Reconnaissance automatique du niveau de langue à l'oral

MANSERI Kéhina[(1)] SILAI Ioana Madalina[(1)]
[(1)]Université Paris Nanterre

1 Introduction

La tâche présentée dans ce rapport consiste en la classification multiclasse du niveau de langue de locuteurs à partir d'enregistrements audio.

Nous allons d'abord exposer la méthode de collecte et de traitement automatique des données, puis introduirons les architectures de deep learning que nous avons sélectionnées. Enfin, nous présenterons et analyserons nos résultats.

1.1 Présentation des données

Les 863 fichiers présentés ci-dessous ont été collectés sur le site Audio Lingua[1] de l'Académie de Versailles. La copie et l'utilisation des données publiées sur le site sont autorisées dans le cadre pédagogique ou scolaire.

Les enregistrements publiés sur Audio Lingua sont mis à disposition afin de servir de supports de cours pour des enseignants de langues étrangères. Ces derniers sont disponibles dans 14 langues différentes, dont le français utilisé ici.

1.2 Labels des enregistrements

Chaque enregistrement dispose d'une description plus ou moins détaillée : genre, tranche d'âge, région ou ville d'origine du locuteur, thématiques abordées sous forme de mots-clés, résumés, et enfin, le niveau de langue. Chaque locuteur est natif à la langue qu'il emploie dans un enregistrement. Ainsi, le niveau de langue permet dans notre cas d'illustrer des constructions syntaxiques, grammaticales ou encore une prosodie attendue pour un niveau de langue donné.

2 Pré-traitement des données

Les enregistrements accompagnés de leurs descriptions ont été collectés automatiquement à l'aide d'un outil de scraping.

[1] https://audio-lingua.ac-versailles.fr/

2.1 Transcription et aligment

Nos données ont été transcrites à l'aide du modèle Tiny de l'outil Whisper d'Open AI. Le modèle Tiny, si moins précis, nous a permis d'obtenir des transcriptions généralement très fidèles avec un temps d'exécution réduit (Negi, 2024). Les enregistrements et transcriptions résultants ont été alignés à l'aide de WebMaus.

2.2 Segmentation

Chaque enregistrement a ensuite été segmenté de manière à obtenir des fichiers audios plus courts (1 seconde) dont la transcription a pu être récupérée à partir des TextGrids produites par WebMaus. Cette dernière permet de conserver un nombre réduit d'unités langagières, illustrant des variations prosodiques et lexicales, mais également d'obtenir des spectrogrammes pouvant être traités plus rapidement et présentant un équilibre raisonnable entre la perte de résolution et le maintien du contexte.

Une fenêtre d'Hanning (Altexsoft, 2022) a également été appliquée aux segments pour réduire les possibilités de spectral leakage, donc de mauvaise interprétation du signal des frontières de segment. Un overlap a enfin été appliqué à chaque segment pour que ces derniers partagent 50% de leurs contenus avec leurs segments voisins. L'overlap permet de rendre compte du contexte linguistique originel du segment, un aspect pouvant s'avérer pertinent lorsque les segments sont mélangés lors de l'entraînement.

3 Distribution des données

Les classes utilisées dans le cadre de ce projet représentent 5 niveaux de langue : A1, A2, B1, B2 et C1. Les échantillons de la classe C2, si bien collectés, n'ont pas pu être utilisés au vu de leur nombre bien trop bas (4 échantillons avant segmentation).

3.1 Statistiques descriptifs

Nos classes ne sont pas représentées de manière égale au sein de notre corpus.

Classe	Enregistrements	Segments
A1	165 (19.11%)	6 025 (8.66%)
A2	312 (36.15%)	21 521 (30.94%)
B1	319 (36.96%)	34 076 (47%)
B2	58 (6.72%)	7 020 (10.09%)
C1	9 (1.04%)	895 (1.28%)

Table 1: Distribution des enregistrements et segments par classe (en nombre et pourcentage).

Les classes B2 et C1 sont sous-représentées avec uniquement 58 et 9 enregistrements respectivement. Au contraire, les classes A2 et B1 sont sur-représentées, rassemblant les deux tiers des enregistrements.

Une différence de pourcentages entre les distributions par enregistrement et segment peut également être observée. Cette dernière indique que si certaines classes sont sous-représentées, elles disposent d'enregistrements plus longs (car plus de segments) et inversement. La classe A1, la troisième classe la plus représentée en terme d'enregistrements, devient la quatrième en terme de segments avec seulement 8.66% d'échantillons (contre 19.11% avant segmentation).

3.2 Division entraînement/test

Afin de prendre en compte ce déséquilibre de classe, nous avons décidé de diviser nos données de manière à obtenir un nombre équivalent d'échantillons de chaque classe dans chaque ensemble.

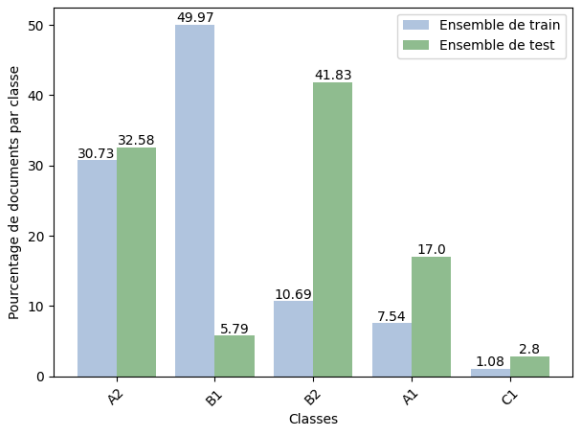

Figure 1: Répartition des segments en fonction des classes dans les ensembles d'entraînement et de test.

Nous avons de plus fait en sorte de ne pas inclure un même locuteur dans les ensembles d'entraînement et de test afin que notre modèle ne se base pas sur l'individu pour effectuer ses prédictions.

Comme le montre la figure 1, cette séparation des locuteurs a empêché la répartition équilibrée entre les deux ensembles, notamment pour les niveaux B1 et B2. Nous avons cependant pu atteindre des répartitions satisfaisantes pour les classes A2 et C1.

3.3 Influence des autres caractéristiques

Enfin, il est important de noter que le niveau de langue peut être associé à plusieurs caractéristiques des locuteurs. Si nous observons la répartition des tranches d'âge en fonction du niveau, nous pouvons remarquer une diminution du nombre de jeunes locuteurs pour les niveaux plus élevés et des pourcentages de locuteurs adultes inégaux pour chaque catégorie.

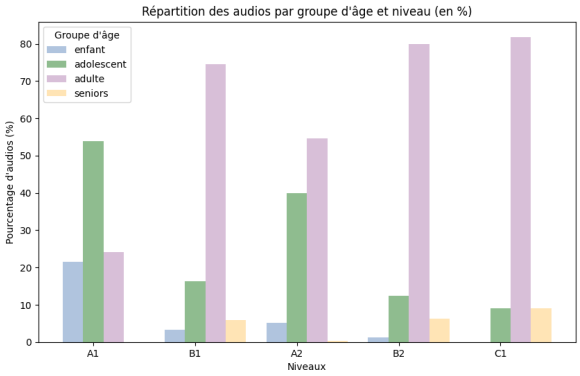

Figure 2: Répartition des tranches d'âge en fonction des niveaux (pour les audios).

Il faut donc garder en tête que les prédictions de nos modèles peuvent être grandement influencées par ces éléments et pourraient, par exemple, attribuer un niveau bas à chaque enregistrement d'enfant. D'autres caractéristiques, comme la région d'origine, pourraient également influencer nos résultats.

4 Entraînements

Les entraînements effectués l'ont été à partir de deux architectures de réseaux de neurones différentes : Dense Neural Network (DNN) et Convolutional Neural Network (CNN). Nous souhaitions tester ces deux architectures afin d'observer de potentielles différences de performance.

4.1 Dense Neural Network

Les DNN sont composés de couches denses complètement connectées et n'utilisent pas de couches convolutives ou récurrentes.

4.1.1 Input des modèles

Nos fichiers audio ont été vectorisés à l'aide de Wav2Vec, un modèle auto-supervisé permettant d'extraire et de représenter une diversité de features concernant la prosodie, la temporalité ou encore certaines informations phonétiques d'un enregistrement donné. Ces embeddings peuvent être composés de plusieurs niveaux d'informations et performent en général convenablement sur des extraits courts comme les nôtres.

Les essais effectués à l'aide des DNN l'ont été à partir d'embeddings obtenus sur les segments initiaux et aussi sur les segments auxquels ont été ajoutés un white noise, donc un bruit aléatoire avec une amplitude basse, afin d'améliorer la généralisation du modèle.

4.1.2 Structure des réseaux

Chaque entraînement a été réalisé à partir de trois structures distinctes de réseaux. La première (dite "complexe") comprend plusieurs couches denses avec des régularisations L2 ainsi que des couches de normalisation par batch. La deuxième (dite "moyenne") utilise également des couches denses avec régularisation L2 et normalisation par batch, mais elle réduit la taille de la première couche dense (512 au lieu de 768 neurones). Finalement, la troisième (dite "simple"), comporte trois couches denses principales avec un dropout agressif (de 0.5) permettant de limiter le surapprentissage.

4.2 Convolutional Neural Network

Les CNN (Convolutional Neural Networks) prennent en input une représentation visuelle dans le cas de l'audio pour lesquels ils sont largement utilisés. Les features sont extraits automatiquement grâce à des filtres convolutifs qui détectent des motifs locaux, comme des contours, des textures ou des structures fréquentielles, permettant ainsi une classification et une analyse précise.

4.2.1 Vectorisation

Le signal de chaque segment audio a été transformé selon l'échelle de Mel, mieux adaptée à la manière dont les humains perçoivent les fréquences sonores. En effet, cette échelle est logarithmique, reflétant le fait que l'oreille humaine est plus sensible aux variations dans les basses fréquences qu'aux hautes fréquences.

Ces signaux ont ensuite été convertis en spectrogrammes de Mel, eux-mêmes transformés à leur tour en images afin de pouvoir servir d'input au CNN. Les images obtenues sont en noir et blanc et en résolution 50x50 et 128x128 pour observer l'impact de la perte d'informations.

4.2.2 Structure des réseaux

L'architecture utilisée pour le CNN commence par deux couches convolutionnelles, une première comportant 32 filtres de taille 3×33×3, et une seconde avec 64 filtres de même taille. Ces couches sont activées par la fonction ReLU, connue pour sa robustesse dans la formation de réseaux profonds. Une couche de pooling est ensuite appliquée pour réduire la taille des données puis suivie d'une couche de dropout de 25%.

Les données sont enfin aplaties puis passent à travers une couche dense suivie d'une régularisation Dropout plus agressive. La dernière couche est une couche d'activation Softmax permettant d'attribuer une probabilité de classification à chaque catégorie pour chaque échantillon.

5 Évaluation des modèles

5.1 Dense Neural Network

Architecture	Batch Size	Test Accuracy	Test Loss
Simple	50	0.57	1.23
Simple	128	0.57	1.15
Simple	256	0.58	1.15
Moyenne	50	0.51	1.35
Moyenne	128	0.43	1.52
Moyenne	256	0.53	1.47
Complexe	50	0.40	1.30
Complexe	128	0.52	1.43
Complexe	256	0.53	1.50

Table 2: Accuracy et loss obtenues avec plusieurs structures et de batch sizes pour le DNN.

Les performances les plus satisfaisantes sont obtenues avec des structures simples de réseaux.

Ces résultats peuvent premièrement s'expliquer par des contraintes trop élevées liées à la normalisation L2, au dropout et à la normalisation par batch pouvant empêcher le réseau de converger vers un minimum global. Les embeddings Wav2Vec, déjà riches en informations, pourraient également causer du surapprentissage lorsqu'ils sont utilisés avec un réseau plus complexe.

De plus, avec un batch size plus important, le modèle dispose de plus d'exemples pour ajuster ses poids, ce qui conduit à une descente de gradient plus stable et donc à une estimation plus précise de ces derniers.

Concernant les tests effectués sur l'ensemble bruité, les résultats obtenus sont légèrement supérieurs et confirment que les modèles simples surpassent systématiquement les architectures plus complexes. Un batch size de 50 entraîne les meilleurs résultats dans toutes les architectures, alors que le pire résultat est obtenu avec une architecture moyenne et un batch size de 256 menant à une accuracy de seulement 0.26.

Architecture	Batch Size	Test Accuracy	Test Loss
Simple	50	0.58	1.16
Simple	128	0.58	1.16
Simple	256	0.55	1.20
Moyenne	50	0.59	1.36
Moyenne	128	0.52	1.37
Moyenne	256	0.26	1.74
Complexe	50	0.58	1.16
Complexe	128	0.58	1.16
Complexe	256	0.55	1.20

Table 3: Accuracy et loss obtenues avec des données bruitées en entraînement, plusieurs structures et valeurs de batch size pour le DNN.

Une autre expérience a été menée en utilisant les transcriptions textuelles des segments audio, vectorisées avec le modèle pré-entraîné BERT, comme entrée du réseau dense. Les résultats obtenus montrent que la structure complexe atteint une accuracy de 0.58 avec une loss de 1.13, tandis que la structure simple produit une accuracy légèrement inférieure de 0.54 et une loss de 1.17. Avec la structure dite "moyenne", l'accuracy est de 0.53 et la loss de 1.09. Ces performances suggèrent que l'utilisation de vecteurs BERT, riches en contextes sémantiques, peut améliorer la convergence du modèle complexe, mais n'apporte pas nécessairement d'avantage significatif par rapport aux embeddings Wav2Vec associés à des architectures plus simples. Par ailleurs, les résultats légèrement meilleurs de la structure moyenne indiquent que des techniques de régularisation plus agressives peuvent aider à généraliser sur des données textuelles, tout en limitant les effets de surapprentissage.

Architecture	Batch Size	Test Accuracy	Test Loss
Simple	50	0.54	1.17
Simple	128	0.53	1.18
Simple	256	0.54	1.19
Moyenne	50	0.53	1.09
Moyenne	128	0.54	1.10
Moyenne	256	0.57	1.09
Complexe	50	0.58	1.13
Complexe	128	0.45	1.18
Complexe	256	0.57	1.13

Table 4: Accuracy et loss obtenues avec plusieurs valeurs de batch size pour le DNN et les embeddings BERT.

5.2 Convolutional Neural Network

Voici les principaux résultats obtenus avec des résolutions d'images et des tailles de batch différentes :

Résolution	Batch Size	Test Accuracy	Test Loss
128 x 128	50	0.50	1.6
50 x 50	50	0.54	1.24
50 x 50	128	0.55	1.22
50 x 50	256	0.56	1.15

Table 5: Accuracy et loss obtenues avec plusieurs valeurs de résolution et de batch size pour le CNN.

Nous observons que la résolution d'entrée 50×50 donne de meilleurs résultats que 128×128, malgré la perte d'informations due à la réduction de taille. De plus, une augmentation de la taille des batches semble stabiliser l'apprentissage et améliorer les performances globales.

La meilleure accuracy obtenue avec une résolution réduite peut premièrement s'expliquer par le fait qu'une résolution élevée peut inclure des détails inutiles, et donc du bruit, pouvant rendre l'apprentissage plus difficile. Il est également possible qu'une résolution plus faible, correspondant donc à moins de données à traiter pour le modèle, empêche, ou du moins réduise, le risque de surapprentissage.

L'amélioration progressive des performances avec des tailles de batch plus grandes s'explique par une meilleure estimation des gradients. Avec un batch size plus important, le modèle dispose de plus d'exemples pour ajuster ses poids, ce qui conduit à une descente de gradient plus stable et efficace.

Les améliorations observées avec l'augmentation des tailles de batch illustrent

une meilleure estimation des gradients mais restent très faibles.

6 Conclusion

Ce projet a exploré la classification automatique des niveaux de langue à partir d'enregistrements audio en combinant des méthodes de traitement audio et textuel avec des architectures de deep learning. Les résultats obtenus montrent que les architectures simples, comme les Dense Neural Networks peu profonds, offrent des performances globalement meilleures pour ce type de tâche. Cette observation s'explique notamment par les propriétés des embeddings utilisés. Les CNN utilisant des Mel spectrogrammes se sont avérés prometteurs pour capturer les variations acoustiques, bien que les données déséquilibrées et la taille réduite de certaines classes aient limité la performance globale. Des solutions telles qu'une augmentation des données ou l'intégration de techniques semi-supervisées pourraient remédier à ces limites. Enfin, l'intégration de vecteurs BERT a démontré un potentiel intéressant pour exploiter le contenu sémantique des transcriptions. Une association des embeddings BERT et Wav2Vec est une piste intéressante pour obtenir de meilleurs résultats. Un intêret particulier doit également être apporté aux caractéristiques des locuteurs et à leurs impacts sur les performances de classification.

References

Altexsoft. 2022. Audio analysis with machine learning: Building ai-fueled sound detection app. *Altexsoft*.

Manish Negi. 2024. Whisper for asr: All you need to know! *Medium*.