
# Rapport de projet de Deep Learning : COVID-19 mRNA Vaccine Degradation Prediction

Travail encadré par Jean-Christophe GELLY et Frédéric GUYON
Université de Paris - Biologie intégrée du globule rouge UMR-S 1134

réalisé par
Théo FERREIRA, Ilyas GRANDGUILLAUME, Maxime KERMARREC,
Ferdinand PETIT et Apollinaire ROUBERT
Université de Paris - Master 2 BI-IPFB

27 octobre 2020

# Table des matières

# Rapport de projet de Deep Learning : COVID-19 mRNA Vaccine Degradation Prediction

Théo FERREIRA, Ilyas GRANDGUILLAUME, Maxime KERMARREC, Ferdinand PETIT et Apollinaire ROUBERT

27 octobre 2020

**Résumé**

Dans le cadre du projet OpenVaccine (université de Stanford et plateforme Eterna) et dans le but de prédire ces points de fragilité, une compétition a été lancée. Ainsi le but de cette compétition publique est d'établir un réseau neuronal d'intelligence artificielle capable de prédire au mieux la fragilité de tous les bases à une position donnée d'une séquence d'ARN. Nous nous sommes proposés de participer à ce projet. Dans ce rapport vous trouverez notre méthodologie et nos résultats.

## 1 Introduction

En décembre 2019, un nouveau coronavirus responsable de détresse respiratoire aiguë sévère est apparu à Wuhan, en Chine. Le SARS-CoV-2 a depuis lors provoqué une pandémie qui touche pratiquement tous les pays. Nous ne connaissons pas avec précision la durée de l'immunité face à ce virus. Des cas suspects de réinfection ont été documentés ces derniers mois.[11, 10] On ne sait pas si cette réinfection est due à un immunité protectrice non durable ou à de différentes souches du même virus, voire les deux. Les dernières études [5, 4] ne trouvent aucune molécule efficace en prévention ou traitement de l'infection par le SARS-CoV-2. Il est alors urgent de trouver un vaccin conduisant à une immunité protectrice durable. Actuellement, de nombreux vaccins candidats sont à l'étude. Certains sont des vaccins à virus inactivés, des vaccins vectorisés par un autre virus, des vaccins protéiques, des vaccins à ADN ou des vaccins à d'ARN messager.[9] Depuis août 2020, les vaccins à ARNm candidats en phase 3 sont au nombre de 2 (mRNA-1273 de Moderna et BNT162b de BioNtech).[8, 3] Les vaccins à ARNm sont particulièrement étudié dans la recherche face au cancer mais n'a cependant jamais montré son efficacité.[12] Les vaccins à ARNm représentent une alternative prometteuse aux approches vaccinales conventionnelles en raison de leur puissance élevée, de leur capacité de développement

rapide et de leur potentiel de fabrication à faible coût et d'administration sûre. Toutefois, leur application est limitée par l'instabilité et l'inefficacité de l'administration in vivo due à l'hydrolyse de l'ARNm [13]. Cette nécessité de conservation à très basse température ( -20°C ou -80°C) est très problématique pour leur utilisation dans les pays en voie de développement et pour l'acheminement dans des zones reculées. Pour augmenter leur stabilité, plusieurs pistes sont envisagées telles que la formation de double brin d'ARN ou encore l'étude de points de fragilités/stabilités sur le squelette de l'ARN.

**Objectif :** Dans le cadre du projet OpenVaccine (université de Stanford et plateforme Eterna) et dans le but de prédire ces points de fragilité, une compétition a été lancée. Ainsi le but de cette compétition publique est d'établir un réseau neuronal d'intelligence artificielle capable de prédire au mieux la fragilité de tous les bases à une position données d'une séquence d'ARN.

# 2 Méthodes & mise au point du réseau de neurones

Pour mener à bien la réalisation de ce projet disponible en ligne https ://www.kaggle.com/c/stanford-covid-vaccine/, nous avions à notre disposition 2 fichiers et 1 dossier.

## 2.1 Données disponibles

Le premier fichier train.json (lignes = 2400, colonnes = 18) qui contient les données nécessaires à l'apprentissage du réseau :

- 2400 séquences d'ARNm (107 nucléotides).
- Leur structure secondaire (les bases appariées sont indiquées par des parenthèses ouvrantes et fermantes). Ce sont des appariements prédits par Eterna.
- Structure "type de boucle" (S : Tige jumelée, M : Multiboucle, I : Boucle interne, B : Renflement, H : Boucle en épingle à cheveux, E : Extrémité pendulaire, X : Boucle externe).

Ainsi que les données de réactivité globale par position pour les 68 premières positions (seq-scored) de la séquence d'ARN :

- réactivité : valeurs de la réactivité.
- deg-Mg-pH10 : valeurs de la réactivité utilisées pour déterminer la probabilité de dégradation à la base/liaison après incubation avec du magnésium à pH élevé.

— deg-Mg-50C : valeurs de la réactivité utilisées pour déterminer la probabilité de dégradation à la base/liaison après incubation avec du magnésium à température élevée (50°C).
— deg-pH10 : valeurs de la réactivité utilisées pour déterminer la probabilité de dégradation à la base/liaison après incubation à pH élevé.
— deg-50C : valeurs de la réactivité utilisées pour déterminer la probabilité de dégradation à la base/liaison après incubation à température élevée (50°C).

Parmi les informations supplémentaires disponibles, une était particulièrement intéressante : SN-filter. Il prenait pour valeur 1 quand l'information était de qualité suffisante. Une fois les données filtrées nous avions à notre disposition 1589 séquences avec un SN-filter $\geq 1$.

Le dossier bpps contenait les matrices de probabilité d'appariement de base pour chaque séquence. Ces matrices donnent la probabilité que chaque paire de nucléotides dans l'ARN forme une paire de bases. Ces informations étaient donc redondantes avec la structure secondaire. Le dernier fichier, test.json comprenait 629 séquences d'ARN de longueur 107 que l'on appellera test public et 3005 séquences d'ARN de longueur 130 nucléotides que l'on appellera test privé.

## 2.2 Prédiction des structures secondaires

Au sein des données mise à disposition par la plateforme, les prédictions des structures secondaire (SS) ont été réalisées via le site Eterna. Un site de résolution de puzzle appliqué à des problèmes biologiques, ici appliqué aux prédiction de structure secondaire sur les mRNA du SARS-CoV-2 en collaboration avec OpenVaccine Team. Cependant, nous n'avons pas plus de détails sur la génération des données. Afin de vérifier si les prédictions sont cohérentes avec les données, nous avons réalisé une prédiction des SS via ViennaRNA [7], un package python (installation conda install -c bioconda viennarna) se basant sur le minimum d'entropie libre, une prédiction séquence par séquence pour les échantillons de train et test via la fonction "fold". Nous observons une homologie de 100% entre les structures prédites par Eterna et celles prédites par ViennaRNA sur le jeu de données train. Cependant, sur le jeu de données test, 51% des séquences prédites par ViennaRNA présente au moins une différence de prédiction par rapport aux données de base. Cette différence semble incohérente. Elle ne s'explique pas par la différence de taille de la séquence, ni par le nombre variant de séquence entre les deux jeux de données. Cependant 78% des séquences divergentes présentent plus de 80% d'homologie, indiquant une faible variabilité des SS pouvant potentiellement jouer sur les données d'apprentissage du modèle.

Ensuite, nous avons voulu mettre en place une seconde prédiction de SS, par SPOT-RNA.[14] Ce programme est basé sur du machine-learning. Cependant, par limitation de

temps et de puissance de calcul, nous n'avons pas eu le temps de mettre en place cette méthode très chronophage (10 minutes pour la prédiction de 5 séquence sur 4 threads). Il serait donc intéressant de mettre en place cette méthode dans le futur, qui semble plus puissante et précise, afin de comparer les résultats.

## 2.3 Visualisation et description des données

Il est important de noter que les séquences du jeu de donnés comportent un préfixe et un suffixe identique (Cf Figure 6 annexe).

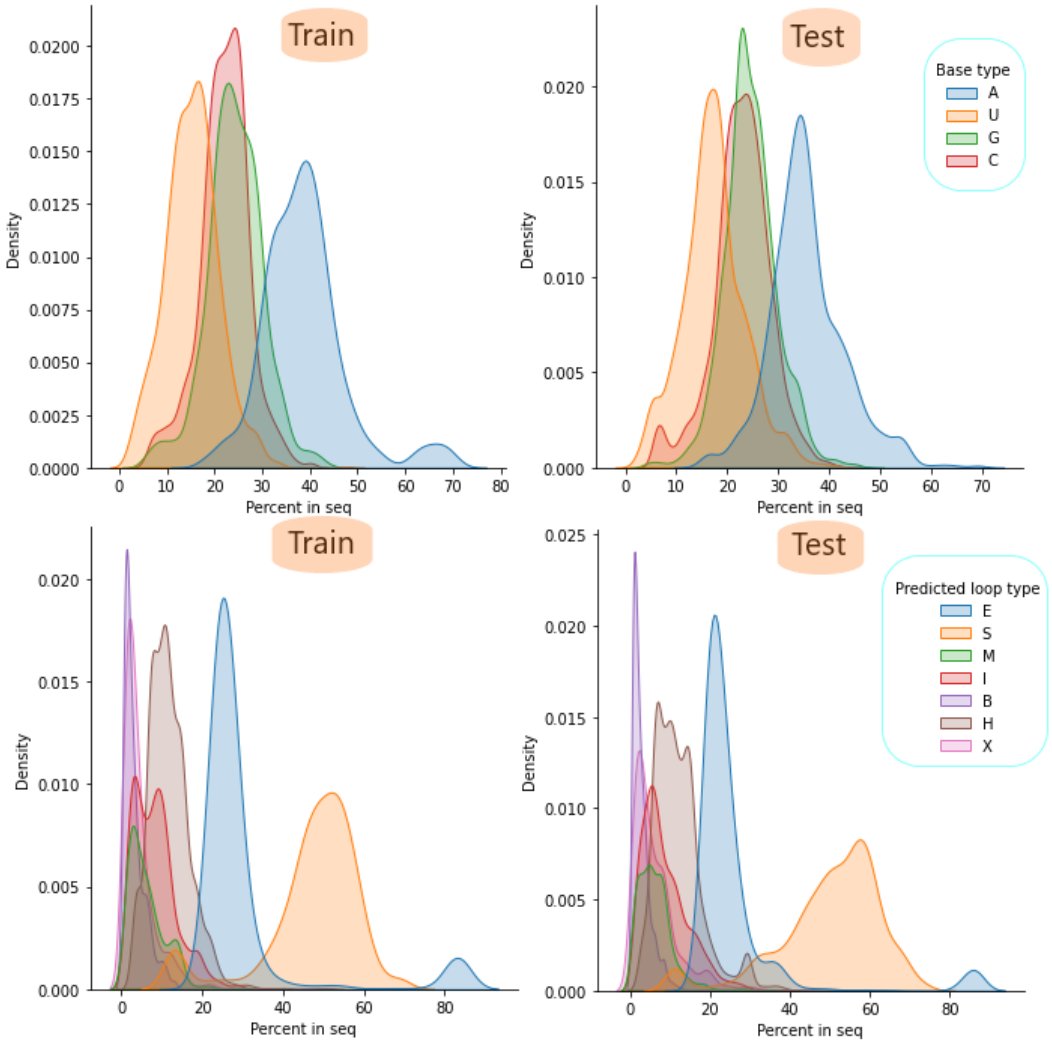


Figure 1 - Répartition des type de bases (haut) et des types de structures prédites (bas) pour les deux jeux de données Train (gauche) et Test (droite). Ces répartition représentent les pourcentage par séquence de base observée ou type de structure prédite (S : paired "Stem" M : Multiloop I : Internal loop B : Bulge H : Hairpin loop E : dangling End X : eXternal loop)

Rien n'est mentionné sur la raison de la présence de ces motifs identiques dans les données, mais nous avons relevé que les cinq premières valeurs de réactivité (mesurées d'après les informations fournies sur le motif GGAAA) n'étaient pas identiques entre les différentes séquences. En revanche les prédictions de structures secondaires et appariements sont toujours identiques sur ces premières bases.

La distribution des bases ainsi que des structures prédites pour ces bases sur les séquences des jeu de données d'apprentissage et de test est représentée figure (2,3abcde). Dans les deux cas les distributions entre les deux jeu de données semblent relativement proches. Cette homogénéité est importante pour que le réseaux puisse prédire correctement, sans ajustement sur le jeu de donnée test.

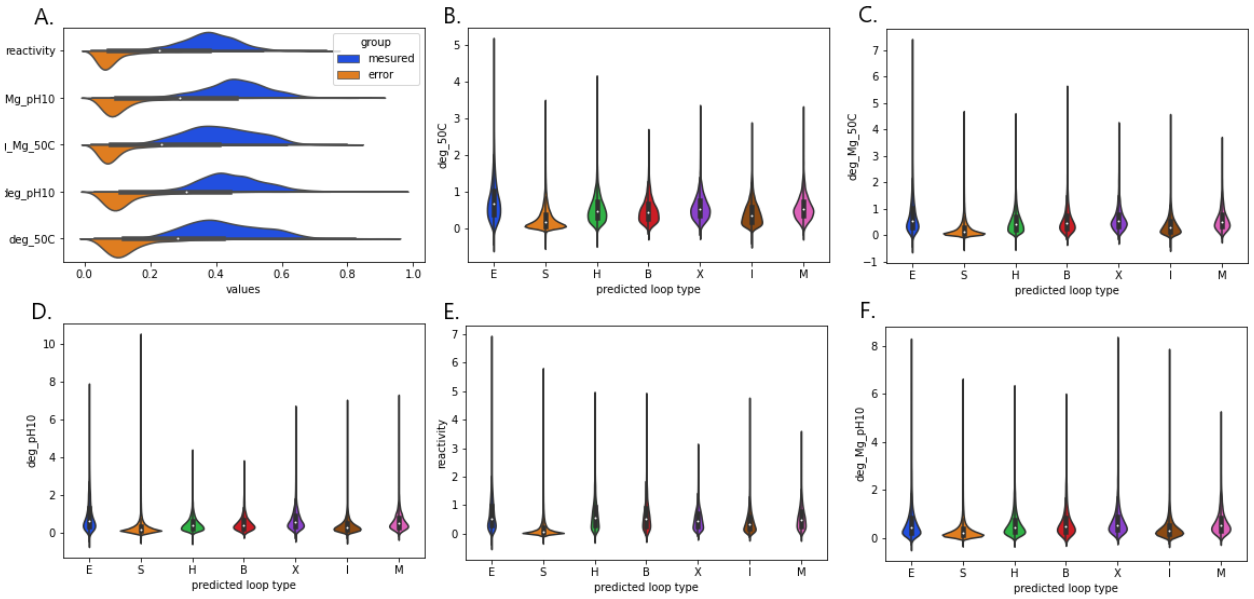


Figure 2 - A :Distribution des réactivités (bleu) et des erreurs (orange) moyennes par séquences pour les cinq variables de réactivité. B,C,D,E,F : Distribution des valeurs de réactivité (respectivement deg-50C, deg-Mg-50C, deg-pH10, reactivity et deg-Mg-pH10) pour chaque type de structure secondaire.

On s'est aussi intéressé aux valeurs moyennes de réactivité du jeu de donnée de test, la distribution de ces valeurs ainsi que des erreurs est représenté figure 2 A. Les erreurs sont faibles et peu dispersées relativement aux valeurs de réactivité. Il faut noter que ces données ont été filtrées préalablement, les erreurs sur les données non filtrées pouvant dépasser les centaines de milliers.

Ces visualisations nous permettent de comprendre comment nos réactivité varient, et si d'autres variables de nos données influent sur elles. Par exemple le type de structure secondaire dans lequel se trouve un nucléotide va déterminer sa valeur de réactivité, plus les nucléotides seront liées entre eux et structurés plus les réactivités de ces nucléotides seront basses. Lorsque l'on regarde les valeurs de réactivité pour les nucléotides dans les structures de type extrémitée pendulaire (E) on remarque une forte variabilité (Figure 2 B,C,D,E). Ceci montre bien que pour les préfixes GGAAA les valeurs sont très dispersées ce qui est probablement dû à la nature flexible et aléatoire de cette structure. On s'attendait à avoir une plus faible réactivité pour les nucléotides appartenant à des structures fortement liées comme la partie tige des tiges boucles (S), ou bien les boucles internes (I). La répartition de la réactivité pour ces types de structures est bien plus centrée et dense que pour les autres, ceci montre bien que la prédiction des structure secondaire est cohérente avec ces valeurs de réactivité. Si l'on voulait obtenir un jeu de donnée homogène il faudrait retirer les valeurs de réactivité extrêmes (très en dehors de la répartition moyenne) que l'on peut identifier sur les graphes B,C,D,E et F qui correspondent aux queues anormalement longues des densités. Cependant ces valeurs peuvent potentiellement représenter de l'information et au vu de la faible quantité de donnée à disposition, on s'est cantonné à ne garder que les données qui ont passé les filtres initiaux (SN filter).

## 2.4 Similarité et homologie des séquences.

On a tout d'abord vérifié que les séquences ne présentaient pas trop de similarité entre elles, pour que le réseaux ne soit pas saturé d'informations redondantes. On a ainsi mené des alignements deux à deux dans un premier temps entre toutes les séquences.

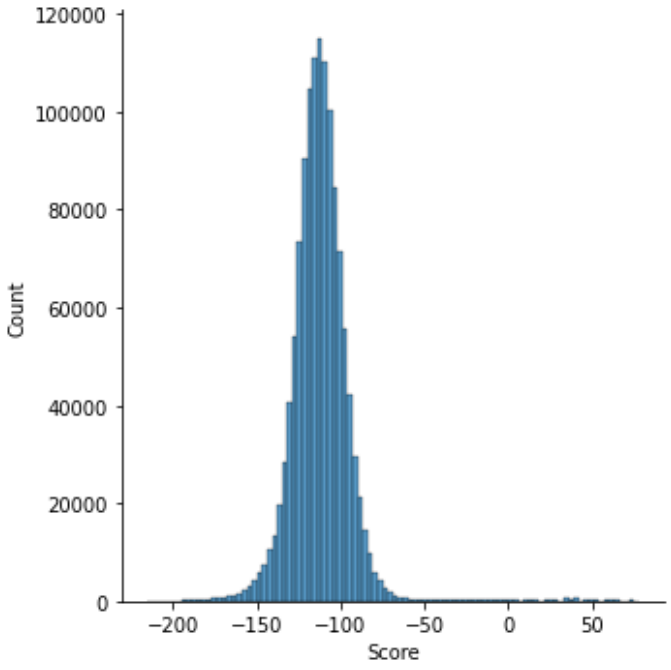

Figure 3 - Distribution des scores d'alignement des séquences deux à deux. Alignements réalisées avec la fonction pairwiseAlignment de BioStrings en mode global sur les séquences (sans préfixes et suffixes).

D'après ces résultats la plupart des alignements mènent à des scores négatifs, donc sont faits sur des séquences non homologues. Une petite minorité d'alignement à mené à des scores supérieurs à 30, ces séquences sont considérées comme homologues. Au vu de leur faible nombre, elles forment probablement de petits groupes. Pour être sûr d'adapter le plus précisément les données à notre réseau, on a construit un arbre de distances d'après un alignement multiple de toutes ces séquences.(Figure 7 - Annexe) Avec le logiciel cdHit [2] on a pu tirer 1319 clusters regroupant maximum 11 séquences. La majorité des séquences sont seules dans leur groupes comme suggéré par l'histogramme plus haut. On a pu ainsi introduire des poids pour chaque séquence, chaque poids étant proportionnel à l'inverse du nombre de séquence dans le cluster.

## 2.5 Choix du réseau à utiliser

Pour choisir le type de réseau optimal à utiliser, nous avons comparé différents types de réseaux neuronaux : (i) nous avons initialement utilisé un réseau composé uniquement de couches "denses" et de couches de "dropout" (Figure 8 - Annexe). Il était cependant incapable de prédire correctement les valeurs attendues (Figure 9 - Annexe) malgré les courbes de "loss" et "val-loss" prometteuses (Figure 10 - Annexe). (ii) Nous avons ensuite, testé un réseau composé de couches de "convolution 1D" qui peuvent prendre des données de taille variable en "inputs" (Figure 11 - Annexe). Les prédictions étaient certes meilleures que le réseau initial mais non suffisantes (Figure 12 - Annexe). (iii) Pour finir nous avons choisi d'utiliser des réseaux récurrents (RNNs), qui permettait de prédire les réactivités d'une position donnée en s'appuyant sur la prédiction précédente. Ce format nous semblait totalement adapté à l'apprentissage sur des séquences nucléiques. En effet les RNNs sont particulièrement adaptés aux données séquentielles/temporelles, qui nécessitent la prise en compte de la relation entre les noeuds des données précédentes. L'utilisation de couches cachées LSTM (Long Short Term Memory Network) [6] ou GRU (Gated Recurrent Unit) [1] montrait une qualité de prédiction supérieure aux réseau précédents (Figure 13 - Annexe). Finalement, les couches GRU bidirectionnelles ont été choisies pour le réseau final. Les RNNs bidirectionnels utilisent à la fois l'information contextuelle passée et future pour prédire la prochaine position. Pour des données "inputs" relativement simples, l'apport du LSTM n'a pas été prouvé ici, nous avons donc opté pour un réseau plus simple avec des couches de GRU.

## 2.6 Architecture du reseau

Le réseau final (Figure 14 - Annexe) prend en entrée des matrices (numpy-array) en 3 dimensions.

— nombre de séquences,
— les positions de chaque séquences
— les variables pour chaque positions.

La 2ème et la 3ème dimensions sont fixées à 130 (nombre de positions) et 14 (la séquence, les appariements et le type des loops encodés en one-hot). En effet, les prédictions sont attendues pour des séquences de 107 et 130 nucléotides. Nos séquences pour l'entraînement du réseau était de 107. Pour pallier à ce souci de dimension, si le nombre de positions dans les séquences est inférieure à 130, nous avons ajouté un "padding" pour atteindre atteindre la taille voulue. Ces positions de "padding" contiennent des valeurs aberrantes et facilement identifiables (ici, -10). Au début du réseau, la couche de "masking" est ici pour prendre en charge ce "padding". Cette couche reconnaît les données en entrée qui contiennent la valeur de "masking" (ici -10) et informe le réseau de ne pas les utiliser lors de l'apprentissage. Une couche de "dropout" est ensuite placée pour éviter le surapprentissage. Les données passent ensuite dans les couches cachées de GRU bidirectionnel pour l'apprentissage. Les données sont coupées pour avoir des séquences de longueur voulues (ici, 68). Cette valeur est expliquée car les données de réactivité ne sont disponibles que pour les 68 premières positions. Une fois que le réseau a fini son apprentissage, nous sauvegardons les poids attribués par le modèle afin de pouvoir les utiliser ultérieurement sur de nouveaux modèles d'apprentissage de séquences de taille différentes (107 et 130 nucléotides)

## 2.7 Paramètres

Le choix des paramètres à utiliser lors de l'apprentissage du modèle sont indispensables. Pour un meilleur apprentissage, nous pouvons utiliser de l'hyper paramétrisation et calculer les performances d'un réseau pour des paramètres données. Cependant les techniques d'hyper paramétrisation sont exigeantes en temps de calcul. Nous avons fait le choix de tester certains paramètres et de les ajuster au mieux pour optimiser l'apprentissage. (i) Ainsi pour l'optimiseur, nous avons sélectionné une extension de la méthode de descente de gradient stochastique, Adam, qui est une des plus couramment utilisée et qui requiert des besoins en mémoire relativement faibles. Elle présente l'avantage de correctement fonctionner même avec un léger réglage des hyperparamètres. (ii) Cet optimiseur prend en argument le learning rate (taux d'apprentissage). Il représente le pas optimal cherchant à minimiser le coût de calcul à chaque étape. Nous avons testé le modèle en faisant varier ce paramètre de 1e-1 à 1e-6 et en fixant les autres paramètres. Le learning rate semblant optimal était de 1e-3 (Figure 4 : Evaluation de la loss en fonction du learning rate et Evaluation du MCRMSE en fonction du learning rate), soit le réglage par défaut

de l'optimiseur. (iii) Les couches cachées GRU possèdent elles aussi un paramètres que nous avons cherché à optimiser. Il s'agit du dropout pour les entrées internes de ces couches. Nous leur avons donné des valeurs allant de 0.1 à 0.5 sans constater de changement significatif dans l'apprentissage ou le sur-apprentissage. Nous avons donc choisi de façon quasi-arbitraire un dropout de 0.2 (Figure 4 : Evaluation de la loss en fonction du dropout et Evaluation du MCRMSE en fonction du dropout). (iv) Pour finir, nous avons ajouté des "callbacks" : EarlyStopping() arrêtant prématurément le réseau en l'absence d'apprentissage. Il utilise la "val-loss" comme moniteur et ReduceLROnPlateau() permettant de diminuer progressivement le learning rate une fois que le réseau atteint un état de plateau. Ce callback est lui aussi monitoré par la "val-loss"

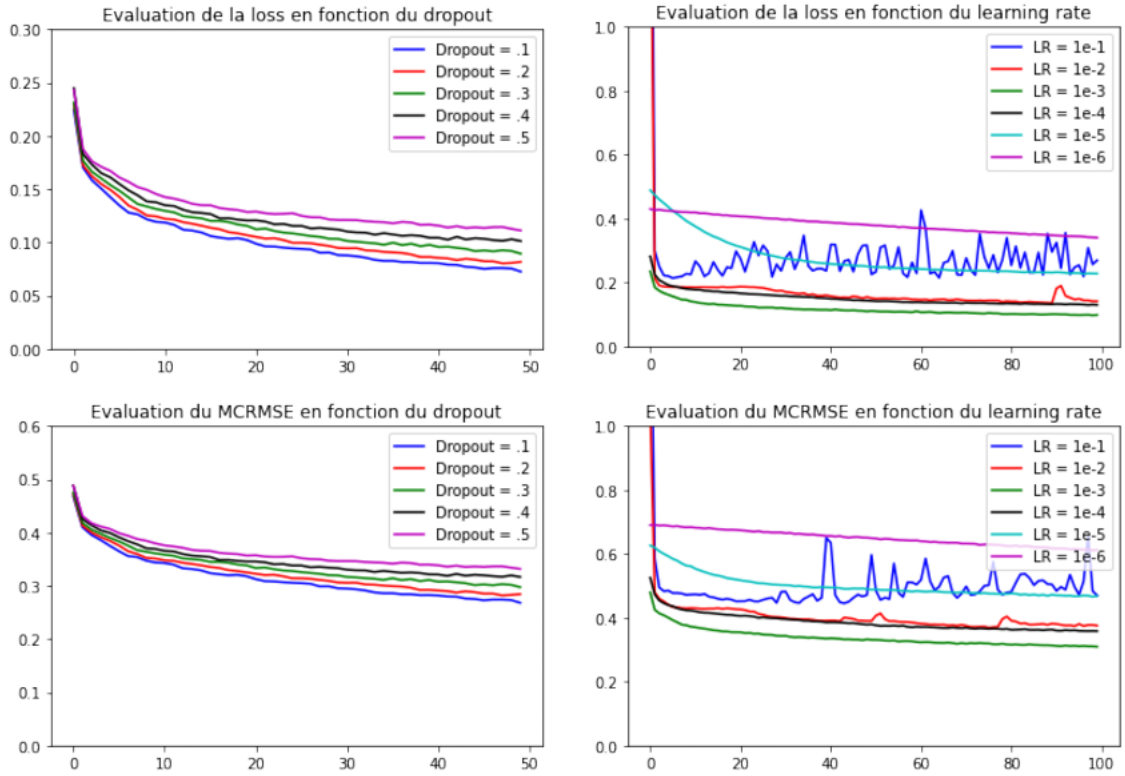


Figure 4 : Graphes de loss et MCRMSE en fonction de l'epoch lors de l'entraînement du réseau sans pré-traitement avec des learning rate et dropout différents.

# 3 Résultats des réseaux RNNs

Chacun des réseaux a été monitoré par la "loss", 'val-loss" et par le MCRMSE (mean columnwise root mean squared error). Ce paramètre d'évaluation a déjà été utilisée dans le cadre d'un concours Kaggle (Africa Soil Property Prediction Challenge) il y a 6 ans. On le preferera au RMSE (Root mean squared error) dans ce cas car nous cherchons à prédire 5 valeurs (réactivité, deg-Mg-pH10, deg-Mg-50C, deg-pH10 et deg-50C) et chacun de ces paramètres aura un RMSE

qui lui sera propre. La MCRMSE est simplement une moyenne de toutes les valeurs RMSE pour chacune de nos colonnes, de sorte que nous puissions toujours utiliser une mesure d'évaluation unique, même en cas de sorties multiples.

Le réseau a été entraîné d'une part avec les données brutes (Figure 5 : graphe de gauche), d'autre part avec les données pré-traitées et clusterisées (Figure 5 : graphe de droite). La clusterisation des données ne permet pas d'amélioration du réseau, les valeurs de MCRMSE en cross-validation ne sont pas en faveur du réseau avec données pré-traitées versus le réseau avec les données brutes (0.38 vs 0.28, respectivement).

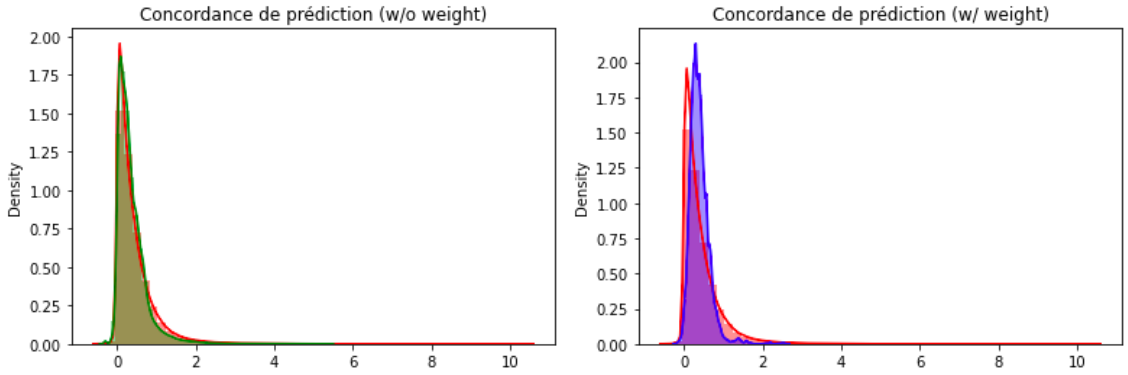


Figure 5 : Densité des prédictions (vert et bleu) superposée sur la densité des valeurs attendues (rouge)

La visualisation des résultats avec le changement de la méthode de prédiction des SS est identique au réseau avec les données brutes. C'est expliqué par l'homologie de 100% entre les deux méthodes de prédiction dans le fichier train.json. Il sera alors intéressant de regarder le score fourni par le site Kaggle pour les fichiers tests (Figure 10 - Annexe). Nous constatons que le réseau sans pré-traitement (private-score de 0.38961, public-score de 0.28602) prédit avec plus de précision que le réseau avec données clusterisées (private-score de 0.46743, public-score de 0.38093). Aucune différence significative n'est observée entre le réseau avec le changement de la méthode de prédiction des SS (private-score de 0.38639, public-score de 0.28474) et le réseau avec données brutes.

# 4 Conclusion

Les prédictions obtenues par le réseau sont très cohérentes avec les valeurs attendues. De tels résultats ont été obtenus en utilisant des données très simples (séquences d'ARN et structures secondaires) et un réseau simple composé de 3 couches de GRU bidirectionnel et une couche de "masking". Lors de la complexification du réseau en prenant en compte la pondération des séquences en fonction de leur similarité, la qualité des prédictions n'a pas été améliorée. Du fait de la petite quantité de données à disposition ainsi que du très faible taux de séquences homologues dans nos données, la pondération de celles ci s'est avérée néfaste pour l'apprentissage. Pour qu'elle soit bénéfique, il aurait fallu qu'il y ai de gros cluster de séquences dans notre jeu de donnée ou bien que l'on utilise un cutoff (pour le clustering) encore plus bas. De plus, le réglage des paramètres a montré des différences minimes sur l'apprentissage et la prédiction lors des variations de ceux-ci (e.g "dropout"). L'étude des prédictions sur la structure secondaire n'a également pas permis d'améliorer le réseau de manière significative, avec des nouvelles prédictions sur les SS très similaires. Cependant, elles ont été faites à partir de ViennaRNA, très proches des prédictions d'Eterna. La méthode de prédiction par machine-learning SPOT-RNA pourrait apporter des prédictions plus proches de la réalité.

Afin d'améliorer notre réseau, il serait peut-être envisageable d'employer un réseau d'inception, qui reprend les données brutes en entrée à chaque couche. Ces données brutes comme nous l'avons vu précédemment sont de bonne qualité. L'inclusion des matrices bpps en parallèle des données actuellement utilisées pourrait peut-être apporter une hausse de la performance. Cela nécessiterait une adaptation du réseau pour lui permettre de prendre en entrée des données en 2 dimensions (matrice bpps) et en 1 dimension (séquences ARN, SS, structure "type de boucle"). Finalement l'intégration des erreurs dans la pondération des données pourrait être envisagée pour éventuellement améliorer le réseau.

# 5 Bibliographie


[1] Yoshua Bengio. *Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation.* 2014.

[2] Adam Godzik. *Cd-hit : a fast program for clustering and comparing large sets of protein or nucleotide sequences.* International society for computational Biology, 2006.

[3] Barney S. Graham. *Evaluation of the mRNA-1273 Vaccine against SARS-CoV-2 in Nonhuman Primates.* The new England journal of medicine, 2020.

[4] RECOVERY Collaborative Group. *Lopinavir–ritonavir in patients admitted to hospital with COVID-19 (RECOVERY) : a randomised, controlled, open-label, platform trial.* The Lancet, 2020.

[5] The RECOVERY Collaborative Group. *Effect of Hydroxychloroquine in Hospitalized Patients with Covid-19.* The new England journal of medicine, 2020.

[6] Sepp Hochreiter. *Long short-term memory.* Neural Computation, 1997.

[7] Ivo L. Hofacker. *The Vienna RNA Websuite.* Nucleic Acids Research, 2008.

[8] Kathrin U. Jansen. *Phase I/II study of COVID-19 RNA vaccine BNT162b1 in adults.* Nature, 2020.

[9] Richard B Kennedy. *SARS-CoV-2 immunity : review and applications to phase 3 vaccine candidates.* The Lancet, 2020.

[10] Mark Pandori. *Genomic evidence for reinfection with SARS-CoV-2 : a case study.* The Lancet infectious diseases, 2020.

[11] Malik Peiris. *Serologic Responses in Healthy Adult with SARS-CoV-2 Reinfection, Hong Kong, August 2020.* The new England journal of medicine, 2020.

[12] Timothy N. C. Wells. *Managing intellectual property to develop medicines for the world's poorest.* Nat Rev Drug Discov, 2017.

[13] X. Hou J. Yan S. Du Y. Xue W. Li G. Xiang Zhao, P. and Y. Dong. *Long-term storage of lipid-like nanoparticles for mRNA delivery*, volume 133. Bioactive materials, 2020.

[14] Yaoqi Zhou. *RNA secondary structure prediction using an ensemble of two-dimensional deep neural networks and transfer learning.* Nature communications, 2019.

# 6 Annexe

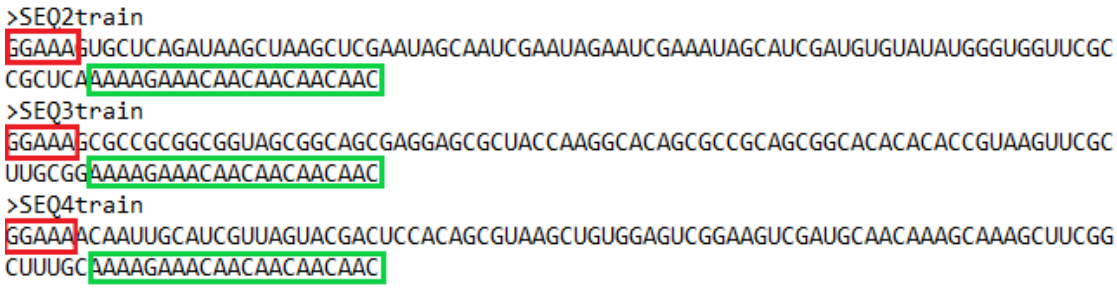


Figure 6 - Préfixes et suffixes identiques présent chez toutes les séquences des deux jeux de données. Le préfixe GGAAA encadré en rouge appartient toujours aux même structures secondaires prédites E (dangling end).

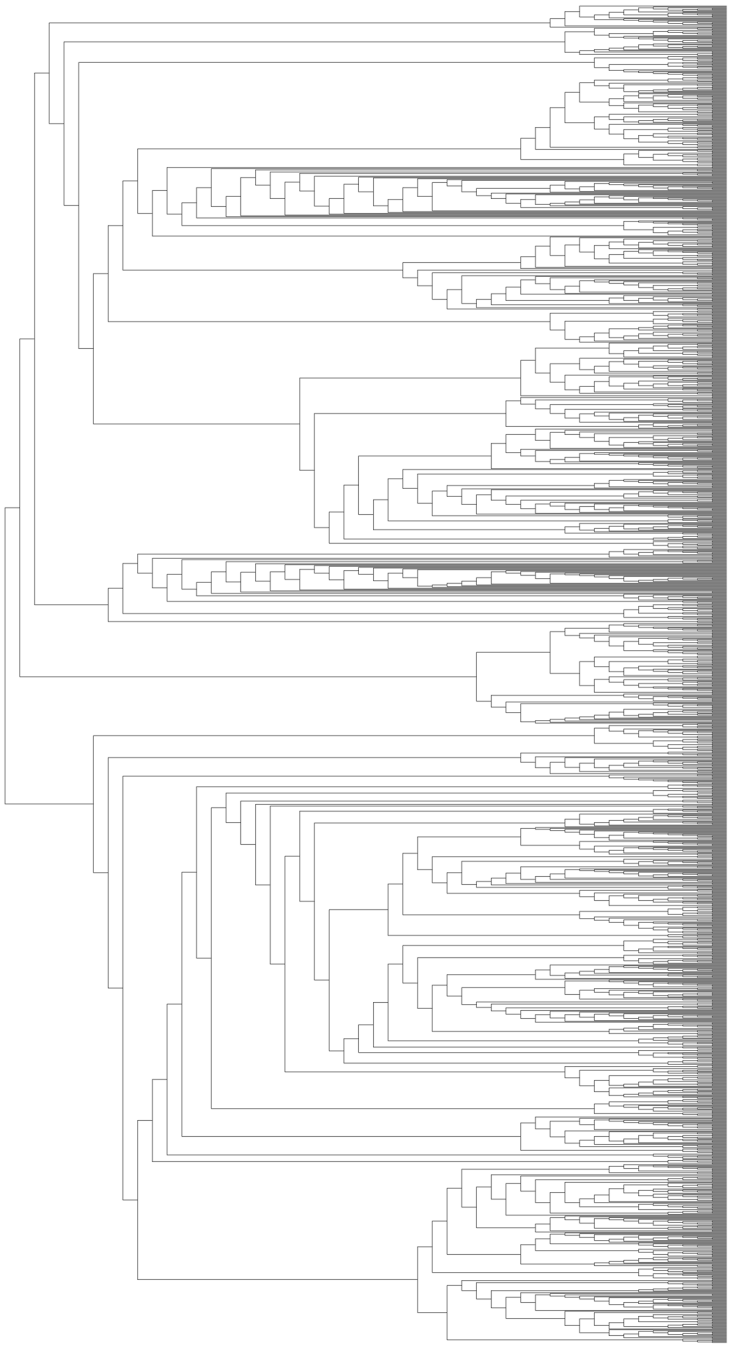

Figure 7 - Arbre non raciné construit avec ClustalW montrant les relations de similarités entre les séquences. Alignment and phylogenetic reconstructions were performed using the function "build" of ETE3 v3.1.1 (Huerta-Cepas et al., 2016) as implemented on the GenomeNet (https ://www.genome.jp/tools/ete/). User provided the multiple sequence alignment. The tree was constructed using FastTree v2.1.8 with default parameters (Price et al., 2009). Values at nodes are SH-like local support.

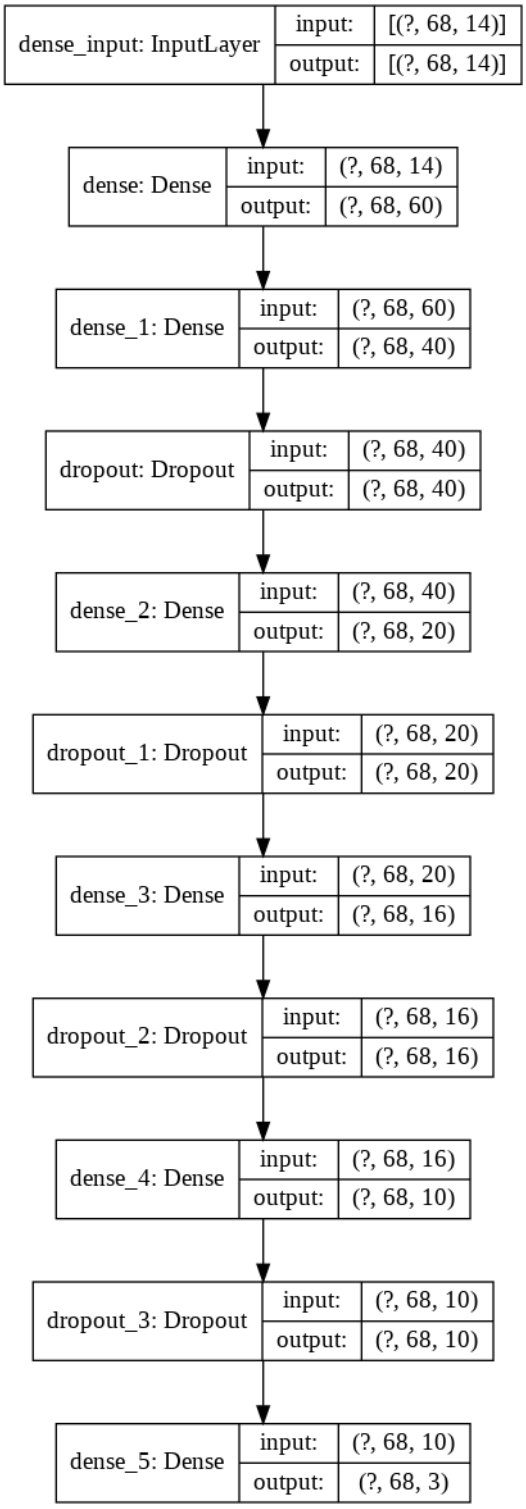


Figure 8 - Notre premier réseau composé uniquement de couches “denses” et de couches de “dropout

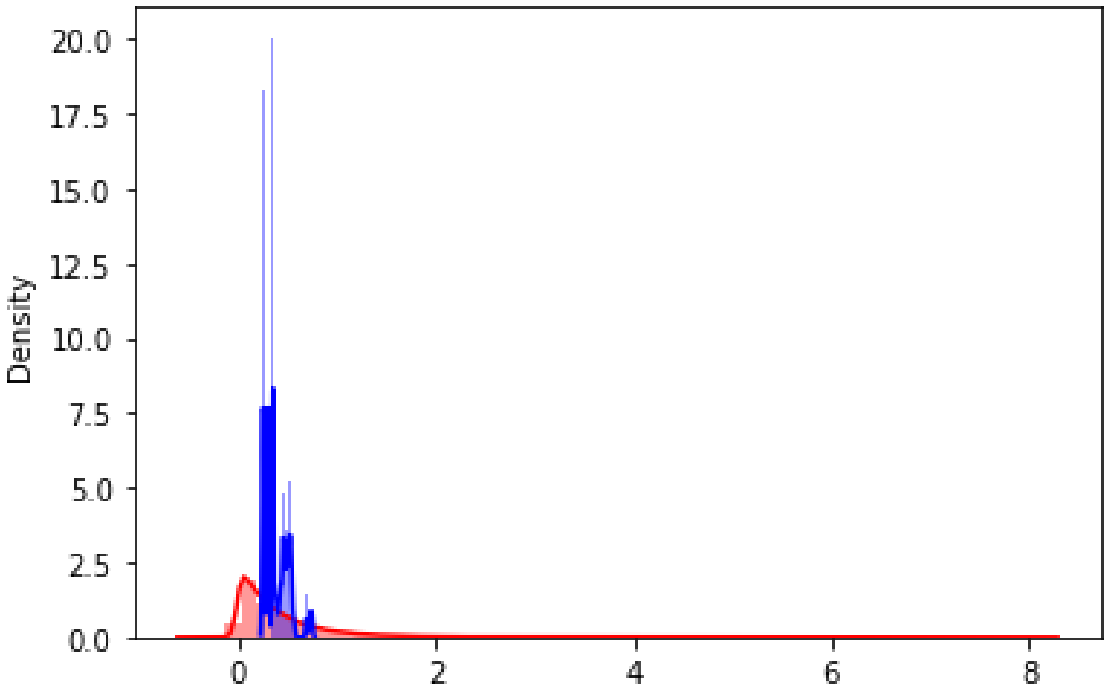


Figure 9 - Densité de prédiction fait par le réseau simple (seulement des couches dense et de dropout en bleu) superposées aux valeurs attendues (rouge).

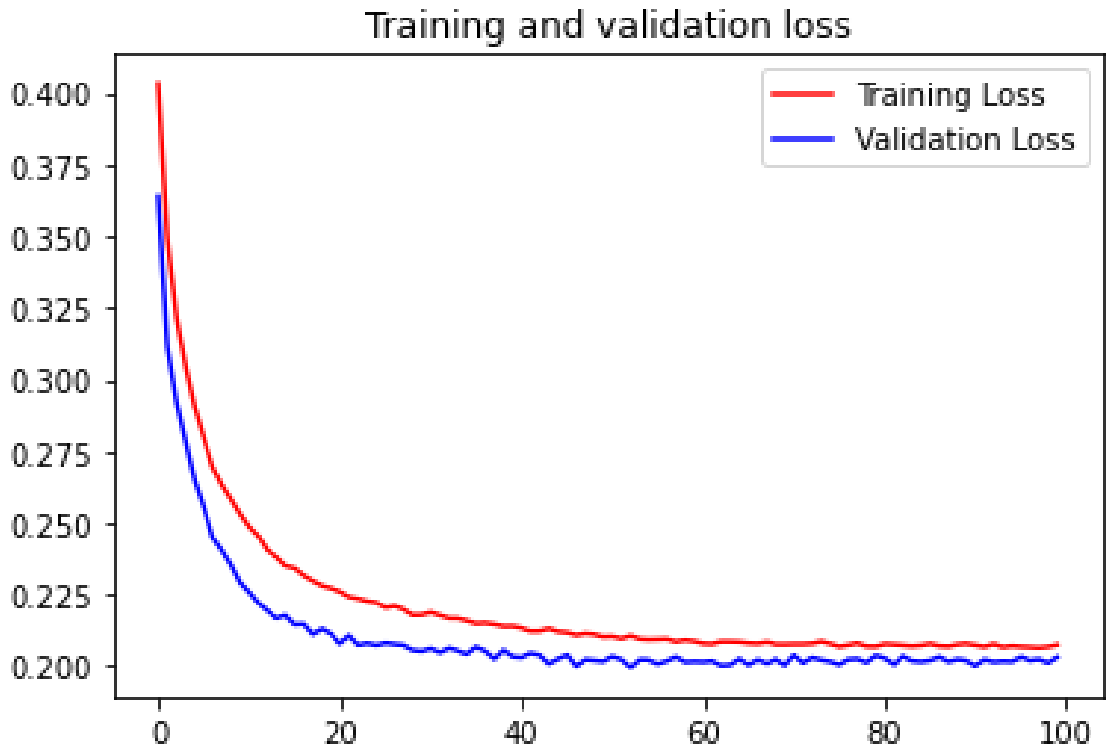


Figure 10 - Graphique de l'évolution de la "val-loss" sur le jeu d'entrainement et de validation sur le premier réseau (seulement des couches dense et de dropout)

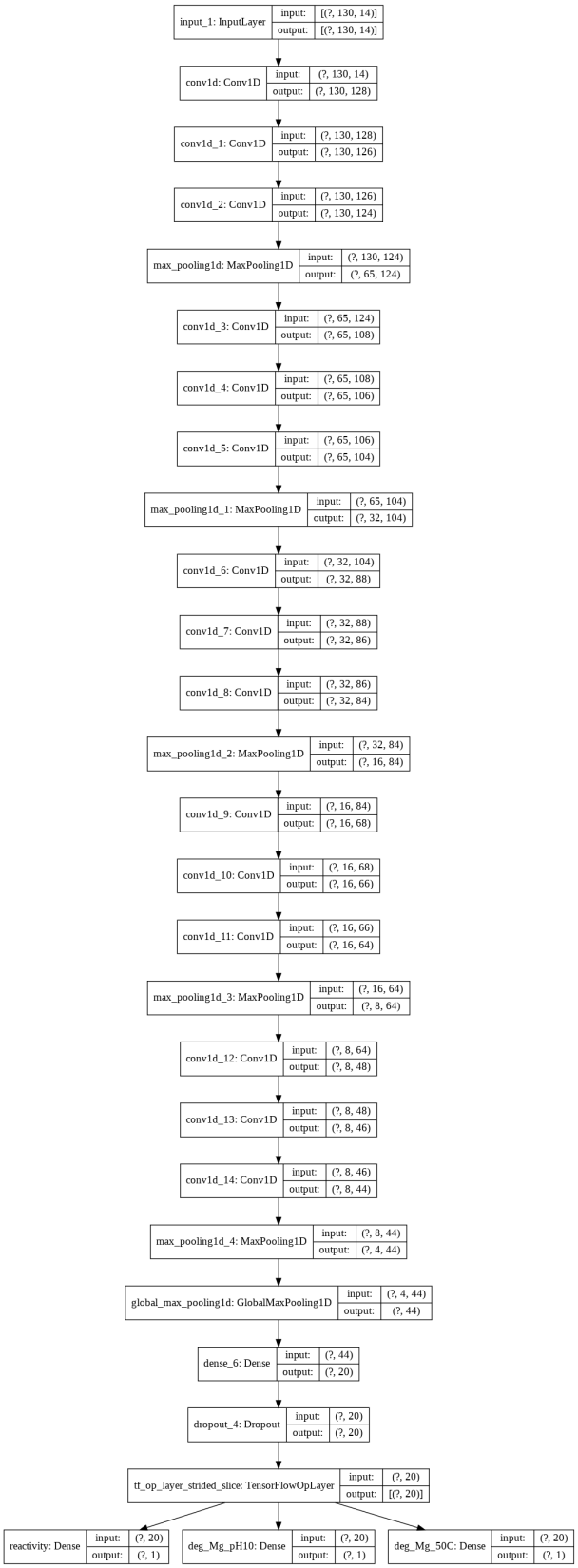


Figure 11 - Notre second réseau composé de couches de “convolution 1D” pouvant prendre des données de taille de variable

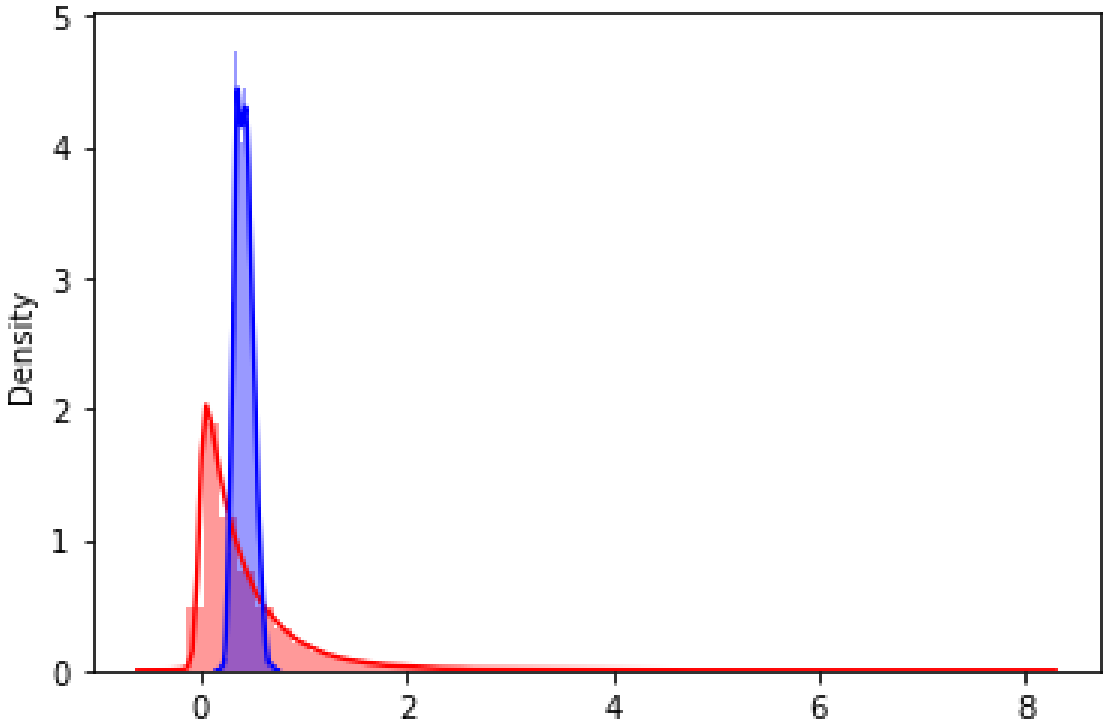


Figure 12 - Densité de prédiction fait par le CNN (bleu) superposées aux valeurs attendues (rouge).

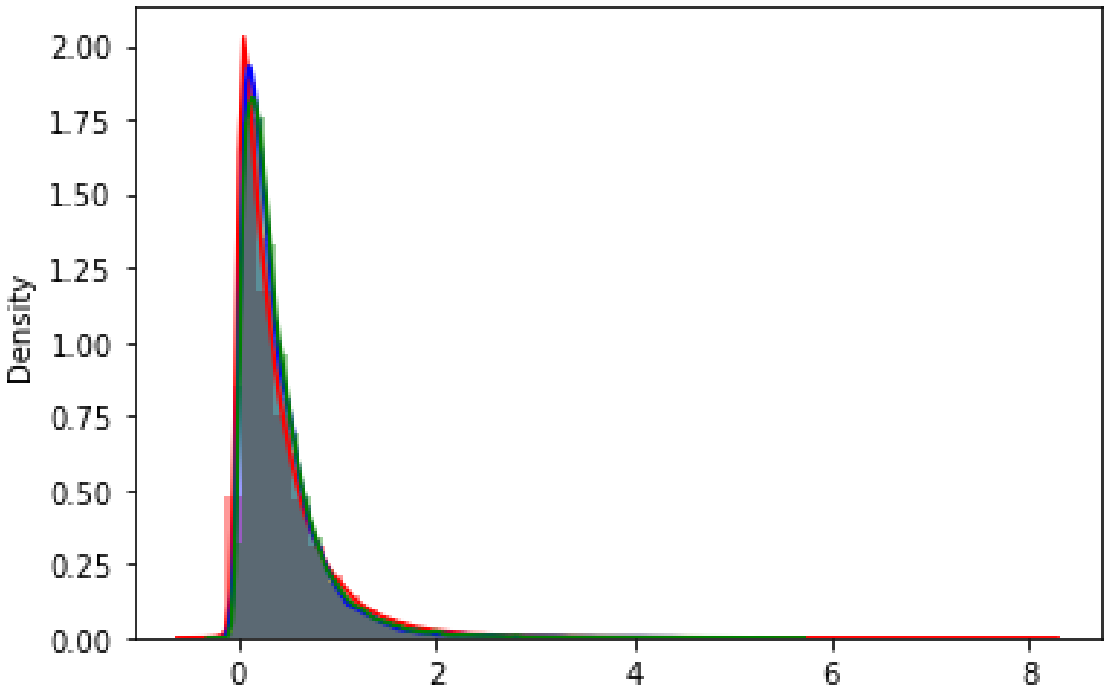


Figure 13 - Densité de prédiction fait par 2 RNNs (en bleu avec des GRU, en vert avec des

LSTM) superposées aux valeurs attendues (rouge).

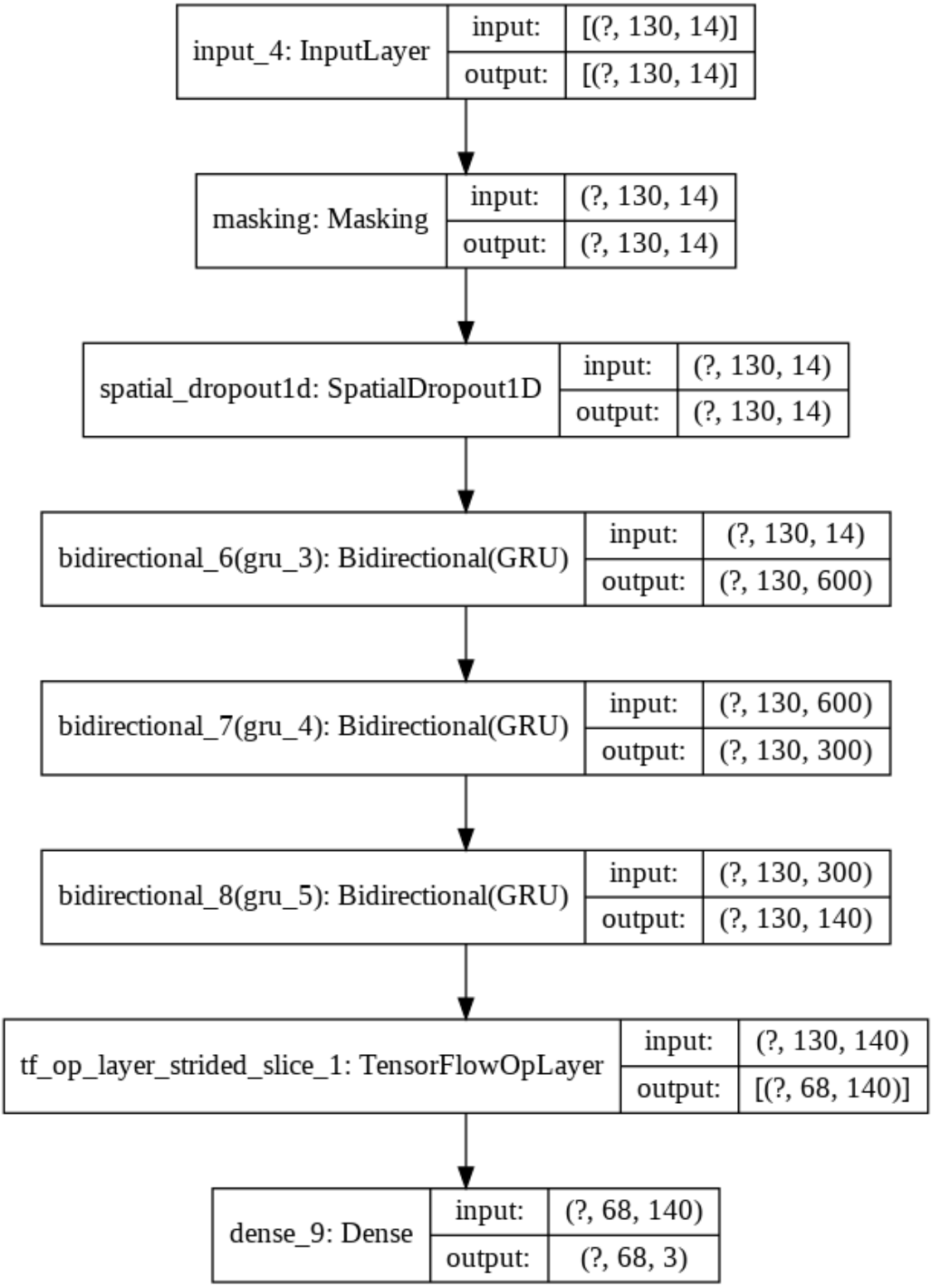


Figure 14 - Le réseau final, matrices en 3 dimensions comme donnée d'entrée.