ARN - Practical Work 3

Supervised Learning and Speaker recognition using Neural Networks

Jonathan Friedli, Valentin Kaelin

12.04.2022

Contents

Introduction

Au cours de ce travail pratique, nous avons appliqué la méthodologie vue en cours afin de développer des algorithmes capables de reconnaître des hommes, des femmes et des enfants via des enregistrements de leur voix. Nous avons utilisé l'approche de la validation croisée pour évaluer les performances des neurones entraînés dans le but de sélectionner un modèle final performant. Finalement nous évaluons la performance finale de nos modèles à l'aide de matrice de confusion.

1. Homme vs Femme

Comme demandé, nous utilisons dans cette première expérience uniquement les voix naturelles des hommes et des femmes afin d'entraîner le réseau de neurones à reconnaître le genre de l'interlocuteur.

Il y a 36 fichiers sonores (.wav) contenant des voix de femmes. Il y en a également 36 contenant les voix d'hommes.

Comme dans le laboratoire numéro 1, nous avons transformé les fichiers audio en valeurs grâce aux MFCC. Nous avons ensuite pris la médiane pour chacune des 13 valeurs de la MFCC. Et ce, pour chaque fichier audio. Nous nous retrouvons donc avec un tableau de 13 valeurs pour chaque fichier audio. Nous les avons ensuite concaténés avant de les normaliser. Nous avons fait cette étape pour toutes les expériences à venir.

Nous avons pris le parti de rajouter une 14ème valeur au tableau afin d'indiquer s'il s'agit de la voix d'un homme ou d'une femme avec le codage suivant:

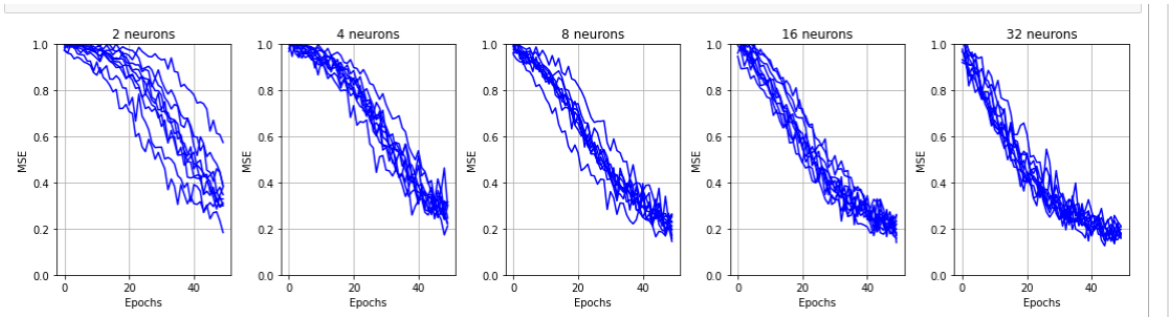
```
1 = Hommes
-1 = Femmes
```

Nous utilisons ces valeurs afin de pouvoir utiliser la fonction d'activation `tanh` qui retourne une valeur entre [-1, 1]. Nous utilisons donc un threshold à 0.0 afin de bien séparer l'intervalle au milieu.

Exploration du nombre de neurones

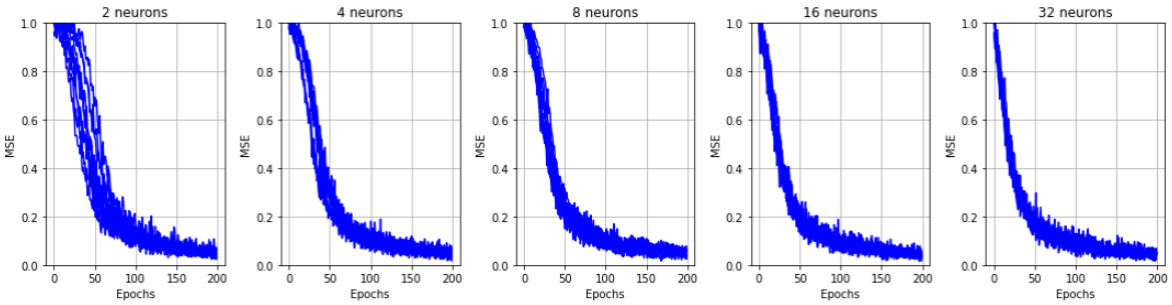
Nous avons ensuite cherché à trouver quel nombre de neurones était le plus adapté à notre situation:

50 epochs:

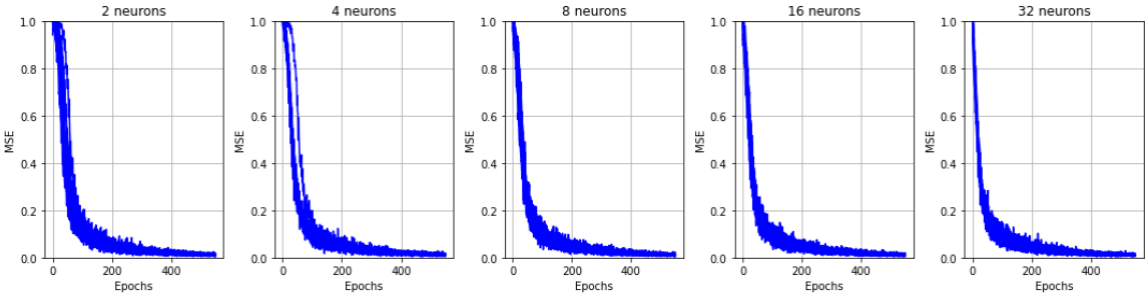


Nous observons que 50 epochs ne sont clairement pas suffisantes.

200 epochs:



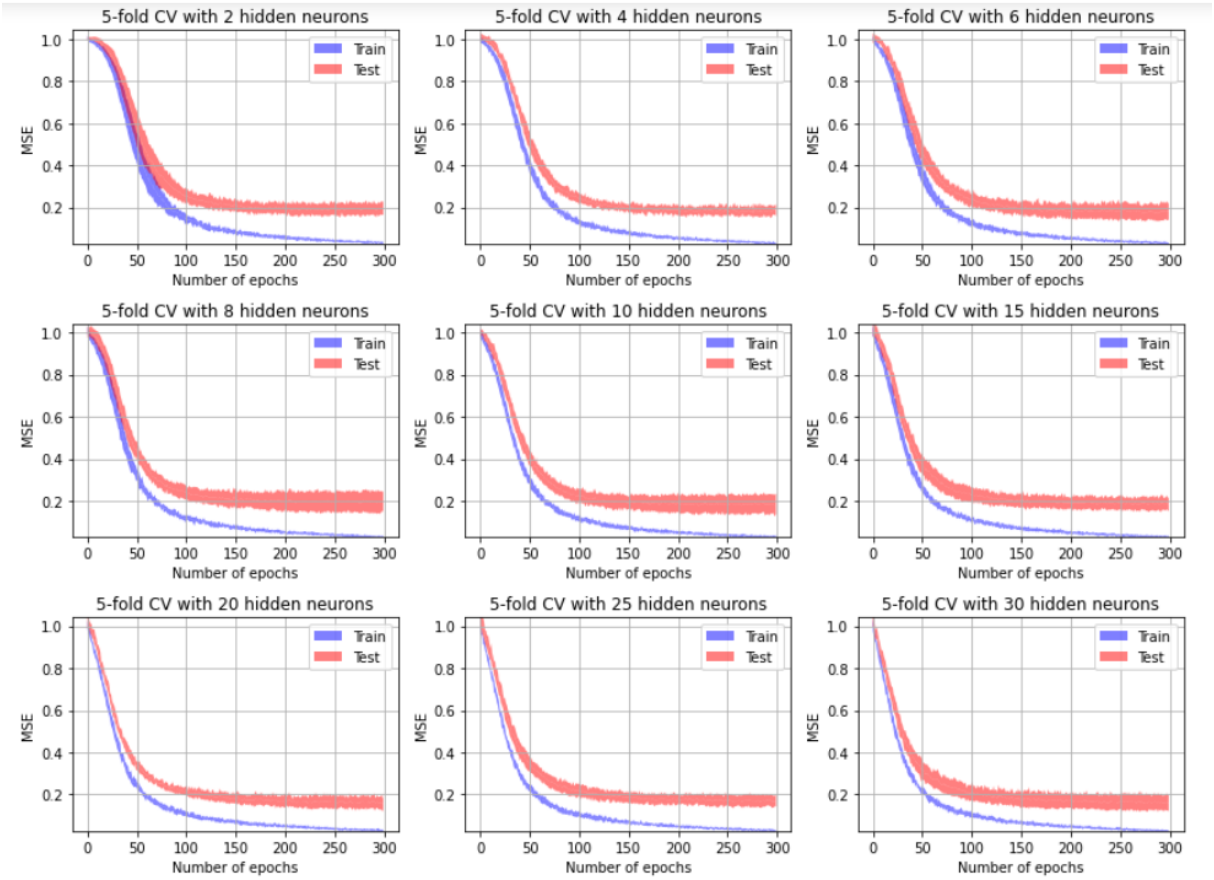
500 epochs:



Nous pouvons voir sur ces trois screen qu'il n'y a pas vraiment d'intérêt à faire plus de 200 epoch. En effet après les 200-250 premières epochs, le résultat change très peu et très lentement, ce qui pourra accélérer nos calculs et observations par la suite. Nous remarquons également qu'il n'y a pas vraiment de différence entre le modèle avec deux neurones et les autres.

Exploration du nombre de neurones dans la couche cachée

Par la suite, nous avons essayé de trouver le nombre de couches de neurones cachées suffisante:



Nous avons tout d'abord essayé 300 epochs afin d'avoir une idée globale des résultats. Nous observons très clairement un overfitting du réseau. Il est donc judicieux de simplifier le réseau bien que les meilleurs résultats soient ceux avec 20 ou 25 neurones. Les deux courbes (train et test) sont assez proches et la MSE est une des plus basses des différents tests réalisés. Cependant, les résultats de 4 neurones sont vraiment proches de ceux avec 25. Il est donc préférable de privilégier ce nombre réduit de neurones à la place de réaliser un réseau très compliqué alors que nous avons très peu de données entrantes (environ 36 données de chaque). Nous choisissons donc 4 par la suite comme nombre de neurones.

Nous pouvons voir que les résultats ne s'améliorent plus vraiment à partir de 200 epochs, c'est donc la valeur que nous allons continuer d'utiliser.

Pour terminer, voici le récapitulatif des hyper paramètres que nous avons trouvé optimaux et que nous avons utilisés pour nos différentes expériences:

```
LEARNING_RATE = 0.001
K = 5
MOMENTUM = 0.8
EPOCHS = 200
N_INIT = 10
N_NEURONS = 4
```

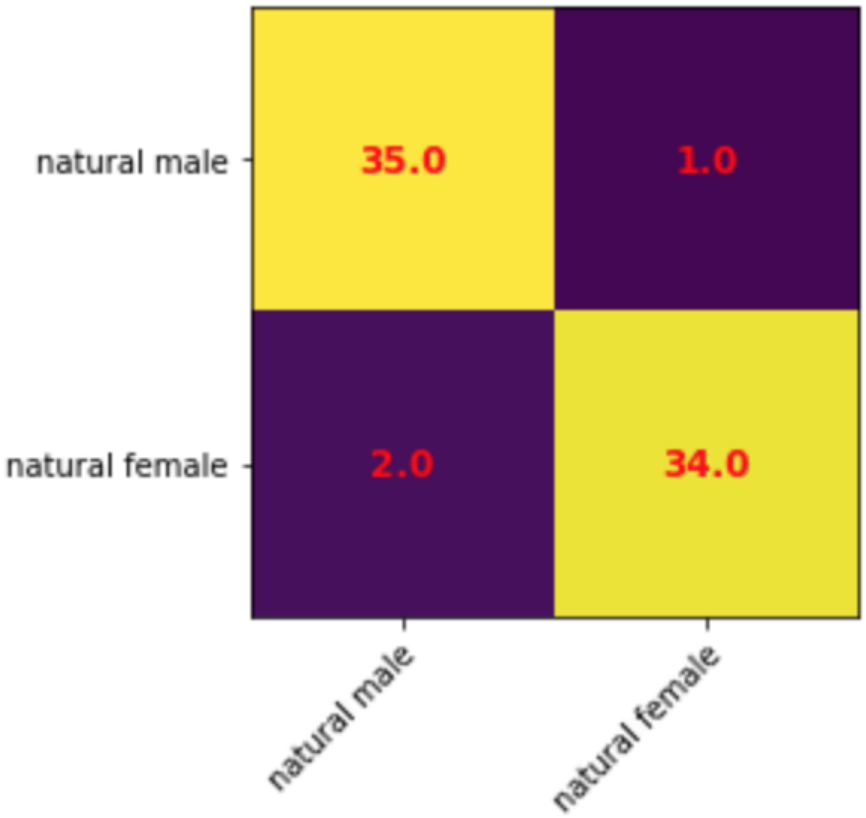
Afin d'arriver à ces résultats, nous avons suivi les consignes vues en cours (ex: modifier le learning rate en multipliant/divisant par un facteur 10, un momentum aux alentours de 0-8, 0.9, etc.)

Matrice de confusion

La matrice de confusion pour cette expérience est la suivante:

```
MSE training : 0.0226
MSE test     : 0.1174

Confusion matrix of selected model:
```




```
f1 score     : 0.9589
accuracy     : 0.9583
```

Nous observons de bons résultats avec un F1-Score très proche de 1 et une erreur très maigre. Ce qui est logique vu les graphiques rencontrés précédemment. Nous pouvons également confirmer l'overfitting avec une erreur environ cinq fois plus grandes lors du test que du training. Le modèle n'a pas plus de peine avec une classe que l'autre, il fonctionne bien avec les deux. Nous sommes

globalement satisfaits des résultats de cette première expérience.

2. Homme vs Femme vs Enfants

Pour cette deuxième étape, nous allons comparer trois classes différentes: les hommes, les femmes et finalement les enfants.

Nous avons toujours 36 entrées hommes, 36 entrées femmes et maintenant en plus 108 enfants.

Afin que le dataset soit équilibré, nous prenons un tiers du dataset des enfants (36 valeurs). Pour faire au mieux, il aurait fallu 36 valeurs aléatoires des 108 initiales mais dans un soucis de rapidité nous avons pris une valeur, toutes les trois. Cela nous permet de prendre 12 valeurs pour chaque type d'enfant (3ans, 5ans et 7ans).

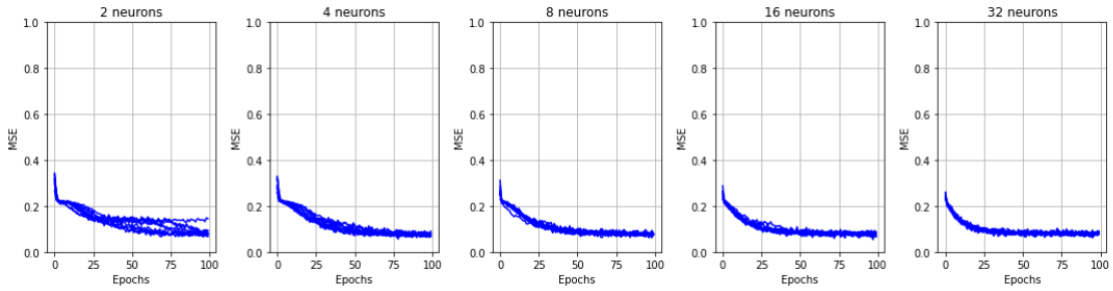
Afin de coder 3 sorties différentes, nous avons utilisé le codage suivant:

(1 quand la condition est vraie, 0 quand ce n'est pas le cas)

Valeur réelle	est homme ?	est femme ?	est enfant ?
Homme	1	0	0
Femme	0	1	0
Enfant	0	0	1

Nous utilisons 1 et 0 car nous avions initialement testé avec 1 et -1 comme pour l'expérience précédente et les résultats étaient très mauvais. Nous avons ajouté un threshold à 0.5 afin de bien séparer l'intervalle en deux lors de la matrice de confusion finale.

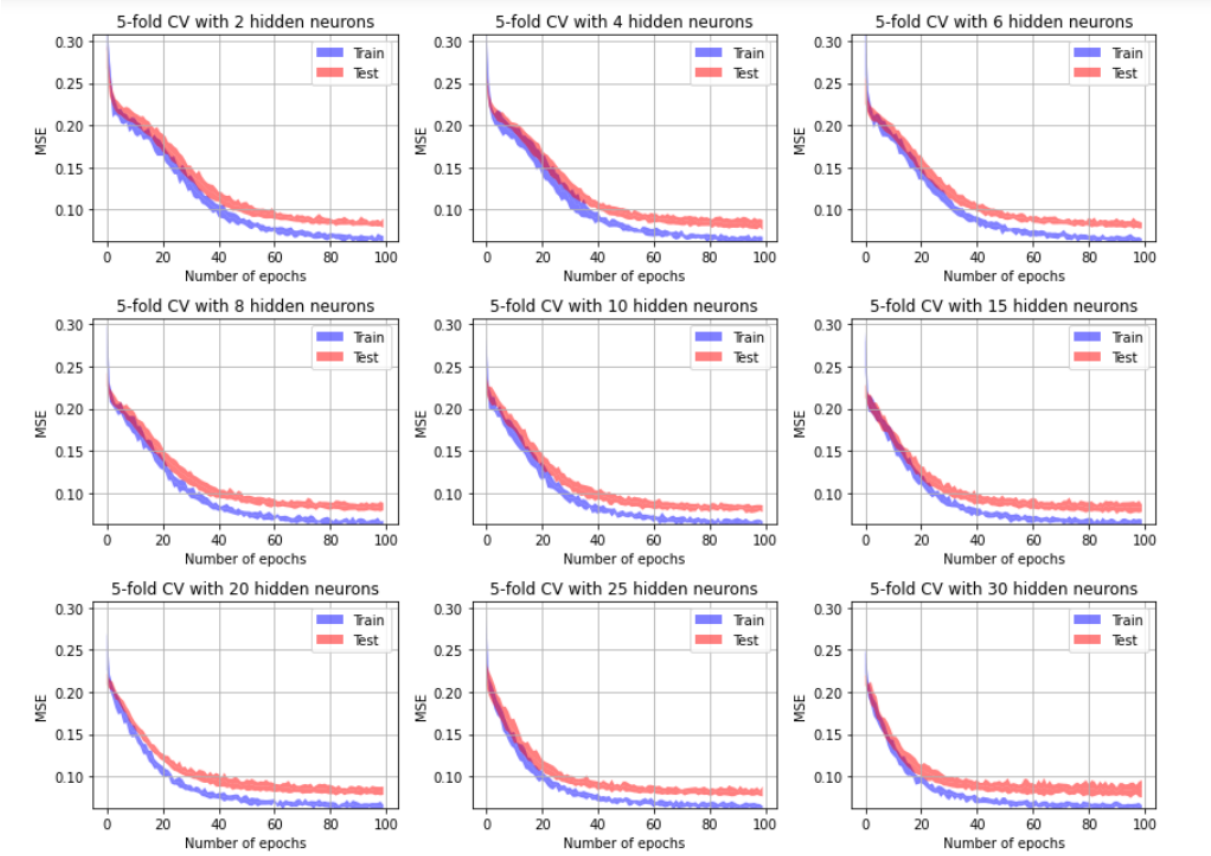
Exploration du nombre de neurones



Ici nous voyons que les résultats sont à partir de 4 neurones. Nous allons donc partir soit sur une

valeurs comprise entre 4 et 8 pour la suite de l'expérience. Nous voyons également que les courbes stagnent aux alentours de 100 épochs.

Exploration du nombre de neurones dans la couche cachée



Ici nous remarquons que tous les résultats sont similaire et légérement overfitté. Par conséquent, nous choisissons de partir avec 8 neurons dans la couche cachée. En effet, c'est ce qui nous semble être la meilleur valeur car nous avons relativement peu de donnée et que ça ne serait pas une bonne idée de prendre une grande valeur alors que nous observons déjà un petit overfitting.

Voici un récapitulatif de nos hyper paramètres sur cette expérience.

```
LEARNING_RATE = 0.001
K = 5
MOMENTUM = 0.9
EPOCHS = 100
N_INIT = 10
N_NEURONS = 8
```

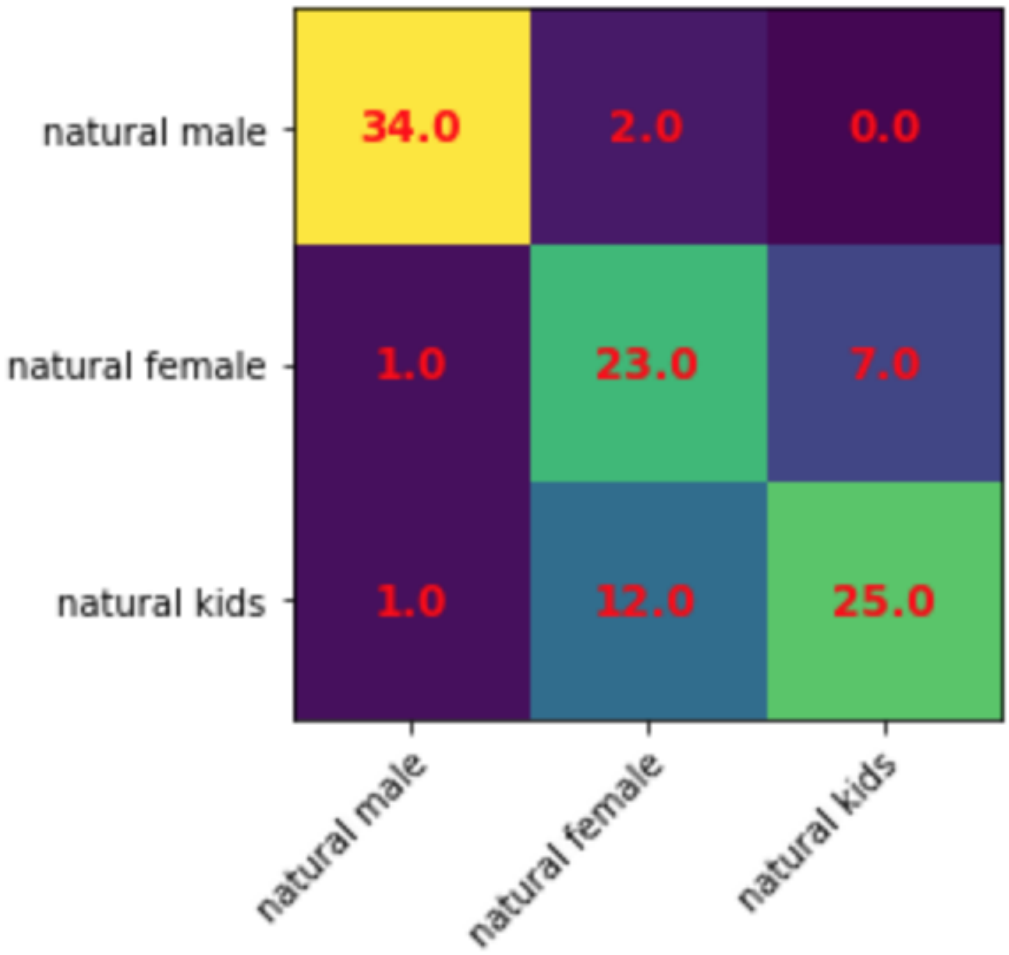
```
THRESHOLD = 0.5
```

Matrice de confusion

Voici la matrice de confusion finale:

```
MSE training : 0.0983
MSE test     : 0.1234

Confusion matrix of selected model:
```



Afin de calculer le F1-score, nous avons tout d'abord calculé le F1-score pour chacune des trois classes de notre expérience (homme, femme et enfant). Cela nous a donc donné trois F1-score. Afin d'avoir

une idée globale finale, nous avons également calculé le f1 score moyen en le pondérant aux nombres d'entrées de chaque classe afin de ne pas trop fausser le résultat. Comme nos 3 entrées sont équilibrées (36 chacune), cette option est inutile mais garantit plus d'évolutivité.

```
f1 score natural male : 0.9444
f1 score natural female : 0.6765
f1 score natural kids : 0.7143

weighted average f1 score : 0.7528
```

Nous observons un f1 score final bon mais pas excellent. Ceci est expliqué car des valeurs sont prédites dans plusieurs classes, notamment à cause du treshold. Si nous augmentons encore le threshold, nous perdons beaucoup de valeurs (environ 20%) que le modèle n'arrive pas à catégoriser, ce qui n'est pas l'objectif souhaité non plus.

3. Voix naturelles vs Voix synthétisées

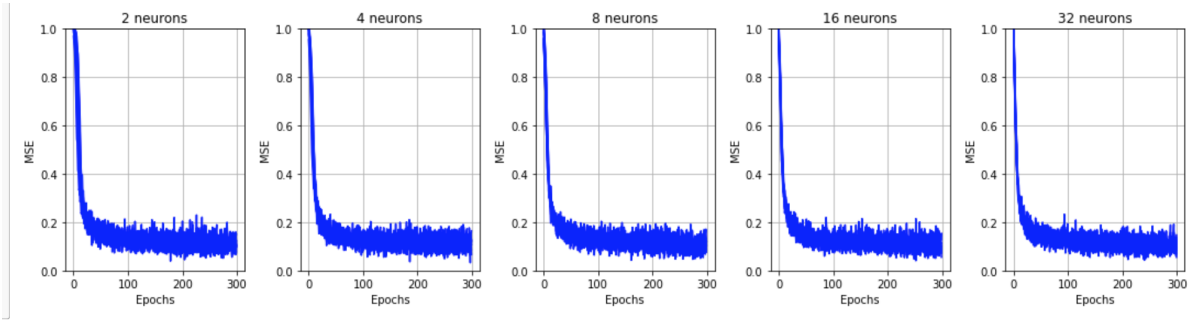
Dans cette dernière étape, nous allons comparer toutes les voix naturelles aux voix synthétisées, afin d'avoir le plus de données dans notre dataset possible. Cette expérience nous permet également de tester des données pas encore utilisées par le passé (voix synthétisées).

Cela nous donne donc une expérience à deux sorties possibles. Nous utilisons donc le même codage que lors de la première (1 pour les voix naturelles, -1 pour les voix synthétisées).

Exploration du nombre de neurones

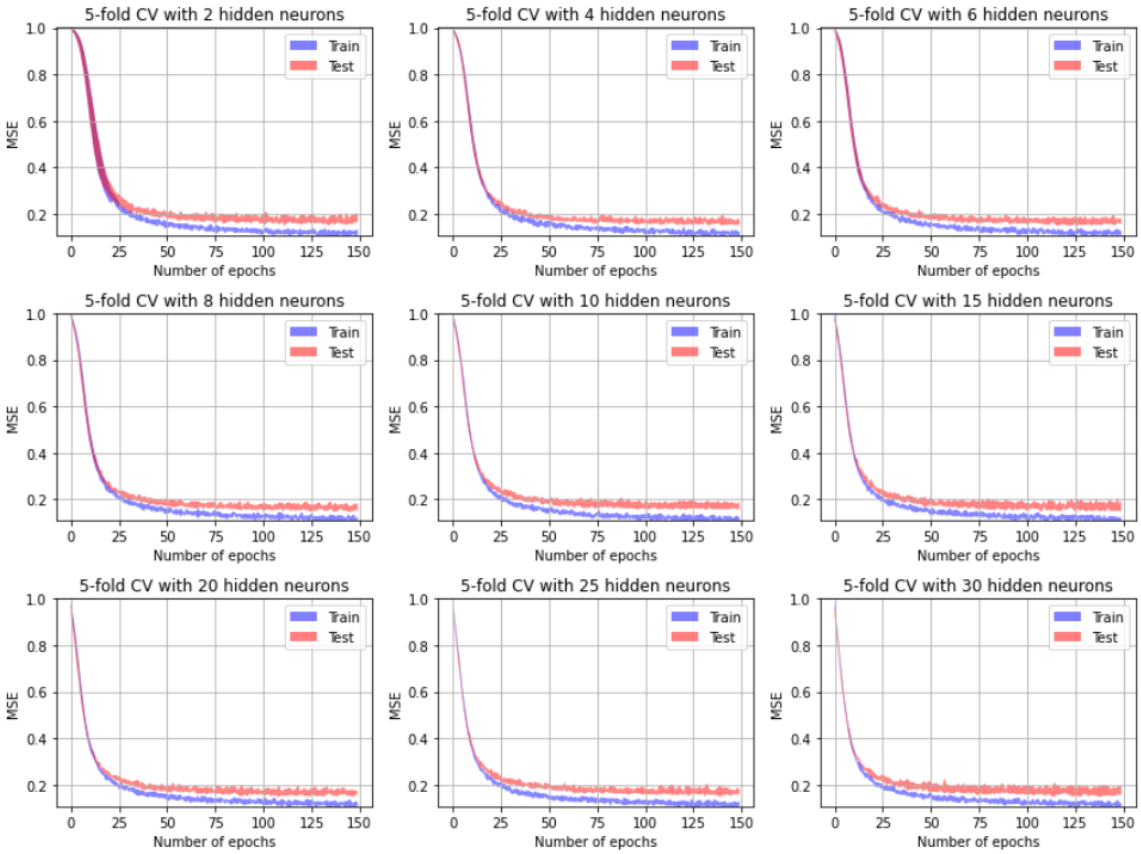
Hyper paramètres utilisés:

```
N_INITS = 10
EPOCHS = 300
N_NEURONS = [2, 4, 8, 16, 32]
LEARNING_RATE = 0.001
MOMENTUM = 0.8
```



Nous observons que l'erreur descend rapidement et que le nombre de neurones n'influence pas énormément le résultat pour le moment. De plus, le nombre d'epochs peut être limité à environ 100 car les courbe se stabilisent déjà. Nous prenons 150 dans la deuxième exploration afin d'avoir un peu de marge.

Exploration du nombre de neurones dans la couche cachée



Nous observons de bons résultats avec un léger overfitting. Les résultats se rapprochent de la première expérience uniquement entre les hommes et les femmes. Afin de simplifier le réseau, nous

choisissons de garder 4 neurones dans la couche cachée. En effet, les résultats sont quasiment équivalents avec plus de neurones et avec 2 il y a un peu plus de bruit.

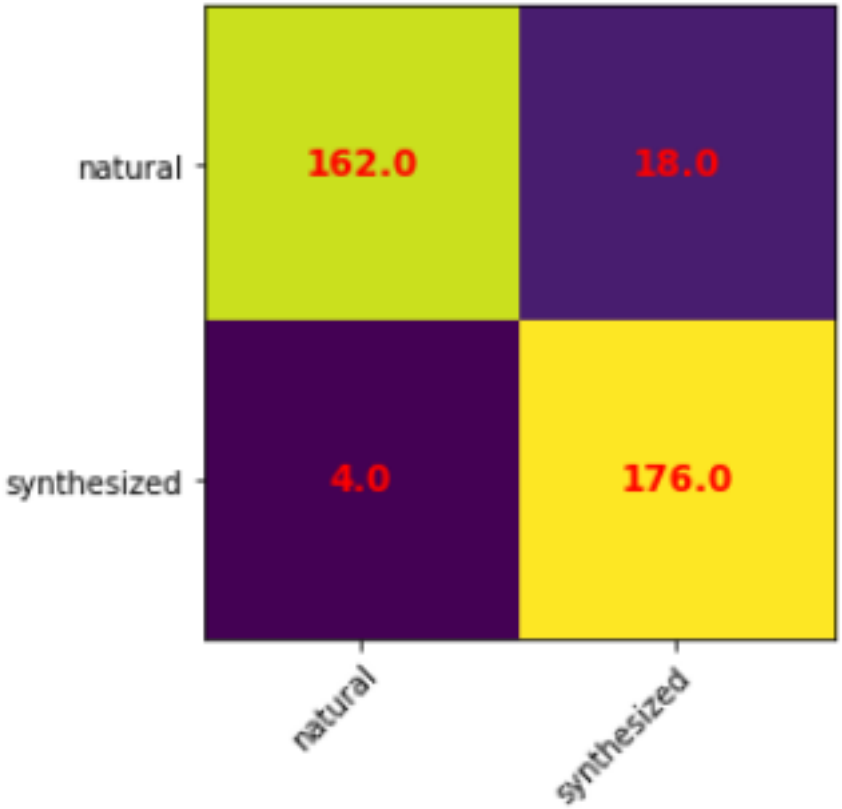
Matrice de confusion

Voici les hyper paramètres utilisés pour ce test final:

```
K = 5
EPOCHS = 100
LEARNING_RATE = 0.001
MOMENTUM = 0.9
TRESHOLD = 0.0
N_NEURONS = 4
```

```
MSE training : 0.1101
MSE test     : 0.1924

Confusion matrix of selected model:
```




```
f1 score     : 0.9364
accuracy     : 0.9389
```

Nous observons de très bons résultats pour cette dernière expérience. Le f1 score est très bon (proche de 1) et la matrice nous le confirme. Nous remarquons également que l'erreur est deux fois plus grande lors du test que lors de l'entraînement, ce qui confirme l'overfitting annoncé précédemment. Nous pouvons en conclure que le réseau a bien été entraîné. Nous pouvons supposer qu'avoir plus de données d'entraînement a aidé le réseau à mieux classifier les différentes valeurs.

Conclusion

Nous sommes plutôt satisfaits des résultats de nos différents réseaux, surtout vu le nombre restreint de données d'entrées. Nous pensions initialement qu'il aurait fallu un jeu de données bien plus grands pour arriver à des résultats aussi bons. D'un autre côté, nous avons bien fait l'expérience de l'overfitting. Nous avons eu du mal à avoir deux courbes bien superposées. Nous imagineons que c'est, en partie, dû au faible nombre de données d'entraînement.

Nous avons apprécié travailler sur un exemple très concret (des fichiers audios) avec lesquels nous pouvions directement. Le seul bémol a été pour nous le temps d'exécution des trainings. En effet, cela était vraiment lent sur nos machines respectives et tester plusieurs valeurs d'hyper paramètres nous a pris beaucoup de temps. Nous avons essayé de voir comment rendre le code multi-thread ou alors utiliser nos cartes graphiques à la place du CPU, mais nous n'avons pas trouvé de solution facile à mettre en place.