	gare, bague	g (occlusive)
e	été	e-fermé		f	feu, neuf	f (fricative)
E	paire, treize	e-ouvert		v	vous, rêve	v (fricative)
°	abordera	schwa élidable		s	sale, dessous	s (fricative)
2	deux	e-fermé		z	zéro, maison	z (fricative)
9	œuf, peur	e-ouvert		S	chat, tâche	ch (fricative)
5	cinq, linge	in (voy. Nasale)		Z	gilet, mijoter	ge (fricative)
1	un, parfum	un (voy. nasale)		m	main, femme	m (cons. nasale)
@	ange	an (voy. nasale)		n	nous, tonne	n (cons. nasale)
§	on, savon	on (voy. nasale)		N	agneau, vigne	gn (c. nasale palat.)
3	parvenu	schwa non élidable		l	lent, sol	l (liquide)
Semi-Voyelles				R	rue, venir	R
j	yeux, paille	y (semi-voyelle)		x	jota	jota (emprunt espagn.)
8	huit, lui	ui (semi-voyelle)		G	camping	ng (emprunt angl.)
w	oui, nouer	w (semi-voyelle)				

FIGURE A.1 – Liste des symboles phonétiques de Lexique3 [1]

A.2 Extrait du script wper.py

Ce script Python est un extrait du script wper.py que nous avons écrit vu dans la section 2.4.3. Il nécessite les téléchargements des modules python suivants : `weighted-levenshtein`, `numpy` et `pandas`.

```
insert_costs = np.ones(128, dtype=np.float64)
delete_costs = np.ones(128, dtype=np.float64)
```

1. http ://www.lexique.org/?page_id=286

```
substitute_costs = np.ones((128, 128), dtype=np.float64)
# dfc : data frame contenant une matrice du poids des
   consonnes
# dfv : data frame contenant une matrice du poids des
   voyelles
# Par exemple, dfc.loc['p','g'] retourne le poids de
   substitution de /p/ par /g/
for i in idxc:
  for idx in dfc.index:
    substitute_costs[ord(i), ord(idx)] = (int(dfc.loc[idx,
       i]) / 10) + 1
for i in idxv:
  for idx in dfv.index:
    substitute_costs[ord(i), ord(idx)] = (int(dfv.loc[idx,
       i]) / 10) + 1
def openfile(fichier):
  dico = {}
  f = open(fichier, 'r')
  for line in f.readlines():
    line = line.strip()
    mot, pron = line.split('\t')
    pron = pron.replace(' ', '')
    dico[mot] = pron
  return dico
refdico = openfile(dict_train)
hypdico = openfile(dict_test)
for k, v in refdico.items():
  ref = v
  hyp = hypdico[k]
  cout = lev(ref,
             hyp,
             insert_costs=insert_costs,
             delete_costs=delete_costs,
             substitute_costs=substitute_costs)
  wper = round((cout / len(ref) * 100.0), 2)
  print(k, ref, hyp, wper)
```

A.3 Liste des abréviations

AM	Acoustic Model
API	Alphabet Phonétique International
ASR	Automatic Speech Recognition
E2E	End-to-End
FTT	Fast Fourier Transform
G2P	Grapheme-to-Phoneme
GMM	Gaussian Mixture Model
GSLM	Generative Spoken Language Model
HMM	Hidden Markov Model
HMM-GMM	Hidden Markov Model-Gaussian Mixture Model
IA	Intelligence Artificielle
IOT	Internet Of Things
LM	Language Model
LV	Large Vocabulary
MFCC	Mel-Frequency Cepstral Coefficient
OCR	Optical Character Recognition
OOV	Out Of Vocabulary
PER	Phoneme Error Rate
PM	Pronunciation Model
SAMPA	Speech Assessment Methods Phonetic Alphabet
WER	Word Error Rate
WFST	Weighted Finite State Transducer
WPER	Weighted Phoneme Error Rate
WWW	World Wide Web