Université Paris Cité
Faculté des sciences du vivant – Campus des grands moulins
Master 2 Biologie-Informatique

UE Apprentissage, Intelligence Artificielle et Optimisation

Thème

Conception de modèles de prédiction des structures secondaires des ARN messager.

Concours Kaggle Stanford Ribonanza RNA Folding

AGSOUS Salim, FAUCHOIS Antoine, HOLO Donovan et YOUJIl ABADI Souad

Promotion : 2022-2024

Introduction

L'épidémie de COVID-19 a été marquée par l'arrivée sur le marché pharmaceutique de vaccins avec une nouvelle approche : l'utilisation d'ARN messager (ARNm). En effet, ce dernier est encapsulé dans des particules lipidiques avant d'être injecté. Cet ARNm, codant pour un antigène, est capable après pénétration dans les cellules d'être traduit, induisant, par la suite, une réponse immunitaire durable. Cette nouvelle approche, pensée par Katalin Karikó, il y a des décennies, ouvre la voie vers une nouvelle génération de traitements préventifs (Zika, Cancer ou encore le VIH). Cependant, cette approche a également ses inconvénients : l'encapsulation des ARNm dans des particules lipidiques et sa stabilité. La fragilité de ces macromolécules nécessite notamment une conservation à température ultra-basse (environ -80°C). La stabilité des ARNm est étroitement liée à leur structure secondaire. On dénombre trois types principaux d'ARN : ARN messager, ARN de transfert (ARNt) et ARN ribosomique (ARNr). Ces derniers sont caractérisés par des structures secondaires diverses formées particulièrement de boucles (terminale, interne, multiple et hernie). Ces éléments structuraux sont essentiellement déterminés par l'existence de motifs répétés et inversés. La détermination expérimentale des structures secondaires des ARN par SHAPE-seq est une approche expérimentale intéressante, mais chronophage et sujette à des erreurs expérimentales. Elle nécessite l'utilisation du Diméthylsulfate (DMS) et de l'acide 2-aminopyridine-3-carboxylique (2A3) se fixant dans les régions flexibles de l'ARN, jouant ainsi le rôle de terminateur de synthèse. La localisation de ces terminateurs après séquençage NGS (Next Generation Sequencing) renseignent sur la localisation des boucles. Afin d'outrepasser ces inconvénients, des équipes de recherche ont mis au point tout un panel d'outils pour la prédiction de leurs structures secondaires. Des outils tels que ViennaRNA, NUPACK ou encore RNAstructure se basent sur la minimisation d'une fonction d'énergie. Ces approches sont cependant gourmandes en puissance de calcul. À la suite de cela, une nouvelle génération d'outils basés sur des réseaux de neurones plus ou moins complexes ont vu le jour à l'instar de Contrafold, Eternafold ou encore LearnToFold. Ils ont des approches variées passant de réseaux convolutifs à la maximisation de la vraisemblance. Cependant, un manque de précision est encore imputable à l'ensemble de ces outils bio-informatiques. Ainsi, dans le cadre d'une compétition Kaggle nommée : Standford Ribonanza RNA Folding, nous nous sommes saisis de cette problématique afin de proposer un modèle de machine learning suffisamment performant pour prédire la réactivité au DMS et 2A3 à chaque position d'une séquence d'ARNm donnée en entrée. Ces informations permettront ensuite une modélisation de la structure secondaire des ARNm. On se propose de présenter la conception et les performances d'un ensemble de six modèles : deux modèles se basant sur les séquences, deux autres se basant sur les séquences et cartes de contacts et enfin les deux derniers modèles inspirés de ceux qui ont remporté le concours Kaggle OpenVaccine: COVID-19 mRNA degradation.

Matériel et Méthodes

1. Présentation des données

Afin d'élaborer nos modèles de machine learning, nous sommes pour cela partis du jeu de données fourni par la plateforme Kaggle pour cette compétition. Ce jeu est uniquement dédié à l'apprentissage de nos modèles et contient les variables suivantes :

Variables	Description
Sequence_id	ID de la séquence d'ARN
Sequence	Séquence d'ARN
Experiment_type	Réactif chimique utilisé (DMS ou 2A3)
Dataset_name	Nom du dataset de séquençage haut débit d'où sont extrait les séquences
Reads	Nombre de reads dans le séquençage
Signal_to_noise	Rapport $\frac{signal}{bruit\ de\ fond}$
SN_filter	Booléen indiquant si $\frac{signal}{bruit\ de\ fond} \geq 1$ et si le nombre de reads ≥ 100
Reactivity_0001 à Reactivity_0256	Réactivités à la molécule utilisée (experiment_type) pour chacune des positions dans la séquence
Reactivity_error_0001 à reactivity_error_0256	Erreurs des réactivités à la molécule utilisée (experiment_type) pour chacune des positions dans la séquence

tableau 1: ***Description des variables***

Les séquences constituent les entrées à utiliser dans nos modèles, tandis que les réactivités de la position n°1 à n°256 constituent les valeurs à prédire par nos modèles.

2. Analyses exploratoires des données :

Une première analyse exploratoire des données apporte une compréhension globale des données fournies et oriente l'étape du data-management (ou feature engineering). Dans un premier temps, nous avons calculé des paramètres de positions et de tendances

centrales sur les variables quantitatives : nombre de reads, rapports $\frac{signal}{bruit\ de\ fond}$, réactivités et les erreurs de réactivités mais également la longueur des séquences uniques. Ensuite, pour les variables qualitatives (SN filter, experiment type), nous avons construit les tableaux de contingences. Ces calculs s'accompagnent de plusieurs graphiques illustratifs.

3. Visualisation des données

Une technique de visualisation t-SNE (t-distributed Stochastic Neighbor Embedding) a été réalisée pour une visualisation globale des relations entre les séquences pour chaque jeu de données. Le t-SNE est une technique de réduction de dimensionnalité utilisée pour visualiser des ensembles de données de haute dimension dans un espace de dimension réduite. Il s'agit d'une approche probabiliste afin de représenter au mieux les probabilités de voisinages des points dans l'espace réduit par rapport à l'espace d'origine. La perplexité est un paramètre critique qui contrôle l'équilibre entre la préservation des points locaux et globaux dans les données d'origine. Ce paramètre a été évalué de façon empirique.

4. Data-management (feature-engineering)

L'étape de feature engineering ou data-management est fondamentale. Elle permet de formater et de nettoyer les données en vue de les fournir à nos modèles. La qualité des données impacte de façon directe les performances des modèles.

On peut noter que le jeu de données fournit des séquences de 23 séquençages NGS différents. Afin d'accélérer le processus de management des données, leur traitement a été optimisé de sorte à traiter toutes les bases de données en parallèle. Il est important de comprendre que chaque séquence doit avoir des valeurs de réactivité pour le DMS et le 2A3. Ainsi, la première étape fût de créer deux blocs de séquences, l'un avec l'utilisation du DMS comme réactif et l'autre avec le 2A3. Puis, nous avons réalisé une jointure complète par la variable "sequence ID" afin de conserver les séquences avec les deux types de réactifs utilisés.

Ensuite, une attention particulière est à apporter aux séquences pour l'utilisation de celles-ci dans un modèle de machine learning. Elles doivent être encodées et de même longueur. Une première étape a été de trimer les séquences en fonction des valeurs de réactivités renseignées et de combler les extrémités des séquences avec une base inconnue, notée N, de sorte que toutes séquences aient une taille standard de 206 bp. La seconde étape a été de les réencoder sous la forme de liste d'entier en utilisant le dictionnaire suivant : {N:0, A:1, U:2, G:3, C:4}.

5. Construction des cartes de contact

Lors du management des données, nous avons remis en question la suffisance des séquences seules pour caractériser les structures secondaires des ARN. Inspirés par Alpha Fold, qui prédit les structures 3D des protéines via des cartes de contact, nous avons élargi notre dataset avec les probabilités d'appariement des bases (BPPS), calculées par EternaFold. Cette extension vise à renforcer notre modèle en intégrant des informations supplémentaires, améliorant ainsi sa compréhension des nuances complexes des structures des ARN.

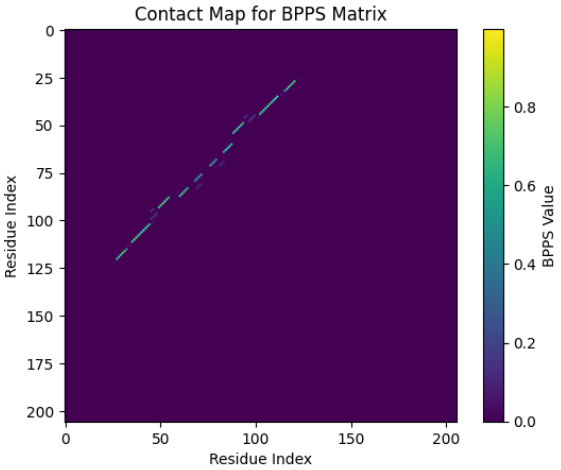

Figure 1 : Carte de contact d'une séquence

Le module Arnie de python permet d'utiliser EternaFold via une Application Programming Interface (API). Ces cartes de contacts de dimension 206 x 206 ont ensuite été aplaties (Flatten) à une dimension de 206^2 = 42 236.

6. Conception des réseaux

Au travers de ces travaux, nous avons voulu appliquer six réseaux différents en faisant varier à la fois la nature des données d'entrées (séquences seules, séquences et cartes de contacts) mais également la structure de ces réseaux liés intrinsèquement à la nature des données d'entrées. La réactivité de chaque base étant supposée dépendante des bases voisines, l'utilisation de réseaux de type RNN (Recurrent Neural Network) est ici privilégiée. Les modèles RNN sont particulièrement intéressants pour les données séquentielles, mais le nombre de paramètres à ajuster devient vite explosif, demandant une mémoire importante et pouvant être à l'origine du problème de vanishing gradient (annulation des gradients). Afin de surmonter ces inconvénients, des couches LSTM (Long Short Term Memory) et GRU (Gated Recurrent Unit) sont utilisées. D'autre part, l'exploitation des cartes de contact s'apparente à analyser des images. L'utilisation de réseaux CNN (Convolutional Neural Network) est ici privilégiée. Le tableau ci-dessous donne les différents modèles testés :

Modèle n°	**Structure globale**	**Entrées du modèle**	
		Séquences d'ARN	Cartes de contact
1	RNN à **LSMT**	✅	❌
2	RNN à **GRU**	✅	❌
3	RNN à **LSTM** combiné à	✅	✅

	un **CNN**		
4	RNN à **GRU** combiné à un **CNN**	✅	✅
5	RNN à **LSTM** combiné à un **CNN** avec **multihead attention**	✅	✅
6	RNN à **GRU** combiné à un **CNN** avec **multihead attention**	✅	✅

tableau 2: ***Vue d'ensemble des modèles***

7. GRU et LSTM des couches réseaux optimisées pour les réseaux RNN

GRU et LSTM sont des types de couches de réseaux RNN. Ils reposent sur le principe de l'utilisation de portes et permettent de définir quelles informations (paramètres et sorties des précédentes itérations) sont à stocker ou à supprimer. Le LSTM comprend trois portes :

- Une forget gate, qui prend en entrée la sortie de la couche à la précédente itération et va définir les éléments à supprimer de la mémoire.
- Une input gate, alimentée par les variables prédictives et définit l'acceptation des sorties de la couche à l'itération actuelle.
- Une output gate, définie si les informations influencent la sortie à l'instant actuel.

Le GRU comprend quant à lui deux portes :

- Une reset gate, détermine les valeurs des sorties de la couche à la précédente itération à oublier,
- Une update gate détermine la quantité d'information à incorporer dans la mémoire.

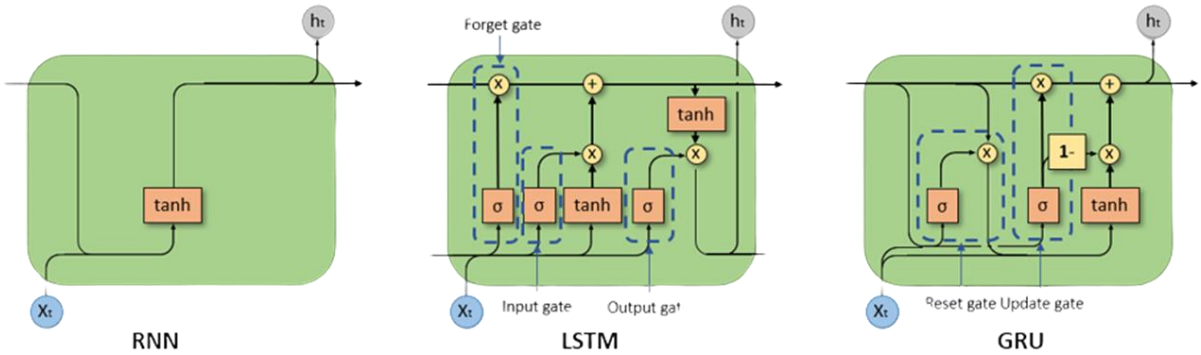

Figure 2 : Schéma des systèmes RNN LSTM et GRU

Ces couches réseaux comprennent une fonction d'activation (tangente hyperbolique) pour le calcul des sorties et une fonction de récurrence (sigmoïde) pour la mise à jour de la mémoire. Ces couches permettent une meilleure modélisation des dépendances à long terme dans les données séquentielles.

8. Convolutional Neural Network (CNN), un réseau pour le traitement des images

Le CNN est un type de réseau utilisé pour l'apprentissage sur des images ou matrices. Il utilise des couches contenant des noyaux de convolutions avec des poids ajustés lors de l'apprentissage. Ces noyaux de convolution (filtres) permettent de capturer des motifs locaux et hiérarchiques lors de l'apprentissage permettant d'ajuster les poids de ces derniers.

9. Prévention du sur-apprentissage

Le sur-apprentissage est un problème majeur en machine learning. Il survient lorsque le modèle développe d'excellentes performances sur le jeu d'apprentissage, mais échoue dans la prédiction des résultats sur un jeu de validation. Pour prévenir ce phénomène, plusieurs solutions peuvent être déployées. D'une part, nous avons opté pour le dropout, permettant de désactiver des neurones tirés aléatoirement d'une couche donnée. Dans notre cas, on fixe le seuil de neurones désactivés à 20%. Ensuite, nous avons également choisi de mettre en place du early-stopping lors des processus d'apprentissage. Ce dernier permet de stopper l'apprentissage de manière précoce dès la détection de sur-apprentissage sur un nombre d'epochs successif. Enfin, nous avons mis en place du batch normalization afin d'harmoniser les batchs entre eux lors de l'apprentissage.

10. Modèles prenant en entrées uniquement les séquences d'ARN (modèles n°1 et n°2):

Dans ce type de modélisation, on propose d'expérimenter deux réseaux, l'un incluant une couche GRU et l'autre une couche LSTM. La structure du réseau se présente de la façon suivante. Une couche GRU ou LSTM prenant en entrées une séquence d'ARN et comprenant 64 neurones. La fonction d'activation utilisée est la tangente hyperbolique ($f(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$) et la fonction de récurrence est une sigmoïde ($f(x) = \frac{1}{1+e^{-x}}$).

Les réseaux incluent ensuite une couche dense à 412 neurones (nombre de réactivités à prédire) et utilise la fonction d'activation ReLU (Rectified Linear Unit définie par $f(x) = max(0; x)$). Le rôle de cette fonction d'activation est d'introduire de la non-linéarité dans le modèle pour accroître la capacité à apprendre des relations complexes et non linéaires entre les caractéristiques des données. Cette couche prend en entrées les sorties de la couche GRU ou LSTM.

Enfin, ces réseaux incluent une couche dense de sortie à 412 neurones. Aucune fonction d'activation n'est utilisée ici, car les valeurs prédites sont quantitatives. Son rôle est d'introduire une linéarité dans le modèle. L'association de ces deux couches offre une approche équilibrée pour effectuer un apprentissage sur des données complexes. Ce modèle reste

simpliste, il offre cependant une première approche de l'exploitation des données. Ces modèles comportent environ 2,8 millions de paramètres.

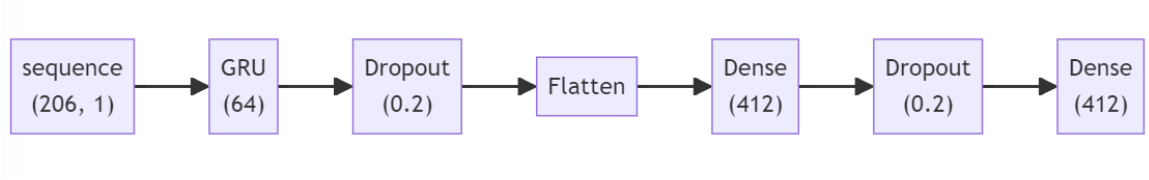

Figure 3 : Structure du modèle RNN à LSTM

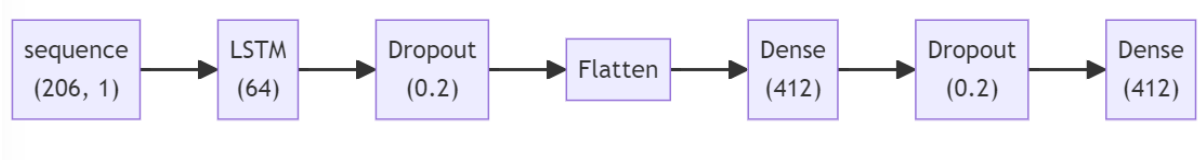

Figure 4 : Structure du modèle RNN à GRU

11. Modèles prenant en entrées les séquences et les cartes de contact (modèles n°3 et n°4)

Ces modèles prennent en entrées les séquences d'ARN et les cartes de contact. Plus élaborés, ces modèles sont une combinaison de réseaux de neurones convolutionnels et de réseaux récurrents. Cette combinaison permet de capturer des motifs locaux et modélise des dépendances à long terme. Ils incluent une couche RNN à LSTM ou GRU comportant 32 neurones avec les mêmes fonctions d'activation et de récurrence que celles utilisées dans le modèle n°1 et n°2. Elles prennent en entrées les séquences d'ARN. Parallèlement à cette couche, nous avons implémenté une couche de convolution 2D à 32 filtres prenant en entrées les cartes de contact. Les sorties de cette dernière sont reliées à une couche de MaxPooling avec des noyaux de dimension 2x2 pour réduire la taille des matrices filtrées. Par la suite, les matrices réduites sont ensuite aplaties sous la forme d'un vecteur avant d'être concaténé à la sortie de la couche LSTM ou GRU. Ce vecteur concaténé constitue l'entrée d'une couche dense à 412 neurones avec pour fonction d'activation ReLU. Enfin, la sortie de cette dernière est connectée à une couche dense de sortie aux mêmes caractéristiques que la précédente couche, mais n'utilisant pas de fonction d'activation. Les matrices ci-dessous présentent la structure des modèles n°3 et n°4. Ces modèles comptent près de 143 millions de paramètres.

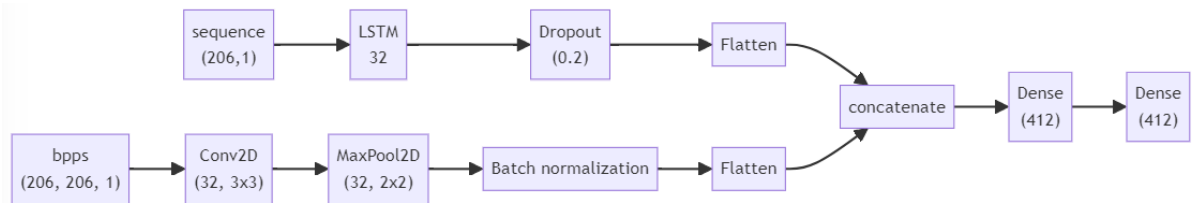

Figure 5 : RNN à LSTM combiné à un CNN

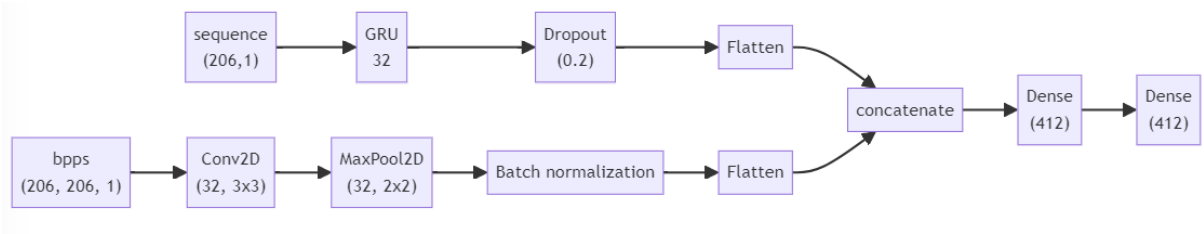

Figure 6 : RNN à GRU combiné à un CNN

12. Modèles inspirés du concours OpenVaccine

Dans le cadre de la compétition Kaggle OppenVaccine, visant à prédire les taux de dégradation des molécules d'ARN pour la stabilité des vaccins à ARNm contre la COVID-19, le modèle gagnant "Nullrecurrent" a utilisé une combinaison de plusieurs types de réseaux de neurones. Intégrant des couches d'attention multitêtes, des réseaux neuronaux récurrents (LSTM ou GRU), et des réseaux CNN, ce modèle a employé une architecture à noyau partagé pour traiter des caractéristiques 1D et 2D. Dans le contexte de ce concours, nous proposons un réseau inspiré du modèle “Nullrecurrent”.

Dans ces modèles, une première ligne de traitement contient une couche d’embedding servant à transformer les séquences d’ARN. Il permet une transformation de la séquence en vecteurs denses dans un espace de caractéristiques continues. Cela permet au modèle d’apprendre des relations nuancées et subtiles entre des nucléotides, au lieu de se baser sur des représentations discrètes (positional et input embedding). Par exemple, si deux nucléotides ont tendance à apparaître dans des contextes similaires ou ont des effets similaires sur la structure de l'ARN, leurs vecteurs d'embedding seront similaires. Chaque nucléotide encodé est transformé en un vecteur de 8 éléments. Ceci offre un compromis entre la complexité du modèle et la capacité à capturer des informations détaillées sur chaque nucléotide. Les sorties de la couche d’embedding sont ensuite reliées à bloc de convolutions 1D successives comprenant 4 couches, avec un nombre de filtres croissant (32, 64, 128 et 256) ayant des tailles de noyau également croissantes (3, 6, 15 et 30), conçues ainsi pour capturer des patterns locaux à différentes échelles. Les noyaux de petite taille capturent des motifs de courte portée, tandis que ceux de plus grandes tailles capturent des motifs de plus longue portée. Cela permettrait au modèle de comprendre à la fois les interactions locales et plus globales entre les nucléotides. Enfin, les sorties du bloc des convolutions 1D entrent dans une couche RNN (soit LSTM ou GRU) de 256 neurones.

La seconde ligne de traitement parallèle contient une succession de deux couches de convolutions 2D prenant en entrées les cartes de contact. Les sorties de ces couches passent ensuite dans une couche dense à 256 neurones. La fonction d’activation utilisée est ici ReLU.

Les sorties des deux lignes de traitement sont ensuite combinées pour former le vecteur de sortie global ensuite intégré dans les couches d’attention évaluant l’importance relative de chaque position dans la séquence d’ARN et dans les données bpps, permettant au modèle de se concentrer sur les interactions les plus pertinentes entre ces deux types de données. Cela aide le modèle à mieux comprendre la structure et les caractéristiques de l'ARN, améliorant ainsi sa

capacité à prédire les réactivités chimiques. Pour finir les sorties de cette couche sont reliées à une couche dense finale de 412 neurones sans utilisation de fonction d'activation. Les modèles RB version LSTM et RB version GRU comptent environ 27 millions de paramètres.

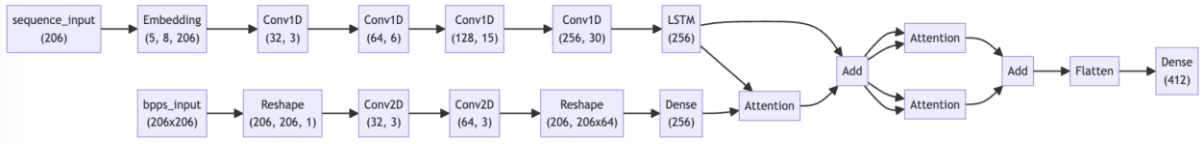

Figure 7 : structure du modèle RB version LSTM

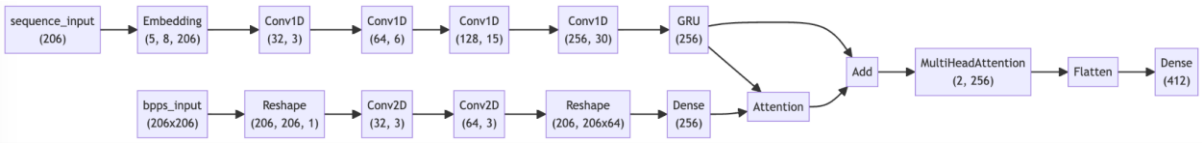

Figure 8 : structure du modèle RB version GRU

13. Hyper paramètres des modèles

En complément de la structure des réseaux, il est aussi nécessaire de fixer les hyper paramètres suivants : le taux d'apprentissages (learning rate), l'optimiseur utilisé pour l'apprentissage, le nombre d'epoch, la fonction de perte utilisée, ou encore la taille des batchs. Le tableau ci-dessous synthétise les hyper paramètres sélectionnés :

Hyper paramètres	**Valeurs**
Optimiseur	Adam
Taux d'apprentissage	1×10^{-3}
Nombre d'epoch	10
Taille des batchs	100
Fonction de perte	Moindres carrés

tableau 3: ***Hyper paramètres***

L'optimiseur Adam, à la fois d'ajuster le taux d'apprentissage en fonction des valeurs des précédents gradients mais il permet également un calcul des gradients à partir des batchs afin d'approcher sa valeur par une moyenne, limitant les temps de calcul. La fonction à minimiser lors de l'apprentissage correspond ici aux moindres carrés ($\sum_{i=1}^{n} \sum_{j=0}^{p} (réactivité_{i,j} - N_w sequence_{i,j})^2$).

En complément de cette métrique nous également mesuré la moyenne des erreurs absolues ($MAE = \sum_{i=1}^{n} \sum_{j=0}^{p} |réactivité_{i,j} - N_w sequence_{i,j}|$). Cette mesure permet une compréhension de l'erreur commise par le modèle.

Résultats

1. Analyses exploratoires des données

Le fichier train_data.csv fourni par la plateforme Kaggle contenait 1 553 465 entrées. Le tableau ci-dessous présente les paramètres de positions et de tendances centrale des variables quantitatives :

Variables	Paramètres
Longueur des séquences uniques	N = 780 418 Min/Max = 115/206 Moy ± sd = 176.68 ± 4.27 Med(IQR) = 177 (177-177)
Nombre de read dans les séquençages (log10)	N = 1 553 465 Moy ± sd = 2.302±0.852 Med(IQR) = 2.225 (1.732-2.761) Min/Max = 0.301/7.169
Rapport signal/bruit de fond	N = 1 553 465 Moy ± sd = 1.26±2.76 Med(IQR) = 0.428 (0.142-1.156) Min/Max = -1.672/256.371

tableau 4: ***Statistiques descriptives des variables quantitatives***

Les paramètres statistiques des longueurs de séquences uniques montrent une grande homogénéité, le tableau ci-dessous présente dans le détail la répartition des longueurs relevés:

Longueurs des séquences uniques	Effectifs
115	2719 (0.35%)
155	2156 (0.28%)
170	14 997 (1.93%)
177	758 047 (97.13%)
206	2499 (0.32%)

tableau 5: ***Distribution des longueurs de séquences***

On peut dire qu'une très large majorité des séquences ont la même taille.Les images ci-dessous présentent les histogrammes respectifs des nombres de reads séquencés ainsi que du rapport $\frac{signal}{bruit\ de\ fond}$.

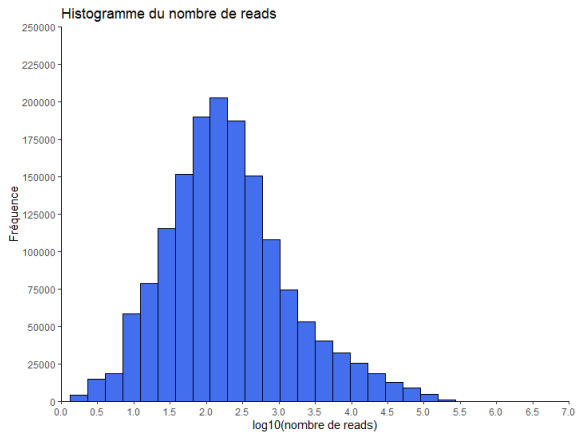

Figure 9 : Histogramme du nombre de reads

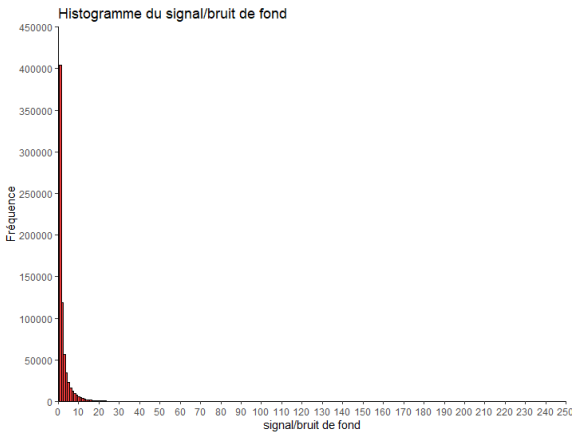

Figure 10 : Histogramme du signal/bruit de fond

Le jeu de données contenait respectivement 768733 (49.48%) de SHAPE-seq avec du 2A3 et 784 732 (50.51%) de SHAPE-seq avec du DMS. On a dénombré également 23 runs séquençages NGS différents desquels sont issus les séquences.

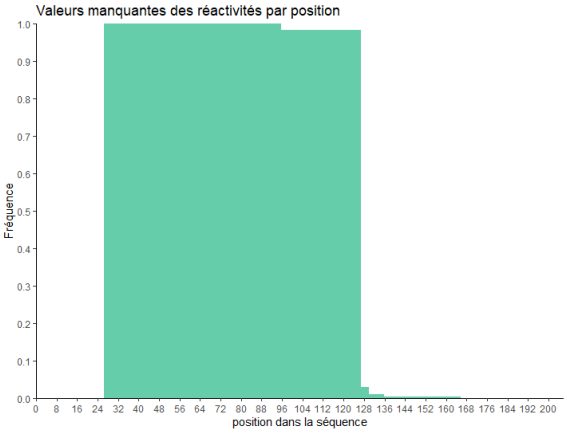

Figure 11 : Valeurs manquantes de réactivités

L'image ci-contre montre les fréquences valeurs non manquantes de réactivités en fonction des positions dans les séquences (de 1 à 206 bp). On observe une importante de valeur manquante sur les extrémités. Ceci est lié au fait qu'une grande partie des séquences ont une taille de 177 bases.

2. Visualisation des données

Les images ci-dessous présentent une observation t-SNE sur deux jeux de données contenant l'un contenant un nombre réduit de 2700 séquences (A) et l'autre contenant un nombre de séquences élevées de 67000 séquences (B). En effet, une visualisation t-SNE sur l'ensemble des séquences n'a pas permis d'obtenir de résultats visualisables.

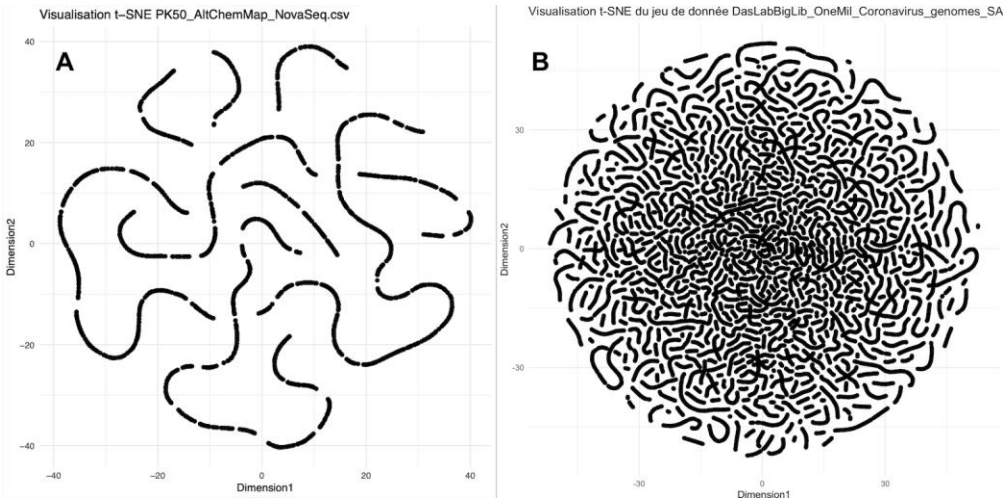

Figure 12 : Visualisation tSNE des sous jeux de données (A) PK50_AltChemMap_NovaSeq et (B) DasLabBigLib_on

Au vu des résultats, nous pouvons indiquer qu'il n'y a pas de cluster apparent. On peut néanmoins observer que les données occupent toute la surface en dimensions réduites. Cette dispersion des données permet de dire que les séquences sont singulières et dissimilaires. Cette singularité des séquences indique une diversité des données, offrant ainsi un bon cadre d'apprentissage pour nos réseaux de neurones.

3. Performances des modèles

Au vu d'un apprentissage des modèles chronophages sur l'ensemble du jeu complet, contenant plus de 760 000 séquences, nous nous sommes restreints à un sous-échantillon de 2800 séquences. Cette taille fût le meilleur compromis entre-temps d'apprentissage et représentativité des données.

- **Modèles prenant en entrées uniquement les séquences d'ARN (modèles n°1 et n°2)**

Les graphiques ci-dessous montrent respectivement l'évolution de la fonction de perte (loss) et de la MAE, pour les échantillons d'apprentissage et de validation, au cours du processus d'apprentissage sur 10 epochs.

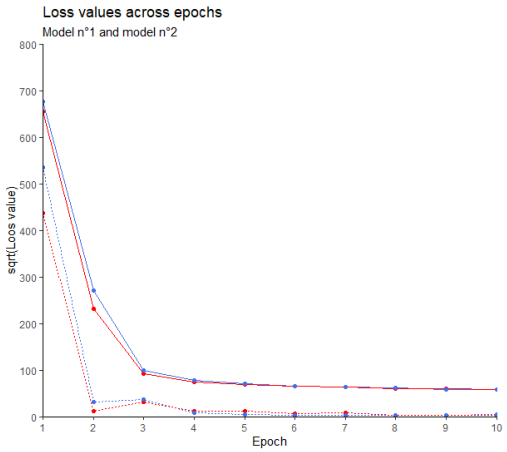

Figure 13 : Évolution de la loss pour les modèles n°1 et n°2

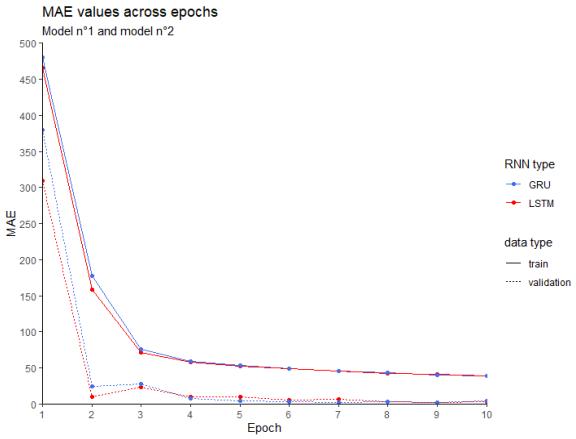

Figure 14 : Évolution de la MAE pour les modèles n°1 et n°2

Premièrement, on peut noter que les valeurs de la loss et de la MAE de l'échantillon de validation sont toujours inférieures à celles de l'échantillon d'apprentissage pour les deux modèles. Ceci peut indiquer un manque de représentativité de l'échantillon de validation. Ensuite, l'allure des courbes d'apprentissage montrent que le choix du taux d'apprentissage s'avère judicieux. On peut observer que les performances du modèle sur l'échantillon d'apprentissage semblent atteindre une asymptote Cela indique qu'il ne parvient pas à apprendre davantage. Ceci peut indiquer qu'il n'est pas suffisamment complexe pour capter les informations pour réaliser de bonnes prédictions.

- **Modèles prenant en entrées les séquences et les cartes de contacts**

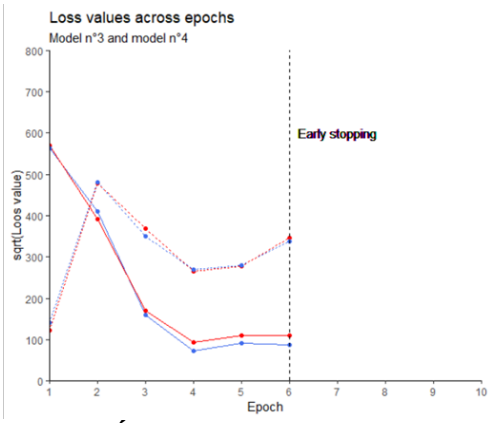

Figure 15 : Évolution de la loss pour les modèles n°3 et n°4

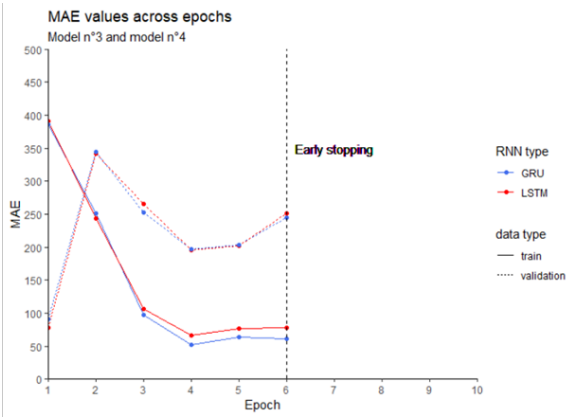

Figure 16 : évolution de la MAE pour les modèles n°3 et n°4

De la même manière que les modèles précédents, on observe un manque de représentativité de l'échantillon de validation. On peut noter un sur apprentissage du modèle. En effet, les performances de l'échantillon de validation deviennent largement inférieures (loss plus élevées) à partir de la troisième epoch. De plus, le early stopping s'est activé à la 6ème epoch à la suite d'un surapprentissage sur 5 itérations successives.

- **Modèles inspirés du concours OpenVaccine**

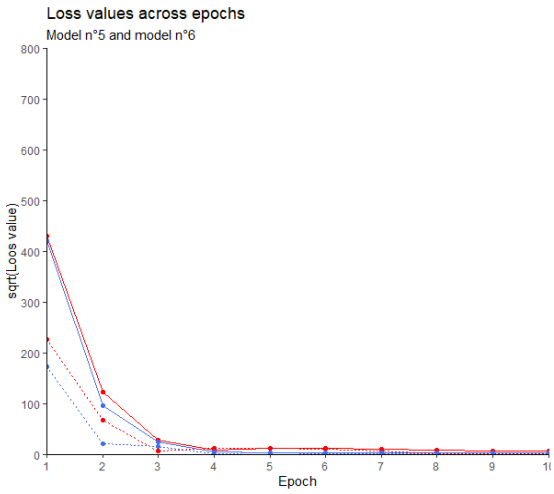

Figure 17 : Évolution de la loss pour les modèles n°5 et n°6

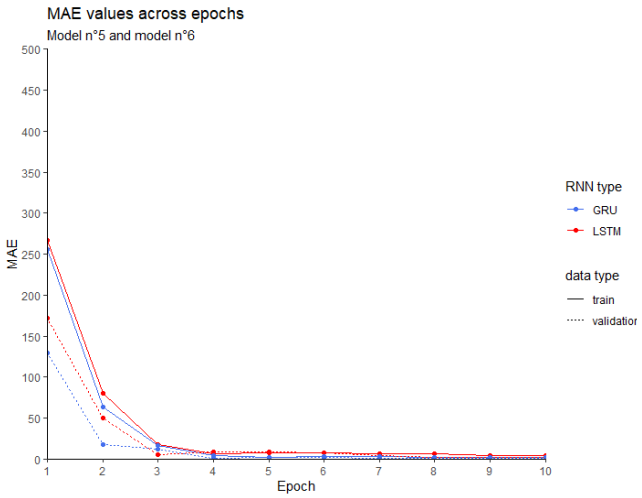

Figure 18 : Évolution de la MAE pour les modèles n°5 et n°6

Une nouvelle fois, il apparaît que l'échantillon de validation n'est pas suffisamment représentatif des données car l'erreur sur cet échantillon est plus faible que sur l'échantillon d'apprentissage. En revanche, on n'observe pas de phénomène de sur-apprentissage et le taux d'apprentissage semble ici bien adapté. De surcroît, lors des phases d'apprentissage, la fonction de perte et la MAE atteignent des valeurs très basses pour les deux modèles, avec une MAE respectivement à 1,72 et 1,21 pour les modèles n°5 et n°6 à la 10ème epoch sur l'échantillon de validation.

Conclusion et discussion

Les modèles utilisant en entrées uniquement les séquences d'ARN constituent une approche pour prédire les réactivités. Cependant souffrent d'un manque de complexité et l'apprentissage finit par atteindre une asymptote de performance pas suffisamment satisfaisante. Ensuite, les modèles GRU/LSTM combiné à du CNN permettent d'incorporer en plus les cartes de contacts des séquences estimées par l'outil EternaFold. Cependant, ces modèles sont caractérisés par un sur-apprentissage pouvant être imputable au nombre important de paramètres. Les modèles RB1 et RB2, inspirés du concours OpnVaccine, inclus en supplément, du multihead attention permettant une focalisation sur les patterns les plus intéressants. Ceci permet de surmonter le surapprentissage rencontré avec les modèles GRIU/LSTM combiné au CNN et offre des performances d'apprentissage et de généralisation inégalées. Ils émergent comme solution plus prometteuse.

Cependant, d'une part, le manque de représentativité du jeu de validation compromet la fiabilité des évaluations. De plus, l'apprentissage sur l'ensemble complet du jeu de données est limité par des contraintes de ressources computationnelles. Pour surmonter ces défis des pistes d'amélioration incluent l'apprentissage des modèles sur le jeu de données complet, le recours à la validation croisée pour renforcer la robustesse des modèles.

De plus, il est important de noter les difficultés rencontrées sur un aspect technique notamment l'entraînement chronophage des modèles qui nous a poussés à réduire drastiquement la taille de nos échantillons d'apprentissage et de validation. De même, l'utilisation des générateurs pour alimenter nos modèles nous a fait éprouver quelques difficultés techniques.

L'expérience acquise dans l'utilisation d'un cluster de calcul offre une compétence précieuse, soulignant la capacité à exploiter des ressources informatiques avancées. De même, la manipulation réussie de fichiers volumineux à l'aide de générateur démontre une aptitude à gérer des données complexes et volumineuses. Enfin pour une approche plus complète et optimale dans cette étude, il aurait été bénéfique d'adopter une stratégie d'enregistrement des poids des modèles entraînés. En les enregistrant il serait possible de réutiliser les modèles pré-entraînés sur un échantillon test, permettant ainsi une évaluation plus cohérente et rapide de la performance des modèles sur des données nouvelles.

Ressources bibliographiques

1. OpenVaccine : COVID-19 mRNA vaccine Degradation Prediction | Kaggle.

(s. d.).https://kaggle.com/competitions/stanford-covid-vaccine

2. Kaggle OpenVaccine Models. (2023).

3. Wayment-Steele, H. K. *et al.* Deep learning models for predicting RNA degradation via dual crowdsourcing. *Nat. Mach. Intell.* **4**, 1174–1184 (2022).

4. Jumper, J., Evans, R., Pritzel, A., Green, T., Figurnov, M., Ronneberger, O., Tunyasuvunakool, K., Bates, R., Žídek, A., Potapenko, A., Bridgland, A., Meyer, C., Kohl, S. A. A., Ballard, A. J., Cowie, A., Romera-Paredes, B., Nikolov, S., Jain, R., Adler, J.,. . . Hassabis, D. (2021). Highly accurate protein structure prediction with AlphaFold. Nature, 596(7873), 583-589. https://doi.org/10.1038/s41586-021-03819-2

Annexe 1 : Distribution des séquences en fonction des différents runs de séquençages

La figure ci-dessous illustre la distribution du nombre de reads par run de séquençage dont sont issus les séquences en fonction du type du réactif SHAPE-seq utilisé (DMS et 2A3).