# Analyse résultats du RNN

Benotsmane Ismat

November 2020

# Table des matières

# 1 Classification de sequence

L'objectif est de construire un modèle qui à partir d'une séquence de température de longueur non prédéfinie infère la ville correspondante. Nous utiliseront ici une architecture RNN *many-to-one*.

Avec les hyperparamétres suivants :
— $epoch = 30$
— $learning\ rate = 1e^{-3}$
— $latent\ dimension = 20$
— $train\ sequence\ length = 100$

Nous avons appris un modéle RNN avec des sequences de taille fixe, et nous l'avons testé sur des sequences de taille fixe(=100).

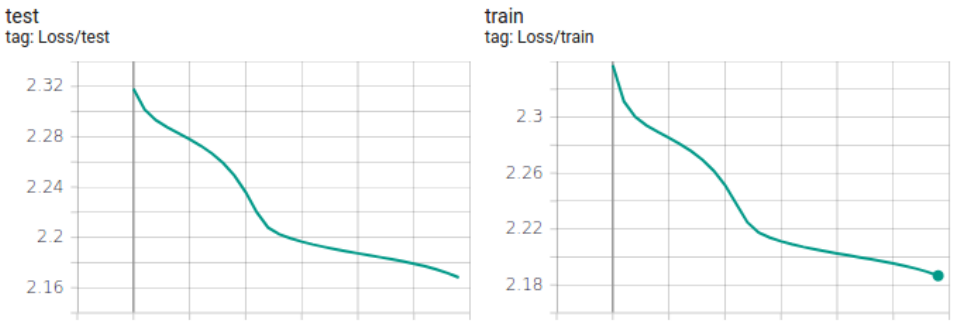


FIGURE 1 – La loss en train et en test

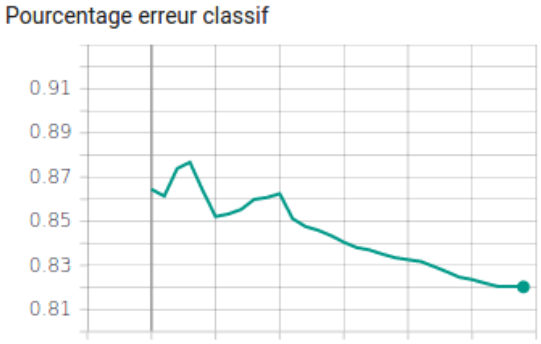


FIGURE 2 – Pourcentage d'erreur en prediction

On remarque que notre modéle s'est bien comporté lors de la phase d'apprentissage, car nous avons réduit le cout en train et en test tout en réduisant le pourcentage d'erreur en classification. Aprés 30 epochs nous étions à **82%** d'erreur.

On remarque aussi que la réduction de la loss en test ne signifie pas obligatoirement une réduction de l'erreur en classification, car on peut trés bien optimiser le coût en test sur des cas déja bien classé mais augmenter légèrement le coût sur des cas qui étaient bien classé lors de l'itération précédente mais mal classé pendant cette itération. Donc le coût global pendant cette itération aura baissé mais l'erreur en classification a augmenté.
D'où l'importance de toujours observer la loss en test et l'erreur en classification.

# 2 Forecast de temperature

L'objectif de cette partie est de faire de la prédiction de séries temporelles : à partir d'une séquence de température de longueur t pour l'ensemble des villes du jeu de données, on veut prédire la température à t + 1, t + 2,....

Nous utiliseront ici une architecture RNN *many-to-many*, avec 2 approches differentes :

— un RNN par série de température propre à une ville, soit n modèles différents qui prennent chacun une série dans R et produisent un décodage dans $\mathbb{R}$.
— un RNN commun à toutes les villes qui prend une série dans R n et prédit une série dans $\mathbb{R}^n$.

Avec les hyperparamétres suivants :
— $epoch = 50$
— $learning\ rate = 1e^{-4}$
— $latent\ dimension = 10$
— $train\ sequence\ length = 100$

Nous avons appris 10 modéles RNN chacun propre à une ville, et un modéle commun aux 10 villes.

On a pu remarquer le modele appris avec une taille de sequence fix est pareil voir parfois meilleur en test sur des sequences inferieur ou egale à 100 que le modele appris avec des tailles de sequences aleatoire.
Mais sur les sequences de tailles superieurs à 100, le modele avec tailles de sequences aleatoire est bien meilleur en forecasting.

Comme l'illustre la figure 3, où la courbe **courbe rose** qui represente la loss d'un modéle appris avec des sequences de tailles fixes(=100) converge plus vite en test sur des séquences de meme taille que le modéle appris avec des séquences de tailles aléatoire(**courbe orange**).

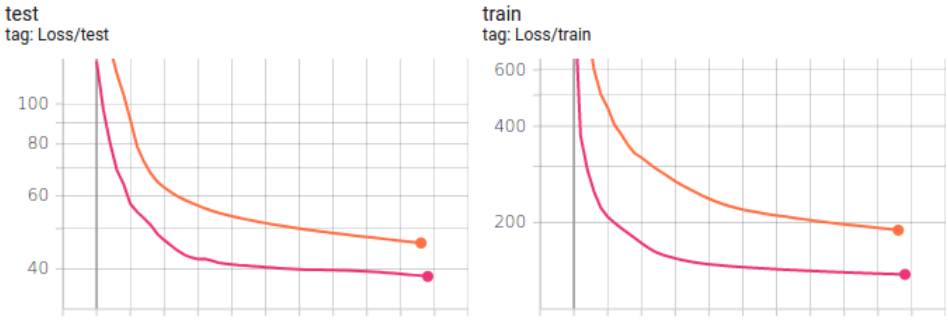


FIGURE 3 – Train avec sequence de taille fixe VS random

On peut en conclure que le choix d'un modele à taille de sequence fixe ou random depend des sequences en prediction. Si elles sont toujours de tailles fixe, on opte pour un modele à taille de sequences fixes, si elles varient on opte pour un modele à tailles de sequences aleatoire. Car ce dernier généralise mieux le problème.

Nous avons effectué un forecasting à t+1 et t+2 sur une sequence de longueur 100 avec le modèle multivarié sur , et les modéles univariés et nous avons obtenu :

| | city 1 | city 2 |
|---|---|---|
| Real temperature | 281.4 | 279.7 |
| Univariate model | 282.2 | 281.0 |
| Multiviate model | 282.2 | 280.2 |

TABLE 1 – Forecast t+1

| | city 1 | city 2 |
|---|---|---|
| Real temperature | 280.6 | 282.8 |
| Univariate model | 283.3 | 286.5 |
| Multiviate model | 282.6 | 284.1 |

TABLE 2 – Forecast t+2

Dans les 2 tableaux, on constate que le modéle multivarié est celui qui approxime le mieux la temperature prédite par rapport aux temperatures réelles.

# 3 Génération de discours

Le but ici est d'apprendre un RNN capable d'engendrer un discours à la Trump en apprenant sur un dataset de ses discours pré-électoraux.

Pour apprendre, nous avons utilisé une architecture RNN $many-to-many$, où une séquence est une suite de lettres, et chaque lettre est encodée en $one-hot$ dans $\mathbb{R}^n$ avec $n$ le nombre de symboles.
Nous avons utilisé une fonction d'activation $tanh$ pour la couche intermédiaire et $softmax$ pour la couche de sortie à laquelle nous avons appliqué une $Cross\ entropy$ loss.

Pour générer un discours à la Trump, on passe au réseau une sequence de longueur $t$ puis il nous retourne les charactéres générer à partir de $t+1, t+2, ...$

Avec les hyperparamétres suivants :
— $epoch = 50$
— $learning\ rate = 1e^{-3}$
— $latent\ dimension = 20$
— $train\ sequence\ length = 100$

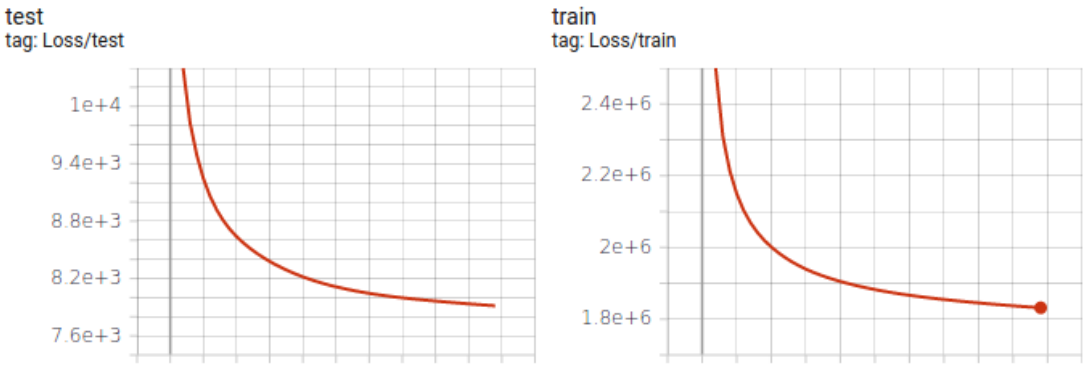


FIGURE 4 – Loss en train et test par itération

La figure 4 illustre le comportement lors de l'optimisation des parametres du modele. On constate que plus on entraine le modele mieux il sera performant.

En choisissant le symbole le plus probable dans la distribution multinomiale décodée à chaque pas de temps, nous avons pu générer une sequence avec ce modéle.

En entrée nous avons fourni la sequence suivante "**I don't kn**" et nous avons généré à la suite de cette sequence "**ownnrnrrrr**".
Comme on peut le voir le resultat n'est pas satisfaisant car la génération n'a aucun sens.

A noter que ce modéle n'est pas du tout optimisé en terme de loss et d'hyperparametres, ceci n'est qu'une illustration de ce qu'on est capable de faire avec un RNN en génération. Avec plus d'entrainement on peut encore réduire la loss en train et en test.

# 4 LSTM et GRU

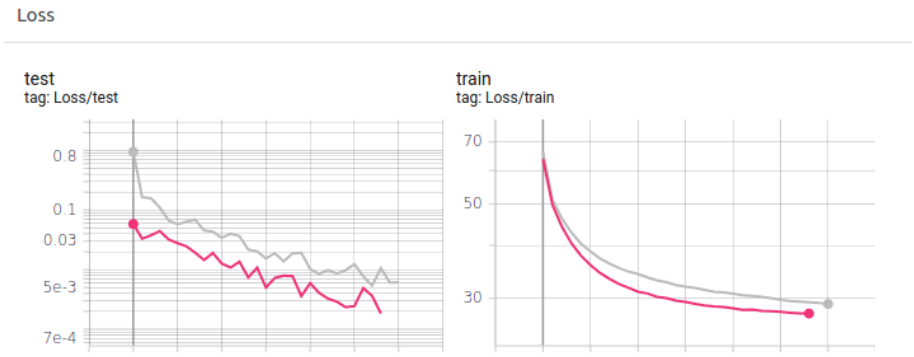


FIGURE 5 – Loss GRU(rose) et LSTM(gris)

On a fait tourner les 2 modéles sur les mêmes hyper-paramétres, et on peut observer qu'il n'y a pas de difference majeure en terme d'optimisation de la loss. Toutefois, on a pu noter que GRU est plus rapide à entrainer que LSTM.

# 5 Génération : RNN VS LSTM

Nous avons voulu comparer la qualité des générations des deux modéles, avec comme début de sequence "**The world is** "

| | **RNN** | **LSTM** |
|---|---|---|
| **Argmax** | the state the state | the people will be a country |
| **Sampling** | our citizens. | used in yours of the regulations of Hillary Clinton |
| **Beam search** | the ountry | the people. |
| **Beam search(Nucleus)** | ,the state the state | not going to be the state of |

TABLE 3 – Comparaison génération RNN et LSTM

# 6 POS-Tagging

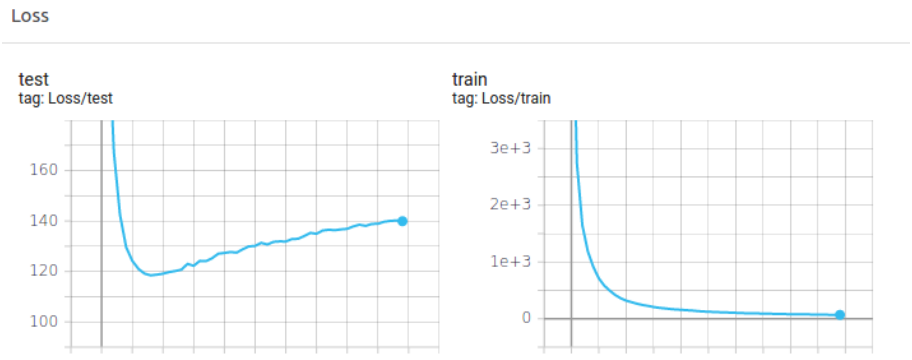


FIGURE 6 – Loss en train et test par itération

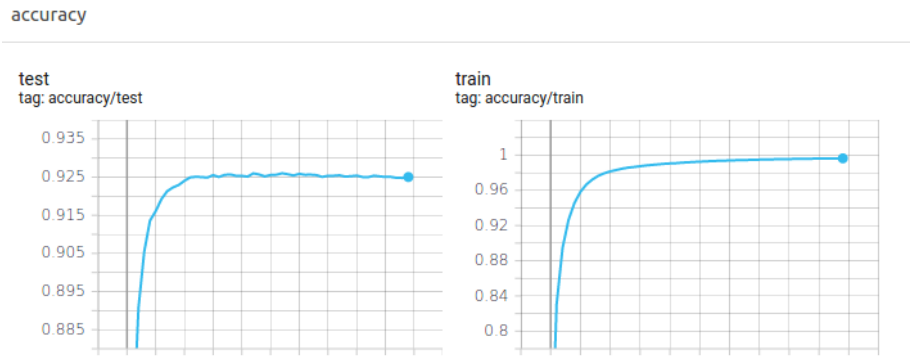


FIGURE 7 – Accuracy en train et test par itération

Les figures 6 et 7 sont les résultats d'un premier modéles qu'on a entrainé. On peut noter qu'à partir de la 20éme epoch le modéle est en sur-apprentissage même s'il y a aucun effet sur l'accuracy en test. Ce phenoméne s'explique par le fait que le modéle en test est confronté à des mots inconnus qu'il n'a jamais rencontré lors de la phase d'apprentissage.

Observons maintenant la qualité de la prédiction :

**Sequence :**
C'est véritablement pour le futur baptisé un changement de cap.

**Targets :**
PRON AUX ADV ADP DET ADJ VERB DET NOUN ADP NOUN PUNCT

**Outputs :**
PRON AUX ADV ADP DET ADJ VERB DET NOUN ADP NOUN PUNCT

On remarque que la qualité de la prédication est trés instéressante ce qui vient confirmer les **92.5%** d'accuracy en test.

Regardons maintenant le comportement avec des mots inconnus qui seront representé par **___OOV___** dans le texte.

**Sequence :**
Le poème en ___OOV___ est né au à le XIXe siècle avec le recueil de poèmes Gaspard de la nuit d'___OOV___ Bertrand.

**Targets :**
DET NOUN ADP NOUN AUX VERB _ ADP DET ADJ NOUN ADP DET NOUN ADP NOUN PROPN ADP DET NOUN ADP PROPN PROPN PUNCT

**Outputs :**
DET NOUN ADP VERB AUX VERB _ ADP DET ADJ NOUN ADP DET NOUN ADP NOUN PROPN ADP DET NOUN ADP NOUN PROPN PUNCT

**Sequence :**
On retrouve ici une touche très ___OOV___ et qui au à le lieu de souligner la thématique du de le livre y ajoute du sens.

**Targets :**
PRON VERB ADV DET NOUN ADV ADJ CCONJ PRON _ ADP DET NOUN ADP VERB DET NOUN _ ADP DET NOUN PRON VERB DET NOUN PUNCT

**Outputs :**
PRON VERB ADV DET NOUN ADV VERB CCONJ PRON _ ADP DET NOUN ADP VERB DET NOUN _ ADP DET NOUN PRON VERB DET NOUN PUNCT

On obsérve dans les séquences ci-dessus que toutes les mauvaises prédictions concernent des mots inconnus du modéle.

Pour résoudre ce probléme, nous avons décider lors de la phase d'apprentissage de masquer aléatoirement certains mots en les considérants comme inconnus afin de rendre le modéle plus robuste pour qu'il puisse mieux généraliser.

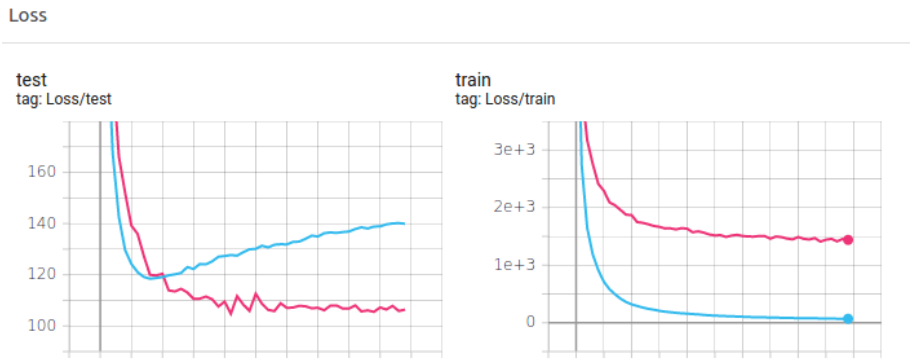


FIGURE 8 – Loss du modéle simple et modéle avec mots inconnus

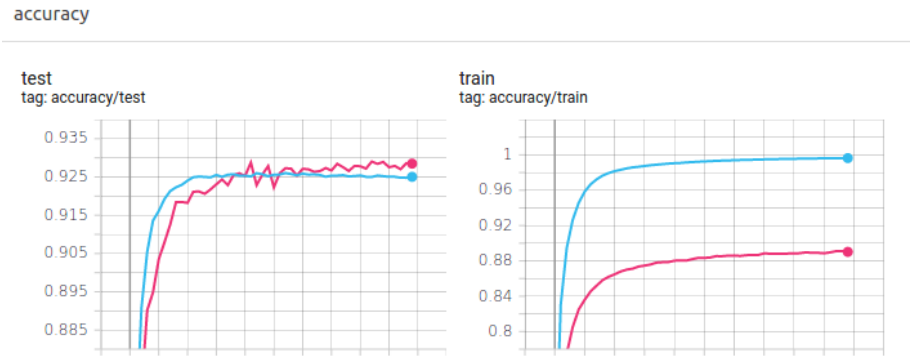


FIGURE 9 – Accuracy du modéle simple et modéle avec mots inconnus

On remarque qu'on a perdu en loss et en accuracy en train pour mieux gagner en test. Et on peut deja observer le résultat sur les mêmes cas traité précédemment.

**Sequence :**
Le poème en ___OOV___ est né au à le XIXe siècle avec le recueil de poèmes Gaspard de la nuit d'___OOV___ Bertrand.

**Targets :**
DET NOUN ADP NOUN AUX VERB _ ADP DET ADJ NOUN ADP DET NOUN ADP NOUN PROPN ADP DET NOUN ADP PROPN PROPN PUNCT

**Outputs :**
DET NOUN ADP NOUN AUX VERB _ ADP DET ADJ NOUN ADP DET NOUN ADP NOUN PROPN ADP DET NOUN ADP PROPN PROPN PUNCT

**Sequence :**
On retrouve ici une touche très ___OOV___ et qui au à le lieu de souligner la thématique du de le livre y ajoute du sens.

**Targets :**
PRON VERB ADV DET NOUN ADV ADJ CCONJ PRON _ ADP DET NOUN ADP VERB DET NOUN _ ADP DET NOUN PRON VERB DET NOUN PUNCT

**Outputs :**
PRON VERB ADV DET NOUN ADV ADJ CCONJ PRON _ ADP DET NOUN ADP VERB DET NOUN _ ADP DET NOUN PRON VERB DET NOUN PUNCT

# 7 Traduction

Pour la tache de traduction notre modéle comporte deux RNN, un pour l'encodage de la phrase dans la langue d'origine et l'autre pour le décodage dans la langue de destination.
Pour l'apprentissage du modéle nous avons utilisé la technique appellée **Curriculum learning**, où pour chaque batch on va appliquer aleatoirement soit de l'apprentissage contraint en passant la phrase cible comme entrée du décodeur, ou l'apprentissage non-contraint en passant la sortie du pas précédent en entrée.

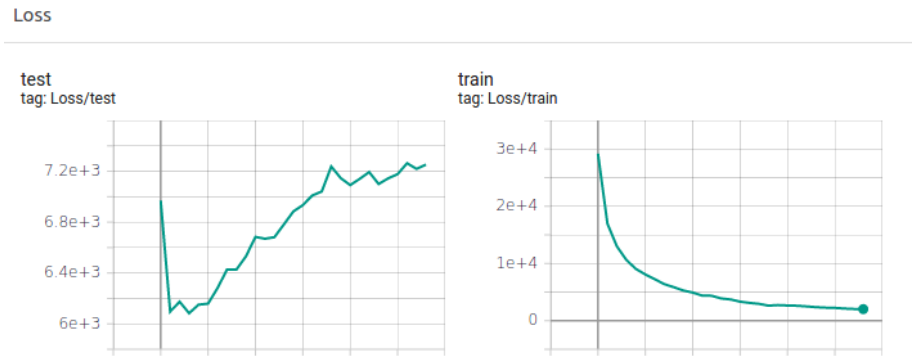


FIGURE 10 – Loss train/test sur 30 itérations

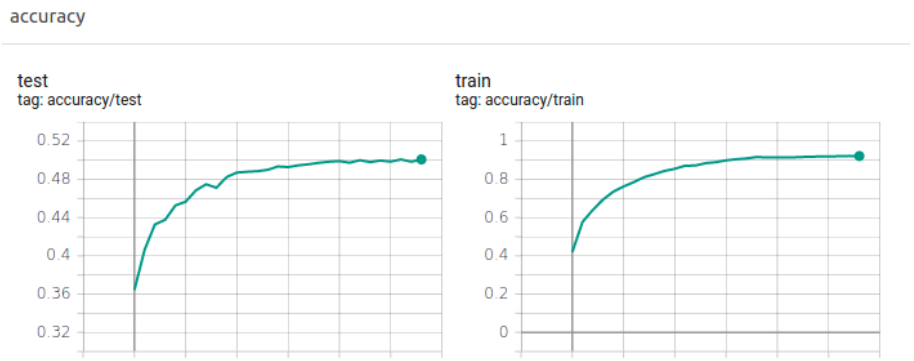


FIGURE 11 – Accuracy train/test sur 30 itérations

Pour la phrase "i am young" le résultat obtenu est "jeunes"