Projet INF8225 – Modèle text to speech

Hugo Petrilli 2306643
Antoine Leblanc 2310186

1 Introduction

Le développement récent des transformers a révolutionné l'intelligence artificielle dans de nombreux domaines, aussi bien dans les NLP, que la computer vision ou les modèles génératifs. Nous cherchons à savoir si le développement de ses transformers a aussi permis de révolutionner les modèles Text To Speech.

Dans ce projet, nous présenterons tout d'abord l'évolution des modèles text to speech à travers le temps. Nous détaillerons les étapes pour transformer un texte en audio et nous expliquerons ensuite en détail le fonctionnement des modèles text to speech les plus récents. Nous implémenterons un modèle de text to speech classique à l'aide de Speech T5 et un modèle de multi speaker et nous observerons les spectrogrammes et les audios de sortie.

2 L'histoire du Text to Speech

Le Text To Speech est une tâche qui consiste à transformer un texte en un audio. Pour ce faire, de nombreuses méthodes ont été développées au cours du temps.

2.1 Le Vocoder

Le premier pas vers la synthèse vocale électronique est le développement du Voder et du Vocoder dans les années 1930. En 1939, l'ingénieur acoustique et électronique Homer Dudley crée le premier synthétiseur vocal électronique. Cette machine, appelé le Voder est manipulée à l'aide d'un clavier et de pédales qui permettent de moduler les effets sonores et de sortir un son de parole. Homer Dudley développe également le vocoder, qui est un système de codage et de décodage de la parole. Le signal vocal reçu par la machine est divisé en bandes de fréquences, et les caractéristiques de la voix sont extraites à l'aide de filtres. L'utilisation des filtres inverses permet de transformer le signal extrait en un audio. Le vocoder est donc le premier mécanisme permettant de transformer un audio en signal et inversement.

2.2 La synthèse à formants

Dans les années 1970, une autre technique pour le text to speech est développée. Il s'agit de la synthèse à formants. C'est une méthode permettant de simuler les caractéristiques des cordes vocales humaines de manière très technique et de prévoir les sons qui en sortent. Cela permet d'obtenir une voix synthétique.

2.3 La synthèse par concaténation

Dans les années 1980, la synthèse par concaténation est développée. Celle-ci repose sur l'enregistrement de petites unités de parole, les phonèmes, qui sont enregistrés. Pour former des phrases entières, il suffit alors de concaténer tous les phonèmes des phrases. Cela améliore la qualité de l'audio.

2.4 La synthèse statistique paramétrique

Dans les années 2000 apparaît le modèle de synthèse statistique paramétrique. Il s'agit d'un modèle reposant sur des modèles statistiques, notamment les modèles de Markov pour générer des ondes sonores à partir d'entrées textuelles. Il s'agit donc de modèles text to speech.

2.5 L'ajout de réseaux de neurones

Dans les années 2010, des modèles basés sur les réseaux de neurones se développent. Des modèles comme Tacotron chez Google ou WaveNet, chez DeepMind se développent. Ils utilisent des réseaux de neurones profonds avec des CNN et des LSTM. Cela permet d'avoir des voix plus naturelles et fluides et expressives.

2.6 Les transformers

Dans les dernières années, avec la découverte des transformers, des modèles comme Transformer TTS ou Tacotron 2 se développent. Ils utilisent des transformers généralement de type Encoder-Decoder et permettent d'augmenter la qualité des audios. C'est ces types de modèles que nous cherchons à étudier dans notre projet.

Les modèles les plus récents sont également basé sur les transformer mais peuvent permettre plus de chose. Ils peuvent synthétiser la voix de quelqu'un à partir d'audios de cette personne, ils peuvent traiter des tâches de multi speaker ou peuvent détecter la langue d'un texte pour fournir un audio avec le bon accent.

3 Les modèles TTS avec transformer

Le premier modèle qui se rapproche le plus du modèle Transformer est Transformer TTS proposé par une équipe de Microsoft en 2019. L'objectif était d'améliorer le modèle Tacotron proposé par Google, qui était un modèle Seq2Seq basé sur des LSTM et de l'attention, en intégrant l'auto-attention multi-tête. L'objectif de ce modèle, comme tous les autres modèles TTS, est de prendre une phrase en entrée et de renvoyer un spectrogramme. Ce spectrogramme est ensuite converti en audio par un vocodeur.

3.1 Le spectrogramme de Mel

Le spectrogramme de Mel est un outil très important dans le TTS puisque c'est lui qui fait le lien entre sortie du modèle et entrée du vocodeur afin de générer l'audio. Tout d'abord, regardons ce qu'est un spectrogramme.

Un audio est échantillonné (avec un taux d'échantillonnage) pour pouvoir le numériser et le rendre compréhensible à l'utilisateur, plus le taux est élevé, meilleure la numérisation de l'audio sera.

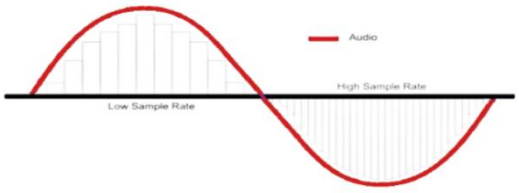


Figure 1 : Audio échantillonné

A partir de cet échantillonnage on peut construire le spectre du signal avec les fréquences calculées par une transformée de Fourier. Le spectrogramme va ainsi être seulement la concaténation des différents spectres calculés sur des petites durées de temps.

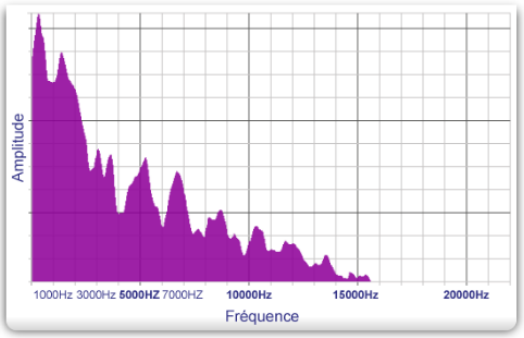


Figure 2 : Spectre du signal

Enfin, la technologie TTS utilise un autre type de spectrogramme, qui est le spectrogramme de Mel. Ce spectrogramme a été créé car l'oreille humaine détecte mieux les sons de basse fréquence que les hautes. Pour construire ce spectrogramme, on vient seulement filtrer chaque spectre et projeter les fréquences sur le l'échelle de Mel (en Hz) qui est plus adapté pour l'oreille humaine.

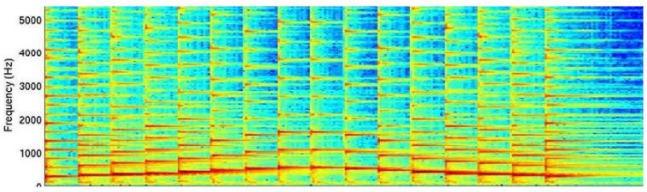


Figure 3 : Spectrogramme de Mel

Ainsi, on comprend que les sorties de notre Transfomer vont être les différentes valeurs des fréquences pour un instant donné (un instant est égale à une très courte période de temps), et cela pour le nombre d'instant souhaité. On peut aussi noter que l'on a ici un hyperparamètres lié à notre modèle qui est lié à la qualité de l'audio : le taux d'échantillonnage. Un modèle est entraîné sur un taux d'échantillonnage unique et ne pourra être utilisé pour du fine tuning ou de l'inférence avec un taux d'échantillonnage différent.

Après cela, le spectrogramme en sortie passe dans un vocoder qui transforme l'image en audio. Le vocodeur utilisé est WaveNet qui est réseau type CNN.

3.2 L'architecture de Transformer TTS

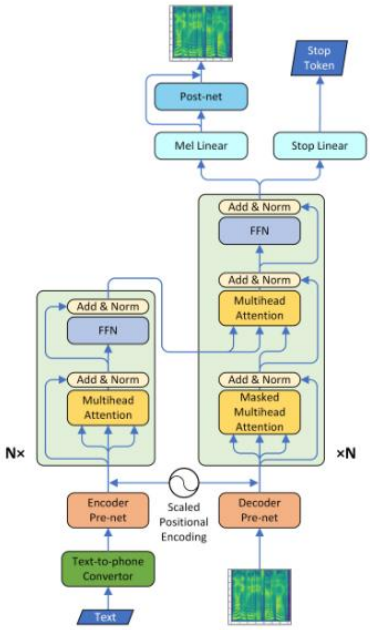


Figure 4 : Architecture de Transformer TTS

Le transformeur TTS reçoit un texte, et renvoie le spectrogramme de Mel de l'audio correspondant. Il est formé autour d'un transformeur de type Encoder-Decoder. L'architecture de l'encodeur et du décodeur est similaire à celle des encodeurs et décodeurs dans les modèles NLP. On observe un

plongement du texte et un plongement positionnel avant d'envoyer les données dans l'encodeur. Une fois dans l'encodeur, l'attention multi tête est calculée puis dans le décoder, il y a calcul de l'attention multi-tête masquée et de la cross attention. En sortie du décodeur, il y a une couche linéaire.

Il y a cependant des différences notables entre les modèles de NLP et le transformer TTS. Le texte est converti en phonème au début du modèle. Les phonèmes sont la manière de prononcer les mots. Cela permet d'être plus précis sur le texte à convertir en audio.

De plus, le décodeur prend et renvoie des images de la forme du spectrogramme de Mel. Cela signifie que les blocs Decoder Pre Net et Decoder Post Net sont des blocs de pré traitement et post traitement de l'image. De manière plus précise, Decoder Pre Net prend en entrée le spectrogramme de Mel de l'input donné par le décodeur et est un réseau de neurones de 3 réseaux connectés, avec une activation ReLu. Le bloc Mel linear et Decoder Post Net est composé de deux modules. Le premier est un layer linear avec comme entrée l'output du décodeur, qui prédit le spectrogramme de Mel. Le deuxième bloc est composé de 5 layers de convolution 1D pour affiner la prédiction du spectrogramme de Mel.

Pour résumer, le modèle Transformer TTS prend en entrée un audio qu'il convertit en phonème et qu'il envoie dans le transformer de type encoder/decoder et qui est centré sur le calcul de l'attention multi-tête. Le modèle prédit alors le spectrogramme de Mel du texte en entrée. Après cela, il faut utiliser un vocoder pur transformer le spectrogramme de Mel en audio.

4 Les modèles implémentées

Pour ce projet, nous avons implémentées différents modèles.

4.1 Speech T5

Le modèle Speech T5 est un modèle développé par Microsoft et inspiré du modèle T5 (Text To Text Transfer Transformer). Le modèle T5 est un modèle de traitement de texte reposant sur un transformer de type encodeur-décodeur. Il repose également sur le principe d'attention. Speech T5 est donc un type de modèle text to speech utilisant les transformers. Il peut gérer tout type de tâches lié à la parole, aussi bien le text to speech que la reconnaissance vocale ou la traduction vocale. Dans ce projet, nous nous intéresserons à l'utilisation text to speech de Speech T5.

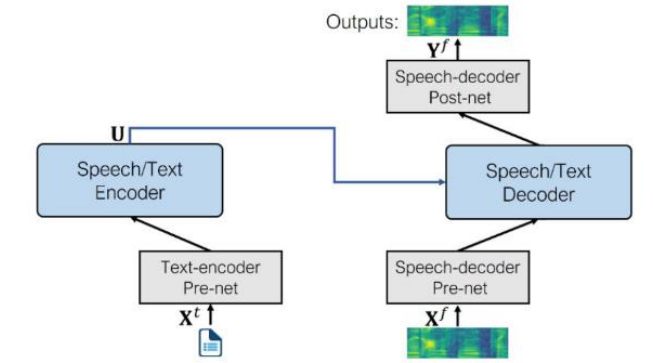


Figure 5 : Architecture du modèle TTS de Speech T5

La version text to speech de Speech T5 est similaire, pour ne pas dire quasi identique à celle du transformer TTS. C'est ce modèle que nous allons implémenter et tester dans la suite du projet.

4.2 Speech T5 multi speaker

Un aspect intéressant des modèles Text-To-Speech est de pouvoir générer des audios mais pour plusieurs voix, accents ou langues différents. Le Transformer TTS est un premier modèle simple qui ne peut générer qu'une seule voix, accent ou langue à la fois. Pour pouvoir générer des variantes il faut récupérer le modèle pré-entraîné et le réentraîner sur la variante de notre choix.

Concernant le modèle SpeechT5 il est possible avec un seul modèle de générer différentes voix, accents ou même langue. La technologie fonctionne en récupérant un plongement de la voix de la personne qui parle et de concaténer ce plongement à la sortie et de passer ce nouveau vecteur dans une couche linéaire pour générer un nouveau spectrogramme de Mel.

4.3 Entrainement de notre voix

Nous nous sommes appuyés sur le tutoriel offert par Hugging Face sur SpeechT5 qui propose l'entraînement du modèle sur une base de données d'audio néerlandais. Nous allons plutôt essayer de faire du voice cloning avec la voix d'A. Leblanc, auteur de ce rapport. Pour constituer la base de données, Antoine a effectué 116 enregistrements entre 5 et 8 secondes de sa voix à partir de phrases générées par ChatGPT. Il s'est enregistré à l'aide de son téléphone personnel et chez lui avec le bruit environnant qu'il pourrait y avoir.

Notre objectif est de faire du voice cloning avec du zero shot, few shots et un entraînement un peu plus conséquent. Le dataset a été divisé en deux : un petit contenant 16 données et un plus grand contenant 102 données.

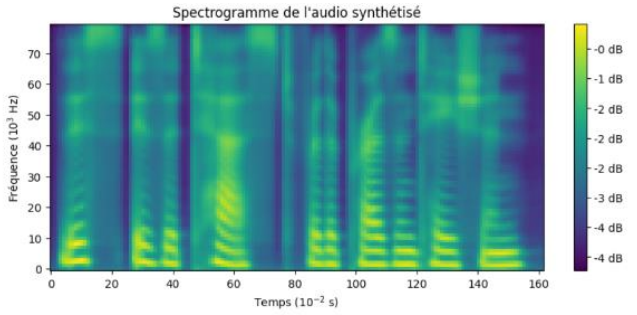
5 Résultats de nos modèles

Nos tests effectués sont disponibles au lien github suivant : https://github.com/AntoineLeblancFr/INF8225/tree/main

5.1 Le modèle text to speech classique (modèle pré-entraîné)

Pour créer notre modèle de text to speech, nous utilisons des modèles pré-entraînés. Nous utilisons la librairie transformers, avec des modules comme SpeechT5Processor, SpeechT5ForTextTospeech ou SpeechT5HifiGan, ce qui permet d'avoir un modèle ou un vocoder. Nous utilisons également des datasets importés.

Après, nous choisissons le texte à transformer en audio. Le modèle nous renvoie un spectrogramme.



Spectrogramme de l'audio synthétisé

En utilisant le vocoder, nous transformons alors ce spectrogramme en audio, que nous pouvons enregistrer.

5.2 Le modèle text to speech multi-speaker

Pour le multi-speaker nous réutilisons le même modèle, seulement il faut générer pour chaque exemple un plongement permettant de représenter la voix du speaker pour cet exemple. Le modèle se charge de faire la concaténation seule. C'est la même chose pour l'inférence, il suffit de transmettre un plongement de la voix que l'on désire.

5.3 Entraînement de notre voix : Première tentative

Les hyperparamètres choisis pour l'entraînement de notre voix sont disponibles dans notre notebook présent à notre lien github.

On peut voir, tout d'abord, que pour le zero shot la voix est compréhensible avec un accent anglais. Cela s'explique par le fait que le modèle a été pré-entraîné sur un dataset d'audio en anglais. Ainsi, le zero-shot fonctionne si on s'exprime dans la bonne langue, sinon on a un audio avec un accent anglais.

Pour le few shot, l'audio ressort presque inaudible. Cela peut s'expliquer par le fait que l'on entraîne le modèle sur un petit dataset français et que l'on a trop peu d'exemples pour pouvoir obtenir un résultat convenable.

Enfin, pour le fine tuning sur le plus gros dataset, l'audio reste de mauvaise qualité mais on discerne tout de même les sonorités les plus importantes et on reconnaît un peu la voix d'A. Leblanc. Tous les audios sont disponibles sur le notebook.

5.4 Entraînement de notre voix : Deuxième tentative

Ainsi, notre plus grand problème était un manque de connaissances du français de la part du modèle. Pour remédier à cela, nous avons récupérer un plus grand dataset et n'avons conservé que 9532 exemples, ce qui fait un entraînement de 3h30 sur GPU T4. Une fois le modèle pré-entraîner sur ces exemples nous avons réitéré les expériences.

La première chose que l'on peut remarquer est que tous les audios sont cette fois-ci audible. En zero-shot learning, l'accent anglais a l'air d'avoir disparu mais le modèle renvoie un audio qui répète notre phrase d'entrée plusieurs fois. Le modèle a du mal à déterminer la fin de la sortie audio.

En few shots learning, notre modèle renvoie une phrae audible mas très robotique. On arrive pas à discerner la voix d'Antoine.

Enfin pour le fine tuning sur notre plus grand dataset de 10 minutes d'audio, on obtient un résultat un peu meilleur et on arrive à discerner la voix d'Antoine – même si l'audio reste très robotique.

Tous les audios sont retrouvables avec le sigle « v2 » à la fin des noms des audios.

6 Conclusion

En conclusion, nous pouvons dire que le résultat du voice cloning est plutôt mitigé puisque l'on obtient un audio proche de notre voix mais il reste très robotique. Cela s'explique par le fait qu'il nous manquait du temps. En effet si on souhaitait un résultat de meilleur qualité, on aurait pu entraîner notre modèle sur le grand dataset que l'on avait trouvé qui comptait 258 000 exemples audio et ainsi obtenir un rendu de bien meilleur qualité. Malheureusement, en extrapolant pour 10 000 exemples qui ont nécessités 3h30 d'entraînement, on trouve une approximation de 87h30 pour entraîner notre modèle sur le dataset au complet – ce qui n'est pas réalisable puisque l'on avait pas assez de crédits GPU sur notre compte Google Collab. Pour améliorer le rendu, une idée aurait été de constituer un plus grand dataset de notre voix, la littérature suggère 20 à 25 minutes d'audio pour un rendu de bonne qualité.

On comprend maintenant pourquoi il est facile de copier la voix de politicien comme on a pu le voir sur les réseaux sociaux, puisque l'on dispose d'un grand nombre de données de leur voix via leur discours avec aucune musique de fond et aucun bruit. A contrario, il est plus compliqué d'effectuer du voice cloning sur quelqu'un dont on a pas assez d'audio, même si aujourd'hui des techniques sont développés en utilisant la technologie auto-encodeur pour faire des représentations plus efficaces des voix et ainsi n'avoir besoin que de peu d'exemples pour faire du voice cloning.

Cette pratique soulève aussi des questions car on voit le danger et les arnaques que l'on peut faire grâce à cette technologie. Cependant elle peut aussi permettre à des personnes qui ont perdu leur voix de la retrouver avec un ancien audio d'eux.

Références

[Li *et al*., 2019] Naihan Li, Shujie Liu, Yanqing Liu, Sheng Zhao, Ming Liu and Ming Zhou. *Neural speech synthesis with transformer network* Guizhou, China, 2019.

[Ao *et al*., 2022] Junyi Ao, Rui Wang, Long Zhou, Chengyi Wang, Shuo Ren,Yu Wu, Shujie Liu, Tom Ko, Qing Li, Yu Zhang, Zhihua Wei,Yao Qian, Jinyu Li and Furu Wei. *SpeechT5: Unified-Modal Encoder-Decoder Pre-Training for Spoken Language Processing* Hong Kong, 2022.

Pratap, V., Xu, Q., Sriram, A., Synnaeve, G., & Collobert, R. (2020). MLS: A Large-Scale Multilingual Dataset for Speech Research. *ArXiv*, *abs/2012.03411*.

Hugging Face. (n.d.). Fine-tuning. *Hugging Face Course: Audio*. Retrieved from https://huggingface.co/learn/audio-course/en/chapter6/fine-tuning.