Réalisé le 26/10/2023

Stanford Ribonanza RNA Folding

Projet Kaggle : Création d'un modèle de prédiction de la structure de molécules d'ARN

UE Apprentissage, Intelligence artificielle et Optimisation 1 - M2BI

OUADAH Lilia - MERABET Anis - BAILLIF Marine - JEAN-MARIE ROUDE

---

1. Introduction

1.1 Contexte et objectif

L'ARN est une molécule présente dans la plupart des organismes vivants et des virus, jouant un rôle crucial dans de nombreux processus, tels que le transfert de l'information génétique et la régulation [1]. Ces fonctions reposent sur la capacité des molécules d'ARN à adopter diverses structures secondaires ou tertiaires. Des études ont montré qu'une mutation entraînant un mauvais repliement de l'ARN pouvait être impliquée dans des pathologies [2]. Une meilleure compréhension de la manipulation de l'ARN pourrait ainsi ouvrir la voie à de nombreuses avancées, notamment dans le domaine médical. Cependant, cette tâche demeure complexe en raison du grand nombre de conformations que peut adopter la molécule.

À l'origine, la résolution de la structure de l'ARN était principalement réalisée par cristallographie aux rayons X ou par résonance magnétique nucléaire (NMR) [1][3]. Cependant, en raison de la grande flexibilité de la molécule d'ARN, sa cristallisation s'avère difficile. Malgré les progrès réalisés grâce à la Cryo-EM4, les bases de données biologiques contiennent peu de structures d'ARN. De plus, ces méthodes rendent complexe l'étude des structures dans des cellules vivantes.

Les approches expérimentales in vivo ou in vitro utilisent généralement des sondes qui se fixent à la molécule d'ARN. L'utilisation de sondes combinées aux techniques de profilage mutationnel (MaP) a montré une certaine efficacité dans la prédiction des structures secondaires des ARN. Dans ces méthodes, des réactifs tels que le DMS (diméthyl sulfate) ou le 2A3 (2-Aminopyridine-3-carboxylic acid imidazolide) se fixent à certaines zones de l'ARN en fonction de la nature des nucléotides et du repliement local de l'ARN. Après fixation, une rétrotranscription est réalisée pour obtenir de l'ADNc à partir de l'ARN. Cependant, la présence de sondes sur certaines zones de l'ARN peut entraîner des mutations lors de la rétrotranscription. L'identification de ces mutations dans la séquence d'ARN connue permet, après séquençage à haut débit, d'obtenir un profil de réactivité de la molécule. Ce profil offre un aperçu du repliement de l'ARN, bien que des difficultés subsistent pour ces méthodes [5][6].

En parallèle des approches expérimentales, la problématique de la prédiction du repliement des ARN a été abordée par des approches computationnelles. Parmi ces approches, on trouve les modèles de deep learning, qui ont permis d'accomplir des avancées significatives dans ce domaine [1]. Néanmoins, des défis subsistent dans la prédiction de la structure d'une molécule complexe telle que l'ARN.

L'objectif de ce projet, proposé par l'Université de Stanford sur la plateforme de compétition Kaggle, est de développer un algorithme permettant de prédire les valeurs de réactivité des nucléotides vis-à-vis du composé chimique habituellement déterminées expérimentalement. La prédiction de ces valeurs permettrait d'obtenir une meilleure compréhension de la structure de l'ARN

2. Matériels et Méthodes

2.1 Présentation des données

Un premier jeu de données nous a été mis à dispositions sur la plateforme Kaggle pour ce projet. Celui-ci est composé au total de 1 643 680 observations, dont 806 578 séquences d'ARN uniques ayant une longueur comprise entre 115 et 206 nucléotides(ANNEXE 1). Ce jeu est composé de différents descripteurs pour chacune de nos observations, parmi celles-ci on retrouve la colonne *sequence* qui correspond à la séquence nucléotidique (A,U,C,G) de la molécule d'ARN, *dataset_name* qui renseigne sur la provenance des informations de la séquence, *reads* qui correspond au comptage des reads aligné à la séquence lors du séquençage et ayant ainsi permis d'obtenir le profil de réactivité de la molécule d'ARN, *experiment_type* correspond au type d'expérience appliqué avec l'utilisation du DMS ou avec l'utilisation du 2A3 permettant d'obtenir pour chaque séquence son profilage mutationnel. Le *signal_to_noise (ANNEXE 2)* correspond à la valeur du signal divisé par le bruit du profil, qui a été définie comme étant la moyenne des mesures sur les nucléotides sondés divisé sur la moyenne des erreurs statistiques des mesures sur les nucléotides sondés. *SN_filter (ANNEXE 3)* est une valeur qui prend la valeur 1 si le profil de la séquence a un *signal_to_noise*≥ 1.00 et une valeur de *reads*≥ 100 renseignant ainsi que la qualité de la séquence, et vaut 0 dans le cas contraire. Les colonnes *reactivity_xxxx*, correspondent aux valeurs de réactivités obtenues à la position nucléotidique *xxxx* au sein de la séquence et les colonnes *reactivity_error_xxxx* correspondent aux erreurs expérimentales associés à la réactivité au sein de la même position.

Le second jeu de données correspond au jeu de test composé de 1 031 888 séquences pour lesquelles les valeurs de réactivités ne sont pas connues, ce jeu est composé de séquences dont la longueur est comprise entre 207 et 457 nucléotides.

Le profil de réactivité d'une séquence d'ARN est composé de l'ensemble de ses valeurs de réactivité pour chacun des nucléotides de la séquence. Ces valeurs sont à prédire pour les expériences au DMS et au 2A3 pour chacun des nucléotides présents dans la séquence. Sont associée aux réactivités pour chaque nucléotides une erreur de réactivité caractérise l'erreur statistique dû à l'expérience.

2.2 Preprocessing

2.2.1 Filtrage du jeu de données

Parmi les 1 643 680 séquences d'ARN un premier filtrage a été effectué en ne conservant que les séquences ayant une valeur de *SN_filter* égale à 1, cette valeur reflète la qualité du profil de réactivité obtenu pour chaque séquence et nous a permis de conserver 437 919 observations (26.6%) dont 242 729 séquences uniques (ANNEXE 3). Parmi les ARN restants, des redondances sont observées car les profils de réactivité proviennent de différents ensembles de données de séquençage, cela signifie que plusieurs expériences ont été menées avec les réactifs pour une même séquence. Afin de ne conserver qu'un profil de réactivité par

expérience au DMS ainsi qu'au 2A3 pour chacune des séquences redondantes, nous avons sélectionné la séquence dont la valeur du *signal_to_noise* était la plus grande, ce qui nous a permis de conserver 410 708 observations (24.9%) avec tout autant de séquences uniques.

À partir de ce jeu de données réduit, nous avons conçu un jeu uniquement composé des séquences dont les valeurs de réactivités ont été obtenues en DMS et 2A3 ce qui nous a permis d'en extraire un jeu de données (cleared_RNA) de 335 958 observations (20.4%) composé de 167 979 séquences uniques. C'est sur ce jeu de données que nous travaillerons.

Après cette étape, le jeu de données a été séparé en train avec 117 585 observations (70%), en un jeu de validation avec 25 196 observations (15%), et un jeu test avec 25 198 observations (~15%).

Un deuxième jeu de données a été réalisé de la même manière en conservant les données ayant un SN filter = 1 et en effectuant le retrait des copies des résultats de réactivités pour une même expérience en ne gardant que celles qui ont un meilleur signal to noise. On ne retient pas ici les séquences pour lesquelles les réactivités sont manquantes pour l'une ou l'autre des expériences. On effectuera un padding et masking ultérieur pour ces dernières. On obtient donc 197258 observations pour 2A3 et 213450 observations pour DMS.

2.2.2 Normalisation

Pour l'entraînement du modèle, il est nécessaire de conserver des échelles de valeurs similaires au sein de nos données afin de limiter les soucis de disparition ou encore d'explosions de gradients lors de l'ajustement des poids du modèle.

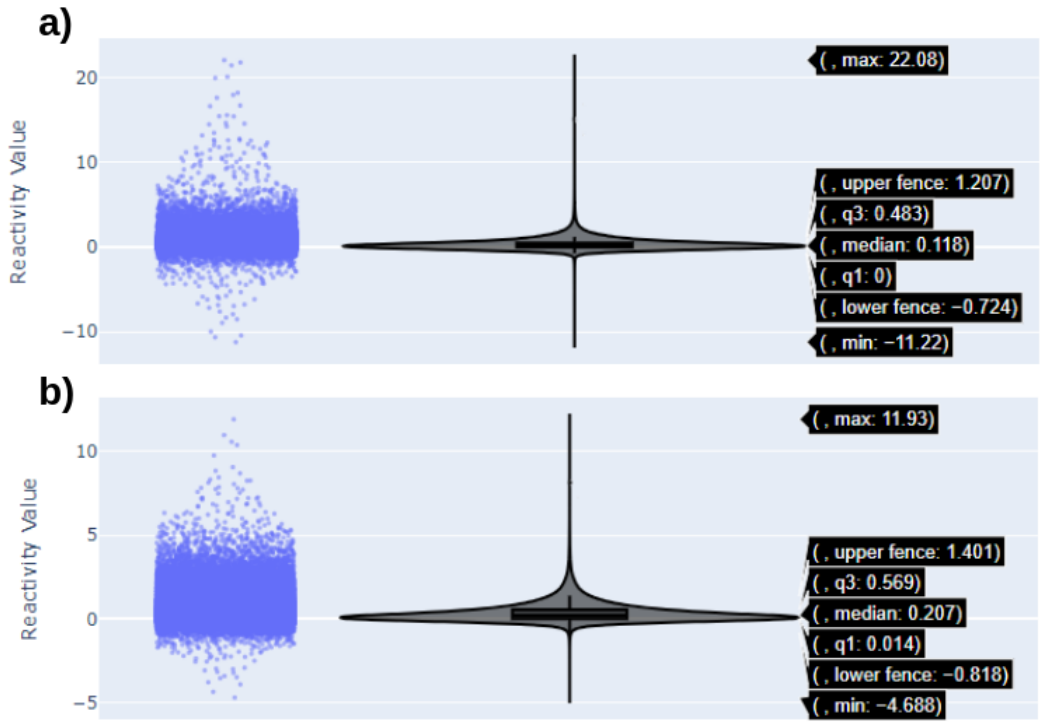


Figure 1: Distribution des valeurs de réactivités pour le DMS et le 2A3 **a)** Violin plot des valeurs de réactivités du DMS. **b)** Violin plot des valeurs de réactivités du 2A3.

Le noyau du graphique en forme de violon (violin plot) adopte une courbe hyperbolique et montre une similitude avec une distribution de loi normale. Cependant, on observe la présence d'outliers. Par conséquent, les valeurs de réactivité, mesurées en DMS et 2A3, ont été normalisées à l'aide d'un z-score robuste pour atténuer leur impact par rapport à un calcul basé sur la moyenne. La formule utilisée pour calculer la valeur du z-score robuste est la suivante :

$$R.Z.score = \frac{0.6745*(X_i - Median)}{MAD}$$

Where MAD = median(|X-median|)

2.2.3 Encodage

Les méthodes basées sur l'apprentissage automatique (machine learning) opèrent avec des données numériques, ce qui rend nécessaire l'encodage de nos séquences. Pour ce faire, nous utilisons un encodage one-hot, qui permet de représenter les bases A, U, C, et G. Chacune de ces bases est associée à un vecteur unidimensionnel (1D) de 4 valeurs. Ainsi, la base A est représentée par le vecteur [1, 0, 0, 0], la base U par [0, 1, 0, 0], la base C par [0, 0, 1, 0], et la base G par [0, 0, 0, 1]. Par conséquent, une séquence d'ARN est représentée par une matrice composée de ces vecteurs correspondant à chaque base de la séquence. Pour une séquence de taille n, la dimension de la matrice est notée (n, 4).

2.2.4 Padding

Les modèles supervisées basés sur le deep learning nécessitent de définir au préalable les dimensions des données (x) que l'on souhaite lui transmettre en entrée pour les prédictions des données (y). Les longueurs des séquences étant de longueur différentes, il est nécessaire d ‘harmoniser ces données pour que chaque représentation matricielle des séquences aient la même dimension (n, m). Pour cela, le padding est appliqué afin d’élargir la longueur des k matrices avec k le nombre de séquences, afin de représenter nos matrices dans une dimension (n, 4) avec n la longueur maximale des séquences et 4 la taille d’un vecteur représentatif de l’encodage des bases. Le padding se résulte par l’ajout de vecteurs remplis de zéro afin d’obtenir des matrices de dimensions (n, 4). La longueur de séquence la plus longue est de 457 nucléotides au sein du jeu de données de soumission de Kaggle, c’est pour cela que la longueur du padding sera de 457. Ce même padding sera appliqué au sein de la matrice de données des valeurs y de dimensions (n, 2) avec la seconde dimension correspondant à l’expérience au 2A3 ainsi qu’au DMS.

2.2.5 Masking

Certaines valeurs de réactivité sont manquantes au sein du jeu de données à prédire (y), parmi certaines positions en amont et en aval de la séquence. De plus, suite au padding des séquences certaines valeurs ne sont pas nécessaires pour l’apprentissage du modèle. C’est pourquoi lors de l’apprentissage, on spécifie au modèle de ne pas prendre en compte les valeurs manquantes ainsi que les valeurs paddées par l’utilisation d’une fonction de perte personnalisée.

2.3 Modèles

Les réseaux Long short-term memory (LSTM), sont des réseaux de neurones qui ont su démontré de bonnes performances dans la prédiction de données liés aux séquences ARN [1]. Ces réseaux ont été conçus pour capturer les dépendances séquentielles et permettent l’intégration de relations à longue distance entre les données, et cela est intéressant quand on sait que la probabilité d’apparition d’une base dépend aussi de son environnement. De ce fait, il nous a semblé judicieux d’utiliser des modèles basés sur les LSTM. Ces critères en font un

modèle permettant de capturer des motifs et de potentielles interactions pouvant potentiellement expliquer les valeurs de réactivité. Pour la création des modèles, ceux-ci prendront des valeurs en entrée de dimensions (457, 4) et prédisent des valeurs en sortie de dimensions (457, 2).

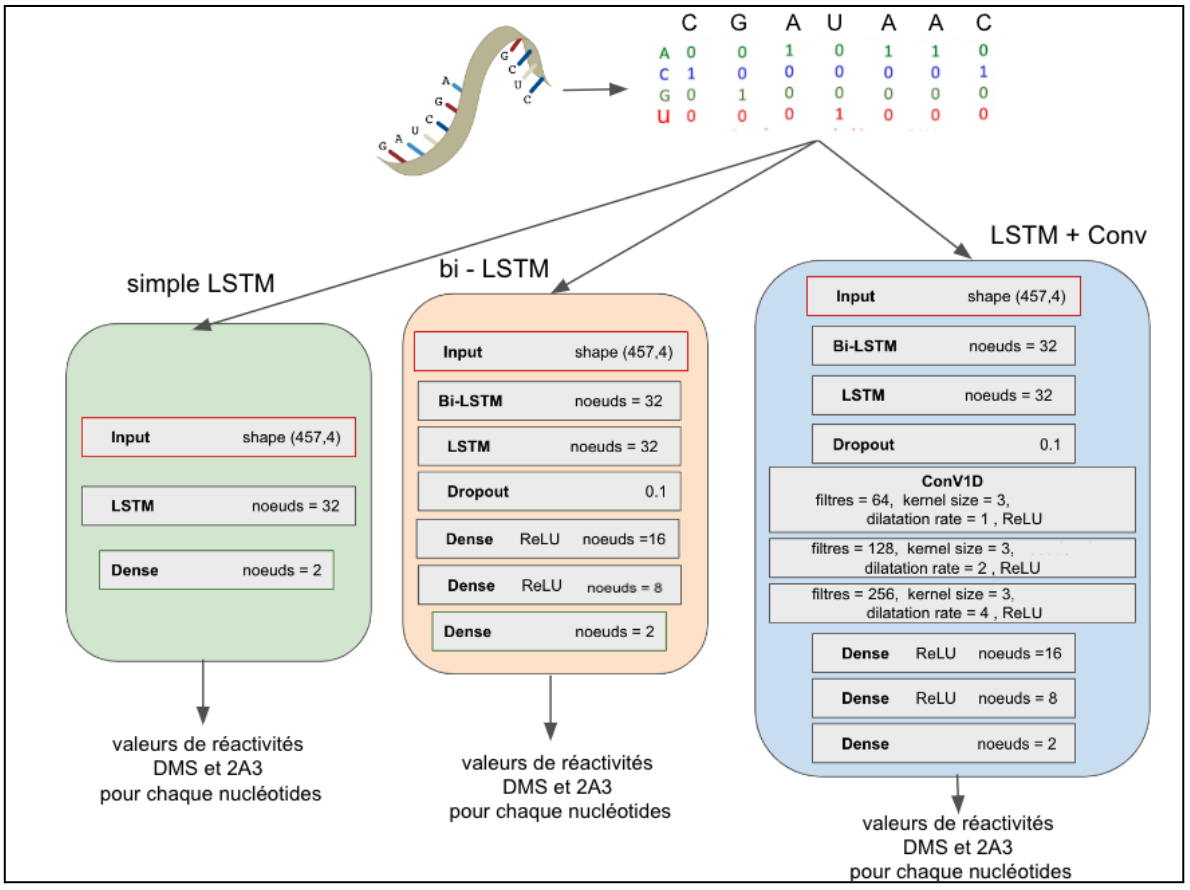


Figure 2: Représentation de l'architecture des modèles simple LSTM, bi - LSTM et Convolutional LSTM

Un premier modèle (voir Figure 2) composé d'une couche cachée LSTM de 32 nœuds a été conçu afin d'évaluer dans un premier temps les performances du modèle. Un second modèle a été réalisé avec l'introduction des Bidirectional LSTM, permettant la prise en compte des informations présentes en amont et en aval de la séquence pour les prédictions. Enfin, en s'inspirant d'architectures plus récentes, nous avons conçu un modèle combinant des couches bidirectionnelles LSTM, des couches LSTM simples ainsi que des couches de convolutions.

Pour la descente de gradient de l'ensemble des modèles, nous avons choisi l'optimiseur adam ainsi qu'un learning rate de 0.005.

2.4 Optimisation des hyperparamètres du modèle

Dans le but de déterminer la structure optimale du modèle et d'identifier l'ensemble d'hyperparamètres d'entraînement le plus performant, nous avons entrepris l'optimisation d'un modèle de type LSTM en utilisant la bibliothèque Python Optuna. L'algorithme d'optimisation choisi est le TPE (Tree-structured Parzen Estimator), qui est l'algorithme par défaut de cette bibliothèque. Le TPE repose sur une approche bayésienne visant à trouver les meilleurs hyperparamètres pour un modèle donné en utilisant un modèle probabiliste pour estimer comment les différentes configurations d'hyper paramètres influencent les

performances du modèle. Son objectif est de réduire le nombre d'essais nécessaires pour identifier les hyperparamètres optimaux.

L'optimisation des hyperparamètres a été réalisée en utilisant un échantillon aléatoire de 5% des données préalablement traitées. La MSE a été définie comme le critère à minimiser lors de l'optimisation du modèle d'apprentissage automatique. Les différents hyperparamètres, qui servent de paramètres d'entrée, sont présentés dans le tableau 1, accompagnés de leurs espaces de recherche respectifs.

La structure du modèle peut être résumée comme suit : il débute par une couche de masquage en entrée, suivie par de 1 à 5 couches de LSTM simples ou bidirectionnelles. Ces couches peuvent être suivies, ou non, par une couche de *batch normalization* et/ou une couche de *dropout*. Ensuite, entre 0 et 3 couches denses peuvent être intercalées avant d'arriver à une couche dense de sortie.

Tableau 1 : Optimisation des hyperparamètres du modèle avec Optuna

Hyperparamètres optimisés		
Hyperparamètres	**Espaces de recherche**	**Caractéristiques**
Nombre de couches LSTM	1 à 5	return_sequences=True
Type de couche LSTM	LSTM simple ou Bidirectionnel	-
Nombre d'unités LSTM	32 à 256	Le nombre d'unité d'LSTM est toujours inférieur à celui de la couche précédente
Usage de couche de *dropout*	Présente ou absente	Après chaque couche LSTM
Taux de dropout	0 à 0.5	-
Usage de couche de *batch normalization*	Présente ou absente	Après chaque couche LSTM
Nombre de couches denses	0 à 3	Se trouvent après les couches LSTM
Nombre d'unités dans les couches denses	Varie du nombre d'unité de la dernière couche LSTM à 16	Le nombre d'unité de la couche est inférieur à celui de la couche précédente
Hyperparamètres fixes		
Hyperparamètres	**Valeurs**	
Couche d'entrée	Couche de masking avec une forme (None, None, 4)	
Couche de sortie	Couche dense avec une forme de (None, None, 2)	
Fonction d'activation	"Relu" pour les couches dense et "Linear" pour la couche de sortie	
Optimiseur	Adam	
Fonction de perte	fonction de perte customisée	
Nombre d'époques	10	

Taille du batch	64

L'optimisation de l'architecture s'est interrompue après 35 itérations et le modèle produisant la plus faible valeur de MSE a été retenu. Ce modèle a été entraîné sur l'ensemble des données obtenues après le prétraitement. À cette fin, un générateur customisé a été utilisé pour appliquer du padding aux séquences d'ARN au sein de chaque batch, et il a chargé les données de manière dynamique du modèle. De plus, une fonction de perte personnalisée, qui renvoie la MSE, a été employée pour masquer les valeurs paddées ainsi que les valeurs NaN dans les réactivités. Le nombre d'époques a été fixé à 1000, avec une mise en place d'un arrêt anticipé (*early stopping*) pour mettre fin à l'entraînement si le modèle ne montre plus d'amélioration pendant 10 époques consécutives.
À chaque itération, le modèle enregistrait ses poids et l'état des paramètres de son optimiseur. En plus de la MSE, nous calculons également la MAE et la RMSE, bien que nous ne les utilisions pas pour l'entraînement du modèle. Une fois l'entraînement terminé, nous retenons l'état du modèle présentant la MSE la plus faible pour évaluer ses performances et générer des prédictions sur les jeux de données de Kaggle en vue de soumettre notre travail dans le cadre de la compétition.

3. Résultats

3.1 Performances des modèles

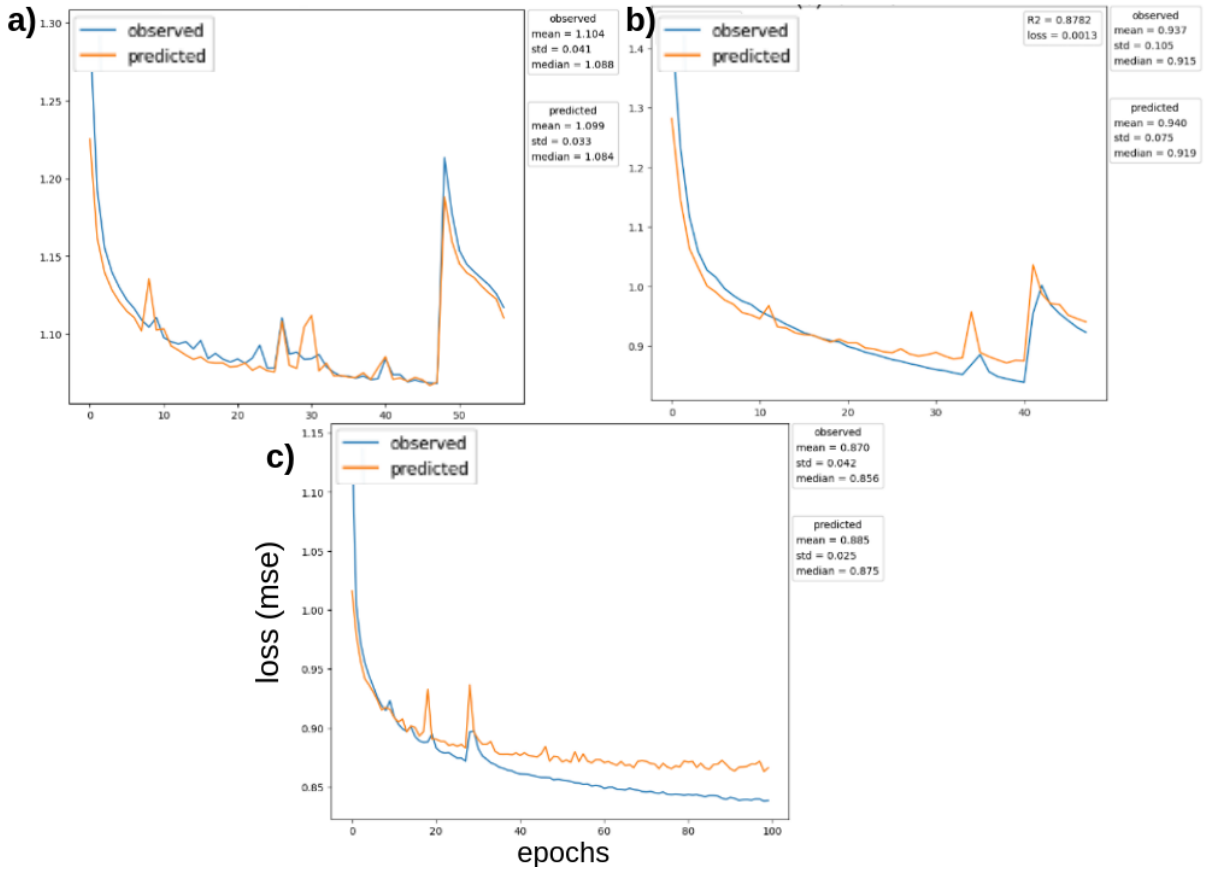


Figure 3: Courbes d'évolution de la fonction de perte (loss) en fonction du nombre d'époques pour les jeux de données train et de validation **a)** courbe de loss pour le modèle LSTM, avec la plus basse valeure de loss atteignant **b)** modèle bidirectionnal LSTM **c)** modèle combinant LSTM et couches de convolution

L’évaluation de nos modèles s’est basée sur l’évaluation de la loss et de la validation loss pour chacun d’entre eux.
Pour le modèle LSTM simple on a une baisse de la fonction de perte des données d’entrainements avec une tendance similaire pour celle des données de validation pour les 40 premières époques avec néanmoins quelques oscillations tout le long et un léger overfitting. Ceci peut notamment s’expliquer par le choix des hyperparamètres potentiellement non optimaux pour notre modèle ou du bruit au sein de nos données. A partir de la 50e époque, on observe une soudaine augmentation de la fonction de perte entraînant l’arrêt de l'entraînement. La moyenne de la fonction de perte pour les données d’entraînement est ainsi de 1.104 et pour les données de validation de 1.099.
Pour le modèle Bi LSTM, on fait face à la même problématique, avec néanmoins moins d’oscillations, potentiellement dû au fait que les hyperparamètres sont plus adaptés dans ce cas . On observe toujours un léger overfitting et une brusque augmentation de fonction de perte à la 40e époque précédée d’une oscillation de celle-ci à la 35e époque provoquant un arrêt prématuré de l'entraînement. Ici, la moyenne de la fonction de perte pour les données d’entraînement est ainsi de 0.937 et pour les données prédites de 0.940. Le modèle est donc légèrement plus performant que le précédent.

Enfin pour le modèle combinant des couches LSTM et des couches de convolutions, on observe une tendance décroissante de la fonction de perte des données d'entraînement et une tendance similaire pour les données de validation. Dans ce cas-ci, il y a un léger underfitting. Il y a quelques oscillations de nos fonctions de perte à la 15e et 30e époque en particulier potentiellement du fait du choix des hyperparamètres. Bien qu’on observe une légère perte d’apprentissage aux alentours de la 45e époque, dans ce cas ci, il n’y a aucune explosion de la fonction de perte. Au contraire, les fonctions de perte des données continuent à décroitre, montrant que le modèle continue à apprendre. Ici, les couches de convolution permettant de repérer certains motifs au sein des séquences ont potentiellement joué un rôle dans l’apprentissage du modèle bien qu’un ajustement du learning rate pourrait potentiellement aider à anticiper la perte d’apprentissage.

Par la suite, nous avons réalisé des prédictions des valeurs de réactivités au sein de séquences présentes dans le jeu test afin de déterminer quelles étaient les séquences les mieux prédites, et quelles étaient celles les moins bien prédites.

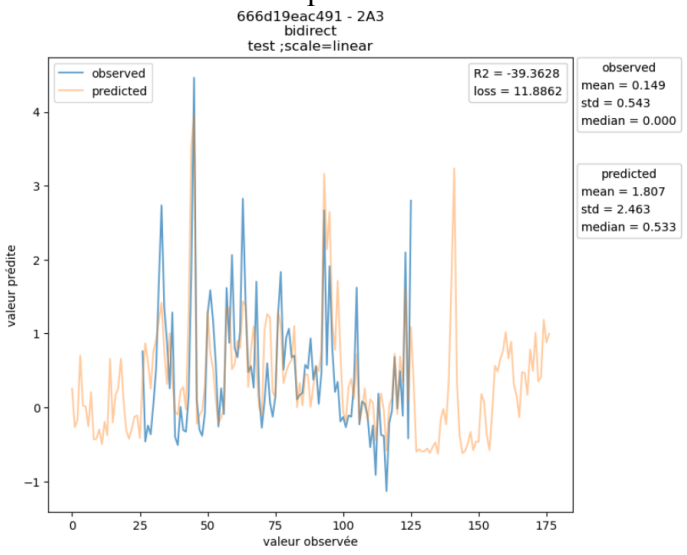


Figure 4: Comparaison des valeurs observées et prédites par le modèle Bi-LSTM pour une séquence (id : 666d19eac491).

Parmi les séquences mal prédites, on retrouve la séquence d'identifiant 666d19eac491 qui présente l'un des scores de corrélation les plus faibles entre les prédictions du modèle bidirectionnel et les valeurs observées. Cette séquence est mal prédite par les 3 modèles.

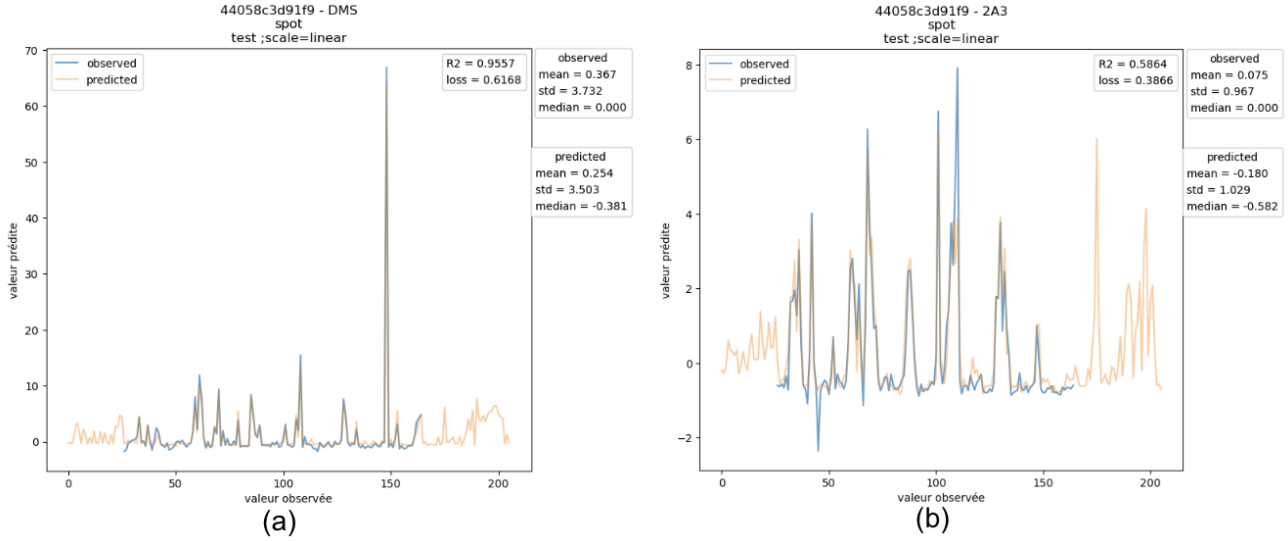
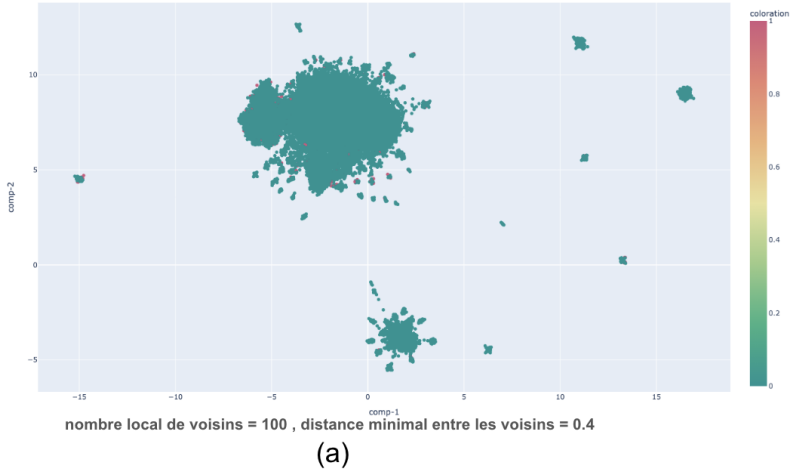


Figure 5: Comparaison des valeurs observées et prédites de la réactivité par le modèle LSTM Convolutionnel pour une séquence (id: 44058c3d91f9). **a)** Comparaison en DMS. **b)** Comparaison en 2A3.

La séquence 44058c3d91f9 est la mieux prédite au sein de l'expérience au DMS par le modèle LSTM Convolutionnel, avec la même séquence, le modèle Bidirectionnel réalise des prédictions tout aussi performantes pour le DMS (voir Annexe 4).



Modèles	$R^2 \geq 0.5$
Simple LSTM 2A3	3
Simple LSTM DMS	42
Bi -LSTM 2A3	19
Bi -LSTM DMS	610
LSTM + Conv 2A3	385
LSTM + Conv DMS	1312

Nombre de prédictions associé à un R ≥ 0.5 pour chaque modèles

(b)

Figure 6: a) Visualisation UMAP des séquences du jeu de données test, coloration des séquences prédites associé à un R^2 supérieure à 0.5 (rose) pour le modèle LSTM + ConV. **b)** Nombre de séquences prédites avec un R^2 supérieure à 0.5 pour les modèles : simple LSTM, bi-LSTM et LSTM+ConV en fonction du type de réactif (DMS ou 2A3)

Afin d'évaluer les performances de prédiction des modèles simples, nous avons compté le nombre de séquences prédites associées à un coefficient de détermination (R2) supérieur ou égal à 0.5 pour chaque modèle (voir la figure 6b). Globalement, on observe un nombre plus élevé de séquences correctement prédites par le modèle LSTM Convolutionnelle. En revanche, les prédictions sont moins bonnes avec le modèle LSTM

simple. Pour déterminer si les séquences correctement prédites présentent des similarités, nous avons représenté les séquences du jeu de données de test dans un espace à deux dimensions à l'aide de l'algorithme UMAP (voir la figure 6a). Sur la représentation, les meilleures prédictions du modèle LSTM + ConV ont été colorées, mais on constate qu'il y a peu de proximité entre ces séquences dans l'espace.

Résultats de l'optimisation des hyperparamètres du modèle

Afin d'explorer de nouvelles architectures de modèles, l'optimisation des hyperparamètres du modèle par l'approche bayésienne a montré une tendance à la réduction de la MSE au cours des 35 itération comme le montre la figure et nous a permis de concevoir une nouvelle architecture.

Figure 7: Evolution de la MSE en fonction des itérations pendant l'optimisation des hyperparamètres du modèle

Le modèle ayant donné la plus faible MSE était enregistré à la vingtième itération. Ce modèle comptait un total de 3 537 886 paramètres, parmi lesquels 3 536 206 étaient entraînables tandis que 1 680 ne l'étaient pas. La présence d'un grand nombre de paramètres suggère une certaine complexité nécessaire pour prédire les valeurs de réactivité. La structure de ce modèle est illustrée dans la figure, et l'inclusion de plusieurs couches LSTM bidirectionnelles suggère que ce type de couche est efficace dans la capture des caractéristiques de chaque séquence d'ARN, assurant ainsi une prédiction précise des valeurs de réactivité pour chaque nucléotide.

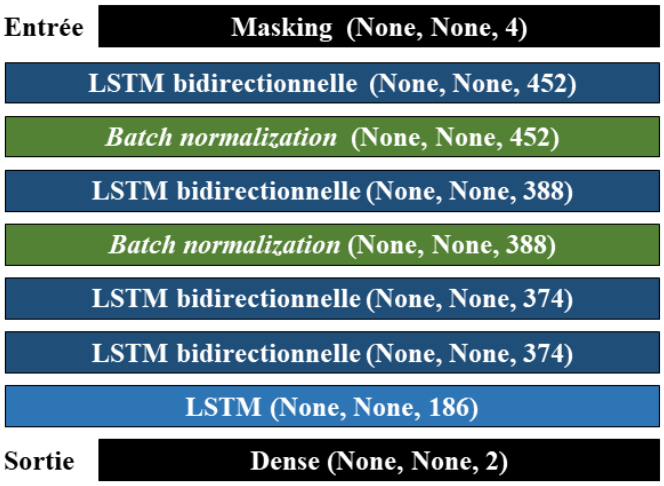


Figure 8: Architecture du meilleur modèle obtenue par l'exploration bayésienne

L'entraînement de ce modèle à l'aide d'un jeu de données suivi de son évaluation par des données de test a montré une réduction progressive de la MSE (Mean Squared Error), de la MAE (Mean Absolute Error) et de la RMSE (Root Mean Square Error) au fil des époques, avec une légère différence entre les valeurs du jeu de données d'entraînement et celles du jeu de données de validation. Cela suggère qu'il y a très peu de sur-apprentissage (overfitting). Cependant, il est important de noter que l'entraînement du modèle n'a pas été poursuivi jusqu'à l'activation de l'arrêt précoce (early stopping) en raison de contraintes de temps. La 24ème époque a enregistré la valeur la plus basse de MSE. Il est à noter que les métriques obtenues avec le jeu de données de test sont légèrement supérieures à celles obtenues avec le jeu de données d'entraînement, ce qui suggère que notre modèle parvient bien à généraliser (voir la figure).

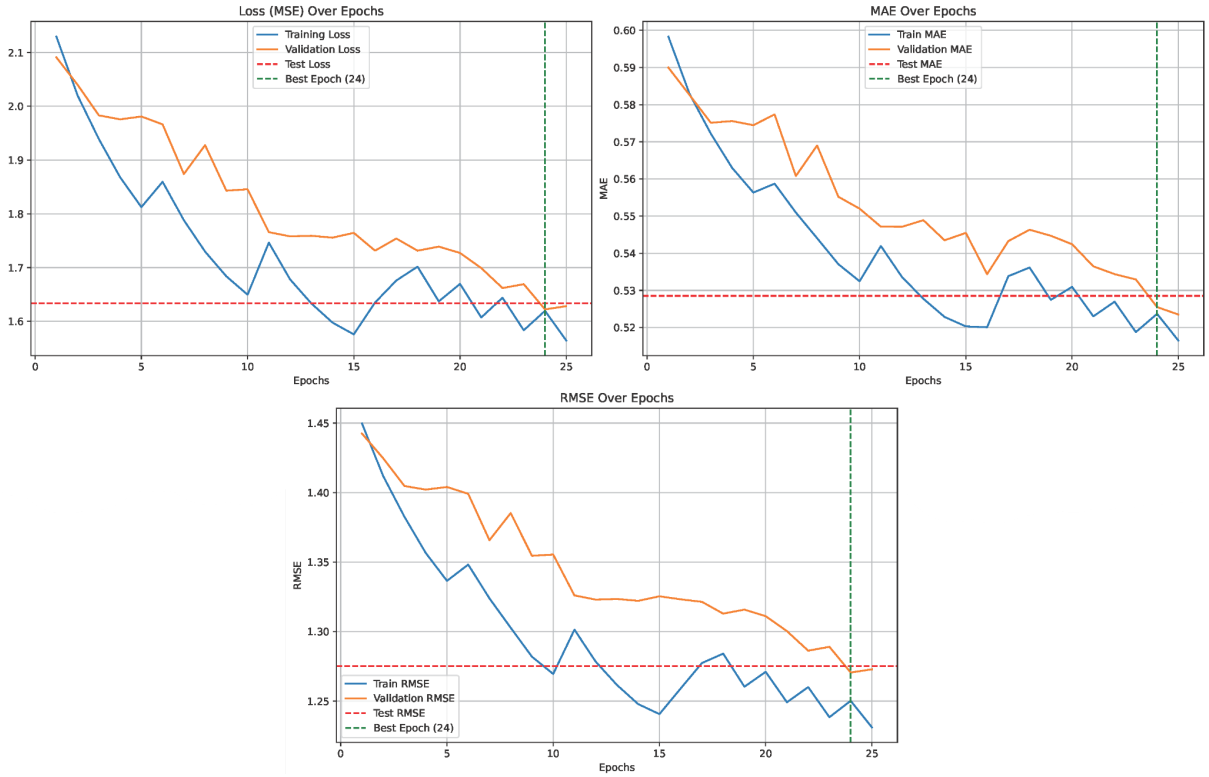


Figure 9: Evolution de la MSE, MAE et la RMSE en fonction des epochs lors de l'apprentissage du modèle

L'analyse de l'écart entre les valeurs réelles et les valeurs prédites a révélé que le modèle parvient à effectuer des prédictions correctes pour la majorité des valeurs de réactivité. En fait, l'erreur de prédiction était quasiment nulle pour la plupart des valeurs, comme le met en évidence la figure. Il convient de noter que les valeurs expérimentales de DMS semblent être prédites de manière plus précise que celles de l'expérience 2A3, ce qui suggère que la quantité de séquences disponibles pour l'expérience de DMS, par rapport à celle de 2A3, a eu un impact sur la qualité des prédictions.

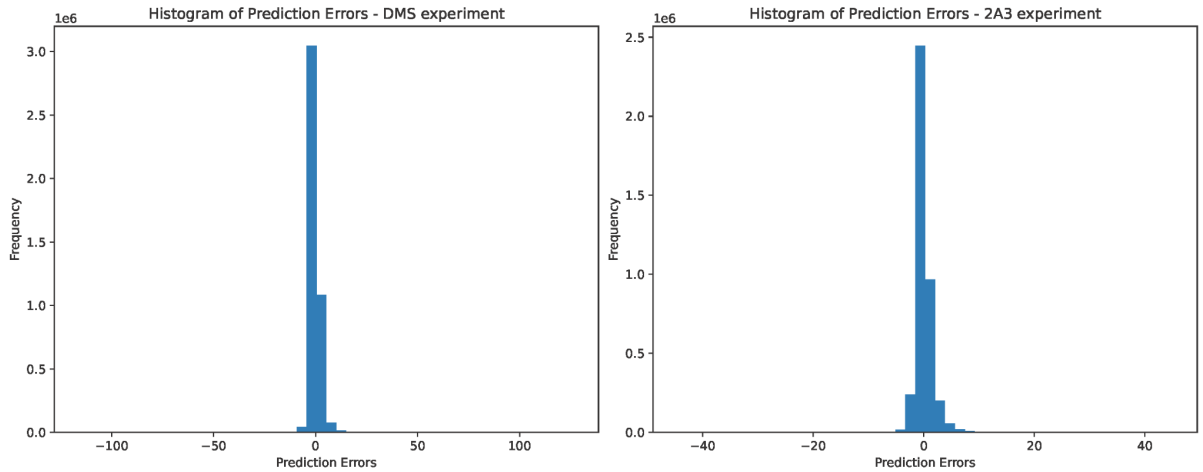


Figure 10: Fréquence des valeurs en fonction de la différence entre les valeurs réelles et les valeurs prédites pour l'expérience de DMS et l'expérience de 2A3

Dans le but de déterminer l'homogénéité de la capacité prédictive de notre modèle pour l'ensemble des valeurs de réactivité, nous avons effectué une comparaison entre les valeurs

réelles et celles prédites en les représentant sous forme d'un graphique en nuage de points. Les résultats mettent en évidence que notre modèle offre des performances équivalentes, qu'il s'agisse de valeurs proches de zéro ou de valeurs extrêmement positives. Il est à noter qu'une fois de plus, les valeurs issues de l'expérience DMS semblent être mieux prédites que celles provenant de l'expérience 2A3. Cependant, il convient de souligner que certaines des valeurs de DMS proches de zéro ont été prédites de manière très négative.

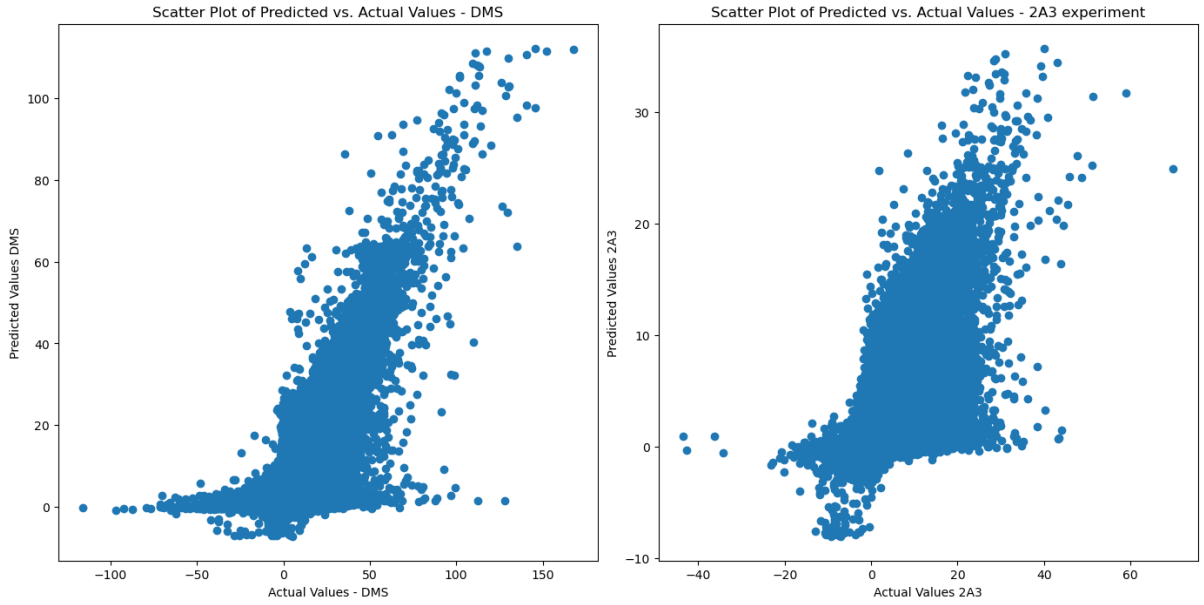


Figure 11: Distributions des valeurs réelles en fonction des valeurs prédites pour l'expérience de DMS et l'expérience de 2A3

4. Conclusion

L'optimisation des hyperparamètres du modèle grâce à une approche bayésienne a permis de trouver un modèle optimal, caractérisé par un grand nombre de paramètres. Ceci suggère qu'un nombre étendu de paramètres est nécessaire pour obtenir de bonnes performances en termes de prédiction de la réactivité dans le cadre des deux expériences. En outre, la prépondérance de couches bidirectionnelles LSTM par rapport aux couches LSTM simples (4 contre 1) suggère que l'utilisation de couches bidirectionnelles peut conférer un avantage significatif dans la capture des caractéristiques des séquences d'ARN, améliorant ainsi la qualité des prédictions.

Ce modèle se distingue par une faible erreur de prédiction, démontrant sa capacité à prédire avec précision à la fois des valeurs proches de zéro et des valeurs extrêmes. De plus, ses performances se sont révélées plus élevées dans le contexte de l'expérience de DMS par rapport à celle de 2A3. Cette disparité peut être attribuée au fait que l'expérience de DMS comportait légèrement plus de séquences d'ARN que l'expérience de 2A3.

Dans l'ensemble, nos résultats semblent prometteurs en ce qui concerne la prédiction des valeurs de réactivité dans le cadre des deux expériences, suggérant que ce modèle peut potentiellement être utilisé pour la prédiction de la structure des ARNs.

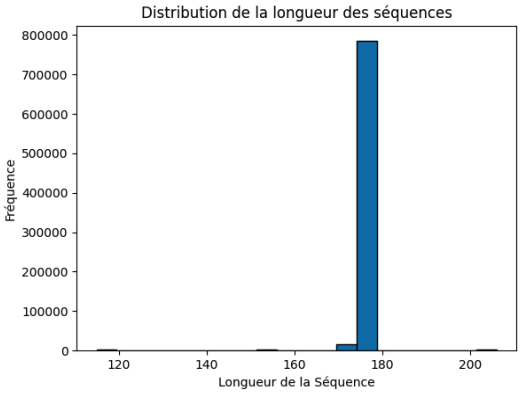
En perspectives, concernant les modèles plus simples testés sur des données plus simplifiées, il serait intéressant d'explorer davantage le modèle combinant couches LSTM

bidirectionnelles ou simples et couches de convolution car celui-ci a pu montrer quelques résultats concluants. Par ailleurs, les modèles actuels de prédiction de structure d'ARN les plus performants connus : SPOTRNA1 et SPOTRNA2 combinent ce genre d'architecture avec des réseaux plus complexes. Il serait donc intéressant de construire un modèle complexifié inspiré de ces derniers afin d'optimiser nos performances [7][8].

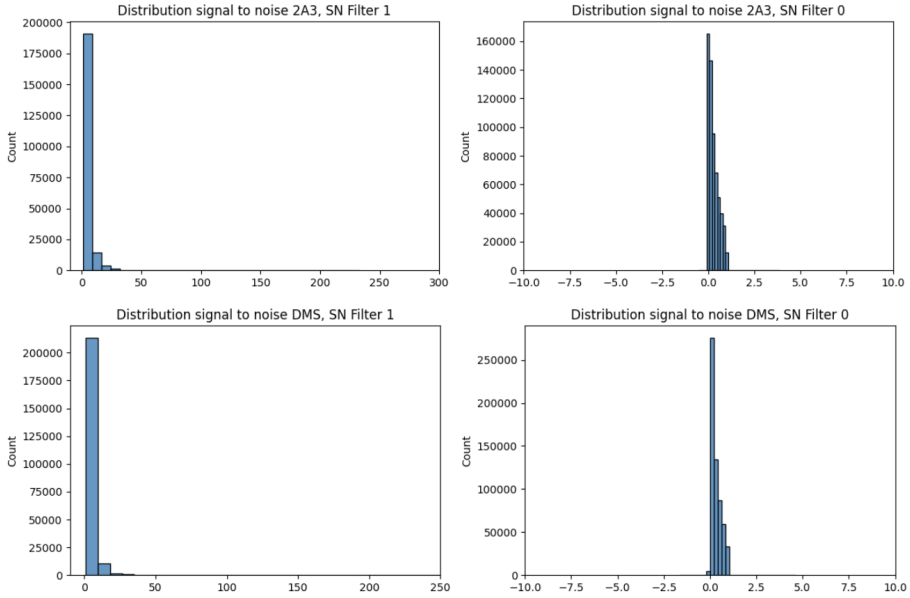
5. BIBLIOGRAPHIE

1. Zhang, J., Fei, Y., Sun, L. & Zhang, Q. C. Advances and opportunities in RNA structure experimental determination and computational modeling. *Nat. Methods* **19**, 1193–1207 (2022).
2. Halvorsen, M., Martin, J. S., Broadaway, S. & Laederach, A. Disease-Associated Mutations That Alter the RNA Structural Ensemble. *PLOS Genet.* **6**, e1001074 (2010).
3. Kim, S. H. *et al.* Three-Dimensional Tertiary Structure of Yeast Phenylalanine Transfer RNA. *Science* **185**, 435–440 (1974).
4. Ma, H., Jia, X., Zhang, K. & Su, Z. Cryo-EM advances in RNA structure determination. *Signal Transduct. Target. Ther.* **7**, 1–6 (2022).
5. Mitchell, D., Cotter, J., Saleem, I. & Mustoe, A. M. Mutation signature filtering enables high-fidelity RNA structure probing at all four nucleobases with DMS. *Nucleic Acids Res.* **51**, 8744–8757 (2023).
6. Marinus, T., Fessler, A. B., Ogle, C. A. & Incarnato, D. A novel SHAPE reagent enables the analysis of RNA structure in living cells with unprecedented accuracy. *Nucleic Acids Res.* **49**, e34–e34 (2021).
7. Justyna, M., Antczak, M. & Szachniuk, M. Machine learning for RNA 2D structure prediction benchmarked on experimental data. *Brief. Bioinform.* **24**, bbad153 (2023).
8. Singh, J. *et al.* Improved RNA secondary structure and tertiary base-pairing prediction using evolutionary profile, mutational coupling and two-dimensional transfer learning. *Bioinformatics* **37**, 2589–2600 (2021).

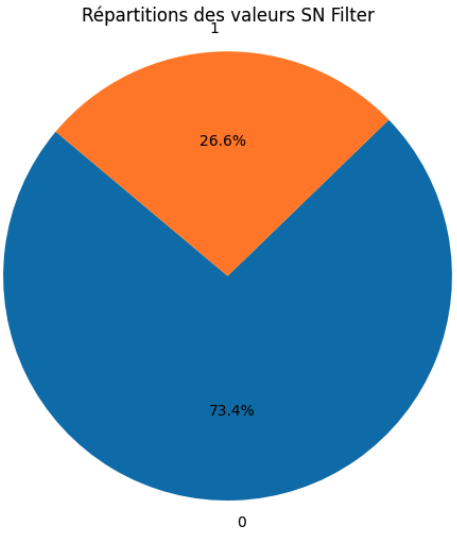
6. Annexe



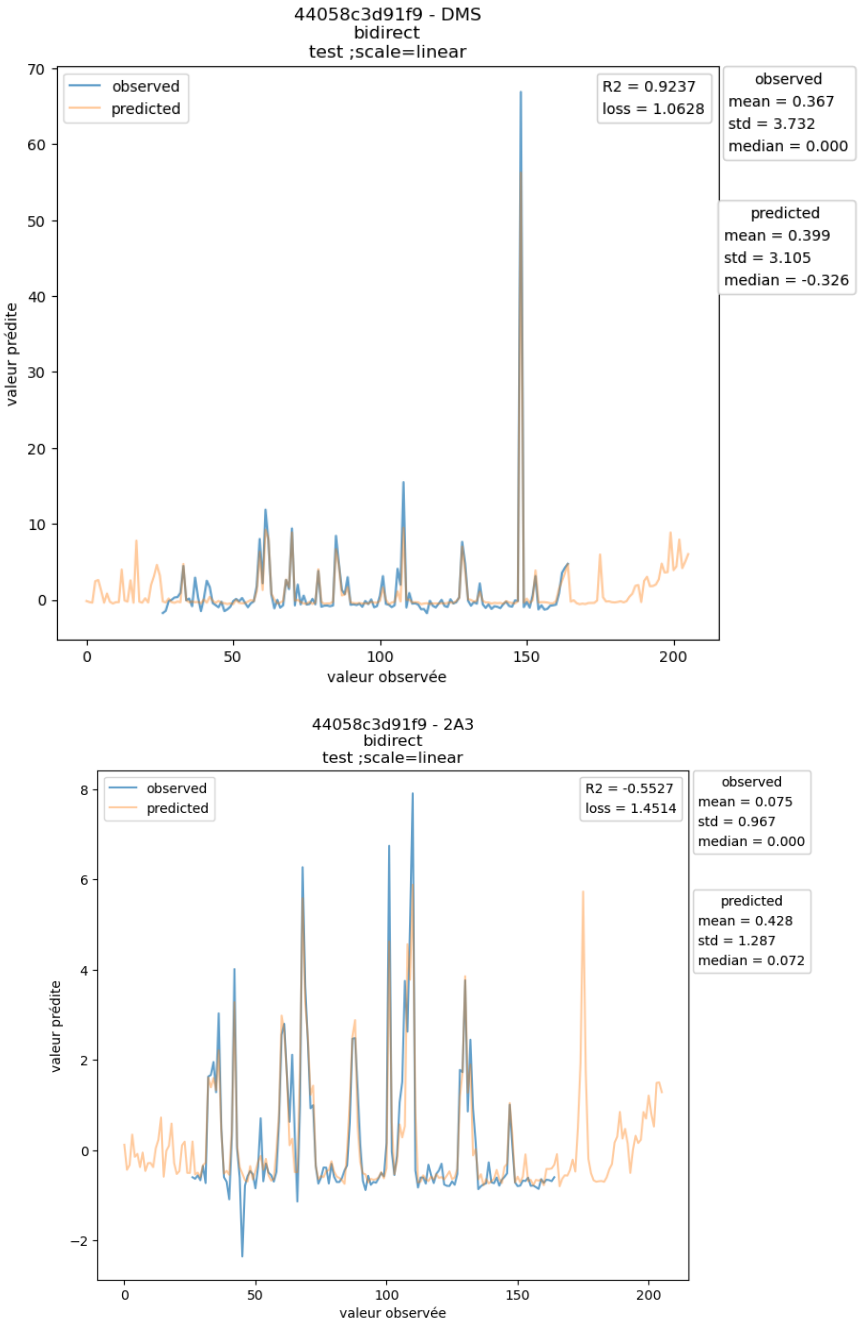
Annexe 1 : Distribution de la longueur des séquences dans le jeu de données



Annexe 2 Distribution des signal to noise selon le réactif et le SN filter



Annexe 3 : Répartition des valeurs de SN Filter (0 ou 1) dans le jeu de données.



Annexe 4 : Valeurs de réactivités prédites obtenus à l'aide du modèle Bidirectionnel sur la séquence d'identifiant 44058c3d91f9.