Université de Montréal

La reconnaissance automatique des brins complémentaires: leçons concernant les habiletés des algorithmes d'apprentissage automatique en repliement des acides ribonucléiques

par

Simon Chasles

Département d'informatique et de recherche opérationnelle
Faculté des arts et des sciences

Mémoire présenté en vue de l'obtention du grade de
Maître ès sciences (M.Sc.)
en informatique

Orientation intelligence artificielle

3 juillet 2023

Université de Montréal
Faculté des arts et des sciences

Ce mémoire intitulé

La reconnaissance automatique des brins complémentaires: leçons concernant les habiletés des algorithmes d'apprentissage automatique en repliement des acides ribonucléiques

présenté par

Simon Chasles

a été évalué par un jury composé des personnes suivantes :

Simon Lacoste-Julien

(président-rapporteur)

François Major

(directeur de recherche)

Sébastien Lemieux

(membre du jury)

Résumé

L'acide ribonucléique (ARN) est une molécule impliquée dans de nombreuses fonctions cellulaires comme la traduction génétique et la régulation de l'expression des gènes. Les récents succès des vaccins à ARN témoignent du rôle que ce dernier peut jouer dans le développement de traitements thérapeutiques. La connaissance de la fonction d'un ARN passe par sa séquence et sa structure lesquelles déterminent quels groupes chimiques (et de quelles manières ces groupes chimiques) peuvent interagir avec d'autres molécules. Or, les structures connues sont rares en raison du coût et de l'inefficacité des méthodes expérimentales comme la résonnance magnétique nucléaire et la cristallographie aux rayons X. Par conséquent, les méthodes calculatoires ne cessent d'être raffinées afin de déterminer adéquatement la structure d'un ARN à partir de sa séquence. Compte tenu de la croissance des jeux de données et des progrès incessants de l'apprentissage profond, de nombreuses architectures de réseaux neuronaux ont été proposées afin de résoudre le problème du repliement de l'ARN. Toutefois, les jeux de données actuels et la nature des mécanismes de repliement de l'ARN dressent des obstacles importants à l'application de l'apprentissage statistique en prédiction de structures d'ARN. Ce mémoire de maîtrise se veut une couverture des principaux défis inhérents à la résolution du problème du repliement de l'ARN par apprentissage automatique. On y formule une tâche fondamentale afin d'étudier le comportement d'une multitude d'algorithmes lorsque confrontés à divers contextes statistiques, le tout dans le but d'éviter le surapprentissage, problème dont souffre une trop grande proportion des méthodes publiées jusqu'à présent.

Mots clés: Intelligence artificielle, apprentissage automatique, réseau de neurones, classification binaire, surapprentissage, acide ribonucléique, repliement, prédiction de structure, nucléotide, complémentarité.

Abstract

Ribonucleic acid (RNA) is a molecule involved in many cellular functions like translation and regulation of gene expression. The recent success of RNA vaccines demonstrates the role RNA can play in the development of therapeutic treatments. The function of an RNA depends on its sequence and structure, which determine which chemical groups (and in what ways these chemical groups) can interact with other molecules. However, only a few RNA structures are known due to the high cost and low throughput of experimental methods such as nuclear magnetic resonance and X-ray crystallography. As a result, computational methods are constantly being refined to accurately determine the structure of an RNA from its sequence. Given the growth of datasets and the constant progress of deep learning, many neural network architectures have been proposed to solve the RNA folding problem. However, the nature of current datasets and RNA folding mechanisms hurdles the application of statistical learning to RNA structure prediction. Here, we cover the main challenges one can encounter when solving the RNA folding problem by machine learning. With an emphasis on overfitting, a problem that affects too many of the methods published so far, we formulate a fundamental RNA problem to study the behaviour of a variety of algorithms when confronted with various statistical contexts.

Keywords: Artificial intelligence, machine learning, neural network, binary classification, overfitting, ribonucleic acid, folding, structure prediction, nucleotide, complementarity.

Table des matières

Liste des tableaux

Liste des figures

Liste des sigles et des abréviations

ARN	Acide ribonucléique
ML	Apprentissage automatique, de l'anglais *machine learning*
DL	Apprentissage profond, de l'anglais *deep learning*
NN	Réseau de neurones, de l'anglais *neural network*
SVM	Machine à vecteurs de support, de l'anglais *support-vector machine*
MLP	Perceptron multicouche, de l'anglais *multilayer perceptron*
CNN	Réseau de neurones convolutif, de l'anglais *convolutional neural network*
BLSTM	de l'anglais *bidirectional long short-term memory*

Remerciements

Je tiens d'abord à remercier François Major pour la confiance, le soutien et les judicieux conseils. Il a su cultiver ma motivation et ma curiosité en encourageant ma participation à de nombreuses conférences. À l'époque de la pensée pressée, c'est une personne avec qui nous pouvons prendre le temps, tout simplement.

Je me dois de remercier les organismes responsables des bourses dont j'ai été récipiendaire au courant de ma maîtrise, soit le conseil de recherches en sciences naturelles et en génie du Canada, les fonds de recherche du Québec - nature et technologies, les études supérieures et postdoctorales de l'Université de Montréal, le département d'informatique et de recherche opérationnelle de l'Université de Montréal ainsi que la Banque Nationale du Canada.

Finalement, un grand merci à ma famille et à mes ami.es qui ont su me tenir compagnie et équilibrer ma vie durant ces deux dernières années.

Introduction

Ce mémoire de maîtrise émane d'une volonté de résoudre le problème du repliement de l'acide ribonucléique (ARN) par une méthode de calcul performante et efficace. Nous aimerions produire un algorithme capable de déterminer de manière précise suffisamment de caractéristiques structurelles d'une séquence d'ARN afin de permettre aux experts des sciences de la santé d'en dériver des traitements et/ou des thérapies. Ce projet s'inscrit donc dans un effort mondial pour comprendre comment la séquence d'un ARN influence sa structure, laquelle détermine sa fonction [**14**, **22**, **26**, **69**, **90**]

Ce projet donne donc suite aux multiples méthodes empiriques classiques comme celles utilisant la thermodynamique pour trouver une structure d'énergie libre minimum [**32**, **49**, **55**, **56**, **61**, **79**, **91**, **92**]. Bien que ces méthodes permettent de modéliser et de prédire une bonne partie des interactions entre les nucléotides, elles peinent à prédire les interactions à longue distance [**1**, **68**] (>300 nucléotides) ainsi que les interactions spéciales [**72**] (pseudonoeuds et paires multiples). Les méthodes statistiques modernes comme celles utilisant l'apprentissage automatique (ML) et l'apprentissage profond (DL) par réseaux de neurones (NN) pour prédire des structures en se basant sur les structures connues se veulent une réelle alternative aux méthodes empiriques [**90**]. Les récents succès d'alphaFold [**36**] en prédiction de structures de protéines laissent croire que le ML ait également un rôle à jouer en prédiction de structures d'ARN, d'autant plus que la taille et la qualité des jeux de données ne font que croître d'année en année [**8**, **15**, **73**, **76**].

Toutefois, ces jeux de données à l'apparence riche recèlent en fait des défauts importants aux yeux de l'informaticien, notamment aux niveaux de la diversité et de la quantité des données. D'abord, parmi les $\sim$ 100 000 séquences du jeu de données le plus largement volumineux [**15**], un grand nombre d'entre elles possèdent la même structure. C'est surtout le cas des

séquences provenant d'une même famille dont la structure a été déterminée par alignement de séquences et modèle de covariance [**5**, **19**, **35**, **39**, **40**, **60**]. Ce manque de diversité est frappant, puisque 93% des séquences de ce jeu de données appartiennent à seulement 4 familles d'ARN [**23**], alors que plus de 4000 familles existent [**37**]. Ensuite, uniquement 7 000 structures connues ont été vérifiées expérimentalement [**8**], la majorité des structures n'étant, en fait, que des prédictions par alignement de séquences. Bien que les méthodes expérimentales comme la cristallographie [**82**] et la résonance magnétique [**25**] puissent établir des liens entre les séquences et les structures, ces méthodes demeurent très coûteuses et inefficaces en temps [**1**, **12**, **90**].

Malgré de tels défis reliés aux données présentement disponibles, de nombreux algorithmes d'apprentissage automatique ont été proposés pour prédire la structure secondaire de l'ARN à partir de la séquence [**1**, **12**, **18**, **24**, **72**, **81**, **86**]. Les techniques utilisées sont variées et vont des grammaires probabilistes hors-contexte aux réseaux de neurones récurrents, en passant par les machines à vecteur de support (SVM). Il n'en demeure pas moins qu'une bonne partie d'entre eux ont présenté des signes de surapprentissage [**62**, **68**, **90**] en plus d'être limités en ce qui a trait à la généralisation entre les différentes familles d'ARN [**75**]. Cette habileté à généraliser entre les familles, ou de manière plus générale, à généraliser entre des molécules d'ARN structurellement distinctes est au coeur de l'apprentissage statistique puisque les structures connues ne représentent qu'une simple fraction de la totalité de l'espace conformationnel de l'ARN.

Il semble donc nécessaire d'en apprendre davantage sur les comportements qu'adoptent les algorithmes de ML et de DL lorsque confrontés aux défis inhérents au problème du repliement de l'ARN. Notamment, on s'attend, ou du moins, on aimerait que les méthodes d'apprentissage statistique aient la possibilité d'apprendre à partir de peu de couples séquence-structure, lesquels peuvent avoir été déterminés d'une multitude de manières (et donc présenter une diversité de représentations), le tout en généralisant à des exemples structurellement différents de ceux actuellement disponibles. Ce projet de recherche s'intéresse donc aux habiletés de certains types de NN et de certains algorithmes de ML sur une tâche fondamentale du domaine de l'ARN: déterminer si deux séquences sont totalement complémentaires. En définissant formellement une telle tâche, il nous est possible de générer des données synthétiques pour évaluer les performances des modèles dans

des contextes statistiques bien précis à la lumière des différents défis reliés aux données structurelles d'ARN. Tel est le sujet de la prochaine section.

Défis concernant les données relatives aux structures d'ARN

L'ARN est un type de molécule impliqué dans l'expression des gènes chez les être vivants. En particulier, les ARN messagers sont responsables d'extraire l'information génétique du noyau des cellules pour permettre la synthèse des protéines, lesquelles jouent divers rôles dans l'organisme. Au même titre que les protéines, les molécules d'ARN peuvent assumer différentes fonctions autre que la traduction génétique. Par exemple, les riboswitches et les micro-ARN sont impliqués dans les mécanismes de régulation de l'expression des gènes [**28**, **50**, **70**].

Or, chaque fonction remplie par une molécule d'ARN est principalement déterminée par sa forme tri-dimensionnelle ainsi que ses groupes chimiques capables d'interagir avec d'autres molécules. Afin de déterminer la fonction d'une molécule d'ARN, il est donc favorable de connaître sa séquence et sa structure. Toutefois, il existe un déséquilibre numérique entre la quantité de séquences connues ($\sim$2 millions [**37**]) et la quantité de structures expérimentalement déterminées ($\sim$7 000 [**8**]). Nous avons donc avantage à développer des méthodes calculatoires efficaces et précises pour estimer la structure d'une molécule d'ARN à partir de sa séquence.

Une molécule d'ARN peut être représentée symboliquement à l'aide d'une séquence de 4 caractères associés aux quatre nucléotides omniprésents dans l'ARN soit l'adénine (`A`), la cytosine (`C`), la guanine (`G`) et l'uracile (`U`). En dénotant l'alphabet de l'ARN par $\Sigma = \{\texttt{A}, \texttt{C}, \texttt{G}, \texttt{U}\}$, les séquences d'ARN peuvent être réduites à des mots $s \in \Sigma^*$ écrits dans la direction de 5' à 3'. Notons toutefois qu'un tel alphabet demeure modeste en comparaison aux nombreuses modifications chimiques que les nucléotides peuvent subir. Par exemple, la pseudouridine (Ψ) est significativement répandue dans l'ARN et bien qu'elle possède des qualités de liaisons différentes de l'uracile, elle est souvent représentée par `U` puisque les propriétés d'appariement de leurs faces *Watson-Crick* sont identiques [**10**, **46**].

Les propriétés d'appariement des nucléotides forment le fondement des mécanismes de repliement de l'ARN. D'après la nomenclature proposée par

Neocles Leontis and Éric Westhof [**43**, **44**], les nucléotides ont une géométrie triangulaire leur permettant de former des ponts hydrogène par l'entremise de trois faces, soit les faces *Watson-Crick*, *Hoogsteen* et *Sugar*. Les liens les plus répandus sont les liens reliant deux faces *Watson-Crick* de deux bases complémentaires ce qui forme les paires dites **canoniques** `A`−`U` et `C`=`G`.

Toutefois, des ponts hydrogène peuvent être formés entre toute paire de nucléotides autre que `A`−`U` et `C`=`G`, auquel cas une paire **non canonique** est formée. Additionnellement, les nucléotides `A` et `U` peuvent interagir par leurs faces *Watson-Crick* et *Hoogsteen* auquel cas la géométrie n'est pas équivalente à celle définissant les paires canoniques. Ces nuances méritent d'être prises en compte lors de la représentation des structures d'ARN, ce qui n'est pas toujours le cas.

La structure de l'ARN réfère à la forme tri-dimensionnelle qu'une molécule adopte lorsqu'elle se replie sur elle-même par le biais de ponts hydrogène entre ses différents nucléotides. Une telle structure peut être représentée à l'aide de coordonnées atomiques décrivant les positions relatives entre les nucléotides d'une molécule. Une telle méthode de représentation se veut grandement descriptive, mais peu compacte, surtout en raison de sa nature géométrique, c'est-à-dire, de l'importance des mesures/distances et orientations entre les nucléotides. Une alternative est de simplement décrire quels nucléotides forment des paires de bases, auquel cas la représentation est compacte, mais potentiellement ambiguë en raison de sa nature topologique, c'est-à-dire que les mesures/distances et orientations ne sont pas prises en compte.

Dans le cas de la description des paires de bases, la structure prend alors la forme d'un graphe non orienté. Étant donnée une séquence $s = s_1...s_L \in \Sigma^L$ pour $L \in \mathbb{N}$, la structure $G = (V, E)$ est le graphe où l'ensemble des sommets $V = \{s_i\}_{i=1}^{L}$ est l'ensemble des nucléotides et où l'ensemble des arêtes E est l'ensemble des paires de nucléotides présentant une interaction d'intérêt. Par exemple, $E = \{\{s_i, s_j\} |\ s_i \text{ et } s_j \text{ forment une paire canonique}\}$ est normalement utilisé pour décrire la structure secondaire d'un ARN. Toutefois, nous pouvons ajouter les paires non canoniques à l'ensemble E des paires d'intérêt. Une ambiguïté inhérente est alors introduite, puisqu'il existe une multitude de types de paires non canoniques, tandis que la représentation par ensemble d'arêtes ne s'intéresse qu'à l'existence de ponts

hydrogène entre deux nucléotides, indépendamment de la nature de ces ponts.

Additionnellement, les paires de bases peuvent se former de manière non imbriquée lorsque $\exists\ \{\{s_i, s_j\},\ \{s_k, s_l\}\} \subseteq E :\ i < k < j < l$. Nous sommes alors en présence d'un **pseudonoeud**. Aussi, puisque les nucléotides peuvent former des ponts hydrogène avec trois faces différentes, des **paires multiples** peuvent exister lorsqu'un nucléotide est apparié à deux nucléotides distincts. Les interactions non canoniques, les pseudonoeuds et les paires multiples sont désignées comme interactions **spéciales** puisqu'elles ne sont pas traditionnellement prises en compte pour décrire la structure secondaire de l'ARN. La description des interactions méritant d'être prédites par les méthodes de prédiction de structures est toujours sujette à débat et change d'une méthode à l'autre, mais surtout d'un jeu de données à l'autre.

Pour ce qui est des données présentement disponibles [**8**, **9**, **15**, **37**], les couples séquence-structure pouvant être utilisés pour de l'apprentissage supervisé diffèrent de par leur technique de représentation, mais aussi par les différentes méthodes utilisées pour déterminer la structure. Notamment, les méthodes expérimentales permettant d'enrichir la *Protein Data Bank* sont capables de saisir les interactions spéciales, surtout pour ce qui est des paires multiples et de la totalité des interactions non canoniques. À l'inverse, les méthodes de covariance et d'alignement de séquence couramment utilisées pour déterminer la structure d'une famille dans RFAM mettent l'emphase sur les paires canoniques, bien qu'il soit possible de déterminer certains pseudonoeuds ou paires non canoniques dans certains cas.

D'ailleurs, il existe plusieurs techniques capables de déterminer les structures expérimentalement [**25**, **82**] et les structures inférées par alignement de séquences et modèle de covariance proviennent de divers algorithmes [**31**, **54**, **60**, **91**]. Il en résulte que l'ensemble des données disponibles pour l'apprentissage supervisé du repliement de l'ARN forme un échantillon $\mathcal{D} = \{(s_i, G_i)\}_{i=1}^{N}$ de couples séquence-structure dont les éléments sont tirés d'une multitude de distributions.

De plus, les structures déterminées à ce jour concernent majoritairement une poignée de familles d'ARN munie d'une certaine distribution de tailles de séquences. Toutefois, les méthodes calculatoires de prédiction de structures d'ARN devraient être performantes pour toute molécule, indépendamment

de sa taille ou de la famille à laquelle elle appartient. Les algorithmes de ML devraient donc être capables de généraliser au sein des familles et des tailles de séquences mieux connues, mais aussi d'extrapoler à des familles et des tailles moins connues.

Aux défis de la variété de distributions/représentations et de la monotonie de familles/tailles s'ajoute le défi de l'abondance des exemples d'entraînement. Avec de tels obstacles à l'apprentissage, il est important de comprendre quel(s) impact(s) ont chacun d'entre eux sur les comportements et les aptitudes des algorithmes de ML et les NN en particulier. De tels modèles ont aussi leur lots de défis, ce sur quoi porte la prochaine section.

Défis concernant les modèles d'apprentissage automatique

Le problème de la prédiction de structures d'ARN est communément modélisé par la prédiction d'un graphe $G = \left(\{s_i\}_{i=1}^{L}, E\right)$ à partir d'une séquence $s_1...s_L$ où E représente l'ensemble des interactions d'intérêt (paires canoniques et/ou non canoniques avec potentiels pseudonoeuds et paires multiples). Ainsi, pour les NN, afin de manipuler des données numériques et tensorielles, il est commode d'utiliser la matrice d'adjacence comme données de sortie et un encodage quelconque des nucléotides comme données d'entrée. Le problème du repliement de l'ARN est alors réduit à une collection de problèmes de classification binaire: pour chaque paire de nucléotides, déterminer s'ils sont liés par des ponts hydrogène significatifs ou pas.

Il s'avère que ce problème soit assez ardu. Les méthodes de programmation dynamique comme celles capables de calculer une structure d'énergie libre minimum en témoignent de par leur complexité algorithmique. La prédiction de structures sans pseudonoeuds se fait en $\mathcal{O}(L^3)$ [**32**, **92**] et la prédiction de structures avec pseudonoeuds peut nécessiter une complexité en temps allant jusqu'en $\mathcal{O}(L^6)$ ou plus [**13**, **48**, **61**]. L'utilisation de NN est donc une alternative raisonnable compte tenu de leur grande expressivité et de leur nature vorace qui convient au paradoxe de Levinthal [**45**], lequel stipule que le temps requis pour explorer la totalité des conformations structurelles possibles est plus grand que le temps utilisé en pratique par les molécules pour adopter leur structure tri-dimensionnelle. En outre, une fois

un réseau entraîné, l'inférence de structure se fait relativement rapidement, surtout lorsqu'exécutée sur `GPU` [**12**, **24**].

Toutefois, le manque de résultats concluants malgré les efforts mondiaux témoigne de la difficulté intrinsèque au problème du repliement de l'ARN. Les mécanismes biologiques à l'oeuvre semblent difficile à saisir et il est donc naturel de bâtir des architectures de grande capacité (très expressives), c'est-à-dire, avec beaucoup de paramètres entraînables. Pourtant, en raison des défis inhérents aux données actuelles en structures d'ARN, les modèles à grande capacité ne sont peut-être pas les mieux adaptés à la résolution du problème du repliement de l'ARN.

Les architectures de NN proposées jusqu'à présent pour la prédiction de structures secondaires contiennent des couches linéaires, des couches de convolution, des couches d'attention et des couches récurrentes [**12**, **24**, **68**, **72**, **81**, **88**]. En effet, les molécules d'ARN ont une nature séquentielle ce qui suggère l'utilisation de réseaux récurrents comme les *bidirectional long short-term memory networks* (BLSTM) [**30**, **64**, **68**, **72**] ou les réseaux à base d'attention [**12**,**80**]. Également, l'ARN se replie de façon hiérarchique de par l'appariement de brins complémentaires d'abord (structure secondaire) suivi d'interactions à longues distances et de pseudonoeuds ensuite (structure tertiaire) [**7**, **78**], ce qui inspire l'utilisation de réseaux à base de couches linéaires et convolutives [**20**,**24**,**27**,**42**,**88**]. Toutefois, chacune de ces familles de NN présente des complications particulières quant aux mécanismes de repliement de l'ARN.

D'abord, en ce qui a trait aux réseaux utilisant l'attention en contexte d'encodeur-transformeur, l'utilisation de couches linéaires de type MLP nécessite un encodage de taille fixe ce qui est un facteur limitant à la représentation des séquences d'ARN en raison de la variabilité de leurs tailles. Le réseau à base de BLSTM se veut alors une alternative considérable compte tenu de sa capacité à traiter des séquences de tailles variables. Cependant, les calculs du BLSTM se font séquentiellement d'une extrémité à l'autre de la séquence, ce qui entre en conflit avec les mécanismes hiérarchiques du repliement de l'ARN: les sites d'initiations (formation d'une structure en épingle à cheveux) peuvent apparaître n'importe où. D'ailleurs, les BLSTM demeurent limités quant à leur capacité à saisir les interactions à longue distance [**89**], ce qui est indispensable à la prédiction de structures d'ARN.

D'un autre côté, nous pouvons faire appel aux réseaux convolutifs afin d'exploiter leur capacité à reconnaître des motifs complexes à partir de motifs plus simples, et ce, sur la totalité de la séquence simultanément. De plus, une fois l'architecture fixée, les réseaux totalement convolutifs ont l'avantage de pouvoir prendre en entrée des séquences de tailles variables [**24**, **47**, **63**], contrairement aux MLP dont les entrées doivent toujours être de même taille. Cependant, cela vient au coût d'être invariant au contexte positionnel dans la séquence ce qui complique le traitement des interactions à longue distance.

En pratique, il s'avère que de nombreux algorithmes de prédiction de structures d'ARN à base de ML présentent des signes de surapprentissage, surtout en contexte de généralisation inter-familiale [**62**, **68**, **75**, **90**]. En effet, les familles d'ARN présentent différents motifs tri-dimensionnels propres à chacune [**59**, **67**]. En assumant que plusieurs motifs demeurent toujours inconnus, nous avons avantage à ce que les algorithmes de ML saisissent les conditions de formation de ces motifs (afin d'extrapoler leurs connaissances pour proposer de nouveaux motifs) au lieu de simplement mémoriser les motifs existants.

Une telle capacité à généraliser concerne d'abord les mécanismes de régularisation des algorithmes de ML [**4**, **29**, **57**, **77**], mais aussi les techniques de représentation des séquences, notamment à ce qui a trait à l'encodage et à la dimensionalité [**17**, **21**, **65**]. De manière plus générale, les défis à l'application de l'apprentissage automatique en sciences biologiques demeurent très vastes en allant du manque d'interprétabilité au fléau de la dimensionalité, en passant par les importants coûts calculatoires [**66**, **83**].

À la lumière des multiples défis énoncés dans les dernières sections, le présent projet est né avec l'intention d'en apprendre davantage sur les comportements et les habiletés des principales familles de réseaux de neurones utilisées pour prédire les structures d'ARN. À titre de motivation, la question suivante nous occupait l'esprit: Comment pouvons-nous nous attendre à ce que les algorithmes de ML puissent apprendre à replier l'ARN, s'ils ne sont même pas capables de reconnaître des *stems* (structure d'hélice)? Nous avons donc défini la tâche fondamentale de la reconnaissance de brins complémentaires, ce qui nous a permis de mesurer les performances d'une multitude de méthodes de ML dans plusieurs contextes précis à l'aide de données synthétiques. Avec l'emphase sur le surapprentissage, les résultats

obtenus permettent d'éclairer la prise de décision quant à la capacité, l'architecture et l'encodage à utiliser en contextes de classification binaire en général.

Premier article.

Automatic recognition of complementary strands: Lessons regarding machine learning abilities in RNA folding

par

Simon Chasles[1] et François Major[1]

([1]) 2900 boul Édouard-Montpetit, Montréal QC H3T 1J4, Canada

Cet article a été soumis dans Frontiers in Genetics: RNA.

Les principales contributions de Simon Chasles à cet article sont présentées.

- Définition formelle du problème
- Conception et organisation du code
- Génération des figures
- Rédaction de l'article

François Major a proposé l'idée originale, orienté le projet et participé à la rédaction et à la soumission de l'article.

Résumé. La prédiction de structures secondaires d'ARN à partir de la séquence nécessite toujours d'importantes améliorations. L'utilisation de l'apprentissage automatique pour résoudre ce problème est devenue une pratique répandue. Toutefois, les algorithmes d'apprentissage automatique peuvent souffrir de surapprentissage, auquel cas la possibilité d'en apprendre davantage sur les mécanismes régissant le repliement de l'ARN se veut limitée. Il est naturel d'utiliser des modèles de haute capacité lorsqu'on tente de résoudre une tâche aussi complexe, mais un trop grand déficit en exemples d'entraînement risque de livrer de piètres habiletés à généraliser. Ici, nous rapportons la relation entre la capacité et la performance sur une tâche fondamentale connexe: déterminer si deux séquences sont totalement complémentaires. Notre analyse s'est concentrée sur l'influence qu'ont l'architecture, la capacité ains que le nombre et la nature des exemples d'entraînement sur la justesse de classification. Nous avons observé que les modèles de basse capacité sont mieux adaptés à l'apprentissage avec étiquettes mal classifiés, tandis que de grandes capacités facilitent la généralisation à des exemples structurellement distincts. Il s'avère que les réseaux de neurones ont du mal à saisir le concept fondamental de complémentarité de bases, en particulier lors d'extrapolation sur la taille des séquences. Étant donné une tâche plus complexe comme le repliement de l'ARN, il n'est pas surprenant que la pénurie d'exemples d'entraînement freine l'application des méthodes d'apprentissage automatique à ce domaine.
Mots clés : Intelligence artificielle, apprentissage automatique, réseau de neuronnes, classification binaire, surapprentissage, acide ribonucléique, repliement, prédiction de structure, nucléotide, complémentarité.

Abstract. Prediction of RNA secondary structure from single sequences still needs substantial improvements. The application of machine learning (ML) to this problem has become increasingly popular. However, ML algorithms are prone to overfitting, limiting the ability to learn more about the inherent mechanisms governing RNA folding. It is natural to use high-capacity models when solving such a difficult task, but poor generalization is expected when too few examples are available. Here, we report the relation between capacity and performance on a fundamental related problem: determining whether two sequences are fully complementary. Our analysis focused on the impact of model architecture and capacity as well as dataset size and nature on classification accuracy. We observed that low-capacity models are better suited for learning with mislabelled training examples, while large capacities improve the ability to generalize to structurally dissimilar data. It turns out that neural networks struggle to grasp the fundamental concept of base complementarity, especially in lengthwise extrapolation context. Given a more complex task like RNA folding, it comes as no surprise that the scarcity of usable examples hurdles the applicability of machine learning techniques to this field.
Keywords: Artificial intelligence, machine learning, neural network, binary classification, overfitting, ribonucleic acid, folding, structure prediction, nucleotide, complementarity.

Introduction

Identifying potential structural candidates for a single RNA sequence is a computationally demanding task. The Zuker-style dynamic programming approach to fold an RNA sequence of length L without pseudoknots requires time complexity in $\mathcal{O}(L^3)$ [**32**, **92**]. Algorithms that take into account pseudoknots are even more complex and have been reported to require significantly more computational power ranging from $\mathcal{O}(L^4)$ to $\mathcal{O}(L^6)$ [**13**, **61**], or higher [**48**].

Machine learning (ML) algorithms offer an alternative to traditional methods for identifying RNA structural candidates. In particular, neural networks can compute structures in an end-to-end fashion, allowing for quick inference in a single feedforward pass, especially when running on `GPU` [**12**, **24**]. However, the training phase can take several days, which can complicate software updates [**71**]. Regardless of the approach, predicting RNA structure requires significant computational resources due to the inherent complexity of RNA folding. As prediction methods must be at least as complex as the problems they aim to solve, it is important in the case of ML to avoid overfitting to this task.

To match the inherent complexity of RNA structure prediction, ML algorithms require high capacity, meaning they should be capable of learning a wide variety of mathematical functions [**27**]. Actually, because statistical learning relies heavily on training data, ML algorithms require high finite-sample expressivity to learn effectively [**87**]. However, high expressivity can lead to overfitting if the neural networks perform well on the training data without being able to generalize to structurally dissimilar testing data [**42**]. Therefore, it is crucial to balance expressivity with generalization to ensure accurate predictions on unseen data.

Several ML algorithms have been developed for RNA secondary structure prediction in recent years [**12**, **24**, **72**, **81**, **86**]. However, many of these algorithms are suspected of overfitting [**62**, **68**, **90**] and have limited ability to generalize across RNA families [**75**]. Generalization is crucial for accurate RNA structure prediction since known RNA structures only represent a small fraction of the entire RNA structure space. Prediction algorithms must be able to accurately predict structures for molecules that are similar and dissimilar to known structures. Therefore, it is important to develop

ML algorithms with stronger generalization properties to accurately predict RNA structures across a wider range of sequences and structures.

The aim of this study is to explore the performance and behavior of ML algorithms on a fundamental RNA-related task: determining if two RNA strands are fully complementary. Specifically, we examined the behavior of four families of neural networks with a focus on overfitting. We tackled three major challenges encountered when applying ML to RNA folding: (1) learning with mislabelled training examples (mislabels), (2) generalizing to structurally dissimilar data, and (3) training with limited examples [**8**, **15**, **23**, **62**].

Our results indicate that low-capacity models are more effective for learning with mislabels, as they have the ability to ignore them. Conversely, high-capacity models demonstrate better generalization performance in length-wise extrapolation context. On top of that, learning with few examples poses challenges for both low and high-capacity models, highlighting the importance of problem representation and architecture choice.

Materials and methods

Learning task

Given the RNA alphabet $\Sigma = \{\texttt{A}, \texttt{C}, \texttt{G}, \texttt{U}\}$, complementary strands are those in which each nucleotide on one strand pairs with its Watson-Crick partner on the other strand (`A` with `U` and `C` with `G`). For example, the RNA strand 5'-`AGUCAG`-3' is complementary to 5'-`CUGACU`-3'. We defined the task of automatic recognition of complementary strands as a binary classification problem that involves comparing and determining whether pairs of RNA strands of the same length are complementary or not. The target is `True` if the strands are fully complementary and `False` otherwise.

To have a fixed-size noisy input and simulate the structure of a hairpin loop, we restricted the maximum length of an RNA strand to 10 nucleotides and inserted an apical loop of 4 random nucleotides between each pair of RNA strands. During the experiments, the lengths of the apical loops varied from 3 to 7 nucleotides, but no significative impact was observed based on apical loop length, so we used the results computed with tetraloops as representatives of our key findings. More specifically, given strands s,

$\bar{s} \in \Sigma^L$ with $L \leq 10$ and $a \in \Sigma^4$, we represented the input as the sequence $0^{10-L}sa\bar{s}0^{10-L}$, where 0 denotes zero-padding.

The target of the classification is $t = 1$ if for all $i \in \{1, ..., L\}$, s_i is paired with its Watson-Crick complement on the other strand, that is, if $s_i = c(\bar{s}_{L+1-i})$, where $c : \Sigma \to \Sigma$ is the Watson-Crick complementarity function defined as $c(\texttt{A}) = \texttt{U}$, $c(\texttt{C}) = \texttt{G}$, $c(\texttt{G}) = \texttt{C}$, and $c(\texttt{U}) = \texttt{A}$. Otherwise, the target is $t = 0$.

Each nucleotide was represented by a vector of size 4, and a word embedding layer was used to update these vectors during training. Therefore, the inputs are fixed-size real-valued matrices $x \in \mathbb{R}^{24\times4}$ and the outputs are real values $y \in (0, 1)$. The output can be interpreted as the probability of the RNA sequence being a positive example, which we also refer to as the positivity score.

The loss function used to train the models is the binary cross-entropy with adjustments made to account for varying ratios of positive and negative examples. Specifically, for a training dataset $\mathcal{D} = \left\{\left(x^{(i)}, t^{(i)}\right)\right\}_{i=1}^{N}$ with positive example ratio $\alpha = \frac{1}{N}\sum_{i=1}^{N} t^{(i)}$, the loss value for a model prediction $y^{(i)} = f\left(x^{(i)}\right)$ was computed by Eq. 1.1.

$$l(y^{(i)}, t^{(i)}) = -\left[\left(\frac{1-\alpha}{\alpha}\right) t^{(i)} \log\left(y^{(i)}\right) + \left(1 - t^{(i)}\right) \log\left(1 - y^{(i)}\right)\right] \quad (1.1)$$

This correction ensured that the loss for positive examples was scaled up or down relative to the loss for negative examples, subject to the dataset's positive example ratio. For instance, if there were twice as many negative as positive examples in a dataset (i.e., $\alpha = 1/3$), the loss value for each positive example would be multiplied by a factor of 2, effectively balancing the contribution of positive and negative examples to the training loss. The parameter α could be controlled when creating synthetic datasets.

Artificial data

Our data generation process aimed to create diverse training and testing datasets that would enable us to evaluate our models in various scenarios. To achieve this, we introduced structural and statistical dissimilarities between the datasets, including differences in quality, sequence length and positivity rate. Mainly, we aimed at simulating the use of all known data (a

training set) to infer predictions on a portion of the unseen data of interest (a corresponding testing set).

To produce a pair of training and testing datasets, we ensured that there was no overlap between the training and testing examples by randomly partitioning the set of all 4^L sequences of length L into two sets. For each example, we concatenated a sequence $s \in \Sigma^L$ with a randomly generated apical loop a and a complementary sequence $\bar{s}$. Specifically, for positive examples, we set $\bar{s}$ to be the exact complement of s (denoted by s^*), while for negative examples, we chose $\bar{s}$ randomly in $\Sigma^L \setminus \{s^*\}$. We carefully controlled the ratio α of positive examples in the training sets and ensured that the testing sets had an equal number of positive and negative examples, thereby avoiding bias when measuring the performance on a test set.

To introduce structural dissimilarities, we varied the sequence length and the positive example ratio between the training and testing datasets. This allowed us to measure the extrapolation abilities of our models. Additionally, to control the quality of a training dataset, we introduced a proportion $\mu \in \left[0, \frac{1}{2}\right]$ of mislabelled examples, where a positive example could have label 0 with probability μ, and a negative example could have label 1 with probability μ. However, we ensured that all examples in the testing datasets were correctly labelled. By doing so, we tested the models' ability to ignore errors and avoid overfitting.

Overall, our data generation process produced realistic and diverse datasets, allowing us to evaluate our models under various conditions. Fully aware of the similarity between the task defined here and the prediction of binding sites of microRNAs, we are mostly interested in the abilities and behaviors of ML algorithms (and neural networks in particular) when trained and tested in various conditions, which we better control using artificial data.

Performance measure

To evaluate the performance of our models, we measured their classification accuracy on the testing datasets using a classification threshold $\theta \in (0,1)$ to distinguish between examples of class 0 and 1. Specifically, given a model $f(\cdot)$ and a dataset $\mathcal{D} = \{(x^{(i)}, t^{(i)})\}_{i=1}^N$, the accuracy of $f(\cdot)$ on $\mathcal{D}$ with threshold θ was calculated by Eq. 1.2.

$$A\left(f, \mathcal{D}, \theta\right) = \frac{1}{N} \sum_{i=1}^{N} \left(t^{(i)} \mathbb{1}\left\{ f\left(x^{(i)}\right) > \theta \right\} + \left(1 - t^{(i)}\right) \mathbb{1}\left\{ f\left(x^{(i)}\right) \leq \theta \right\} \right) \quad (1.2)$$

where $\mathbb{1}\{\cdot\}$ is the indicator function that equals 1 if the condition inside the bracket is true, and 0 otherwise.

While threshold θ could be optimized through methods such as SVM [**74**] or maximizing accuracy on the training dataset [**72**], we set $\theta = 1/2$ to study the behavior of our neural networks independently of any additional optimization steps. We considered the mean accuracy over 50 simulations, each of which generated datasets as described above, trained a model as described in the next section, and evaluated its performance using the aforementioned accuracy metric.

Architectures and training

The four tested models are relatively small representatives of four types of neural networks that have been recently used to predict RNA structures [**12**,**68**,**72**]. All models have only three layers, where the first layer is unique to each architecture and the last two layers have the form `linear - batchnorm - ReLU - dropout - linear-sigmoid` [**27**, **29**, **34**, **53**]. We used a dropout rate of 0.1 and a weight decay of 10^{-3} for all models, and the number of epochs was set to $\frac{8\times10^4}{N}$ to ensure that all models were trained for the same number of iterations, regardless of the number of training examples N. We used the `Adam` optimizer with a learning rate of 10^{-3} and default parameters in `PyTorch` [**38**] with a batch size of 256.

The four tested models and their capacity control are summarized in Fig. 1.1. The multi-layer perceptron model (**MLP**) [**27**] has a first layer of the form `linear - batchnorm - ReLU - dropout`, and its capacity is controlled by the number H_1 of hidden units in the linear module. For the multi-head self-attention model (**Att**), the first layer uses a skip connection of the form $h_1 = \texttt{conv}(x + \texttt{positional encoding})$ and $h_2 = \texttt{batchnorm}(h_1 + \texttt{dropout}(\texttt{multi-head self-attention}(h_1)))$, where the multi-head attention and positional encoding are described by Vaswani and co-workers [**80**]. The capacity of this layer is controlled by the number H_2 of heads in the multi-head attention, so the 1D-convolution uses a kernel of size 1 to project the 4-dimensional embedding into a H_2-dimensional input for the attention module. For the long-short term memory model (**LSTM**) [**30**,**64**],

the first layer is a one-layer bidirectional LSTM without dropout, and its capacity is controlled by the number H_3 of hidden units in the LSTM module. Finally, for the convolutional neural network (**CNN**) [**20**,**42**], the first layer has the form `outer concatenation - 3x3conv - batchnorm - ReLU - dropout` with a stride and padding of size 1. The outer concatenation operation takes the 24×4 input matrix and returns a $24 \times 24 \times 8$ tensor, where the 8-dimensional vector at position i, j is the concatenation of the 4-dimensional encodings at positions i and j in the initial matrix. The capacity of the CNN is controlled by the number H_4 of feature maps produced by the convolution module.

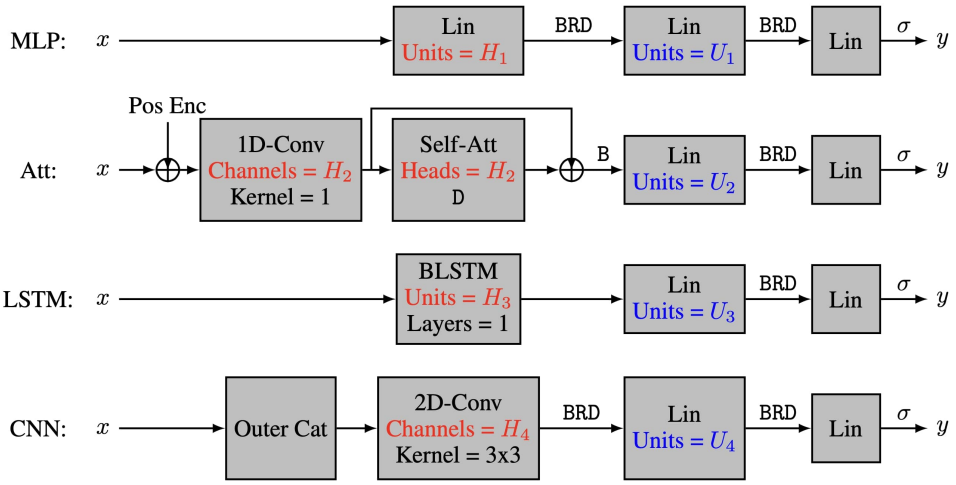


Fig 1.1. The four neural network architectures. The 4 tested neural network architectures take nucleotide encodings as input and output the positivity score. All models have three layers, with the first layer being a characteristic layer, and the last two layers having the form Lin-`B`-`R`-`D`-Lin-σ, where Lin refers to a linear layer, Conv refers to a convolutional layer, Self-Att refers to a multi-head self-attention layer, BLSTM refers to a bidirectional long-short term memory layer, Pos Enc refers to positional encodings, and letters σ, `B`, `R` and `D` refer respectively to sigmoid activation, batch normalization, ReLU activation and dropout regularization. The capacity of the models is controlled by hyperparameters H_i and U_i.

To further manipulate the capacity of the models, we also varied the number U_i of hidden units in the second layer of our four architectures. Our main interest lies in the number C of parameters in each model, and more specifically in $\log_{10} C$. The actual number of heads, hidden units, and feature maps for each value of C is provided in Table 1.1. As discussed earlier, we trained 50 instances of each model on 50 different training

datasets and reported the corresponding mean test accuracy. Our goal was to investigate how the test accuracy is affected by N and C for different values of L, μ and α.

Table 1.1. Hyper-parameters used to control the capacity of each model.

$\log_{10} C$	≈ 2.70	≈ 3.00	≈ 3.50	≈ 4.15	≈ 4.75	≈ 5.50
MLP (H_1, U_1)	(5, 5)	(10, 10)	(30, 30)	(80, 80)	(200, 200)	(500, 500)
Att (H_2, U_2)	(2, 9)	(4, 12)	(6, 24)	(12, 60)	(24, 120)	(80, 180)
LSTM (H_3, U_3)	(2, 5)	(3, 8)	(4, 18)	(10, 35)	(16, 78)	(48, 145)
CNN (H_4, U_4)	(1, 1)	(1, 2)	(2, 4)	(4, 9)	(8, 16)	(20, 36)

H_i refers to the capacity of the characteristic layer and U_i indicates the number of units in the second to last linear layer. The quantity $\log_{10} C$ gives an order of magnitude for the number C of trainable parameters.

Results and discussion

Learning with mislabels

The concept of mislabelling introduced in this section can be seen as a form of double standard. Our goal was to complicate the learning process deliberately by labelling some positive examples as class 0 and some negative examples as class 1. Thus, the term mislabelling is used because a ground truth classification is defined based on sequence complementarity. However, mislabelling on its own is not necessarily a problem for the learning process. A model $f(\cdot)$ that generalizes well on a completely mislabelled dataset could perform equally well on a correctly labelled dataset sampled from the same distribution by simply outputting $1 - f(\cdot)$. Therefore, the learning process is complicated when a proportion $\mu \in (0, 1)$ of the training dataset is mislabelled, creating a double standard as different pairs of complementary strands are labelled differently.

Moreover, since the labels can be interchanged without loss of generality, we focused on mislabelling probabilities $\mu \in \left(0, \frac{1}{2}\right)$. The special case where $\mu = \frac{1}{2}$ behaves as if the labels were randomly assigned [**87**]. In this study, we used a mislabelling probability of $\mu = 0.2$, which we find to be a good representative of the key findings presented in this section.

In comparison to the actual RNA structure prediction task, learning with mislabels can be seen as analogous to learning on a dataset that contains structures with non-canonical base pairs as well as structures

within which non-canonical base pairs are ignored, even though such pairs exist and can be identified. Similarly, the same concept applies when training on a dataset aimed at predicting pseudoknots and triplets when these kinds of interactions are only reported for some structures but not for others. Without defining the structural elements that need to be predicted, situations of double standard can arise when certain types of base pairs can be labelled as present (positive) as well as absent (negative) in a single dataset.

We were interested in investigating how the number of parameters C and the number of training examples N affect the test accuracy of the models for the automatic recognition of complementary strands. In particular, we focused on fixed sequence length $L = 8$ and mislabelling probability $\mu = 0.2$. The results for the MLP and Att models are presented in Fig. 1.2, where heatmaps depict the performance of the models for different values of C and N. Shades of red depict train accuracies; blue testing. See Fig. S1 for the equivalent LSTM and CNN results. As expected, the accuracy increases with N, and overfitting behavior is observable for $\log_{10} C \geq 4.15$.

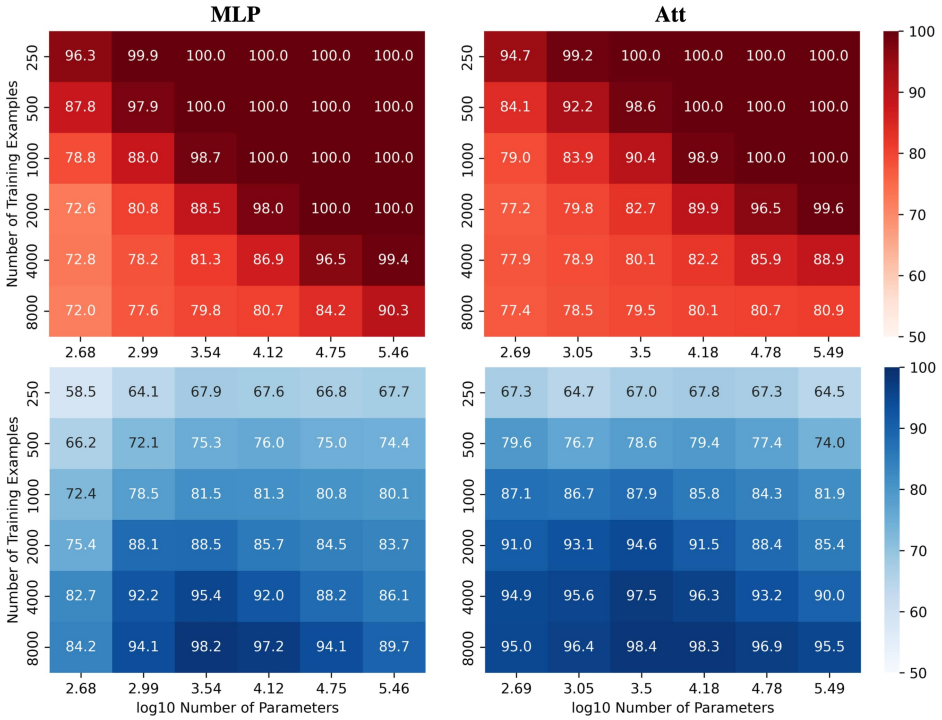


Fig 1.2. Performance of MLP and Att models when learning with mislabels. Train (red) and test (blue) mean accuracies over 50 simulations reported for MLP and Att models. Sequence length and mislabelling probability are fixed to $(L, \mu) = (8, 0.2)$.

Remarkably, these results demonstrate the models' ability to handle mislabels, as they achieve a test accuracy of over 80% even when 20% of the training set is mislabelled. Moreover, the models can achieve near-perfect test accuracies as long as they are trained on a sufficiently large dataset. This finding suggests that models with relatively low capacity can still learn effectively when trained on low-quality high-quantity datasets.

Indeed, it appears that low-capacity models ($\log_{10} C \approx 3.5$) are more likely to achieve test accuracies above $1 - \mu$ than high-capacity models ($\log_{10} C \approx 5.5$). This is likely due to the fact that mislabels introduce irregularities into the sample space that are difficult for low-capacity models to account for. Low-capacity models tend to compute smoother functions than high-capacity models. They are thus less capable of capturing the intricate patterns that arise from mislabelling, especially in large training datasets. They have however enough capacity to estimate the function of interest, yielding test accuracies near 100% and train accuracies around $1 - \mu$.

On the other hand, high-capacity models can account for the irregularities introduced by the mislabels for larger training datasets, delaying the cross-over point of train and test accuracies to larger N. They thus require a broader view of the sample space to attain high test accuracies, coupled with sufficient regularization to prevent overfitting. The graphs in Fig. 1.3 illustrate these behaviors. As in Fig. 1.2, shades of red and blue depict train and test accuracies respectively.

However, it is important to note that these reported accuracies may be too optimistic because the models were trained and tested on sequences of the same length with the same ratios of positive and negative examples. Although no training sequence was repeated in the testing set, this setup only evaluates a model's ability to generalize within structurally similar data, but not to extrapolate to structurally dissimilar data. While this section highlights the effectiveness of low-capacity models in learning with mislabels, the following section presents a contrast as high-capacity models appear to be more suitable for generalizing to structurally dissimilar data.

Generalizing to structurally dissimilar data

In the context of the automatic recognition of complementary strands, the ability to generalize to structurally dissimilar data refers to the ability

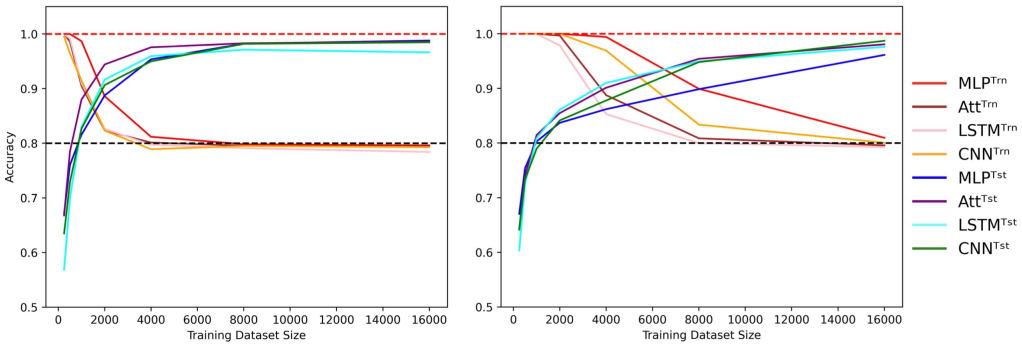


Fig 1.3. Cross-over behavior when learning with mislabels. Influence of the training dataset size over train (Trn) and test (Tst) accuracies for low-capacity models (left) and high-capacity models (right). Sequence length is fixed to $L = 8$ and mislabelling probability is fixed to $\mu = 0.2$, with low capacity meaning $\log_{10} C \approx 3.5$ and high capacity meaning $\log_{10} C \approx 5.5$. Dotted lines indicate the 100% and 80% accuracy marks since $\mu = 20\%$.

of a model to make accurate predictions over a subspace of the sample space that it has not or poorly seen during training. This means that models could be trained and tested on datasets containing different sequence lengths and positivity rates to evaluate their understanding of sequence complementarity. For example, a model can be trained on sequences of length 5 or 6 and then tested on sequences of length 8, or trained on datasets with few or a lot of positive examples and then tested on balanced datasets. This approach allows us to investigate how well a model can generalize and extrapolate its understanding of the problem. In light of current discussions regarding extrapolation in ML conditions [**2**, **6**], the concept of extrapolation is used here to convey how sequences of length 8 can't belong to the convex hull of a set of sequences of length 6 with zero-padding.

In contrast, ML algorithms used for predicting actual RNA structure face different challenges. For instance, hardware limitations may restrict the maximum length of training sequences, but the model is still expected to predict the structure of longer sequences. The presence of non-canonical base pairs, pseudoknots, and base triples can increase the number of base pairs per nucleotide, making it difficult for models trained on sequences with a lower base pair density to generalize to more sophisticated structures. Moreover, different RNA families may have varying frequencies of certain structural motifs [**52**], further complicating the generalization of models to unseen families. Despite these challenges, the ultimate goal of predicting

RNA structure for all families remains the same, regardless of their structural similarity to previously known RNA structures.

To better visualize the impact of training example count N and the number of model parameters C on test accuracies for the automatic recognition of complementary strands, we report experiments on CNN and Att models. Specifically, the models were trained on correctly labelled sequences of length 6 and then tested on the full set of 4^8 sequences of length 8. We present heatmaps (Fig. 1.4) that illustrate the performance of the models as the number of parameters and training examples are varied. See Fig. S2 for the equivalent MLP and LSTM results. We observed that higher model capacity tends to yield better performance, regardless of the number of training examples.

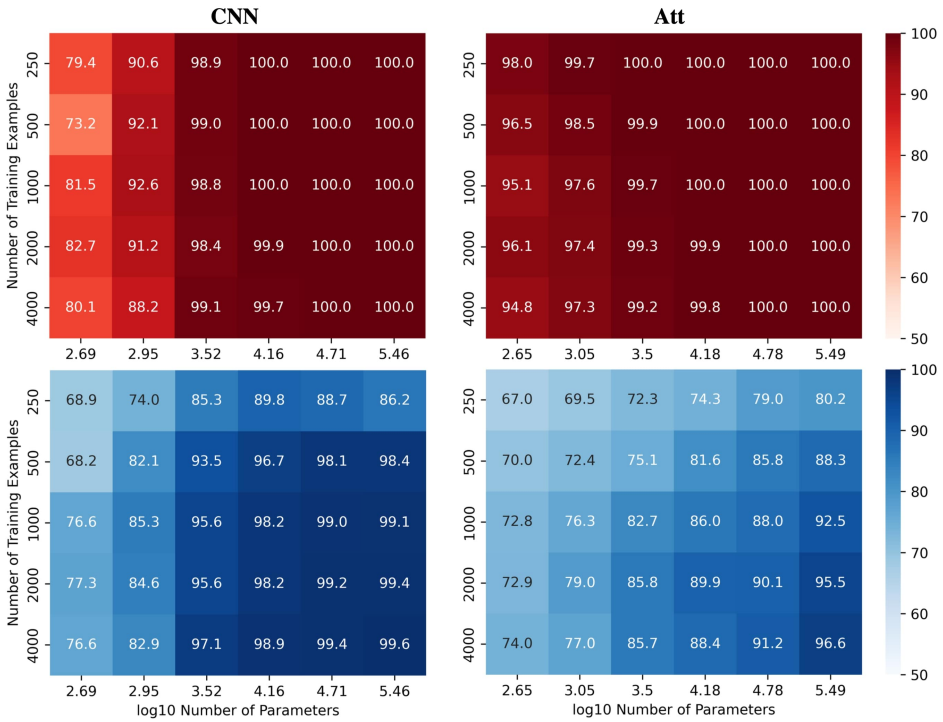


Fig 1.4. Performance of CNN and Att models in length-wise extrapolation context. Train (red) and test (blue) mean accuracies over 50 simulations reported for CNN and Att models. Models were trained on sequences of length 6 before being tested on sequences of length 8, without mislabelling in the training set.

Nevertheless, as we increase the capacity, signs of overfitting can still be observed. Notably, in the graphs presented in Fig. 1.5, we observe signs of overfitting from the MLP model when $\log_{10} C \geq 4$, for both $(L, N) = (5, 500)$ on the left and $(L, N) = (6, 2000)$ on the right.

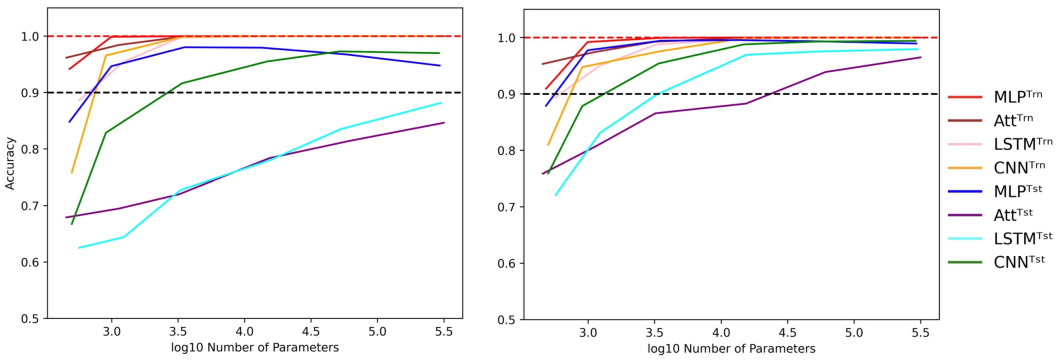


Fig 1.5. Influence of capacity in length-wise extrapolation context. Impact of the number of trainable parameters over train (Trn) and test (Tst) accuracies in length-wise extrapolation context. Results when training with 500 examples of length 5 is shown on the left and 2000 examples of length 6 on the right. Sequence length is set to 8 in the testing datasets and mislabelling probability is set to 0 in the training datasets. Dotted lines indicate the 100% and 90% accuracy marks to highlight acceptable test performance.

Interestingly, we observe that the MLP and CNN models appear to be better suited for extrapolation than the Att and LSTM models. The former can achieve test accuracies of over 90% with ease, while the latter struggle to do so. Additionally, the Att and LSTM models exhibit greater variability in their performances. For instance, the Att model of capacity $\log_{10} C \approx 4.15$ with $(L, N) = (6, 2000)$ has a mean test accuracy of 84.8% over 50 simulations. However, its best accuracy is over 99% and its worst accuracy is around 50%. In Fig. 1.5, the error bars have been omitted to better highlight the performance of the four models. Furthermore, as it is challenging to perform well on bigger sequences than those on which the models were trained, the reverse can be equally challenging. It is notably the case for the low-capacity LSTM which presents poorer generalization performances as the absolute difference between train and test sequence lengths gets bigger (see Fig. S4).

Despite the impressive statistical performance of the MLP model, it fails to capture the nuances that make two RNA sequences complementary. Specifically, when trained on 500 sequences of length 5, the MLP model tends to mistake a negative example with mismatches in the upper three base pairs for a positive example. The MLP model with $\log_{10} C \approx 3.5$ achieves a statistical accuracy of 98.1%, yet it incorrectly classifies the sequence `ACGUACGUGAAAACGUAGCA` as a positive example with a mean positivity score

of 0.9810 (Fig. 1.6). Interestingly, all other models exhibit a similar trend, as they assign a comparable (albeit slightly lower) mean positivity score to this negative sequence as they do to its corresponding positive sequence.

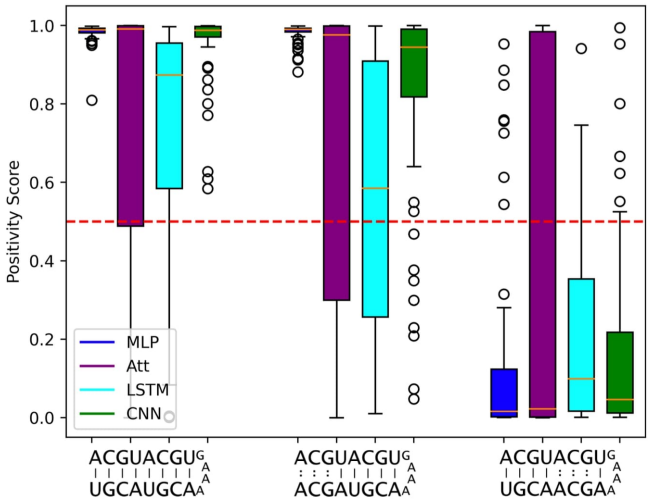


Fig 1.6. Limits to length-wise extrapolation with zero-padding and fixed-size inputs. Distribution of model outputs (positivity score) on specific sequences of length 8 when trained on 500 sequences of length 5. The first sequence is a positive example and the second and third ones are the same positive example where 3 mismatches have been introduced respectively at base pair positions 6 to 8 and 2 to 4. We use the best tested capacity for each model so $\log_{10} C \approx 3.5$ for the MLP and $\log_{10} C \approx 5.5$ for all other models. The classification threshold is represented by the red dotted line.

Furthermore, the negative example with mismatches in the lower five base pair positions is correctly classified most of the time by all models. This suggests that the use of zero-padding to have fixed-size intputs limits the length-wise extrapolation abilities of ML models to the nucleotide positions seen during training. The MLP model seems to be particularly affected by the zero-padding, while the LSTM tends to produce lower positivity scores for the negative sequence with mismatches in the upper three base pair positions, probably due to its sequential calculation.

It is worth noting that all results presented so far were obtained by training and testing on datasets with an equal number of positive and negative examples. However, we can manipulate the proportion α of positive examples in the training dataset while still testing on a balanced dataset. Despite the loss function correction that accounts for an equal number of positive and negative examples, when the concentration of positive examples

is too low or too high, all four models perform poorly (Fig. 1.7); even when classification threshold θ is equal to α.

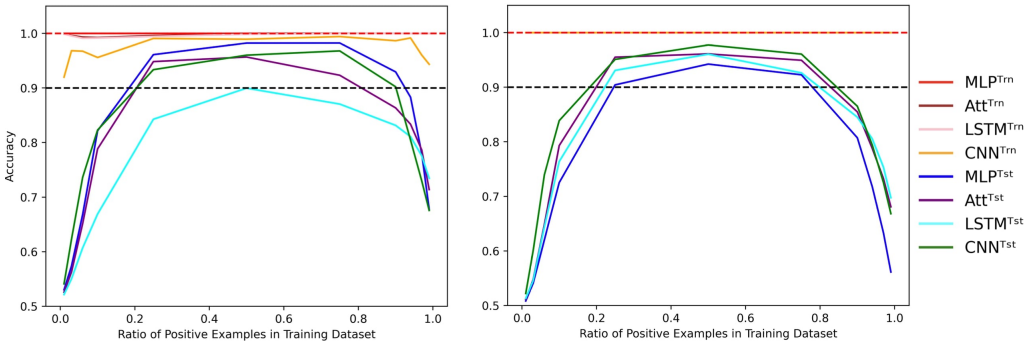


Fig 1.7. Evolution of performance in the extrapolation context of various positivity rates. Influence of the concentration of positive examples in the training set on the train (Trn) and test (Tst) accuracies for low-capacity models (left) and high-capacity models (right). Parameters were fixed to $(N, L, \mu) = (500, 6, 0)$. Dotted lines indicate the 100% and 90% accuracy marks to highlight acceptable test performance.

It is worth mentioning that in Fig. 1.7, the test accuracy curves are skewed to the left, indicating that the models perform better with too many positive examples than too few. This behavior can likely be due to the fact that the positive examples require complete Watson-Crick complementarity. To be fair to the models, a distinction could be made between negative examples with 0% Watson-Crick pairs and 90% Watson-Crick pairs. In this line of thought, we also formulated the problem as a regression task where we wanted to predict the concentration of Watson-Crick pairs within each example. Even so, the main conclusions are not affected by such formulation of the task, so we stick to the binary classification formulation since we are motivated by structured prediction problems that can be modeled using a collection of binary classification tasks.

If we aim to extrapolate accurately without mislabels, high-capacity models are favorable, which means that the training datasets must be sufficiently large to support this level of expressiveness. Moreover, when our training dataset contains mislabelling, the models require a substantial number of training examples before they can disregard the errors. In the next section, we will address the challenge of learning from a small training dataset while also consolidating the insights from the previous two sections on capacity.

Learning with few training examples

Here we focus on learning with small training datasets, but this at the same time provides us with an opportunity to explore how all our initial challenges interact with each other, both for neural networks and some classical ML methods. Thus, it is an important section that consolidates the insights gained from the previous sections.

We can note from last sections that learning with high capacity is advantageous for extrapolating without mislabels, while learning with low capacity is necessary to account for mislabels. However, when both challenges are combined, model behaviors become more complex.

To illustrate this complexity, we present heatmaps showing the behavior of the four tested architectures when trained with various quantities of sequences of length 6 with a mislabelling probability of 20% and a positivity ratio of 40% (Fig. 1.8), the later being an arbitrary value to further challenge the extrapolation abilities of the models. The test accuracies are reported over the whole set of sequences of length 8 with as many positive and negative examples. See Fig. S3 for the corresponding train accuracies. These heatmaps demonstrate that the ability to ignore errors is conserved even when extrapolating, provided enough training examples are supplied. However, the required capacity to maximize generalization performance varies among the models and can be influenced by the number of training examples. For instance, the MLP and CNN require low capacity to attain their best generalization performance, while the Att and LSTM require high capacity. Additionally, the CNN's best generalization performance requires high capacity with fewer training examples, but low capacity with more training examples.

However, achieving acceptable performances with test accuracy over 80% requires a substantial number of training examples ($N \geq 2000$) when extrapolating with mislabels. When only a few training examples are available ($N \leq 500$), test accuracies over 80% become challenging to achieve. Fig. 1.9A shows the influence of capacity on the accuracies when trained on 500 sequences of length 6 with 20% mislabelling and 40% positivity rate, tested on the whole balanced set of correctly labelled sequences of length 8. It appears that variations in capacity have a low impact on the model's performance, although a general slight improvement can be observed as capacity increases.

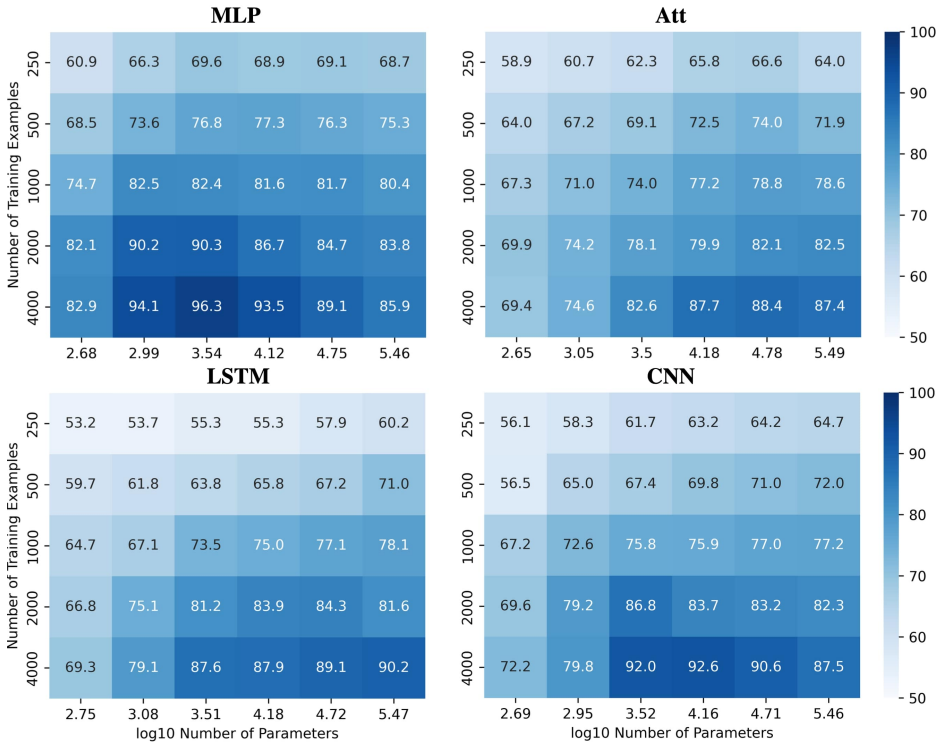


Fig 1.8. Performances for all models when learning with mislabels in extrapolation context. Test accuracies for all models when tested on sequences of length 8 after being trained on sequences of length 6 with 20% mislabelled training examples and 40% positivity rate.

In order to gain more insight into the behavior of the four models when dealing with few or many training examples, we compared their distributions of performances (Fig. 1.9B). To produce this figure, we used the capacities that maximize accuracy based on the heatmaps presented in Fig. 1.8. As a result, a wide range of capacities was needed for the models, with the MLP using $C \approx 3500$ and the CNN using $C \approx 15\ 000$ when trained on many examples. On the other hand, the Att and LSTM required much higher capacities, with the Att using $C \approx 60\ 000$ and the LSTM using $C \approx 300\ 000$ regardless of the number of training examples. Analyzing these distributions, we observe that the MLP model appears to be the most suitable for extrapolating with mislabels since its test accuracies have a higher mean and lesser variance than the other three models.

With such a fundamental binary classification task, the behaviors of some specific classical ML algorithms can put the results on neural networks into perspective. Focusing on the k-nearest neighbors (KNN), the support vector machine (SVM), the decision tree (tree) and the random forest

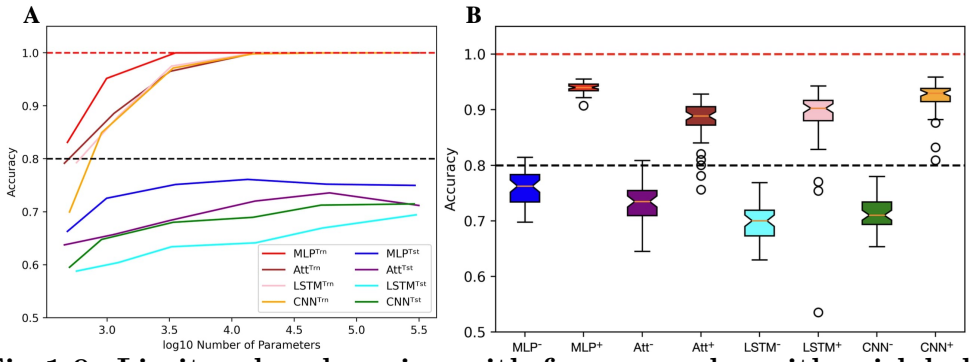


Fig 1.9. Limits when learning with few examples with mislabels in extrapolation context. (A) Influence of capacity on train (Trn) and test (tst) accuracies. The training sets are unbalanced and mislabelled as parameters (N, L, μ, α) are fixed to $(500, 6, 0.2, 0.4)$. The testing sets on the opposite are balanced and correctly labelled with parameters (N, L, μ, α) being fixed to $(4^8, 8, 0.0, 0.5)$. (B) With the same parameters, distributions of test accuracies over 50 simulations are reported, with models being trained on 500 examples (e.g. MLP$^-$) or 4000 examples (e.g. MLP$^+$). The capacity used for each situation is the best capacity achieved on the test sets with respect to the heatmaps in Fig. 1.8. Dotted lines indicate the 100% and 80% accuracy marks since $\mu = 20\%$.

(forest) algorithms, we measured the performances of such methods when learning on few examples with mislabels in a length-wise extrapolation context. All algorithms use default `sklearn` implementation except the KNN uses 35 neighbors, the tree uses a maximum depth of 12 decisions and the forest uses 400 classification trees. These parameters were determined by grid search maximizing the generalization performance for most training dataset sizes.

First of all, we measured the ability of these algorithms to account for mislabels in the training dataset (Fig. 1.10A). In comparison to the four tested neural networks, the method that behaves in the most similar way is the SVM, as the test accuracy tends to 100% while the train accuracy tends to $1 - \mu$ as $N \to \infty$. The decision tree behaves similarly, but its generalization performances are not as good. The KNN also performs poorly, but its train accuracies are stuck at 100%. The most surprising behavior is held by the random forest algorithm: Even tough it fits 100% of the training datasets, which include 20% of mislabelled examples, the test accuracies can simultaneously reach above 99%, which shows an ability to minimize both test and training risks [**4**, **57**]. The accuracy distributions presented in Fig. 1.10B allow to further visualize this diversity of behaviors

when algorithms are trained with 500 and 4000 examples of length 6 before being tested on the whole set of sequences of length 8. Note that despite the relatively good performances of the SVM and random forest algorithms, they also suffer from the same misclassification problem presented in Fig. 1.6.

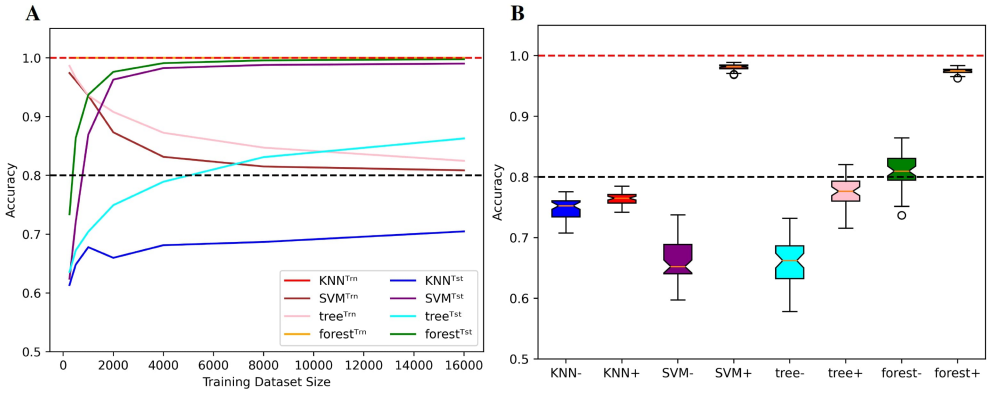


Fig 1.10. Behaviors of classical ML methods when extrapolating with mislabels. (A) Influence of the training dataset size over train (Trn) and test (Tst) accuracies for specific classical ML algorithms. Sequence length and mislabelling probability are fixed to $(L, \mu) = (8, 0.2)$. (B) With $(\mu, \alpha) = (0.2, 0.4)$, the models are trained on sequences of length 6 before being tested on the whole set of sequences of length 8. Distributions of test accuracies over 50 simulations are reported. The dataset size varies between $N = 500$ (e.g. KNN$^-$) and $N = 4000$ (e.g. KNN$^+$). Dotted lines indicate the 100% and 80% accuracy marks since $\mu = 20\%$.

Overall, this section highlights the importance of considering multiple challenges simultaneously when measuring the impact of capacity and training dataset size on the generalization performance of a variety of ML models. It also underscores the need for a better understanding of the trade-offs involved in choosing appropriate models and capacity for specific tasks, especially when dealing with small training datasets.

Conclusion

In conclusion, the use of statistical learning through neural networks holds great promise for gaining insight into the complex mechanisms that govern RNA folding. Even though over-parameterized models may be adequate for learning useful representations, the fact remains that the quality of a ML model highly depends on the data on which it has been trained. Our study highlights three main challenges researchers face when

working with current RNA structure data and provides suggestions for overcoming them.

A binary classification task has been defined to measure the abilities of a variety of ML algorithms to learn Watson-Crick complementarity. The definition of the task has allowed us to generate synthetical datasets to properly test our models in light of specific challenges one can encounter when dealing with RNA structure prediction. An emphasis has been put on four types of neural networks that act as representatives of four families of commonly used neural networks in the field.

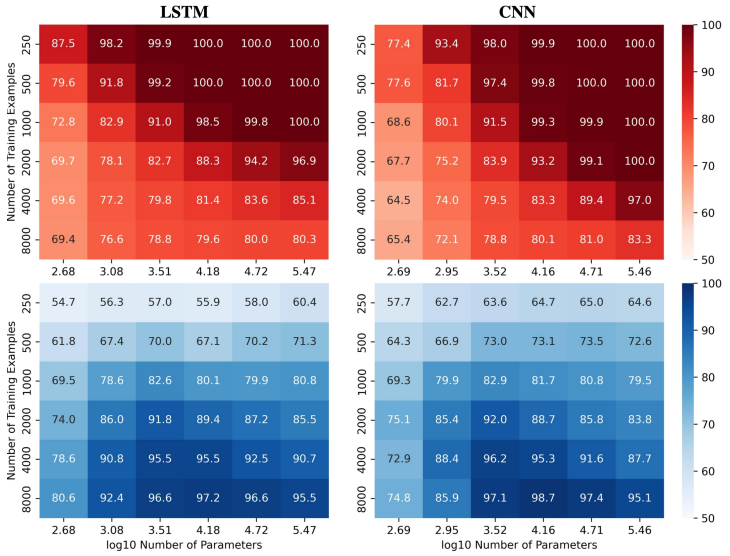
Specifically, when dealing with mislabels, low-capacity models may be preferable, as long as enough training examples are provided. Moreover, for tasks that require extrapolating to structurally dissimilar data, high-capacity models may provide better performance. With the fixed-size input involving limitations regarding length-wise generalization, we propose using models that can adapt to different RNA sequence lengths like recurrent neural networks or fully convolutional networks. Finally, we recommend exploring the behavior of a variety of neural networks on synthetic data to better understand their specific risks and benefits in predicting RNA structure. Overall, by addressing these challenges, machine learning could provide valuable insights into RNA folding and contribute to the development of new approaches for studying biological systems involving RNA.

Acknowledgments

We would like to express our sincere gratitude to Sébastien Lemieux and all the participants of the Computational Approaches to RNA Structure and Function conference held in Benasque, Spain in 2022 for their valuable suggestions and feedback.

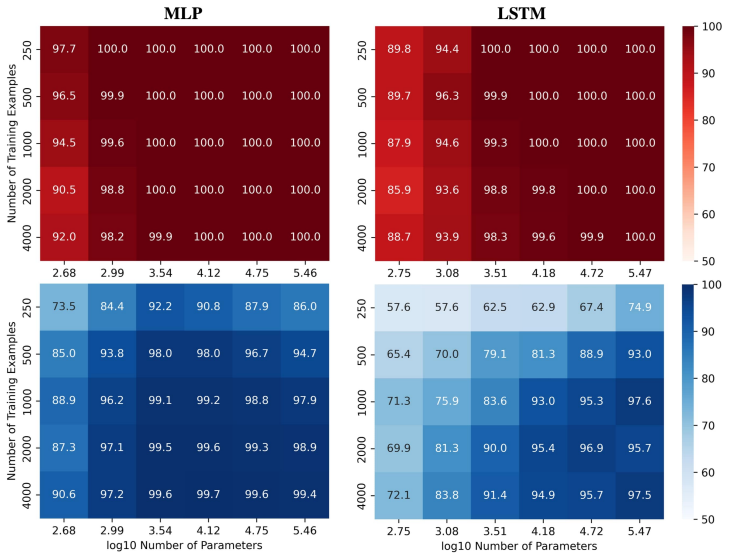
Supplementary material

Fig. S1.



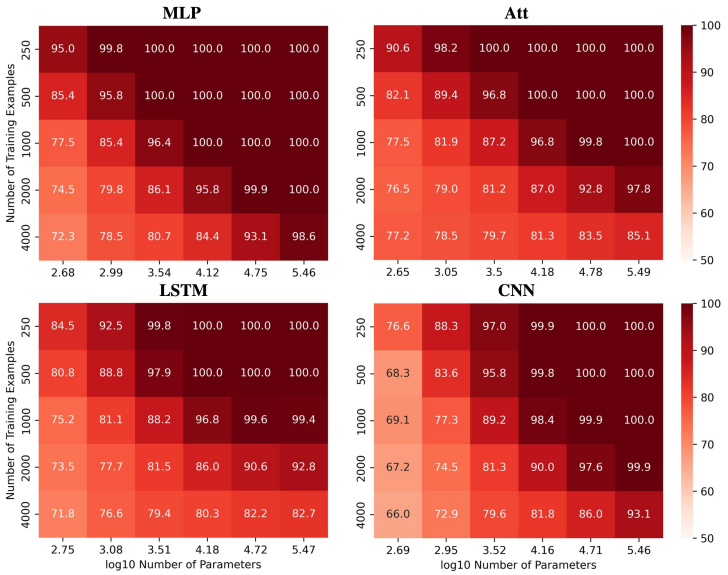
Performance of LSTM and CNN models when learning with mislabels. Train (red) and test (blue) mean accuracies over 50 simulations reported for LSTM and CNN models. Sequence length and mislabelling probability are respectively fixed to $L = 8$ and $\mu = 0.2$.

Fig. S2.



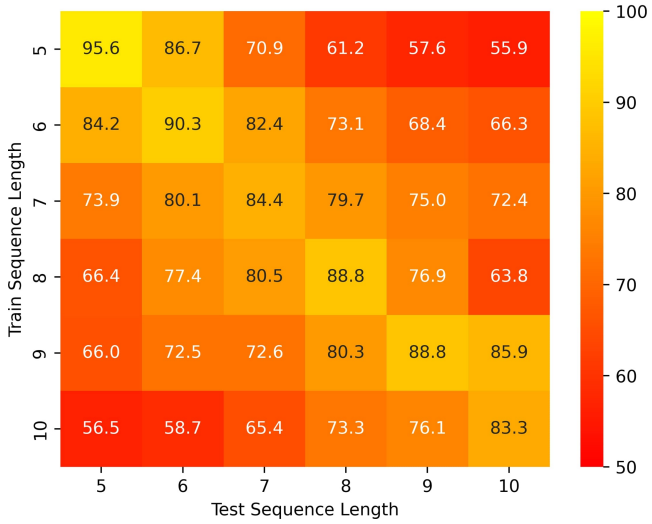
Performance of MLP and LSTM models in length-wise extrapolation context. Train (red) and test (blue) mean accuracies over 50 simulations reported for MLP and LSTM models. Models were trained on sequences of length 6 before being tested on sequences of length 8, without mislabelling in the training set.

Fig. S3.



Training performances for all models when learning with mislabels in extrapolation context. Train accuracies for all models when trained on sequences of length 6 with 20% mislabelled training examples and 40% positivity rate.

Fig. S4.



Performance for low-capacity LSTM when trained and tested on a variety of sequence lengths. Test accuracies for the LSTM model with $\log_{10} C \approx 3.5$ when trained and tested on 500 sequences of length 5 to 10 without mislabelling with a 50% positivity rate.

Conclusion

Dans l'ensemble, le travail effectué dans le cadre de ce mémoire de maîtrise témoigne d'une grande importance accordée aux potentiels risques et dangers à éviter lorsqu'on s'attaque au problème du repliement de l'ARN par ML. Autant au niveau de la nature des données que des rudiments des algorithmes, certaines précautions se doivent d'être prises. Avec une telle conscience des inconvénients, la table est mise pour les projets futurs.

D'abord, une attention particulière se doit d'être portée à l'encodage des séquences d'ARN. Les méthodes actuelles se contentent d'un encodage *one-hot* ou d'un encodage de style *word embedding* basé sur les séquences dont on connaît la structure. Toutefois, la majorité des molécules d'ARN séquencées n'ont pas de structure connue, mais ces séquences recèlent de l'information pertinente pouvant être exploitée pour en apprendre davantage sur les caractéristiques des séquences d'ARN existantes. En effet, les molécules d'ARN courantes ne sont pas le fruit d'un simple hasard puisqu'elles dérivent d'un processus évolutif bien particulier.

Ensuite, l'étude des aptitudes des algorithmes de ML et de DL ne se limite pas à la reconnaissance automatique des brins complémentaires. Une variété d'autres tâches peuvent être formulées pour mesurer leurs performances et leurs comportements sur des données synthétiques. Notamment, nous pouvons considérer la reconnaissance de motifs comme les boucles terminales, la prédiction d'appariement, et ultimement, la prédiction de structures. En addition aux architectures présentement utilisées en prédiction de structures d'ARN, les réseaux de type *denseNet* [**33**] devraient être évalués. Ces réseaux totalement convolutifs présentent un potentiel intéressant si on considère le progrès qu'ils ont apporté en reconnaissance d'objets, notamment en segmentation d'images [**11**, **41**, **47**], soit une façon de formuler le problème de la prédiction structurée en raison de la nature hiérarchique du repliement et de sa juxtaposition en segments (motifs).

De plus, les connaissances acquises par ce genre de travaux nous permettrons d'aborder des problèmes biologiques concrets présentant des données ribonucléiques à analyser. Par exemple, davantage d'efforts pourraient être dirigés sur l'analyse de la régulation de l'expression des gènes par microARN. Notamment, nous aimerions développer une méthode calculatoire capable de déterminer comment l'appariement des nucléotides entre un microARN et sa cible influence le degré d'extinction d'un gène [**3**, **51**, **84**].

Finalement, l'ensemble des travaux futurs s'effectueront en gardant le cap sur l'objectif ultime soit la prédiction de structures d'ARN à partir de la séquence. Des contributions majeures peuvent être apportées en ce qui a trait à l'utilisation de la recherche opérationnelle par le biais de la programmation stochastique et de la programmation en nombres entiers, techniques jusqu'à présent peu utilisées en optimisation structurée d'ARN [**58**,**85**]. Également, la dynamique structurale des ARN se doit d'être prise en compte sachant que les molécules d'ARN sont loin d'être statiques puisqu'elles alternent plutôt entre différentes conformations [**14**, **16**, **26**].

C'est avec l'intention de faire progresser ces travaux que le projet de doctorat intitulé «Apprentissage automatisé pour modéliser les réseaux d'interaction microARN-ARNmessager et leur interférence sur la production des protéines» sera entamé dès l'an prochain sous la direction de François Major.

Références bibliographiques

1. Manato Akiyama, Kengo Sato et Yasubumi Sakakibara : A max-margin training of rna secondary structure prediction integrated with the thermodynamic model. *Journal of bioinformatics and computational biology*, 16(06):1840025, 2018.
2. Randall Balestriero, Jerome Pesenti et Yann LeCun : Learning in high dimension always amounts to extrapolation. *arXiv preprint arXiv:2110.09485*, 2021.
3. Winston R Becker, Benjamin Ober-Reynolds, Karina Jouravleva, Samson M Jolly, Phillip D Zamore et William J Greenleaf : High-throughput analysis reveals rules for target rna binding and cleavage by ago2. *Molecular cell*, 75(4):741–755, 2019.
4. Mikhail Belkin, Daniel Hsu, Siyuan Ma et Soumik Mandal : Reconciling modern machine-learning practice and the classical bias–variance trade-off. *Proceedings of the National Academy of Sciences*, 116(32):15849–15854, 2019.
5. Stephan H Bernhart, Ivo L Hofacker, Sebastian Will, Andreas R Gruber et Peter F Stadler : Rnaalifold: improved consensus structure prediction for rna alignments. *BMC bioinformatics*, 9:1–13, 2008.
6. Leonard Berrada, Andrew Zisserman et M Pawan Kumar : Training neural networks for and by interpolation. *In International conference on machine learning*, pages 799–809. PMLR, 2020.
7. Philippe Brion et Eric Westhof : Hierarchy and dynamics of rna folding. *Annual review of biophysics and biomolecular structure*, 26(1):113–137, 1997.
8. Stephen K Burley, Charmi Bhikadiya, Chunxiao Bi, Sebastian Bittrich, Li Chen, Gregg V Crichlow, Jose M Duarte, Shuchismita Dutta, Maryam Fayazi, Zukang Feng *et al.* : Rcsb protein data bank: Celebrating 50 years of the pdb with new tools for understanding and visualizing biological macromolecules in 3d. *Protein Science*, 31(1):187–208, 2022.
9. Jamie J Cannone, Sankar Subramanian, Murray N Schnare, James R Collett, Lisa M D'Souza, Yushi Du, Brian Feng, Nan Lin, Lakshmi V Madabusi, Kirsten M Müller *et al.* : The comparative rna web (crw) site: an online database of comparative sequence and structure information for ribosomal, intron, and other rnas. *BMC bioinformatics*, 3:1–31, 2002.

10. Michael Charette et Michael W Gray : Pseudouridine in rna: what, where, how, and why. *IUBMB life*, 49(5):341–352, 2000.
11. Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy et Alan L Yuille : Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE transactions on pattern analysis and machine intelligence*, 40(4):834–848, 2017.
12. Xinshi Chen, Yu Li, Ramzan Umarov, Xin Gao et Le Song : Rna secondary structure prediction by learning unrolled algorithms. *arXiv preprint arXiv:2002.05810*, 2020.
13. Anne Condon, Beth Davy, Baharak Rastegari, Shelly Zhao et Finbarr Tarrant : Classifying rna pseudoknotted structures. *Theoretical Computer Science*, 320(1):35–50, 2004.
14. Paul Dallaire, Huiping Tan, Keith Szulwach, Christopher Ma, Peng Jin et François Major : Structural dynamics control the microrna maturation pathway. *Nucleic acids research*, 44(20):9956–9964, 2016.
15. Padideh Danaee, Mason Rouches, Michelle Wiley, Dezhong Deng, Liang Huang et David Hendrix : bprna: large-scale automated annotation and analysis of rna secondary structure. *Nucleic acids research*, 46(11):5381–5394, 2018.
16. Elizabeth A Dethoff, Katja Petzold, Jeetender Chugh, Anette Casiano-Negroni et Hashim M Al-Hashimi : Visualizing transient low-populated structures of rna. *Nature*, 491(7426):724–728, 2012.
17. Jacob Devlin, Ming-Wei Chang, Kenton Lee et Kristina Toutanova : Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.
18. Chuong B Do, Daniel A Woods et Serafim Batzoglou : Contrafold: Rna secondary structure prediction without physics-based models. *Bioinformatics*, 22(14):e90–e98, 2006.
19. Robin D Dowell et Sean R Eddy : Efficient pairwise rna structure prediction and alignment using sequence alignment constraints. *BMC bioinformatics*, 7(1):1–18, 2006.
20. Vincent Dumoulin et Francesco Visin : A guide to convolution arithmetic for deep learning. *arXiv preprint arXiv:1603.07285*, 2016.
21. Mathieu J Dupont et François Major : D-orb: a web server to extract structural features of related but unaligned rna sequences. *Journal of Molecular Biology*, page 168181, 2023.
22. Jörg Fallmann, Sebastian Will, Jan Engelhardt, Björn Grüning, Rolf Backofen et Peter F Stadler : Recent advances in rna folding. *Journal of biotechnology*, 261:97–104, 2017.
23. Christoph Flamm, Julia Wielach, Michael T Wolfinger, Stefan Badelt, Ronny Lorenz et Ivo L Hofacker : Caveats to deep learning approaches to rna secondary structure prediction. *Biorxiv*, pages 2021–12, 2021.

24. Laiyi Fu, Yingxin Cao, Jie Wu, Qinke Peng, Qing Nie et Xiaohui Xie : Ufold: fast and accurate rna secondary structure prediction with deep learning. *Nucleic acids research*, 50(3):e14–e14, 2022.
25. Boris Fürtig, Christian Richter, Jens Wöhnert et Harald Schwalbe : Nmr spectroscopy of rna. *ChemBioChem*, 4(10):936–962, 2003.
26. Laura R Ganser, Megan L Kelly, Daniel Herschlag et Hashim M Al-Hashimi : The roles of structural dynamics in the cellular functions of rnas. *Nature reviews Molecular cell biology*, 20(8):474–489, 2019.
27. Ian Goodfellow, Yoshua Bengio et Aaron Courville : *Deep learning*. MIT press, 2016.
28. Paul G Higgs et Niles Lehman : The rna world: molecular cooperation at the origins of life. *Nature Reviews Genetics*, 16(1):7–17, 2015.
29. Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever et Ruslan R Salakhutdinov : Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*, 2012.
30. Sepp Hochreiter et Jürgen Schmidhuber : Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.
31. Ivo L Hofacker, Martin Fekete et Peter F Stadler : Secondary structure prediction for aligned rna sequences. *Journal of molecular biology*, 319(5):1059–1066, 2002.
32. Ivo L Hofacker, Walter Fontana, Peter F Stadler, L Sebastian Bonhoeffer, Manfred Tacker, Peter Schuster *et al.* : Fast folding and comparison of rna secondary structures. *Monatshefte fur chemie*, 125:167–167, 1994.
33. Gao Huang, Zhuang Liu, Laurens Van Der Maaten et Kilian Q Weinberger : Densely connected convolutional networks. *In Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.
34. Sergey Ioffe et Christian Szegedy : Batch normalization: Accelerating deep network training by reducing internal covariate shift. *In International conference on machine learning*, pages 448–456. pmlr, 2015.
35. Bryan D James, Gary J Olsen et Norman R Pace : [18] phylogenetic comparative analysis of rna secondary structure. *In Methods in Enzymology*, volume 180, pages 227–239. Elsevier, 1989.
36. John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko *et al.* : Highly accurate protein structure prediction with alphafold. *Nature*, 596(7873):583–589, 2021.
37. Ioanna Kalvari, Eric P Nawrocki, Nancy Ontiveros-Palacios, Joanna Argasinska, Kevin Lamkiewicz, Manja Marz, Sam Griffiths-Jones, Claire Toffano-Nioche, Daniel Gautheret, Zasha Weinberg *et al.* : Rfam 14: expanded coverage of metagenomic, viral and microrna families. *Nucleic Acids Research*, 49(D1):D192–D200, 2021.

38. Diederik P Kingma et Jimmy Ba : Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
39. Bjarne Knudsen et Jotun Hein : Rna secondary structure prediction using stochastic context-free grammars and evolutionary history. *Bioinformatics (Oxford, England)*, 15(6):446–454, 1999.
40. Bjarne Knudsen et Jotun Hein : Pfold: Rna secondary structure prediction using stochastic context-free grammars. *Nucleic acids research*, 31(13):3423–3428, 2003.
41. Philipp Krähenbühl et Vladlen Koltun : Efficient inference in fully connected crfs with gaussian edge potentials. *Advances in neural information processing systems*, 24, 2011.
42. Yann LeCun *et al.* : Generalization and network design strategies. *Connectionism in perspective*, 19(143-155):18, 1989.
43. Neocles B Leontis et Eric Westhof : Conserved geometrical base-pairing patterns in rna. *Quarterly reviews of biophysics*, 31(4):399–455, 1998.
44. Neocles B Leontis et Eric Westhof : Geometric nomenclature and classification of rna base pairs. *Rna*, 7(4):499–512, 2001.
45. Cyrus Levinthal : Are there pathways for protein folding? *Journal de chimie physique*, 65:44–45, 1968.
46. Xiaoyu Li, Shiqing Ma et Chengqi Yi : Pseudouridine: the fifth rna nucleotide with renewed interests. *Current Opinion in Chemical Biology*, 33:108–116, 2016.
47. Jonathan Long, Evan Shelhamer et Trevor Darrell : Fully convolutional networks for semantic segmentation. *In Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015.
48. Bertrand Marchand, Sebastian Will, Sarah Berkemer, Laurent Bulteau et Yann Ponty : Automated design of dynamic programming schemes for RNA folding with pseudoknots. *In WABI 2022 - 22nd Workshop on Algorithms in Bioinformatics*, Potsdam, Germany, septembre 2022.
49. David H Mathews, Jeffrey Sabina, Michael Zuker et Douglas H Turner : Expanded sequence dependence of thermodynamic parameters improves prediction of rna secondary structure. *Journal of molecular biology*, 288(5):911–940, 1999.
50. Gunter Meister et Thomas Tuschl : Mechanisms of gene silencing by double-stranded rna. *Nature*, 431(7006):343–349, 2004.
51. Yale S Michaels, Mike B Barnkob, Hector Barbosa, Toni A Baeumler, Mary K Thompson, Violaine Andre, Huw Colin-York, Marco Fritzsche, Uzi Gileadi, Hilary M Sheppard *et al.* : Precise tuning of gene expression levels in mammalian cells. *Nature communications*, 10(1):818, 2019.
52. Peter B Moore : Structural motifs in rna. *Annual review of biochemistry*, 68(1): 287–300, 1999.
53. Vinod Nair et Geoffrey E Hinton : Rectified linear units improve restricted boltzmann machines. *In Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 807–814, 2010.

54. Eric P Nawrocki et Sean R Eddy : Infernal 1.1: 100-fold faster rna homology searches. *Bioinformatics*, 29(22):2933–2935, 2013.
55. Ruth Nussinov et Ann B Jacobson : Fast algorithm for predicting the secondary structure of single-stranded rna. *Proceedings of the National Academy of Sciences*, 77(11):6309–6313, 1980.
56. Marc Parisien et Francois Major : The mc-fold and mc-sym pipeline infers rna structure from sequence data. *Nature*, 452(7183):51–55, 2008.
57. Evan Peters et Maria Schuld : Generalization despite overfitting in quantum machine learning models. *arXiv preprint arXiv:2209.05523*, 2022.
58. Vladimir Reinharz, François Major et Jérôme Waldispühl : Towards 3d structure prediction of large rna molecules: an integer programming framework to insert local 3d motifs in rna secondary structure. *Bioinformatics*, 28(12):i207–i214, 2012.
59. Vladimir Reinharz, Antoine Soulé, Eric Westhof, Jérôme Waldispühl et Alain Denise : Mining for recurrent long-range interactions in rna structures reveals embedded hierarchies in network families. *Nucleic acids research*, 46(8): 3841–3851, 2018.
60. Elena Rivas : Rna structure prediction using positive and negative evolutionary information. *PLoS computational biology*, 16(10):e1008387, 2020.
61. Elena Rivas et Sean R Eddy : A dynamic programming algorithm for rna structure prediction including pseudoknots. *Journal of molecular biology*, 285(5):2053–2068, 1999.
62. Elena Rivas, Raymond Lang et Sean R Eddy : A range of complex probabilistic models for rna secondary structure prediction that includes the nearest-neighbor model and more. *RNA*, 18(2):193–212, 2012.
63. Olaf Ronneberger, Philipp Fischer et Thomas Brox : U-net: Convolutional networks for biomedical image segmentation. *In Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*, pages 234–241. Springer, 2015.
64. Hasim Sak, Andrew W. Senior et Françoise Beaufays : Long short-term memory recurrent neural network architectures for large scale acoustic modeling. *In INTERSPEECH*, pages 338–342, 2014.
65. Shimi Salant et Jonathan Berant : Contextualized word representations for reading comprehension. *arXiv preprint arXiv:1712.03609*, 2017.
66. Nicolae Sapoval, Amirali Aghazadeh, Michael G Nute, Dinler A Antunes, Advait Balaji, Richard Baraniuk, CJ Barberan, Ruth Dannenfelser, Chen Dun, Mohammadamin Edrisi *et al.* : Current progress and open challenges for applying deep learning across the biosciences. *Nature Communications*, 13(1):1728, 2022.
67. Roman Sarrazin-Gendron, Vladimir Reinharz, Carlos G Oliver, Nicolas Moitessier et Jérôme Waldispühl : Automated, customizable and efficient

identification of 3d base pair modules with bayespairing. *Nucleic acids research*, 47(7):3321–3332, 2019.

68. Kengo Sato, Manato Akiyama et Yasubumi Sakakibara : Rna secondary structure prediction using deep learning with thermodynamic integration. *Nature communications*, 12(1):941, 2021.
69. Matthew G Seetin et David H Mathews : Rna structure prediction: an overview of methods. *Bacterial regulatory RNA: methods and protocols*, pages 99–122, 2012.
70. Alexander Serganov et Evgeny Nudler : A decade of riboswitches. *Cell*, 152(1-2):17–24, 2013.
71. Tao Shen, Zhihang Hu, Zhangzhi Peng, Jiayang Chen, Peng Xiong, Liang Hong, Liangzhen Zheng, Yixuan Wang, Irwin King, Sheng Wang *et al.* : E2efold-3d: End-to-end deep learning method for accurate de novo rna 3d structure prediction. *arXiv preprint arXiv:2207.01586*, 2022.
72. Jaswinder Singh, Jack Hanson, Kuldip Paliwal et Yaoqi Zhou : Rna secondary structure prediction using an ensemble of two-dimensional deep neural networks and transfer learning. *Nature communications*, 10(1):5407, 2019.
73. Michael F Sloma et David H Mathews : Exact calculation of loop formation probability identifies folding motifs in rna secondary structures. *RNA*, 22(12):1808–1818, 2016.
74. Alex J Smola et Bernhard Schölkopf : A tutorial on support vector regression. *Statistics and computing*, 14:199–222, 2004.
75. Marcell Szikszai, Michael Wise, Amitava Datta, Max Ward et David H Mathews : Deep learning models for rna secondary structure prediction (probably) do not generalize across families. *Bioinformatics*, 38(16):3892–3899, 2022.
76. Zhen Tan, Yinghan Fu, Gaurav Sharma et David H Mathews : Turbofold ii: Rna structural alignment and secondary structure prediction informed by multiple homologs. *Nucleic acids research*, 45(20):11570–11581, 2017.
77. Yingjie Tian et Yuqi Zhang : A comprehensive survey on regularization strategies in machine learning. *Information Fusion*, 80:146–166, 2022.
78. Ignacio Tinoco Jr et Carlos Bustamante : How rna folds. *Journal of molecular biology*, 293(2):271–281, 1999.
79. Douglas H Turner, Naoki Sugimoto et Susan M Freier : Rna structure prediction. *Annual review of biophysics and biophysical chemistry*, 17(1):167–192, 1988.
80. Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser et Illia Polosukhin : Attention is all you need. *Advances in neural information processing systems*, 30, 2017.
81. Linyu Wang, Yuanning Liu, Xiaodan Zhong, Haiming Liu, Chao Lu, Cong Li et Hao Zhang : Dmfold: A novel method to predict rna secondary structure with pseudoknots based on deep learning and improved base pair maximization principle. *Frontiers in genetics*, 10:143, 2019.
82. Eric Westhof : Twenty years of rna crystallography. *Rna*, 21(4):486–487, 2015.

83. Chunming Xu et Scott A Jackson : Machine learning and complex biological data, 2019.
84. Yifei Yan, Mariana Acevedo, Lian Mignacca, Philippe Desjardins, Nicolas Scott, Roqaya Imane, Jordan Quenneville, Julie Robitaille, Albert Feghaly, Etienne Gagnon *et al.* : The sequence features that define efficient and specific hago2-dependent mirna silencing guides. *Nucleic acids research*, 46(16):8181–8196, 2018.
85. Jason Yao, Vladimir Reinharz, François Major et Jérôme Waldispühl : Rnamoip: prediction of rna secondary structure and local 3d motifs from sequence data. *Nucleic acids research*, 45(W1):W440–W444, 2017.
86. Shay Zakov, Yoav Goldberg, Michael Elhadad et Michal Ziv-Ukelson : Rich parameterization improves rna structure prediction. *Journal of Computational Biology*, 18(11):1525–1542, 2011.
87. Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht et Oriol Vinyals : Understanding deep learning (still) requires rethinking generalization. *Communications of the ACM*, 64(3):107–115, 2021.
88. Hao Zhang, Chunhe Zhang, Zhi Li, Cong Li, Xu Wei, Borui Zhang et Yuanning Liu : A new method of rna secondary structure prediction based on convolutional neural network and dynamic programming. *Frontiers in genetics*, 10:467, 2019.
89. Jingyu Zhao, Feiqing Huang, Jia Lv, Yanjie Duan, Zhen Qin, Guodong Li et Guangjian Tian : Do rnn and lstm have long memory? *In International Conference on Machine Learning*, pages 11365–11375. PMLR, 2020.
90. Qi Zhao, Zheng Zhao, Xiaoya Fan, Zhengwei Yuan, Qian Mao et Yudong Yao : Review of machine learning methods for rna secondary structure prediction. *PLoS computational biology*, 17(8):e1009291, 2021.
91. Michael Zuker : Mfold web server for nucleic acid folding and hybridization prediction. *Nucleic acids research*, 31(13):3406–3415, 2003.
92. Michael Zuker et Patrick Stiegler : Optimal computer folding of large rna sequences using thermodynamics and auxiliary information. *Nucleic acids research*, 9(1):133–148, 1981.