UNIVERSITÄT
DUISBURG
ESSEN

Open-Minded

Dissertation

RNA Structure Prediction Guided by Co-Evolutionary Information—Method Development and Applications

From the Faculty of Biology
the University of Duisburg-Essen
approved dissertation
to obtain the degree
Dr. rer. nat.
from

M. Sc. Mehari Bayou Zerihun
from Ethiopia

Advisor:
Prof. Dr. Alexander Schug
Co-advisor:
Prof. Dr. Elsa Sánchez-García

Essen, Germany
April 2021

The experiments underlying the present work were conducted at Steinbuch Center for Computing, Karlsruhe Institute of Technology and Institute for Advanced Simulations, Jülich Research Center.

1. Examiner: Prof. Dr. Alexander Schug

2. Examiner: Prof. Dr. Elsa Sánchez-García

Chairman of the Examination Committee: Prof. Dr. Daniel Hoffmann

Date of the Oral Examination: 2021-05-14

DOI: 10.17185/duepublico/74472
URN: urn:nbn:de:hbz:464-20210625-080038-6

UNIVERSITÄT
DUISBURG
ESSEN

Offen im Denken

Dissertation

Vorhersage der RNA-Struktur anhand koevolutionärer Informationen – Methodenentwicklung und Anwendungen

Von der Fakultät für Biologie
der Universität Duisburg-Essen
genehmigte Dissertation
zur Erlangung des Grades
Dr. rer. nat.
von

M. Sc. Mehari Bayou Zerihun
aus Äthiopien

Gutachter:
Prof. Dr. Alexander Schug
Gutachter:
Prof. Dr. Elsa Sánchez-García

Essen, Deutschland
April 2021

Die der vorliegenden Arbeit zugrunde liegenden Experimente wurden am Steinbuch Centre for Computing, Karlsruher Institut für Technologie und Institute for Advanced Simulations, Forschungszentrum Jülich durchgeführt.

1. Gutachter: Prof. Dr. Alexander Schug

2. Gutachter: Prof. Dr. Elsa Sánchez-García

Vorsitzender des Prüfungsausschusse: Prof. Dr. Daniel Hoffmann

Tag der mündlichen Prüfung: 2021-05-14

Hiermit versichere ich, die vorliegende Dissertation selbstständig, ohne fremde Hilfe und ohne Benutzung anderer als den angegebenen Quellen angefertigt zu haben. Alle aus fremden Werken direkt oder indirekt übernommenen Stellen sind als solche gekennzeichnet. Die vorliegende Dissertation wurde in keinem anderen Promotionsverfahren eingereicht. Mit dieser Arbeit strebe ich die Erlangung des akademischen Grades Doktor der Naturwis- senschaften (Dr. rer. nat.) an.

Essen; April 5, 2021

Ort, Datum	Mehari Bayou Zerihun

Acknowledgments

First and foremost, I would like to thank my supervisor Prof. Dr. Alexander Schug, for accepting me to work on this multidisciplinary and exciting research topic. Thank you for believing in me and for your kind and unlimited support. Next, I would like to thank Prof. Dr. Elsa Sánchez-García for agreeing to review this work.

I also like to express my gratitude to Dr. Fabrizio Pucci for many reasons. A lot of this work is made possible because of you. I am fortunate to meet you, and it is a great honor and pleasure to work with you. I also thank Dr. Emanuel K. Peter for the collaborative work we had while working in Prof. Dr. Schug's research group.

I am very grateful for the kind and professional support I got from the international advisory offices in Karlsruhe Institute of Technology and Jülich Research Center during my first arrivals and throughout my stay there. I also thank the support I received from the secretarial offices at Steinbuch Centre for Computing, Karlsruhe Institute of Technology, and Institute for Advanced Simulations, Jülich Research Center. I also thank the support I got from the Faculty of Biology secretarial and international offices of the University of Duisburg-Essen.

I would like to thank all individuals who have been my colleagues in the research group of Prof. Dr. Alexander Schug. I thank Claude and Ines for their kind support and guidance during my arrival. I also thank Arthur, Oskar, Momin, Marie, and Fathia for being kind and supportive colleagues. I am incredibly grateful to Jakob for the kind support you provided me, particularly during my arrival in Jülich and around the end of my work. I also thank Jan, Julian, and Linda for being friendly colleagues, although our meeting was relatively short.

Finally, thanks to my brothers and sisters Aemro and Birte, Gize and Meazi—for the support and motivation you provided me and the time and hospitality you gave me on my first visit to the United States to attend a biophysical society annual conference. Thanks to my sisters, Abe and Wube Andualem—your support played a significant role in reaching this point in my life.

Mehari Bayou Zerihun
2021

Abstract

Ribonucleic acids (RNAs) are critical players in cellular activities. They are, e.g., involved in coding, decoding, regulation, and expression of genes. Their function is related to their three-dimensional (3D) structure. Consequently, understanding their structure is critical for understanding their function. Technological advances in high-throughput sequencing methods have made it possible to sequence many RNAs. The sequenced RNAs are stored in public databases, and the number of sequenced RNAs keeps growing. Nevertheless, the vast majority of them lack the corresponding three-dimensional structure—since RNA molecules are incredibly flexible, experimental RNA structure determination is challenging. A complementary approach is to use computer simulations to model RNA 3D structure starting from the sequence.

Such computer simulations to predict bio-molecular 3D structure can be highly challenging as the energy landscape is enormous and complex. Including a priori information in molecular modeling tools can help guide structure prediction more accurately by reducing the search space to the energy landscape. A particular example is providing pairs of nucleobases known to be spatially proximal as restraints. While several experimental approaches exist, a theoretical approach uses sophisticated statistical and machine learning algorithms to mine information about nucleobase pairs from sequences.

During the course of evolution, RNAs undergo mutations. Mutations that do not adversely affect survival take place randomly. However, others must occur in tandem–a change in nucleobase of an RNA in one place can trigger a complementary change in sequentially far region in the RNA sequence but in proximity within the 3D structure—to preserve the structure and function of RNA and ensure the survival of organisms.

Coordinated mutations leave imprints of nucleotide pair co-evolution, and this co-evolutionary information may be extracted from multiple sequence alignment (MSA) of homologous RNAs using sophisticated algorithms. In the last decade, inverse statistical methods based on generative models known as direct-coupling analysis (DCA) have shown tremendous success in predicting spatially adjacent residue pairs of proteins from MSA data. These pairs are incorporated with molecular modeling tools resulting in accurate protein 3D structure prediction at the level of experimental resolution. Inverse statistical methods are also recently started to be used in RNA 3D structure prediction, but their success is somewhat limited compared to protein structure prediction.

This thesis presents a new and improved RNA contact prediction method and its application for RNA 3D structure prediction. In particular, the thesis (i) presents software

implementation of state-of-the-art DCA algorithms that are contained in a light-weight, stand-alone, and open-source software; (ii) makes available a curated RNA dataset to test and compare the performance of contact prediction algorithms on the dataset; (iii) introduces a new and improved RNA contact prediction algorithm based on a combination of DCA and convolutional neural network that improves RNA contact prediction from MSA and; (iv) finally, provides a workflow for the RNA 3D structure prediction using putative contacts obtained from the new algorithm as restraints with a molecular modeling tool based on coarse-grained replica-exchange Monte-Carlo method.

Keywords: Ribonucleic acid (RNA), co-evolution, generative models, direct-coupling analysis (DCA), convolutional neural networks, multiple sequence alignment (MSA), RNA 3D structure.

Zusammenfassung

Ribonukleinsäuren (RNAs) spielen eine entscheidende Rolle bei zellulären Aktivitäten. Sie sind z.B. an der Kodierung, Dekodierung, Regulation und Expression von Genen beteiligt. Ihre Funktion ist direkt mit ihrer dreidimensionalen (3D) Struktur verbunden. Folglich ist das Verständnis ihrer Struktur entscheidend für das Verständnis ihrer Funktion. Technologische Fortschritte bei Hochdurchsatz-Sequenzierungsmethoden haben es ermöglicht, viele RNAs zu sequenzieren. Die sequenzierten RNAs werden in öffentlichen Datenbanken gespeichert, und die Anzahl der sequenzierten RNAs wächst weiter. Trotz der riesigen Datenmenge meisten fehlt oft die entsprechende dreidimensionale Struktur - da RNA-Moleküle unglaublich flexibel sind, ist die experimentelle Bestimmung der RNA-Struktur eine Herausforderung. Ein komplementärer Ansatz besteht darin, Computersimulationen zu verwenden, um die RNA-3D-Struktur ausgehend von der Sequenz zu modellieren.

Computersimulationen zur Vorhersage der biomolekularen 3D-Struktur ausgehend von der Sequenz können eine große Herausforderung darstellen, da die Energielandschaft groß und komplex ist. Das Einbeziehen von A-priori-Informationen in molekulare Modellierungswerkzeuge kann dabei helfen, die Strukturvorhersage besser zu steuern durch Reduzieren des Suchraums der Energielandschaft. Ein besonderes Beispiel ist die Bereitstellung von Paaren von Nukleobasen, von denen bekannt ist, dass sie in räumlicher Näher befinden. Während es dazu mehrere experimentelle Ansätze gibt, verwendet ein theoretischer Ansatz ausgefeilte statistische und maschinelle Lernalgorithmen, um Informationen über Nukleobasenpaare aus Sequenzen zu gewinnen.

Im Verlauf der Evolution unterliegen RNAs Mutationen. Mutationen, die das Überleben nicht beeinträchtigen, finden zufällig statt. Andere müssen jedoch gleichzeitig auftreten - eine Änderung der Nukleobase einer RNA an einer Stelle löst eine komplementäre Änderung in großer Entfernung in der RNA-Sequenz, jedoch proximal innerhalb der 3D-Struktur aus, um Struktur und Funktion der RNA zu erhalten und damit das Überleben von Organismen zu ermöglichen. Koordinierte Mutationen hinterlassen daher Abdrücke der Koevolution von Nukleotidpaaren, und diese koevolutionäre Information kann unter Verwendung ausgefeilter Algorithmen aus dem Multiple Sequence Alignment (MSA) homologer RNAs extrahiert werden. In den letzten zehn Jahren haben inverse statistische Methoden, die auf generativen Modellen basieren, die als Direct Coupling Analysis (DCA) bekannt sind, enorme Erfolge bei der Vorhersage räumlich benachbarter Proteinrestpaare aus MSA-Daten gezeigt. Diese Paare können von molekularen Modellierungswerkzeugen genutzt werden und führen zu einer genaueren Vorhersage der Protein-3D-Struktur. Inverse statistische Methoden werden seit

kurzem auch für die Vorhersage der RNA-3D-Struktur eingesetzt, ihr Erfolg ist jedoch im Vergleich zur Vorhersage der Proteinstruktur bisher begrenzt.

In dieser Arbeit wird eine neue und verbesserte Methode zur Vorhersage von RNA-Kontakten und ihre Anwendung zur Vorhersage der RNA-3D-Struktur vorgestellt. In der Arbeit (i) wird insbesondere die Open Source Software-Implementierung von DCA-Algorithmen auf dem neuesten Stand der Technik vorgestellt; (ii) ein kurierten RNA-Datensatz zur Verfügung gestellt, um die Leistung von Kontaktvorhersagealgorithmen für den Datensatz zu testen und zu vergleichen; (iii) wird ein neuer und verbesserter Algorithmus zur Vorhersage von RNA-Kontakten basierend auf einer Kombination von DCA und Faltungs-Neuronales Netzwerk vorgestellt, der die Vorhersage von RNA-Kontakten verbessert; (iv) stellt schließlich einen Workflow für die Vorhersage der RNA-3D-Struktur unter Verwendung vorhergesagter Kontakte vor.

Schlüsselwörter: Ribonukleinsäure (RNA), Koevolution, generative Modelle, Direktkopplungsanalyse (DCA), Faltungs-Neuronale Netze, Mehrfachsequenz-Alignment (MSA), RNA 3D Struktur.

List of Publications

In the context of this doctoral work the following articles were published

- **Mehari B Zerihun**, Fabrizio Pucci, Emanuel K Peter, Alexander Schug
  pydca v1.0: a comprehensive software for direct coupling analysis of RNA and protein sequences. *Bioinformatics, Volume 36, Issue 7, 1 April 2020, Pages 2264–2265, https://doi.org/10.1093/bioinformatics/btz892*

- Fabrizio Pucci*, **Mehari B. Zerihun***, Emanuel K Peter and Alexander Schug
  Evaluating DCA-based method performances for RNA contact prediction by a well-curated dataset. *RNA (2020) Vol. 26, No. 7, pages 794–802, https://doi.org/10.1261/rna.073809.119*

- **Mehari B. Zerihun** and Alexander Schug
  Biomolecular coevolution and its applications: Going from structure prediction toward signaling, epistasis, and function. *Biochem Soc Trans (2017) 45 (6): 1253–1261, https://doi.org/10.1042/BST20170063*

- **Mehari B. Zerihun** and Alexander Schug
  Biomolecular Structure Prediction via Coevolutionary Analysis: A Guide to theStatistical Framework. *NIC Symposium 2018. Forschungszentrum Jülich GmbH, John von Neumann Institute for Computing (NIC), Schriften des Forschungszentrums Jülich, NIC Series, Vol. 49,ISBN 978-3-95806-285-6, pp. 15. http://hdl.handle.net/2128/17544*

In the context of this doctoral work the following article has been submitted

- **Mehari B. Zerihun***, Fabrizio Pucci* and Alexander Schug
  CoCoNet—Boosting RNA contact prediction using convolutional neural networks. *submitted, bioRxiv version https://doi.org/10.1101/2020.07.30.229484*

*These authors had equally contributed.

List of Symbols

Abbreviations

Abbreviation	Denomination
API	Application Programming Interface
CNN	Convolutional Neural Network
DCA	Direct Coupling Analysis
ELU	Exponential Linear Unit
FN	False Negative
FP	False Positive
MCC	Mathews Correlation Coefficient
$mfDCA$	Mean-field Direct Coupling Analysis
$mRNA$	Messenger Ribonucleic Acid
MSA	Multiple Sequence Alignment
MI	Mutual Information
PPV	Positive Predictive Value
PDB	Protein Data Bank
$Pfam$	Protein family
$plmDCA$	Pseudo-likelihood Maximization Direct Coupling Analysis
$ReLU$	Rectified Linear Unit
RNA	Ribonucleic Acid
$rRNA$	Ribosomal Ribonucleic Acid
$Rfam$	RNA family
$3D$	Three-dimensional
TN	True Negative
TP	True Positive
$tRNA$	Transfer RNA

Contents

Acknowledgments i

Abstract iii

Zusammenfassung v

List of Publications vii

List of Symbols ix

1. Introduction to Ribonucleic Acids 1
- 1.1. RNA structure . . . 1
- 1.2. RNA structure determination methods . . . 4
- 1.3. Summary . . . 5

2. Probabilistic Models of Contact Prediction 6
- 2.1. Introduction . . . 6
- 2.2. Global probability model . . . 8
- 2.3. Mean-field approximation . . . 11
- 2.4. Pseudo-likelihood maximization . . . 13
- 2.5. Gaussian approximation . . . 14
- 2.6. Boltzmann learning . . . 15
- 2.7. Estimating interaction scores . . . 16
- 2.8. Summary . . . 17

3. Convolutional Neural Networks 19
- 3.1. Introduction . . . 19
- 3.2. Architecture of Convolutional Neural Networks . . . 19
- 3.3. Activation Functions . . . 21
- 3.4. Traning convolutional neural networks . . . 21
- 3.5. Summary . . . 22

4. Implementing DCA Algorithms into Computer Software 24
- 4.1. Introduction . . . 24
- 4.2. Architecture of *pydca* . . . 25

4.3. Sequence reweighting . . . 27
4.4. Mean-field DCA implementation . . . 28
4.5. Pseudo-likelihood maximization DCA implementation . . . 29
4.6. Mapping reference sequences to its family . . . 29
4.7. Trimming multiple sequence alignment data . . . 30
4.8. Contact map and positive predictive value visualization . . . 31
4.9. Algorithmic complexities in *pydca* . . . 32
4.10. Usage examples of pydca . . . 32
4.11. Summary and conclusion . . . 37

5. Comparing DCA Algorithms for RNA Contact Prediction 38
5.1. Introduction . . . 38
5.2. Dataset formation . . . 39
5.3. Effect of multiple sequence alignment data . . . 41
5.4. Contact prediction by nucleotide pair . . . 46
5.5. Overall contact prediction accuracy . . . 50
5.6. Tertiary contact prediction accuracy . . . 56
5.7. Summary and conclusion . . . 61

6. Enhancing RNA Contact Prediction by Convolutional Neural Networks 63
6.1. Introduction . . . 63
6.2. Coevolutional structural features . . . 63
6.3. Network architecture . . . 65
6.4. Learning parameters of filter matrices . . . 66
6.5. Results . . . 66
6.6. Summary and conclusion . . . 74

7. RNA 3D Structure Prediction 76
7.1. Introduction . . . 76
7.2. Materials and methods . . . 76
7.3. Results . . . 79
7.4. Summary and conclusion . . . 80

8. Summary and Outlook 83
8.1. Summary . . . 83
8.2. Outlook . . . 85

A. RNA Dataset Description 86

B. Mathematical Details Related to DCA 91

Bibliography 100

List of Figures

1-1. Chemical structure of the five nucleobases. Adenine(A) and guanine(G) are purine bases. Uracil(U), thymine(T), and cytosine(C) are pyrimidine bases[92]. 2

1-2. Chemical structures of cytidine, and guasonine nucleotides[92]. The nucleotides are formed chemical bond between the C1' atom of the ribose sugar with N1 atom of pyrimidines or the N9 atom of purines. The ribose sugar is also connected to the phosphate group by gylcosidic covalent bonding. 3

1-3. Figure showing the secondary and tertiary structures of the adenine riboswitch RNA (PDB id 4tzx). The secondary structure contains three stems, two loops, and two junctions. A cartoon representation of the tertiary structure shows the backbone in thick loop paths, whereas sticks represent the side chains. . 4

2-1. Mutations in protein/RNA sequences through evolution leave traces of correlated mutations between residues/nucleotides. These coordinated mutations are necessary to preserve a protein/RNA molecule structure while functionally important sites remain conserved. Statistical methods are used to infer coevolving site-pairs such as i and j that are in proximity within the three-dimensional structure of proteins/RNAs. Functionally essential sites may stay conserved during coevolution. 7

3-1. Architecture of convolutional neural networks[49]. The diagram shows pooling, convolution, and dense layers. The pooling layer involves the downsampling of input data using methods such as max-pooling or average-pooling. The convolution layer is a layer where convolution operations are performed using kernel matrices. The dense layer consists of fully connected layers. In this diagram, the final 1×10 layer is the output layer that classifies a given image, e.g., handwritten digits. 20

3-2. Illustration of convolution operation (Figure adopted from[75]). In the context of CNNs, convolution involves sliding a kernel (matrix of weights) on input data and multiplying the corresponding elements of the section of the data that the kernel overlaps. The final result is the sum of all those multiplied numbers. 20

3-3. Linear unit activation functions (the Figure adopted is from[6]). The ReLU function is biologically interpretable. However, it filters out negative input values and is not differentiable at zero. The leaky ReLU function allows small positive gradients when the input is negative; however, it is not differentiable at zero. The ELU function allows small positive gradients when the input is negative; also, it is differentiable at zero. 22

4-1. *pydca* modular architecture. The software is composed of sub-packages, each encapsulating a specific task. The current version of *pydca* has sub-packages for pseudo-likelihood and mean-field approximation DCA computation algorithms and sequence back-mapping, MSA trimming, and visualization sub-packages. 26

4-2. Figure showing a snapshot of logging messages while executing *pydca* from the command line. Each logging message begins with a logging level (e.g., INFO). Then follows the date and time that message was displayed, the name of the module where the logging message originated, and the function (method) that emitted the logging message. Beneath this line is the logging message, e.g., the message "computing single site frequencies." 36

5-1. Figure showing the percentage of nucleotide pairs in the dataset. C-G and A-G pairs are the most prevalent pairs, each representing 15% of the entire nucleotide pairs. The least abundant pair is U-U representing only 4% of the nucleotide pairs in the dataset. 41

5-2. Figure showing the average number of contacts in the RNA dataset PDB structure as a function of contacts' distance. The top represents all contact types, i.e., tertiary contacts or otherwise, and the bottom represents only tertiary contacts. At $r_c = 10$ Å there are on average about half tertiary contacts compared to the entire contacts in the dataset. 42

5-3. Figure comparing MSA methods performance on mean-field DCA. The vertical axes represent average PPV values, and the horizontal axes represent ranks at which the PPVs are taken. In the topmost, middle and bottommost figures the averages are performed over datasets $\mathcal{D}$—all RNA; $\mathcal{D}^H$—RNA with $M_{eff} > 70.0$; and $\mathcal{D}^L$—RNA with $M_{eff} \leq 70.0$, respectively. In all dataset categories, MSA obtained using Infernal results in best PPV across all ranks. 43

5-4. Dependence of nucleotide contact positive predictive value $\langle PPV \rangle$ on effective number of sequences (M_{eff}) and BIT score. The averages are taken through the four DCA-based algorithms implemented in pydca, mfDCA; EVCouplings, plmDCA; Boltzmann learning; and PSICOV, graphical LASSO. A) $\langle PPV \rangle_{algos}$ vs M_{eff}:green RNAs whose BIT score is– greater than 50.0, and red less than or equal to 50.0. B) $\langle PPV \rangle_{algos}$ vs BIT score:green RNAs whose M_{eff} is– greater than 70.0, and red less than or equal to 70.0. 45

5-5. The relative percentage of correctly predicted nucleotide pairs computed at rank L of each RNA family. In all of the four DCA algorithms, secondary structure base pairs dominate the fraction of correctly predicted base pairs– canonical base pairs accounting for about half of the dataset's total predictions. 47

5-6. Average positive predictive values by nucleotide pair type computed using the four DCA algorithms. Top—the PPV is computed at rank L, and bottom—the PPV is computed at $2L$. In both cases, secondary structure pairs A-U and G-C are more accurately predicted by the algorithms than other nucleotide pairs. 49

5-7. Average PPV computed using the four DCA algorithms as a function of RNA sequence length L. In the topmost Figure, PPVs are averaged over all RNAs considered in this study; the middle Figure is only for RNAs whose effective number of sequences greater than 70.0; and in the bottom Figure are RNAs whose effective number of sequences less than or equal to 70.0. There is a negligible difference between mean-field pydca, EVCouplings, and Boltzmann learning DCA algorithms, while PSICOV performs slightly less accurate than the other three. 51

5-8. Average positive predictive values $\langle PPV \rangle$ at rank L of the DCA algorithms as a function of contact distance r_c measured in Angstrom. The averages are taken over all RNAs in the dataset $\mathcal{D}$. All four algorithms show they improve $\langle PPV \rangle$ as r_c is increased. Below 10Å the algorithms except PSICOV show comparable performances, although EVCouplings is slightly more accurate than mean-field pydca and Boltzmann learning. 53

5-9. Contact prediction accuracy comparison of mean-field DCA with other algorithms. The PPVs are computed at rank L, where L is the RNA sequence length of each RNA in dataset $\mathcal{D}$. The horizontal axes are values of mean-field pydca PPV, whereas the vertical axes those of EVCouplings (A); Boltzmann learning (B); PSICOV (C); and pseudo-likelihood DCA of pydca (D). The diagonal lines indicate the values that the horizontal and vertical axes are equal. When there is no difference in PPV between mean-field pydca and other DCA algorithms, the PPVs lie on the diagonals. Note that, in this comparison, contacts are taken without making a distinction between tertiary contacts or otherwise. 55

5-10.Tertiary contacts average positive predictive values $\langle PPV \rangle$ as a function of rank measured in terms of sequence's length L. Topmost, middle and bottom plots are for datasets $\mathcal{D}$, $\mathcal{D}^H$, and $\mathcal{D}^L$, respectively. At higher ranks, EVCouplings and Boltzmann learning perform better than mean-field pydca and PSICOV, particularly for dataset $\mathcal{D}^H$. 57

5-11.Tertiary contacts average positive predictive values at rank L as a function of contact distance cut-off value r_c. At $r_c = 4, 6, 8$ and 10Å, mean-field pydca, EVCouplings, and Boltzmann learning perform very closely, whereas PSICOV Is slightly less accurate. Above 10Å there are significant differences between the algorithm's performance; however, these values are too large to be considered contact distances. 59

5-12.Contact prediction accuracy comparison of mean-field DCA with other algorithms for tertiary contacts. The PPVs are computed at rank L, where L is the RNA sequence length of the RNAs. The results show that the algorithms can have significant PPV variation for individual RNAs. 60

6-1. Patterns of average DCA scores in a 7×7 window for a range of distances, r. (A) $r \leq 4.0$ Å; (B) $4.0 < r \leq 10.0$ Å; (C) $r > 10.0$ Å. Brighter patterns show strong DCA scores and darker ones show weak DCA scores. 64

6-2. Network architecture of **CoCoNet**. There are four main layers of the network: i) the input layer containing input MSA data; ii) the coevolution layer involving DCA contact maps; iii) the convolution layer that results from doing convolution on the coevolution layer; iv) the output layer that contains contact maps of nucleotide pairs with high scores. 65

6-3. Average positive predictive values, $\langle PPV \rangle$, as a function of the number of contacts for (A) all the 57 RNAs; (B) RNAs with $M_{eff} > 70.0$; and (C) RNAs with $M_{eff} \leq 70.0$. In all of the three dataset categories, **CoCoNet** significantly outperforms mean-field DCA as the average PPV curves of **CoCoNet** are closer to the theoretical curve than mean-field DCA's cures are. 67

6-4. Tertiary contacts average positive predictive values, $\langle PPV \rangle$, as a function of the number of contacts (rank) for (A) all the 57 RNAs; (B) RNAs with $M_{eff} > 70.0$; and (C) RNAs with $M_{eff} \leq 70.0$. For the highest-ranked nucleotide pairs, mean-field DCA shows slightly better performance than **CoCoNet**; however, **CoCoNet** overtakes mean-field DCA below rank 10. 68

6-5. Average MCC at rank L for all contacts types. The averages are performed over datasets (i) all RNAs, $\mathcal{D}$, top Figure (ii) RNAs whose $M_{eff} > 70.0$, $\mathcal{D}^H$, middle Figure and (iii) RNAs whose $M_{eff} \leq 70.0$, $\mathcal{D}^L$, bottom Figure. The MCC values of **CoCoNet** are significantly higher than those of state-of-the-art DCA algorithms, showing twice as much MCC as DCA for dataset $\mathcal{D}$. 73

6-6. Average MCC at rank L for tertiary only contacts. The averages are performed over datasets (i) all RNAs, $\mathcal{D}$, top Figure (ii)RNAs whose $M_{eff} > 70.0$,$\mathcal{D}^H$, middle Figure and (iii) RNAs whose $M_{eff} \leq 70.0$, $\mathcal{D}^L$, bottom Figure. The MCC values of **CoCoNet** are significantly higher than those of state-of-the-art DCA algorithms, showing twice as much MCC as DCA for dataset $\mathcal{D}$. 74

7-1. Figure showing the workflow diagram for RNA 3D structure prediction using co-evolutionary information obtained by the CoCoNet algorithm and the 3D modeling done using the SimRNA RNA modeling software. In this workflow, nucleotide pair ranking is done by computing the co-evolutionary interaction scores using the CoCoNet algorithm that uses multiple sequence alignment of a target sequence with homologous sequences in the Rfam database. Then, top L tertiary putative contacts computed relative to a secondary structure are used as restraints in the SimRNA software. 78

7-2. Figure showing the alignment of predicted structures(in red) with the corresponding structures obtained from the PDB database(in green). The predicted structures are obtained using top L tertiary restraints of CoCoNet 3×3 algorithm plus consensus secondary structure from the Rfam database. The structures are rendered using the open-source version of PyMol[86]. No atom is rejected while aligning the structures. 81

List of Tables

5-1. Table showing the numerical value of average positive predictive values $\langle PPV \rangle$ as a function of rank expressed in terms of RNA sequence length L. The averages are done over the three categories of datasets—$\mathcal{D}$, $\mathcal{D}^H$, and $\mathcal{D}^L$—that are grouped based on the number of effective sequences M_{eff}. Across all ranks, all of the four DCA algorithms perform better in dataset D^H. At rank, L mean-field pydca, EVCouplings, and Boltzmann learning show comparable performance. PSICOV is the least accurate method by up to about 5% than the other three. 52

5-2. Table showing the average positive predictive values $\langle PPV \rangle$ as a function of rank expressed in terms of RNA sequence length L. The averages are done over the three categories of datasets—$\mathcal{D}$, $\mathcal{D}^H$, and $\mathcal{D}^L$—that are grouped based on the number of effective sequences M_{eff}. Across all ranks, all of the four DCA algorithms perform better in dataset D^H. At rank, L mean-field pydca, EVCouplings, and Boltzmann learning show comparable performance. PSICOV is the least accurate method by up to about 5% than the other three. 58

6-1. Average positive predicted value, $\langle PPV \rangle$, at rank L for all RNAs in the dataset $\mathcal{D}$. The first two columns indicate the number and size of filter matrices used, respectively. The third column corresponds to the number of free parameters to learn. The fourth and last columns show $\langle PPV \rangle$ at rank L for all and tertiary contacts, respectively. The bottom five rows represent the $\langle PPV \rangle$ of DCA algorithms including, (i) mean-field DCA implemented in pydca, mfDCA-pydca; (ii) pseudo-likelihood maximization of EVCouplings, plmDCA-EVC; (iii) pseudo-likelihood maximization of pydca, plmDCA-pydca; (iv) Boltzmann learning and (v) the graphical LASSO algorithm implemented in PSICOV. Columns four and five represent $\langle PPV \rangle$s for all contacts and tertiary only contacts, respectively. The $\langle PPV \rangle$s show that CoCoNet significantly outperforms state-of-the-art DCA algorithms. 70

6-2. Average positive predictive value, $\langle PPV \rangle$, at rank L when the dataset is split into two categories—$\mathcal{D}^H$ and $\mathcal{D}^L$ based on the effective number of sequences. The first two columns indicate the number and size of filter matrices used, respectively. The third and fifth columns represent $\langle PPV \rangle$s for all contact types, whereas the fourth and sixth columns represent tertiary only contacts $\langle PPV \rangle$s. The bottom five rows represent the $\langle PPV \rangle$ of DCA algorithms including, (i) mean-field DCA implemented in pydca, mfDCA-pydca; (ii) pseudo-likelihood maximization of EVCouplings, plmDCA-EVC; (iii) pseudo-likelihood maximization of pydca, plmDCA-pydca; (iv) Boltzmann learning and (v) the graphical LASSO algorithm implemented in PSICOV. The $\langle PPV \rangle$s show that CoCoNet significantly outperforms state-of-the-art DCA algorithms. 71

7-1. The root-mean-squared deviation (RMSD) of simulated RNA 3D structures obtaining by aligning the simulated structures with their experimental structures taken from the PDB database. The first and second columns label PDB IDs and family names of the RNAs, respectively. The third, fourth, fifth, and sixth columns contain the average/minimum RMSD values when simulations are done using the SimRNA modeling software using (i) the sequence (ii) sequence plus secondary structure (labeled on the table as Secondary struc.) (iii) sequence plus secondary structure plus mfDCA tertiary contacts and (iv) sequence plus secondary structure plus CoCoNet 3×3 tertiary contacts. The number of tertiary contacts used is L, where L is the RNA length. 80

A-1. Table showing the dataset of RNA used in our analysis. The second column is the RNA type. The third column contains the family in Rfam the RNA belongs. The fourth column refers to the PDB code in the PDB database. The fifth is the resolution (in Å) of the PDB structure. The sixth is column contains the sequence length of the RNA. The seventh is the BIT score for multiple sequence alignment, and the last column is the number of effective sequences in the family. 86

A-2. Table showing the RNA dataset with the number of PDB contacts annotated for contact distance cut-off value 10Å. 88

1. Introduction to Ribonucleic Acids

Ribonucleic acid (RNA) plays key roles within cells. It is involved in coding, decoding, expression and regulation of genes[91, 93, 59, 58, 63]. For example, messenger RNA (mRNA) carries genetic information coded from DNA; transfer RNA (tRNA) is involved in transferring residues during translation, and ribosomal RNA (rRNA) is involved in gene expression involving protein synthesis. Most eukaryotic genes although are not translated to proteins, the resulting non-coding RNAs are attributed to crutial functions[14, 68, 73, 7, 31]. There exist many non-coding RNAs whose functions are still to be understood[81, 25]. In the recent Covid-19 crisis, RNA is used in vaccine development[39, 69]

The function of RNA is usually related to its structure[3, 70]. It exhibits hierarchies of structures varying complexity. In this chapter, a brief introduction of RNA structures is presented. The chapter is organized as follows. First various levels of RNA structure are briefly highlighted. Then experimental and theoretical methods of RNA structure prediction are presented before summarizing the chapter.

1.1. RNA structure

There are various hierarchical levels of the structure of RNA molecules. The simplest of all the structures is the RNA sequence, a string of basic RNA units called nucleotides composed of a pentose ribose sugar, phosphate group, and nucleobases. There are five standard nucleobases—adenine(A), cytosine(C), guanine(G), uracil(U) and thymine(T). Uracil is found in RNAs, and thymine is a DNA nucleobase. The nucleobases are categorized into two as purines and pyrimidines (see Fig. **1-1**). Adenine and guanine are purines, whereas cytosine, uracil, and thymine are pyrimidines.

Nucleotides are formed from when a nucleobase forms a glycosidic bond with the ribose sugar forming a nucleoside. When a phosphate group is attached with the ribose sugar through a condensation reaction, a nucleotide is created(see Fig. **1-2**). The ribose sugar's carbons are numbered from 1' to 5' to distinguish them from those in the nucleobases. In pyrimidines nucleosides, the glycosidic bonding occurs between the C1' carbon of the ribose sugar and the N1 of the nucleobases. The glycosidic bonding in the case of purines is between the C1' atom and the N9 atom. The phosphate group attaches to the ribose sugar at the C5' atom. The phosphate group and the ribose sugar together form the backbone of an RNA chain.

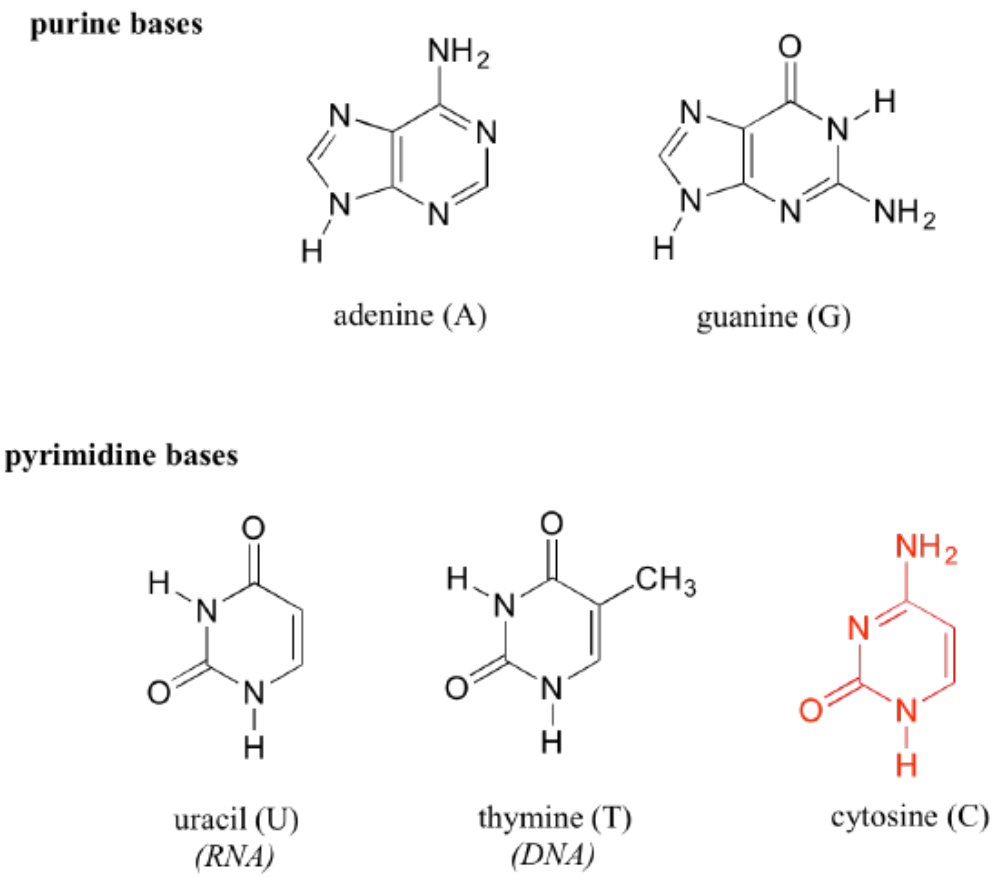
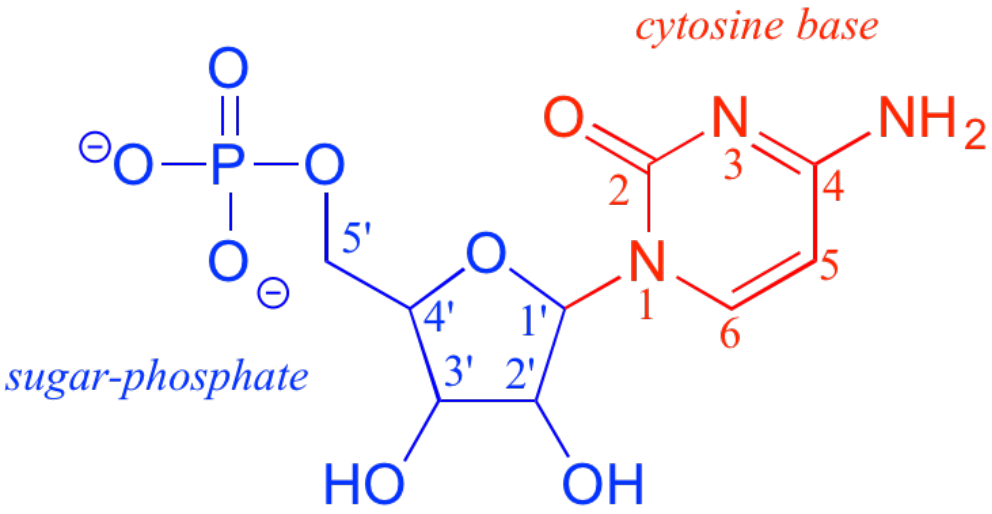


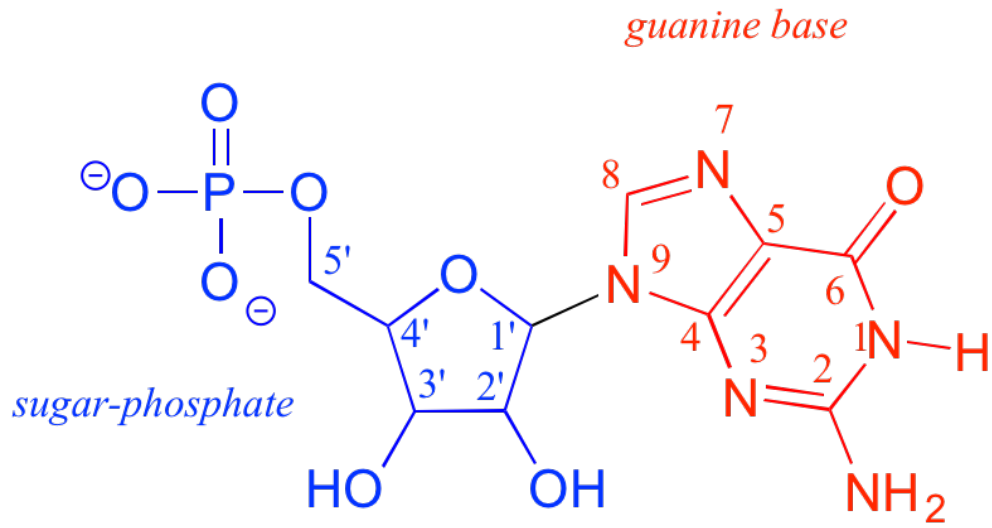
Figure **1-1**.: Chemical structure of the five nucleobases. Adenine(A) and guanine(G) are purine bases. Uracil(U), thymine(T), and cytosine(C) are pyrimidine bases[92].

An RNA sequence is a polymeric chain of nucleotides. The directionality of the sequence from the C5' to the C3' of the ribose sugar, written as "5'-to-3'". Although RNA is naturally a single-stranded chain of nucleotides, it can fold onto itself and form a two-dimensional structure as in Fig.**1-3**A. The two-dimensional structure, often known as the secondary structure, can have stems and loops. An RNA secondary structure stem is typically formed from a base-pairing of pyrimidine with a purine. For instance, an adenine can be paired with uracil and guanine with cytosine. These are called canonical base-pairs. However, a non-canonical base-pairing can be formed, e.g., between guanine and uracil[54].

The three-dimensional structure of an RNA, also known as the tertiary structure, is represented by the atom's position in the molecule. Fig.**1-3**B shows a cartoon visualization of the adenine riboswitch RNA whose tertiary structure is experimentally solved and deposited in the PDB database with PDB id 4TZX. Although being non-coding, this RNA is known to play roles in the translational machine during gene expression. The backbone of the riboswitch is represented by loops and the side chains by sticks. While the adenine riboswitch's secondary structure looks like a simple Y-shape, the tertiary structure is folded in a complex



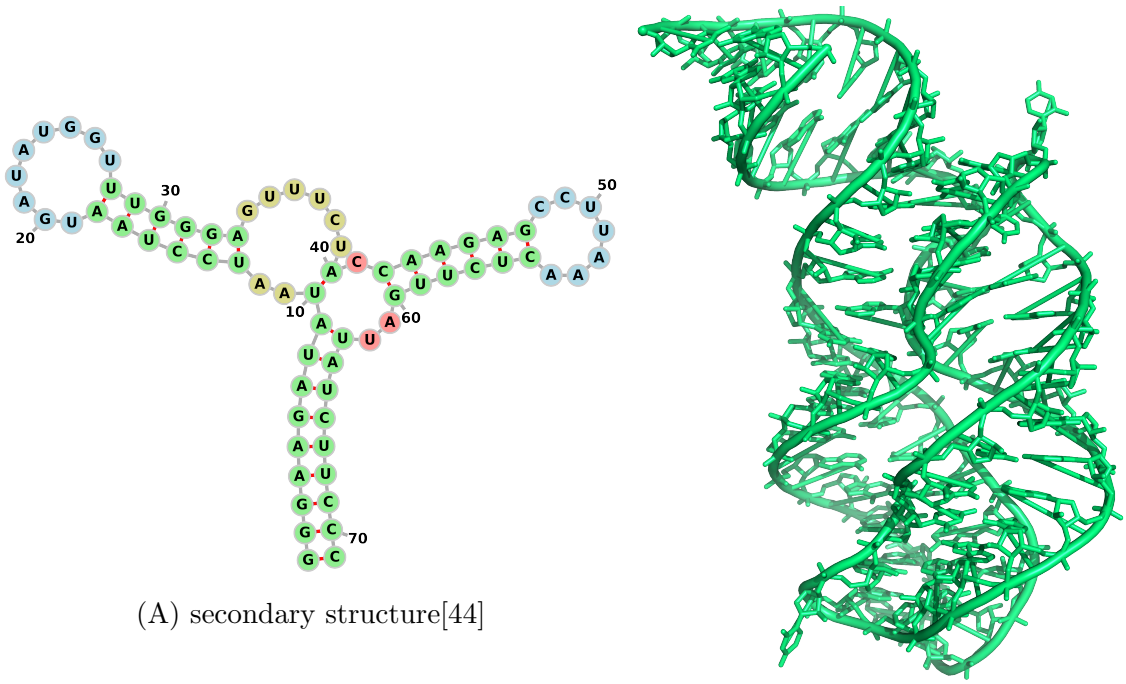
(A) Chemical structure of cytidine nucleotide



(B) Chemical structure of guanosine nucleotide

Figure **1-2**.: Chemical structures of cytidine, and guasonine nucleotides[92]. The nucleotides are formed chemical bond between the C1' atom of the ribose sugar with N1 atom of pyrimidines or the N9 atom of purines. The ribose sugar is also connected to the phosphate group by gylcosidic covalent bonding.

way. Indeed, the tertiary structure of RNAs is massively challenging to resolve[55]. The backbone formed from the ribose sugar and phosphate group is highly flexible, allowing a myriad of possible three-dimensional structures for an RNA to assume.



(A) secondary structure[44]

(B) tertiary structure

Figure **1-3**.: Figure showing the secondary and tertiary structures of the adenine riboswitch RNA (PDB id 4tzx). The secondary structure contains three stems, two loops, and two junctions. A cartoon representation of the tertiary structure shows the backbone in thick loop paths, whereas sticks represent the side chains.

1.2. RNA structure determination methods

There are experimental and theoretical ways of determining the structure of RNAs. Experimental methods of RNA structure determination include X-ray crystallography, nuclear magnetic resonance (NMR) spectroscopy, small-angle scattering (SAS), and fluorescence resonance energy transfer (FRET)[15, 13, 56, 98]. These methods provide structures of varying resolution. For instance, X-ray methods can provide atomic resolution of RNA structures, but they are often very challenging techniques where are SAS can be an easier method, but at the expense of resolution as it provides structure at nanometer-level resolution.

Complementary approaches of RNA structure determination involves molecular modeling technique such as Monte-Carlo methods and molecular dynamics methods. Often, these theoretical methods incorporate restraints obtained from chemical mapping experiments or theoretical methods such as statistical and machine learning methods[23, 104]. Methods such as direct-coupling analysis (DCA) aim to predict pairs of nucleotides spatially proximal within the tertiary structure of RNA from multiple sequence alignment (MSA) of homologous sequences.

Experimental RNA tertiary structure is arduous. For instance, looking at the Rfam ver-

sion 14.5 database, 3,825 families have no annotated tertiary structure, whereas only 115 do have [42]. Theoretical methods using computer simulations have shed light on improving RNA structure prediction, mainly when done in tandem with restraints obtained from chemical probing experiments or coevolutionary approaches. However, RNA structure prediction's success using these methods lags compared to the field of protein structure prediction. This thesis aims to improve RNA contact prediction from MSA data and provide a workflow of tertiary structure modeling guided by information about spatial proximity of nucleotide pairs.

1.3. Summary

Ribonucleic acids (RNA) are polymeric biomolecules composed of basic units called nucleotides. Nucleotides have three main components—a nitrogenous nucleobase, a ribose sugar, and a phosphate group. There are five nucleobases, adenine, cytosine, guanine uracil, and thymine. Thymine is found in DNA, whereas uracil is found in RNA. Nucleobases are categorized into two—purines and pyrimidines. Adenine and guanine are purines, and cytosine, uracil, and thymine are pyrimidines.

The sequence of nucleotides represents the simplest structure of RNA. The nucleotides are attached through the phosphate group and the ribose sugar. The carbon atoms in the ribose sugar are numbered from 1' to 5' to distinguish them from other carbon atoms. An RNA sequence is read from the 5' to the 3' end. RNA can also fold onto itself to form a secondary structure. The secondary structure is typically composed of loops, stems, and junctions. The three-dimension (3D) structure of an RNA is represented by the coordination of all atoms in the molecule. RNA can form a myriad of complex 3D structures.

There are experimental and theoretical methods of RNA structure determination. Experimental methods include X-ray crystallography, nuclear magnetic resonance(NMR), fluorescence resonance energy transfer(FRET), and small-angle scattering(SAS). These methods provide information about RNA structure of varying resolution. Theoretical methods include molecular dynamics and Monte-Carlos simulations. These methods benefit from information about spatial proximity between nucleotide pairs as constraints.

2. Probabilistic Models of Contact Prediction

2.1. Introduction

Biomolecules such as proteins and RNA change their sequence while preserving their three-dimensional structure. During the course of evolution, residues/nucleotides of protein/RNA can mutate and result in a new sequence of a protein or an RNA. However, to preserve the biomolecule structure and function, mutations may occur in tandem instead of separately. Mutation occurring at one place in the sequence may trigger another to ensure the correct functioning of a biomolecule[5, 4].

Statistical models of contact prediction presume that homologous sequences collected in protein or RNA families contain information about coevolved residue/nucleotide pairs (see Fig. **2-1**). However, a correlation between pairs may occur as a result of direct or indirect effects. For instance, consider three residues/nucleotide A, B and C. Suppose that B is in direct contact with both A and C in the three-dimensional structure, whereas A and C are not. While the correlation B with A or C is a result of direct effect, A and C may also show a strong correlation albeit as a result of indirect contact propagated through B.

Contact prediction using methods like mutual information (MI) relies on the correlation between pairs of residues/nucleotides without considering the cause, i.e., whether it is as a result of direct or indirect effect. To see this, consider two sites i and j in a protein/RNA sequence. The mutual information between these two sites is defined as[20]

$$MI_{ij} = \sum_{a} \sum_{b} f_{ij}(a,b) \log \frac{f_{ij}(a,b)}{f_i(a) f_j(b)} \tag{2-1}$$

where a and b are residues/nucleotide at sites i and j, respectively; f_{ij} and f_i are pair- and single-site frequency counts. Eq.2-1 quantified mutual information between the two sites i and j without considering the effect of other sites in a sequence—it is a local measure, not a global one.

In recent years, contact prediction algorithms that take into account not only quantifying correlation between pairs of residues/nucleotide but also identify its source—direct or indirect—are introduced. These algorithms collectively known as direct coupling analysis (DCA) methods are shown to be more accurate than local measures such as MI and have showed tremendous success in the field or protein structure prediction[101, 61, 62, 37, 97,

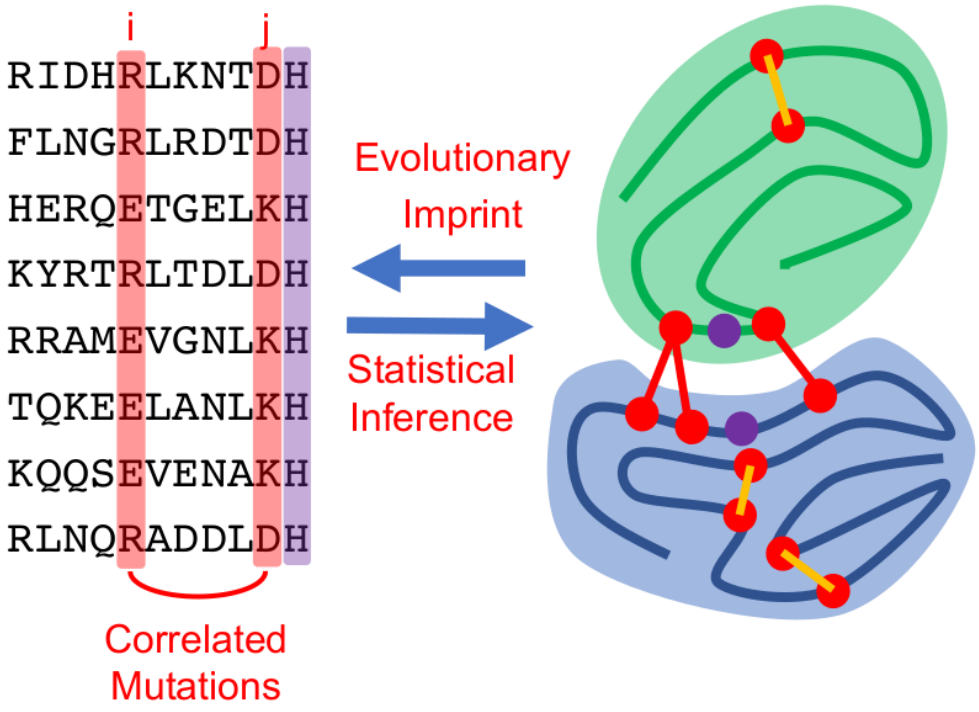


Figure **2-1**.: Mutations in protein/RNA sequences through evolution leave traces of correlated mutations between residues/nucleotides. These coordinated mutations are necessary to preserve a protein/RNA molecule structure while functionally important sites remain conserved. Statistical methods are used to infer coevolving site-pairs such as i and j that are in proximity within the three-dimensional structure of proteins/RNAs. Functionally essential sites may stay conserved during coevolution.

35, 32, 88, 22, 29, 99, 94, 65, 67, 17, 85, 8, 16, 76], and recently started being utilized in the corresponding field of RNA structure prediction[23, 104].

In this chapter, popular DCA algorithms are reviewed. The chapter is organized as follows: In Section 2.2 I highlight the global probability model's derivation by maximizing entropy subject to some constraints. Next, review approximate algorithms for solving the global probability distribution. In particular, mean-field DCA[66]—obtained by using a mean-field approximation in Section2.3; pseudo-likelihood DCA[27]—obtained by using a pseudo-likelihood instead of the full-likelihood in Section 2.4; Boltzmann learning[21]—that uses the full-likelihood of the global probability method in Section 2.6. I also review the DCA algorithm that is derived from multi-variable Gaussian distribution[10] in Section 2.5.

2.2. Global probability model

Maximizing entropy subject to constraints

Here, an expression for a probability function is obtained using the principle of maximum entropy. In the absence of constraints, one expects that each state is equally likely, and the result of maximizing entropy would be a uniform probability for each state. However, the introduction of constraints skews the uniform probability, and some states will be more likely than others.

In the context of contact prediction from biological sequences, two constraints, in particular, are introduced. The first constraint considers the effect of local-states (residues/nucleotide) at a site in the sequence. Second, each residue/nucleotide at a site can interact with every other residue/nucleotide. In other words, the global probability function should be able to reproduce single-site and pair-site marginal probabilities. Besides, each probability must be normalized to unity when summed over all possible outcomes.

Consider a sequence $\sigma = a_1 a_2 ... a_L$ of length L in multiple sequence alignment data of homologous sequences where a_i is a residue or gap at site i. Let $p(\sigma)$ denote the probability that this sequence to be sampled during the course of evolution. The entropy as a functional of p is given by[60]

$$s[p] = -\sum_{\{a_k\}} p(a_1, a_2, ..., a_L) \ln p(a_1, a_2, ..., a_L). \tag{2-2}$$

Single- and pair-site marginals, respectively, are obtained as

$$\sum_{\{a_k | k \neq i\}} p(a_1, a_2, ..., a_L) = p_i(a_i), \tag{2-3}$$

and

$$\sum_{\{a_k | k \neq i,j\}} p(a_1, a_2, ..., a_L) = p_{ij}(a_i, a_j). \tag{2-4}$$

In addition, normalization condition is given by

$$\sum_{\{a_k\}} p(a_1, a_2, ..., a_L) = 1. \tag{2-5}$$

Note that Eq.2-3 represents Lq equations—q is the number of residues plus gap state—since i runs from one to L and a_i runs from one to q. Likewise, Eq.2-4 is a short-hand notation of $\frac{1}{2}L(L-1)q^2$ equations since there are $\frac{1}{2}L(L-1)$ non-redundant site-pairs and there are q^2 ways of choosing residue/nucleotide pairs at site-pairs.

Eqn.2-2 is maximized subject to the constraints in Equations 2-3, 2-4 and 2-5. Introducing Lagrange's multipliers and combining similar constraints together results in

$$\sum_{i=1}^{L} \sum_{a_i} \left(\sum_{\{a_k | k \neq i\}} h_i(a_i) p(a_1, a_2, ..., a_L) - c_1 \right) = 0, \tag{2-6}$$

$$\sum_{i=1}^{L-1}\sum_{j=i+1}^{L}\sum_{a_i}\sum_{a_j}\left(\sum_{\{a_k|k\neq i,j\}} J_{ij}(a_i,a_j)p(a_1,a_2,...,a_L) - c_2\right) = 0, \tag{2-7}$$

$$\lambda\left(\sum_{\{a_k\}} p(a_1,a_2,...,a_L) - 1\right) = 0, \tag{2-8}$$

where $h_i(a_i)$ and $J_{ij}(a_i,a_j)$ are Lagrange's multipliers and c_1 and c_2 are constants.

The Lagrangian is obtained by adding Equations 2-6, 2-7 and 2-8 to Equation 2-2.

$$\begin{aligned}\mathcal{L}[p] = &-\sum_{\{a_k\}} p(a_1,a_2,...,a_L)\ln p(a_1,a_2,...,a_L) + \\ &\sum_{i=1}^{L}\sum_{a_i}\left(\sum_{\{a_k|k\neq i\}} h_i(a_i)p(a_1,a_2,...,a_L) - c_1\right) + \\ &\sum_{i=1}^{L-1}\sum_{j=i+1}^{L}\sum_{a_i}\sum_{a_j}\left(\sum_{\{a_k|k\neq i,j\}} J_{ij}(a_i,a_j)p(a_1,a_2,...,a_L) - c_2\right) + \\ &\lambda\left(\sum_{\{a_k\}} p(a_1,a_2,...,a_L) - 1\right)\end{aligned} \tag{2-9}$$

The variation of $\mathcal{L}$ with respect to p is

$$\frac{\delta\mathcal{L}}{\delta p} = -\ln p(a_1,a_2,...,a_L) - 1 - \lambda + \sum_{i=1}^{L} h_i(a_i) + \sum_{i=1}^{L-1}\sum_{j=i+1}^{L} J_{ij}(a_i,a_j). \tag{2-10}$$

The entropy is maximized for values of probabilities such that the variation in Eq. 2-10 is zero. This results in the expression for the probability given by

$$p(a_1,a_2,...,a_L) = \frac{1}{Z}\exp\left(\sum_{i=1}^{L-1}\sum_{j=i+1}^{L} J_{ij}(a_i,a_j) + \sum_{i=1}^{L} h_i(a_i)\right), \tag{2-11}$$

where $Z = (\exp(\lambda-1))^{-1}$ is the normalization constant, also known as partition function in statistical physics. The Lagrange multipliers, $h_i(a_i)$ and $J_{ij}(a_i,a_j)$, are known as fields and couplings, respectively. Comparing with Boltzmann distribution, i.e., $p(a_1,a_2,...,a_L) \propto \exp(-\beta H(a_1,a_2,...,a_L))$—where β is the inverse temperature—the energy function corresponding to Eq. 2-11 is given by

$$\beta H = -\sum_{i=1}^{L-1}\sum_{j=i+1}^{L} J_{ij}(a_i,a_j) - \sum_{i=1}^{L} h_i(a_i). \tag{2-12}$$

Eq.2-11 is also called the global probability function because, unlike mutual information given in Eq.2-1, it relates each site in a sequence with every other through the energy function.

Over-parametrization and gauge fixing

In Eq.2-11 the fields and couplings are functions of residue or gap states of an RNA or protein sequence. The total number of residues plus a gap is denoted by q. There are four standard nucleotides in RNAs ($q = 5$) and 20 standard residues for proteins ($q = 21$). For a sequence of length L, there are Lq fields and $\frac{1}{2}L(L-1)q^2$ couplings. As a result, the total number of parameters is $Lq + \frac{1}{2}L(L-1)q^2$. However, the model has fewer equations relating to these parameters and is over-parameterized. That is, there is no unique set of parameters that define equation 2-11.

To find out the total number of unique parameters, One needs to look at the constraints (equations 2-3 and 2-4) a bit closely. The single-site probabilities $p_i(a_i)$ can be normalized by summing over a_i fore every site i, i.e., $\sum_{a_i} p_i(a_i) = 1$. As a results there are $Lq - L = L(q-1)$ unique parameters. Similarly, considering Eq. 2-4, single-site marginals can be obtained by summing over either a_i or a_j. Furthermore, the single-site marginals can be normalized. Summing over either a_i or a_j results in $\frac{1}{2}L(L-1)q$ equations relating the parameters. Therefore, there are $L(L-1)q$ equations relating the parameters; however, again each single-site probability can be normalized. It implies that, $L(L-1)$ equations are actually redundant. As a consequence, the total number of unique parameters that can be obtained from Eq.2-4 is $\frac{1}{2}L(L-1)q^2 - [L(L-1)q - L(L-1)]$ which can be written as $\frac{1}{2}L(L-1)(q-1)^2$.

Over-parametrization of the model is circumvented using the so called gauge fixing. DCA algorithms that estimate the fields and couplings for biological sequences do the gauge fixation by setting the fields and coupling involving gap states to be zero. Numbering standard residues/nucleotides 1, 2, ..., $q-1$ and the gap state by q the fields and couplings involving q become $h_i(q) = J_{ij}(a_i, q) = J_{i,j}(q, a_j) = 0$. This is also known as the lattice-gas gauge. Another approch is to use the zero-sum gauge as $\sum_{a_i} h_i(a_i) = \sum_{a_i} J_{ij}(a_i, q) = \sum_{a_j} J_{ij}(q, a_j) = 0$[103, 66].

In the context of biomolecular coevolution, the model is utilized to predict pairs of coevolving residues/nucleotides. The fields and couplings are estimated from input sequence data as sampled by nature through evolution. However, fitting the global probability model as it stands is computationally costly. Computing the partition function scales as $\mathcal{O}(q^L)$. For instance, for a protein of length $L = 100$, the algorithmic complexity is $\mathcal{O}(10^{100})$ which is a formidable task for the computing capability of current computes. Consequently, inverse statistical approaches in the context of protein/RNA sequence data use approximation methods to estimate the fields and couplings.

A few algorithms that approximately estimate the fields and couplings have been proposed and utilized. The message passing algorithm was successfully used for predicting protein-protein interactions[103]. Latter, an efficient algorithm based on the mean-field approximation was introduced[66]. This algorithm, called mean-field direct coupling analysis (mfDCA), is widely used because of its efficiency. Simultaneously, a method based on a mul-

tivariate Gaussian model was introduced[40]. Then, another algorithm based on maximum likelihood is shown to improve residue-residue contact prediction capability for proteins[27]. The derivation of these algorithms is reviewed in the following sections. The mathematical details are found in Appendix B.

2.3. Mean-field approximation

The mean-field approximation DCA algorithm estimates couplings by inverting the covariance matrix computed from MSA data. Here, expressions for the mean-field DCA parameters—fileds and couplings—are derived. The derivations are based on[66]. The fields are calculated using a self-consistent equation. The starting point is to express the energy function of the global probability model as function of a parameter that can be varied. Then, this parameter is chosen to be such that the interactions via the couplings are small. The expression for this new energy function becomes

$$\mathcal{H}_\alpha(\{J_{ij}(a_i,a_j)\},\{h_i(a_i)\};\alpha) = \alpha\sum_{i=1}^{L}\sum_{j=i+1}^{L} J_{ij}(a_i,a_j) + \sum_{i=1}^{L} h_i(a_i). \tag{2-13}$$

And the corresponding partition function

$$\mathcal{Z}_\alpha = \sum_{\{a_i\}} \exp(\mathcal{H}_\alpha), \tag{2-14}$$

whereas the Helmholtz free energy is

$$\mathcal{F}_\alpha = -\ln \mathcal{Z}_\alpha. \tag{2-15}$$

The next step is to obtain Gibbs free energy from Helmholtz free energy. This task is achieved by performing Legendre transformation of the Helmholtz free energy. The Helmholtz free energy depends on fields and couplings in addition to the parameter α. That is $\mathcal{F}_\alpha = \mathcal{F}_\alpha(\{J_{ij}(a_i,a_j)\},\{h_i(a_i)\};\alpha)$ The Legendre transformation is performed for the fields as.

$$\mathcal{G}_\alpha = \mathcal{F}_\alpha - \sum_{i=1}^{L}\sum_{a_i=1}^{q-1} h_i(a_i)\frac{\partial \mathcal{F}_\alpha}{\partial h_i(a_i)}. \tag{2-16}$$

In equation 2-16, the inner summation is up to $q-1$ to indicate that gap states are not included due to gauge fixation. Not also that $\mathcal{G}_\alpha$ depends on the couplings, $\{\frac{\partial \mathcal{F}_\alpha}{\partial h_i(a_i)}\}$ and α. That is $\mathcal{G}_\alpha = \mathcal{G}_\alpha(\{J_{ij}(a_i,a_j)\},\{\frac{\partial \mathcal{F}_\alpha}{\partial h_i(a_i)}\};\alpha)$. Using the results of Appendix B, One has $\frac{\partial \mathcal{F}_\alpha}{\partial h_i(a_i)} = -p_i(a_i)$. The Gibbs free energy then takes the form

$$\mathcal{G}_\alpha(\{J_{ij}(a_i,a_j)\},\{p_i(a_i)\};\alpha) = \mathcal{F}_\alpha + \sum_{i=1}^{L}\sum_{ai=1}^{q-1} h_i(a_i)p_i(a_i). \tag{2-17}$$

Performing Taylor expansion of $\mathcal{G}_\alpha$ and approximating it up to the linear order in α for small couplings the Gibbs free energy becomes,

$$\mathcal{G}_\alpha = \mathcal{G}(0) + \frac{\partial \mathcal{G}_\alpha}{\partial \alpha}\Big|_{\alpha=0} \alpha + \mathcal{O}(\alpha^2). \tag{2-18}$$

At $\alpha = 0$, the Hamiltonian $\mathcal{H}_\alpha$ has terms only involving singe-site fields. In addition the Legendre transformation has removed the contribution of internal energy via fields—the Gibbs free energy is a functional of couplings and single-site probabilities. This implies that the first term in equation 2-18 is equivalent to the contribution from the (negative) entropy. For independent sites the entropy $\mathcal{S}_0$ can be written as

$$\mathcal{S}_0 = -\sum_{i=1}^{L}\sum_{a_i=1}^{q} p_i(a_i) \ln p_i(a_i). \tag{2-19}$$

The second term in equation 2-18 has contribution from the Helmholtz free energy which is given by

$$\frac{\partial \mathcal{F}_\alpha}{\partial \alpha}\Big|_{\alpha=0} = -\sum_{i=1}^{L-1}\sum_{j=i+1}^{L}\sum_{a_i=1}^{q-1}\sum_{a_j=1}^{q-1} J_{ij}(a_i, a_j) p_i(a_i) p_j(a_j). \tag{2-20}$$

The Gibbs free energy for small couplings up to linear order can be expressed as

$$\begin{aligned}\mathcal{G}(\{J_{ij}(a_i, a_j)\}, \{p_i(a_i)\}) = &\sum_{i=1}^{L}\sum_{a_i=1}^{q} p_i(a_i) \ln p_i(a_i) \\ &- \sum_{i=1}^{L-1}\sum_{j=i+1}^{L}\sum_{a_i=1}^{q-1}\sum_{a_j=1}^{q-1} J_{ij}(a_i, a_j) p_i(a_i) p_j(a_j).\end{aligned} \tag{2-21}$$

Note that in Eq. 2-21 α is absorbed into the couplings, i.e., it is re-written $\alpha J_{ij}(a_i, a_j)$ simply as $J_{ij}(a_i, a_j)$. Performing the derivative of Gibbs free energy in equation 2-21 with respect to single-site probabilities one obtains a self-consistent equation for the fields

$$\frac{p_i(a_i)}{p_i(q)} = \exp\left(h_i(a_i) + \sum_{(j=1|j\neq i)}^{L}\sum_{a_j=1}^{q-1} J_{ij}(a_i, a_j) p_j(a_j)\right). \tag{2-22}$$

The second derivative $\mathcal{G}$ with respect to single-site probabilities, i.e., $\frac{\partial^2 \mathcal{G}}{\partial p_i(a_i) \partial p_j(a_j)}$ is related to the inverse of the correlations as

$$\frac{\partial^2 \mathcal{G}}{\partial p_i(a_i) \partial p_j(a_j)} = C_{ij}^{-1}(a_i, a_j). \tag{2-23}$$

Computing the second derivatives from equation 2-21 the couplings are related to the correlations as

$$C_{ij}^{-1}(a_i, a_j) = \begin{cases} J_{ij}(a_i, a_j), & j \neq i \\ \frac{\delta_{a_i, b_i}}{p_i(a_i)} - \frac{1}{p_i(q)}, & i = j. \end{cases} \tag{2-24}$$

To sum up, Equations 2-22 and 2-24 form the basic algorithms to estimate fields and couplings, respectively, using mfDCA. Estimation of the fields requires information about single site probabilities, $p_i(a_i)$. In practice, these probabilities are substituted by single-site frequencies obtained from MSA data. Similarly, estimating the couplings requires finding the inverse of the correlation matrix whose elements are $C_{ij}(a_i, a_j) = p_{ij}(a_i, a_j) - p_i(a_i)p_j(a_j)$. Like in the estimation of fields, the single site-probabilities are substituted by single-site frequencies of MSA data. However, pair-site probabilities are also required for computing the correlation matrix elements. Again, the pair-site probabilities are substituted by MSA data frequency counts.

2.4. Pseudo-likelihood maximization

In this section, another popular DCA algorithm based on likelihood maximization is reviewed. The derivations are based on[27]. The pseudo-likelihood direct coupling analysis (plmDCA) estimates fields and couplings by introducing a new energy function—instead of using the full energy function of the global probability—hence, the name pseudo-likelihood. Each site i in a sequence m is assigned a probability independently such that the effect of other sites on i is taken into account through the couplings. The probability that site i to be occupied by a residue/gap a is given by

$$p_i^m(a^m) = \frac{1}{\mathcal{Z}_i^m} \exp\left(h_i(a^m) + \sum_{i=1|i\neq j}^{L} J_{ij}(a^m, a_j^m)\right), \tag{2-25}$$

where $\mathcal{Z}_i^m$ is the normalization constant given by

$$\mathcal{Z}_i^m = \sum_{a^m=1}^{q} \exp\left(h_i(a^m) + \sum_{i=1|i\neq j}^{L} J_{ij}(a^m, a_j^m)\right) \tag{2-26}$$

The new energy function

$$\mathcal{H}_i(h_i(a^m), \{J_{ij}(a^m, a_j^m)\}) = h_i(a^m) + \sum_{i=1|i\neq j}^{L} J_{ij}(a^m, a_j^m) \tag{2-27}$$

contains only a single summation over the couplings which makes it simpler compared to the full Hamiltonian in equation 2-12. In addition, the partition function as given by equation 2-26 contains only a single sum and the computational complexity is significantly reduced compared to the partition function of the global probability model (see Eq. 2-11).

The likelihood is defined over a multiple sequence alignment data of homologous protein-s/RNA. If there are M sequences in the alignment the pseudo-likelihood is

$$l(\{h_i(a)\}, \{J_{ij}(a, a_j)\}) = \prod_{m=1}^{M} \prod_{i=1}^{L} p_i^m(a^m). \tag{2-28}$$

The fields and couplings are obtained by maximizing the logarithm of the pseudo-likelihood in equation 2-28. The log pseudo-likelihood is

$$\log l(\{h_i(a)\},\{J_{ij}(a,a_j)\}) = \sum_{m=1}^{M}\sum_{i=1}^{L}\left(h_i(a^m) + \sum_{i=1|i\neq j}^{L} J_{ij}(a^m,a_j^m) - \ln \mathcal{Z}_i^m\right). \tag{2-29}$$

The fields and couplings are estimated using gradient decent algorithms. The gradients are obtained as

$$\frac{\partial \log l(\{h_i(a)\},\{J_{ij}(a,a_j)\}))}{\partial h_i(a)} = \sum_{m=1}^{M}(\delta_{a^m,a} - p_i^m(a)), \tag{2-30}$$

$$\frac{\partial \log l(\{h_i(a)\},\{J_{ij}(a,a_j)\})}{\partial J_{ij}(a,b)} = \sum_{m=1}^{M}(\delta_{a^m,a} - p_i^m(a))\delta_{a_j^m,b}. \tag{2-31}$$

In the pseudo-likelihood approximation algorithm, the overparameterization problem is circumvented by adding a regularizer to it's objective function.

2.5. Gaussian approximation

The mean-field approximation method in Section 2.3 obtains the couplings using matrix inversion. The relationship between inverse correlation matrix and couplings is established using physical arguments to approximate the global probability model parameters. This approach also helped to derive self-consistent equations for computing fields. Another way of estimating couplings from the inverse correlation matrix is using multivariate Gaussian probability distribution to describe residue/nucleotide co-evolution.

The Gaussian approximation method of contact prediction uses multivariate probability distribution. Each sequence m in MSA is represented by a vector $\boldsymbol{x}^m$ of dimension $Lq \times 1$ where L is the length of sequences in MSA and q is the number of residues/nucleotides plus gap. Denoting the mean by $\boldsymbol{\mu}$ and the $Lq \times Lq$ covariance matrix by $\boldsymbol{\Sigma}$ the multivariate Guassian probability is written as

$$p(\boldsymbol{x}^m;\boldsymbol{\mu},\boldsymbol{\Sigma}^{-1}) = \frac{1}{(\sqrt{2\pi})^{L^2q^2}\sqrt{\det\boldsymbol{\Sigma}}}\exp\left(-\frac{1}{2}(\boldsymbol{x}^m-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}^m-\boldsymbol{\mu})\right) \tag{2-32}$$

where $\det\boldsymbol{\Sigma}$ denotes determinant. Due to sampling effect the covariance matrix may be singular. In addition Gauge fixing should be taken into account to while computing the inverse. One way of circumventing these issues is to use maximum likelihood with regularization as in[40] or using a conjugate prior so as to find a closed form solution as in[10]. The strategy used in PSICOV, for example, is to compute the inverse covariance matrix using sparse regularizer also known as graphical LASSO—acronym for *least absolute shrinkage and selection operator*. The likelihood function given M sequences in MSA data is written as

$$l(\boldsymbol{\mu}, \boldsymbol{\Sigma}^{-1}) = \prod_{m=1}^{M} \frac{1}{(\sqrt{2\pi})^{L^2(q-1)^2}\sqrt{\det\boldsymbol{\Sigma}}} \exp\left(-\frac{1}{2}(\boldsymbol{x}^m - \boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}^m - \boldsymbol{\mu})\right). \tag{2-33}$$

The log-likelihood is

$$\log l(\boldsymbol{\mu}, \boldsymbol{\Sigma}^{-1}) = -\frac{M}{2}\left(L^2(q-1)^2\log 2\pi + \det\boldsymbol{\Sigma}\right) - \sum_{m=1}^{M}\frac{1}{2}(\boldsymbol{x}^m - \boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}^m - \boldsymbol{\mu}). \tag{2-34}$$

Dropping the constant term one can write the log-likelihood as

$$\log l(\boldsymbol{\mu}, \boldsymbol{\Sigma}^{-1}) = -\frac{M}{2}\det\boldsymbol{\Sigma} - \sum_{m=1}^{M}\frac{1}{2}(\boldsymbol{x}^m - \boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}^m - \boldsymbol{\mu}). \tag{2-35}$$

The gradients of the log likelihood are given by

$$\frac{\partial \log l(\boldsymbol{\mu}, \boldsymbol{\Sigma}^{-1})}{\partial \boldsymbol{\mu}} = \sum_{m=1}^{M}\boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu} - \boldsymbol{x}^m) \tag{2-36}$$

and

$$\frac{\partial \log l(\boldsymbol{\mu}, \boldsymbol{\Sigma}^{-1})}{\partial \boldsymbol{\Sigma}^{-1}} = \frac{M}{2}\boldsymbol{\Sigma}^T - \frac{1}{2}\sum_{m=1}^{M}(\boldsymbol{x}^m - \boldsymbol{\mu})(\boldsymbol{x}^m - \boldsymbol{\mu})^T. \tag{2-37}$$

2.6. Boltzmann learning

The previous sections highlight DCA algorithms are obtained by approximating the global probability in Eq. 2-11 since the direct numerical solution is computationally expensive. This challenge is particularly true for protein sequences as the complexity is in the order of 20^L. However, recently Cuturello *et al.* used the global probability model to infer parameters for RNA sequences[21]. This algorithm—known as Boltzmann learning—uses log-likelihood of the global probability and estimates parameters using Monte Carlo sampling.

The global probability function is given by Eq. 2-11. The likelihood is given by

$$l(\{J_{ij}(a_i, a_j)\}, \{h_i(a_i)\}) = \prod_{m=1}^{M}\frac{1}{\mathcal{Z}}\exp\left(\sum_{i<j}J_{ij}(a_i^m, a_j^m) + \sum_i h_i(a_i^m)\right). \tag{2-38}$$

The log-likelihood is

$$\log l(\{J_{ij}(a_i, a_j)\}, \{h_i(a_i)\}) = -M\log\mathcal{Z} + \sum_{m=1}^{M}\left(\sum_{i<j}J_{ij}(a_i^m, a_j^m) + \sum_i h_i(a_i^m)\right). \tag{2-39}$$

The gradients of the log-likelihood are given by

$$\frac{\partial \log l(\{J_{ij}(a_i, a_j)\}, \{h_i(a_i)\})}{\partial h_i(a_i)} = M\left(f_i(a_i) - p(a_i)\right) \tag{2-40}$$

and

$$\frac{\partial \log l(\{J_{ij}(a_i,a_j)\},\{h_i(a_i)\})}{\partial J_{ij}(a_i,a_j)} = M\left(f_{ij}(a_i,a_j) - p(a_i,a_j)\right). \tag{2-41}$$

Eq.2-40 and Eq. 2-41 show that when the log-likelihood of the global probability is maximized, the empirical frequencies obtained data exactly match the corresponding marginal probabilities of the global probability model.

2.7. Estimating interaction scores

In the previous sections, various algorithms have been seen to compute DCA parameters, i.e., couplings and fields. The couplings and fields are not directly used to measure coevolutionary signal strength from families of homologous proteins/RNAs. Instead, a quantity that summarizes these parameters is introduced. Throughout DCA algorithm development, two popular quantities have been utilized. one is called direct information–similar in structure with that of mutual information. The other is taking the Frobenius norm of the couplings.

The direct information approach of computing DCA coevolutionary scores was introduced in mean-field approximation[66]. The direct information between two sites is computed from direct probability defined as

$$p_{ij}^{dir}(a,b) = \frac{1}{\mathcal{Z}_{ij}} \exp\left(J_{ij}(a,b) + \tilde{h}_i(a) + \tilde{h}_j(b)\right) \tag{2-42}$$

where $\mathcal{Z}_{ij}$ is normalization constant, and $\tilde{h}_i(a)$ and $\tilde{h}_j(b)$ are new fields are sites i and j respectively. The new fields are obtained from the requirement that marginal probabilities of p_{ij}^{dir} correspond to single-site frequencies of MSA data, i.e., $\sum_a p_{ij}^{dir}(a,b) = f_j(b)$ and $\sum_b p_{ij}^{dir}(a,b) = f_i(a)$. The direct information score between sites i and j is denoted by D_{ij}^{dir} and obtained from

$$D_{ij}^{dir} = \sum_{a=1}^{q-1}\sum_{b=1}^{q-1} p_{ij}^{dir}(a,b)\log\frac{p_{ij}^{dir}(a,b)}{f_i(a)f_j(b)}. \tag{2-43}$$

The advantage of direct information is that its values are gauge independent—shifting the couplings and fields by constant values will not alter the direct probability used to compute the direct information.

Another way of quantifying interaction scores is using the Frobenius norm of couplings written as

$$F_{ij} = \sqrt{\sum_{a}^{q-1}\sum_{b=1}^{q-1} |J_{ij}(a,b)|^2}. \tag{2-44}$$

However, the Frobenius norm is gauge dependent—shifting the couplings by constant values changes the value of Frobenius norm. In addition, to minimize the effect of biased data, e.g., due to entropic or phylogenetics bias, the couplings are transformed to[27]

$$J_{ij}^{new}(a,b) = J_{ij}(a,b) - J_{ij}(a,\cdot) - J_{ij}(\cdot,b) + J_{ij}(\cdot,\cdot) \tag{2-45}$$

where the $\cdot$ denotes average over that argument. Furthermore, the score computed from Frobenius norm is transformed to a new score by using

$$F_{ij}^{new} = F_{ij} - F_{i\cdot} - F_{\cdot j} + F_{\cdot\cdot} \tag{2-46}$$

which is also known as average product corrected (APC) score. In addition to direct information, introduced[66] and Frobenius norm with an average product correction that commonly used in plmDCA[27] to compute interaction scores, the other way of quantifying coevolutionary signal from couplings obtained from DCA algorithms is to use l^1-norm of the couplings. This method has been used in PSICOV[40] which uses Gaussian approximation for the probability model. The l^1-norm is mathematically written as $\sum_{a,b} |J_{ij}(a,b)|$ where the summations run over the residues/nucleotide.

2.8. Summary

During evolution, protein/RNA sequences experience mutations. To keep the three-dimensional structures maintained, the mutations should not occur randomly; instead, they coevolve, i.e., the mutations happen in a coordinated manner. Statistical methods such as mutual information (MI) has been used to infer traces of residue/nucleotide pairs from multiple sequence alignments of homologous sequences. While MI can measure correlations between pairs of residues/nucleotides, it cannot identify whether the correlations are a result of direct interaction or due to indirect interactions propagated through other residues/nucleotides.

A decade ago, a class of contact prediction algorithms based on a global probability model was developed and applied particularly for proteins. These algorithms, collectively known as direct coupling analysis (DCA), can predict spatially adjacent residue/nucleotide pairs from MSA data. One of the algorithms is based on mean-field approximation—an idea borrowed from statistical physics—computes the global probability model's parameters by matrix inversion and self-consistent equation results from an approximation of the model's energy function[66]. A similar approach that uses matrix inversion is based on multivariate Gaussian approximation for the probability model[40]. Latter, a DCA algorithm based on pseudo-likelihood approximation is developed[27]. This algorithm is shown to be more accurate than the mean-field one, particularly for proteins, although mean-field approximation is computationally more efficient. Recently, the global probability model is used directly without approximation for RNA sequences[21]. It estimates the parameters using Monte Carlo sampling.

The global probability model that DCA algorithms are based on contains more parameters than the number of equations. The DCA algorithms circumvent this problem using two ways. One is using Gauge fixing. This task involves setting DCA parameters or their combination involving gap states to zero[103, 66]. Another way is using a regularizer that takes care of the overparameterization[27, 40]. Finally, it has been seen that the coevolution interaction

score is a summary of parameters of the global probability. It is usually done by computing the norm of couplings such as l^1-norm[40] or the Frobenius norm[27]. Also, a procedure called average product correction (APC) is carried out in order to reduce effect entropic and phylogenetic effects[40].

3. Convolutional Neural Networks

3.1. Introduction

n machine learning and artificial intelligence, convolutional neural networks (CNN) are among the most popular deep learning algorithms. Their success for tasks such as image and video recognition has made them to be widely used in many disciplines[90, 48, 47, 108, 51]. Unlike conventional neural networks, convolutional neural networks are shift/space invariant algorithms since they use shared parameters/weights to be learned from data. CNNs are supervised machine learning algorithm as they learn features guided by labeled data.

This chapter provides a brief introduction to CNN. It describes their architecture, such as layers, presents how these algorithms work using convolution operation, and provides some examples of non-linear functions that are recently becoming very popular. For a rigorous theoretical explanation of these powerful algorithms can be found in[49, 50, 45, 52, 24, 105] . Software implementation of CNNs includes the open-source TensorFlow software written in the Python programming language using C++ as its backend and is optimized to run on several platforms[1]

3.2. Architecture of Convolutional Neural Networks

Traditional artificial convolutional neural networks can be diagrammatically represented in several ways. A diagram that shows how these algorithms are represented is shown in Fig.**3-1**. In this representation, there are several layers, such as the pooling, convolution, and dense layers. At each layer, mathematical operations are carried out accordingly.

At the pooling layer, a down-sampling of input data is carried out. This operation can be done in a variety of ways[12]. Typical down-sampling techniques are max-pooling and average-pooling. In max-pooling, the data size is reduced by taking the maximum pixel element in a section of the input data. In contrast, in average-pooling, the average of the pixel elements in that section is taken. The convolution layer is the layer where convolution operation is performed. In the context of CNNs, convolution operation involves multiplicating a section of input data pixels with a weight matrix—the kernel. The multiplication is carried out element-by-element of the data section and the kernel. The final result is the sum of all these multiplications (see Fig.**3-2**).

In addition to the pooling and convolution layers, CNNs can contain dense layers and an

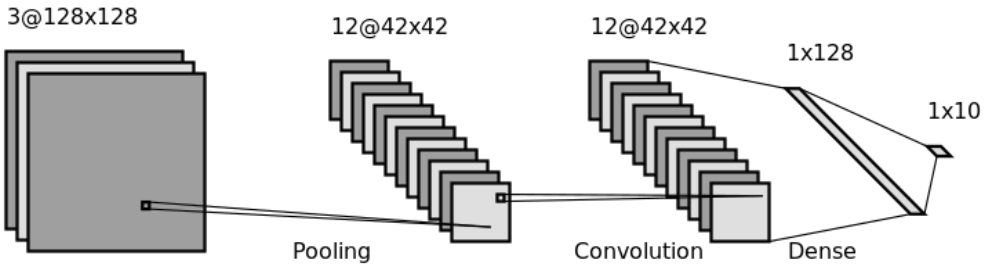


Figure **3-1**.: Architecture of convolutional neural networks[49]. The diagram shows pooling, convolution, and dense layers. The pooling layer involves the downsampling of input data using methods such as max-pooling or average-pooling. The convolution layer is a layer where convolution operations are performed using kernel matrices. The dense layer consists of fully connected layers. In this diagram, the final 1×10 layer is the output layer that classifies a given image, e.g., handwritten digits.

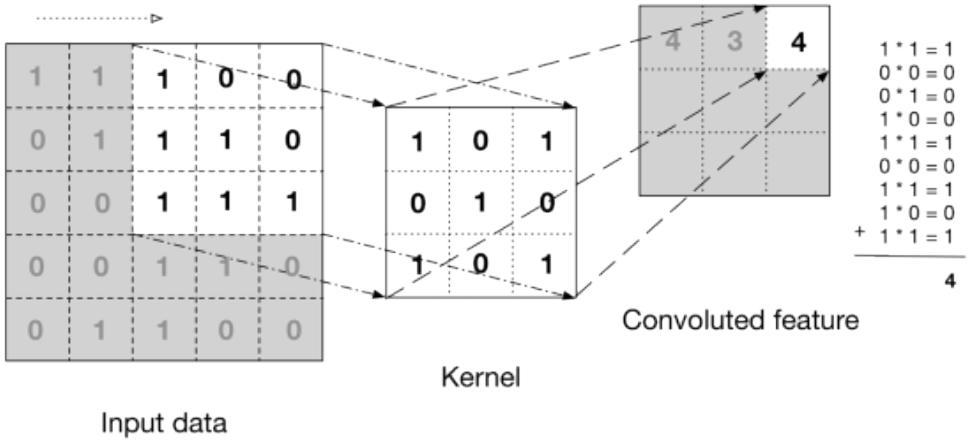


Figure **3-2**.: Illustration of convolution operation (Figure adopted from[75]). In the context of CNNs, convolution involves sliding a kernel (matrix of weights) on input data and multiplying the corresponding elements of the section of the data that the kernel overlaps. The final result is the sum of all those multiplied numbers.

output layer. The dense layers are fully connected to conventional neural networks. The output layer contains a vector of final output results; for instance, in the case of hand-written digit classification, it is a 1×10 vector containing the corresponding probabilities of digits from zero to nine.

CNNs have several hyper-parameter—for instance, width and depth of the network, number, and size of kernel matrices. The width refers to the size of input data, whereas the depth refers to the number of layers involved. Also, a particular layer may contain several input data matrices, e.g., the three color channels of a colored image. Since the kernel matrices are

shared across a particular input data, the width does not increase the number of trainable parameters. The number of free parameters is mainly affected by the number and size of kernel matrices used in a particular layer and the network's depth.

3.3. Activation Functions

Activation functions allow the introduction of nonlinearities in neural networks and help learn complex features. There is a wide range of activation functions used in neural networks, each having its advantages and disadvantages. Recently, rectified linear unit (ReLU) activation function is widely used[71]. The ReLU activation is also used in a slightly modified manner, such as the Leaky ReLU and the exponential leaner unit (ELU) (see Fig.**3-3**).

The ReLU functions filter out negative values. This characteristic makes ReLU activation functions biologically inheritable[33]. This function is scale-invariant, meaning multiplying the values ReLU functions take by a constant results in scaling of the entire function, i.e., $g(az) = ag(z)$ where $g(z) = max(0, z)$. Unlike traditionally used tangent hyperbolic and sigmoid functions, the ReLU function is also computationally efficient. However, the ReLU function is not differentiable at zero. Also, because negative values are not activated, it can result in a problem called dying ReLU, meaning ReLU functions may be completely inactive, causing a problem while training the network as a consequence.

A variant of the ReLU function is the Leaky ReLU defined as $g(z) = max(\epsilon z, z)$ where ϵ is a small number much greater than one. The leaky ReLU function has an advantage over the ReLU function as the leaky ReLU allows small positive gradients when the unit is not active, i.e., when the value of z is less than zero. However, like the ReLU function, the leaky ReLU function is not differentiable at zero.

The ELU function is given by the expression $g(z) = max(\alpha(e^z - 1), z)$. The ELU activation function addresses ReLU and leaky ReLU activation functions' weaknesses– it is differentiable at zero (ReLU and leaky ReLU are not). Also, ELU allows small positive gradients when the input is less than zero (ReLU does not allow it).

3.4. Traning convolutional neural networks

There are two main stages while feedforward convolutional neural networks; forward pass and backpropagation[49]. These operations involve a series of numerical operations across the network layers.

During the forward pass, data is passed through the input layer and proceeds to the subsequent layers. Filter matrices are initialized, and convolution operations are performed on the convolutional layers. Furthermore, non-linearities are applied to the results of convolutional operations. Finally, the output results are compared with the anticipated values. The cost function is computed from the output values suggested by the network and the labeled

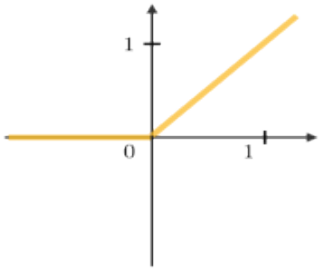
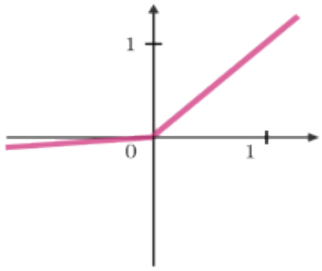
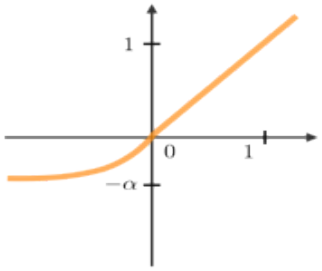
ReLU	**Leaky ReLU**	**ELU**
$g(z) = \max(0,z)$	$g(z) = \max(\epsilon z,z)$ with $\epsilon \ll 1$	$g(z) = \max(\alpha(e^z - 1),z)$ with $\alpha \ll 1$
		
Non-linearity complexities biologically interpretable	Addresses dying ReLU issue for negative values	Differentiable everywhere

Figure **3-3**.: Linear unit activation functions (the Figure adopted is from[6]). The ReLU function is biologically interpretable. However, it filters out negative input values and is not differentiable at zero. The leaky ReLU function allows small positive gradients when the input is negative; however, it is not differentiable at zero. The ELU function allows small positive gradients when the input is negative; also, it is differentiable at zero.

data.

The backpropagation process starts from the output layer. This task involves adjusting the weights based on the gradients of the output layer and the preceding layers[53, 84, 50]. Instead of computing the gradient of the resulting composition function, backpropagation uses the chain rule to update the parameters layer by layer. The values of these parameters is updated using gradient descent method[83].

3.5. Summary

Convolutional neural networks (CNN) are powerful algorithms used in machine learning and artificial intelligence. CNNs are supervised machine learning methods that are popular in the field of computer vision. They can extract complex features of images and videos using kernels—matrices of weight parameters learned using gradient descent and back-propagation.

A typical CNN contains several layers, including pooling, convolution, and dense layers. The pooling layer involves down-sampling of input data, e.g., using max-pooling and average pooling. In max-pooling, the largest pixel element in a section of data is taken among all elements in that section, whereas average pooling involves the average of those elements. The convolution layer involves performing convolution operations using kernels—matrices whose

elements are the weights that will be learned. The dense layer contains fully connected neural networks.

In order to learn various features of input data such as images, CNNs use activation functions. These functions introduce non-linearity and help to learn complex features from data. Typical examples of non-linear activation functions include rectified linear unit (ReLU), leaky ReLU, and exponential linear unit (ELU). Each of these activations has advantages and disadvantages. For instance, while the ReLU function is biologically interpretable, it can cause an algorithmic problem during training as it filters out input values less than zero.

4. Implementing DCA Algorithms into Computer Software

4.1. Introduction

In the previous chapter, I have revised a class of algorithms called direct coupling analysis (DCA) based on a global probability function derived using the entropy maximization principle with a minimal set of constraints. Using single-site and pair-site probabilities—in addition to the principle of normalization of probabilities—as constraints, an expression for the probability that a protein/RNA sequence to be sampled during evolution was obtained. This global probability model relates every site in a sequence to another through its couplings and considers the local effect of a residue/nucleotide through fields. Due to the global probability model's complexity, estimating the fields and couplings is mainly carried out using approximate algorithms, e.g., the mean-field and pseudo-likelihood approximations.

There exist some software implementations for DCA algorithms in the scientific community[36, 40, 9, 89]. However, most of this DCA software either focuses on one algorithm or mainly focuses on proteins. Also, although some of them provide a high-level application programming interface (API), e.g., the EVCouplings Python framework[36], the main parameter-inference algorithms are implemented in low-level programming languages that need to be manually compiled and integrated into the high-level API. Furthermore, coevolutionary information obtained from DCA algorithms may be used as a crucial input to machine-learning algorithms in order to improve protein/RNA contact prediction[2]. Consequently, building DCA algorithms software API that can seamlessly be integrated with other statistical/machine-learning algorithms is useful.

Here I present a software implementation of mean-field and pseudo-likelihood DCA algorithms. The DCA software implemented using the multipurpose open-source Python programming language provides DCA computations to RNA sequences and proteins. It uses low-level languages such as C and C++ to carry out computationally costly parameter inference part of the pseudo-likelihood DCA algorithm. Even though the several programming languages to build the software are used, it is light-weight and stand-alone; installing is as easy as installing any other stand-alone Python-based software. Currently, the software provides the following main functionalities:

- Computation of mean-field DCA scores summarized by Frobenius norm or direct-

information as well as their average product corrected forms.

- Computation of pseudo-likelihood maximization DCA scores summarized by Frobenius norm direct-information as well as their average product corrected forms.
- Computation of the parameters of the energy function, i.e., the fields and couplings.
- Trimming MSA data by gap percentage or reference sequence.
- Mapping DCA scores of a reference sequence to the corresponding site-pair scores in the MSA.
- Visualization of contact map and true positive rate of DCA ranked site-pairs.

The material in this chapter is organized as follows. First, the architecture of pydca is presented. Then, sequence reweighting is explained before explaining the implementation of two popular DCA algorithms. The following sections are dedicated to how each module component of the software works. After that, a discussion about algorithmic complexities. Finally, the software's usage examples are presented before concluding the chapter.

4.2. Architecture of pydca

The implementation of *pydca* follows modular architecture. It is composed of subpackages each responsible for a specific task (see Fig.**4-1**). In current version—*pydca v1.x*—the main subpackages are `msa_trimmer` for trimming MSA data before DCA computation; `sequence_backmapper` for mapping a reference sequence sites to the corresponding sites in the MSA; `contact_visualizer` for visually comparing contact maps and true positive rates; `meanfied_dca` for mean-field algorithm based DCA computation and `plmdca` for pseudo-likelihood approximation based DCA computation.

The `msa_trimmer` subpackage implements MSA data trimming functionalities so that trimming is performed in either of two ways. The first is trimming by gap percentage in MSA data columns. It allows removing all columns in MSA if they contain gaps more than a specified percentage of gaps. The second is trimming by reference sequence. This trimming method uses a supplied reference sequence to perform trimming. A sequence in MSA that best matches the reference sequence is searched using pair-wise sequence alignment to accomplish this task. Then sequence gaps in MSA corresponding to the best matching sequence are removed. This removal can be done in two ways: by removing all the gaps or removing by gap percentage of the gaps in the best matching sequence.

The main goal of DCA in the context of contact prediction is to find out which pairs of residues of a particular protein/RNA sequence are likely to be in spatial proximity. Sometimes, a sequence of interest (reference/target sequence) may be shorter/longer than the

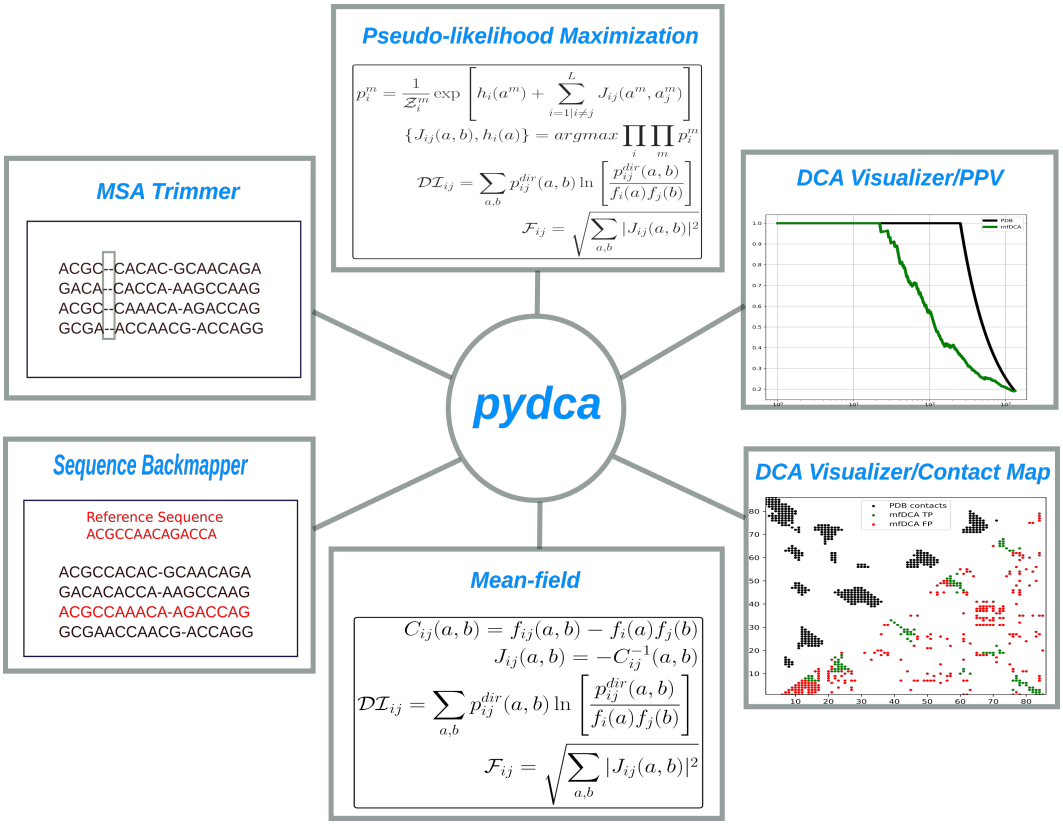


Figure **4-1**.: *pydca* modular architecture. The software is composed of sub-packages, each encapsulating a specific task. The current version of *pydca* has sub-packages for pseudo-likelihood and mean-field approximation DCA computation algorithms and sequence back-mapping, MSA trimming, and visualization sub-packages.

number of columns in the MSA data. The `sequence_backmapper` subpackage provides functionality for mapping the target sequence sites to the MSA's corresponding sites. *pydca* performs this task first by finding the best matching sequence among all MSA sequences to the target sequence.

Sometimes, one might be interested in quickly evaluating the mean-field and pseudo-likelihood algorithms' contact prediction capabilities within *pydca*. This utility is implemented in `contact_visualizer` sub-package. The subpackage is mainly intended for comparing contact maps of DCA predicted residue-pairs with that of a known PDB structure and evaluating the true positive rates of residue pairs ranked by their DCA score.

The mean-field and pseudo-likelihood maximization algorithms are implemented within `meanfield_dca` and `plmdca` sub-packages, respectively. These two comprise the core of *pydca* software contact prediction sub-packages. They also involve the numerically intensive parts of the software in its current version. The mean-field DCA algorithms are implemented entirely in the Python programming language. Since Python is a high-level interpreted language, DCA computation can be slower than implementations using low-level compiled

programming languages like C and C++. In *pydca* we have taken advantage of just in time (JIT) compiler known as Numba[46]. In addition to compiling the Python code, we have used Numba to compute weights of sequences and frequency counts using multiple threads. Unlike the mean-field DCA algorithm, the pseudo-likelihood maximization algorithm parameter inference in *pydca* is entirely implemented using the C++ backend. Its implementation allows multi-threading using OpenMP (https://www.openmp.org/). Once the parameters are computed in the C++ backend, they are copied back to the Python interface for computing DCA scores.

To sum up, *pydca* follows modular architecture implementation. This feature has resulted in less coupling among the various sub-packages, thereby enabling smooth future extension of the software. Furthermore, it takes advantage of modern computers' multi-processor nature by implementing computationally time-consuming parts of DCA using the mean-field and pseudo-likelihood maximization algorithms in a multi-threaded fashion.

4.3. Sequence reweighting

Biological sequences are usually classified into families and stored in databases such as Pfam [28], and Rfam [41]. The sequence families contain homologous sequences—sequences that have similar structure and function. The sampling process may be biased, e.g., due to the sequencing of some organisms more often than others. It may result in the over-representation of some sequences in databases which can negatively affect statistical analysis based on MSA of protein/RNA sequences.

A common procedure that is used in DCA-based algorithms is to reweight the sequences. Reweighting is usually done by assigning a cut-off value for sequence similarity. In this procedure, sequences whose similarity is beyond the cut-off value are considered redundant, i.e., they represent very similar sequences but are counted several times as though they were distinct. A weight is assigned to each of the sequences in an MSA to avoid over-counting of the sequences. The values of the weight are less than or equal to one. It can be seen as follows. Suppose that there are m sequences that are similar for a given similarity cut-off value. Then each of these sequences are assigned a weight $\omega = \frac{1}{m}$. The value of ω is maximum (which equals one) when a sequence is unique in MSA data.

Without sequence reweighting, each sequence in MSA is equally important, i.e., it receives a weight equal to one regardless of the number of similar sequences that may present in the MSA. However, due to reweighting, each sequence receives a new weight based on the number of similar sequences. It also changes the way sequences are counted. The total number of sequences when reweighting is carried out is counted by summing the weights. The resulting value—also known as the number of effective sequences—is given by $M_{eff} = \sum_{i=1}^{M} \omega_i$ where M is the total number of sequences and ω_i is the weight of the i^{th} sequence. *pydca* uses this procedure to assign weights to each sequence in the MSA.

4.4. Mean-field DCA implementation

In the previous chapter, I have reviewed that the mean-field DCA algorithm computes the coupling by matrix inversion. More precisely, the inverse of the correlation matrix obtained from MSA data is related to the negative of the couplings. However, the correlation matrix constructed from raw MSA data may be singular. This problem in mean-field DCA is circumvented by addition pseudo-counts to the MSA data [66].

When pseudo-counts are included and sequences are reweighted single-site and pair-site frequencies are given by

$$f_i(a_i;\lambda) = \frac{1}{M_{eff}+\lambda}\left(\frac{\lambda}{q} + \sum_{m=1}^{M}\omega_m \delta_{a_i^m,a_i}\right), \tag{4-1}$$

and

$$f_{ij}(a_i,a_j;\lambda) = \frac{1}{M_{eff}+\lambda}\left(\frac{\lambda}{q^2} + \sum_{m=1}^{M}\omega_m \delta_{a_i^m,a_i}\delta_{a_j^m,a_j}\right). \tag{4-2}$$

where λ is a parameter that controls the pseudo-count; $\delta_{x,y}$ is an indicator function that takes a value of one when x is equal to y and zero otherwise; ω_m is the weight of the m^{th} sequence in MSA; and q is the number of standard residues/nucleotides plus gap state. For convenience, Equations 4-1 and 4-2 can be rewritten as

$$f_i(a_i;\theta) = \frac{\theta}{q} + (1-\theta) f_i(a_i;\lambda=0) \tag{4-3}$$

and

$$f_{ij}(a_i,a_j;\theta) = \frac{\theta}{q^2} + (1-\theta) f_{ij}(a_i,a_j;\lambda=0) \tag{4-4}$$

where $\theta = \frac{\lambda}{M_{eff}+\lambda}$ is the parameter that controls the pseudo-count. My implementation of mean-field DCA in *pydca* uses a default pseudo-count—as suggested in [66]—$\lambda = M_{eff}$ which amounts to setting $\theta = 0.5$. This ensures the invertibility of the correlation matrix.

The mean-field DCA algorithm in *pydca* is entirely implemented using the Python programming language. To speed up execution and carry out matrix operations, it uses the Scipy[100] and Numpy[34] libraries. Also, execution speed is enhanced by utilizing the Numba just in time (JIT) compiler[46].

In mean-field DCA the couplings are related to inverse of the correlation matrix (see Chapter 2 Eq. 2-24). In *pydca* the correlation matrix is constructed from regularized frequencies in Equations 4-3 and 4-4. From the correlation matrix inverse the couplings are obtained. Typically, in mean-field DCA the final interaction scores are computed using the direct information given by Eq. 2-43. However, *pydca* also provides methods for computing the interaction scores using the Frobenius norm of couplings (Eq. 2-44) with average product correction (APC). In addition to computing DCA interaction scores, the mean-field algorithm implemented in *pydca* provides Python API to compute fields and couplings. This

task is carried out by using Eq. 2-22 which relates the single-site frequencies with fields and couplings.

4.5. Pseudo-likelihood maximization DCA implementation

The pseudo-likelihood maximization DCA algorithm in pydca computes the fields and couplings using gradient descent. The objective function is computed from the pseudo-likelihood's negative logarithm plus l^2 regularization for fields and couplings. This procedure of using gradient descent with l^2 regularization is also known as ridge regression. The objective function and its gradients are presented in Chapter 2 Section 2.4.

Adding l^2 regularization terms and taking into account sequence reweighting the gradient of the negative log-likelihood are given by

$$\frac{\partial \log l(\{h_i(a)\},\{J_{ij}(a,a_j)\};\lambda_h)}{\partial h_i(a)} = -\sum_{m=1}^{M}\omega_m(\delta_{a^m,a} - p_i^m(a)) + 2\lambda_h \sum_{i=1}^{L}\sum_{a_i=1}^{q} h_i(a_i), \quad (4\text{-}5)$$

$$\frac{\partial \log l(\{h_i(a)\},\{J_{ij}(a,a_j)\};\lambda_J)}{\partial J_{ij}(a,b)} = -\sum_{m=1}^{M}\omega_m(\delta_{a^m,a} - p_i^m(a))\delta_{a_j^m,b} + 2\lambda_J \sum_{i=1}^{L-1}\sum_{j=i+1}^{L}\sum_{a_i=1}^{q}\sum_{a_j=1}^{q} J_{ij}(a_i,a_j). \quad (4\text{-}6)$$

where λ_h and λ_J are parameters that determine the magnitude of regularizing terms for fields and coupling, respectively. In *pydca* the gradient decent is performed by using limited-memory BFGS algorithm [74, 57].

The gradient descent method is an iterative algorithm that minimizes the negative log-pseudo-likelihood cost function with l^2 regularization for fields and couplings. This iterative procedure can be computationally costly, particularly for large proteins in length and depth (raw number of sequences) in MSA data. Consequently, *pydca* uses a C/C++ backend to carry out the iterative optimization task of inferring fields and couplings.

Coevolutionary interaction scores in pseudo-likelihood maximization DCA are typically computed using the Frobenius norm of couplings, i.e., Eq. 2-44. For proteins, it has been shown that this interaction score, together with average product correction, can improve contact prediction [27]. Nevertheless, *pydca* software provides methods to compute interaction scores using direct information, i.e., Eq. 2-43 with or without average product correction.

4.6. Mapping reference sequences to its family

The primary input data for DCA-based algorithms is multiple sequence alignment data of homologous proteins/RNAs. Typically, MSA data is obtained as follows. First, a reference/-target protein/RNA sequence of particular interest is chosen. A query to databases such as

Pfam[28] or Rfam[41] is searched against the target sequence. This process can result in the matching of a particular protein/RNA. Protein/RNA families contain a set of protein/RNA sequences that are related. These protein/RNA can have varying sequence lengths.

Aligned sequences have an equal length in MSA. It is done by alignment algorithms introducing gaps at various columns. Also, some of the gap columns may be removed from the MSA data to increase the input MSA quality by reducing noise that negatively affects contact prediction accuracy. Consequently, a reference/target sequence can be shorter or longer than the length of sequences in the MSA.

DCA algorithms compute interaction scores of each site in the MSA column with every other. If the target sequence is shorter than the MSA columns, only those columns corresponding to the target sequence are relevant to predicting the target sequences' contacts. On the other hand, if the target sequence is more extended than MSA columns, not all sites in the target sequence can be included for contact prediction, i.e., the excess sites in the target sequence will not be part of predicted contacts.

To map the target sequence sites to the corresponding columns in the MSA, *pydca* provides a sequence back-mapper method. The sequence back-mapper works as follows. First, the best matching sequence among the MSA sequences is chosen by pairwise alignment with the target sequence. The best matching sequence is the MSA sequence that results in the highest score aligned with the target sequence. Then the residues/nucleotides of the best matching sequence interaction scores are mapped to the corresponding residues/nucleotides of the target sequence. These interaction scores are sorted in descending order, and top-ranked pairs represent putative contact pairs of the target sequence.

A particular case is when the target sequence itself present in the MSA. In this case, there is a one-to-one correspondence between the target sequence residues/nucleotides and that of the sequence itself in the MSA. Another possibility is the existence of more than one matching sequence in the MSA. In this case, the first matching sequence (according to the order of the MSA sequences) is taken.

4.7. Trimming multiple sequence alignment data

Curating input data for statistical models can enhance prediction accuracy. It is also true for DCA algorithms of protein/RNA contact prediction. When sequences are aligned, gaps are introduced or extended for shorter sequences. Too many gaps may contribute to noise that can negatively influence contact prediction accuracy. One way of curating MSA data before DCA computation is to trim columns in the MSA that contain too many gaps. *pydca* provides two ways of MSA trimming: by gap percentage or reference/target sequence.

MSA trimming by gap percentage is a simple task. It just removes all columns in MSA that contain a gap percentage more than particular cut-off values. For instance, setting the gap cut-off value to 0.9 removes columns that contain more than 90% gaps. On the other hand, trimming by reference sequence is a bit involved as it requires searching for the best

matching sequence from the MSA to that of the reference sequence. Besides, trimming by reference sequence can be done in two ways. One is by gap percentage, and the other is by removing all gaps in the MSA that do not involve columns in the MSA whereby the best matching sequence sites are residue/nucleotides.

Trimming by gap percentage guided by reference sequences works as follows. First, the best matching sequence to the reference is searched from the MSA. Then columns that contain gaps beyond gap cut-off values are removed from the MSA. However, if those columns do involve residues/nucleotides of the target sequence (instead of gaps), they will not be removed. On the other hand, trimming by reference sequence can also be used to remove all MSA gaps– regardless of the percentage of gaps in the columns–that are also gaps within the best-matching sequence to the reference/target sequence.

Finally, it should be noted that pydca does not perform trimming during DCA computation. Instead, it uses whatever MSA data is given to it and carries out computations. It is at the disposal of the user whether to trim the MSA or not. Based on this notion, the trimming utilities in *pydca* are supplied as stand-alone API that can be used before DCA computation is performed. Users of *pydca* can trim their MSA data, save it into a new file and then carry out DCA computation using this trimmed MSA.

4.8. Contact map and positive predictive value visualization

Other utilities that pydca provides include contact map and positive predictive value (PPV) visualization. These visualizations help to evaluate the accuracy of DCA-based contact prediction quickly. Given a known three-dimensional (PDB) structure of a protein/RNA, the contact map of DCA predicted site-pair could be compared with the PDB contact map. Likewise, positive predictive values can be computed and visualized to assess their deviation from theoretical values.

To visualize contact maps of DCA predicted site-pairs and compare them with the PDB structure contact map, *pydca* requires the PDB file or PDB id, the target/reference sequence, and the type of biomolecule (protein or RNA). The DCA interaction scores file–the output file containing sorted DCA interaction scores–is used to read DCA predicted site-pairs. The target/reference sequence helps determine the segment of protein/RNA sequence in the PDB structure that matches the target/reference. This task is done by the sequence back-mapper API of *pydca*. Specifying the type of biomolecule as protein or RNA allows *pydca* to select the correct scoring matrix while mapping the reference/target sequence with that of the PDB. Furthermore, the user is required to supply the contact distance cut-off values—the maximum distance that two residues/nucleotides to be considered as contacts—and the number of DCA predicted site-pairs to be considered as putative contacts. Alternately, for RNAs, a file containing non-nested secondary structure in dot-bracket notation may be

supplied to highlight the secondary structure in the contact map.

Visualization of positive predictive values also requires similar input data to that of contact map visualization. However, the number of putative contacts is not required as the positive predictive values are computed for all pairs of predicted contacts. Typically the positive predictive value at rank L—where L is the reference sequence length—is used as a benchmark rank to assess contact predictive accuracy. Nevertheless, the *pydca*'s positive predictive value visualization API provides information for the entire rank of predicted site-pairs.

4.9. Algorithmic complexities in pydca

The implementation of *pydca* software significantly takes into account performance speed. Computationally time-consuming parts of the DCA algorithms are optimized for performance. It is done in *pydca* by using just in time (JIT) compilers for python source codes or using compiled programming languages C and C++. Nevertheless, here we summarized the significant algorithmic complexities of the software.

The space complexity is $\mathcal{O}(L^2q^2)$ for both mean-field and pseudo-likelihood maximization algorithms which amount to the storage of fields and couplings, as well as single- and pair-site frequency counts. In the mean-field algorithm, the frequency counts need to be stored to construct the correlation matrix, which is used to obtain the couplings. In the pseudo-likelihood maximization, both fields and couplings and their gradients need to be stored for the subsequent iteration.

The time complexities scale as $\mathcal{O}(M^2L^2)$ for sequences weight computation; $\mathcal{O}(L^3q^3)$ for matrix inversion in the mean-field algorithm; $\mathcal{O}(ML_{ref}L_{min})$—where L_{ref} is the length of the reference sequence and L_{min} is the length of the shortest sequence after gaps are removed in the MSA—for sequence back-mapping if a reference sequence is supplied. In the plmDCA implementation of *pydca* the gradient descent algorithm has a time complexity of $\mathcal{O}(L^4q^3MN)$ where N is the number of gradient descent steps.

4.10. Usage examples of pydca

pydca is implemented so that it can be used as a Python library or from the command line. As a Python library, *pydca* or its sub-packages can be imported into other Python source codes. This feature is useful as the Python programming language is becoming more and more popular in the scientific computing community in general. The command-line interface of *pydca* allows the execution of commands flexibly as it allows the setting of the values of parameters right from the command line. In this section, I show examples of *pydca* usage as a Python library and from the command line.

Using pydca as a Python library

To use *pydca* from other Python source codes it needs to be imported into the source code. For example in Listings 1 the `plmdca, meanfield_dca, sequence_backmapper` and

```
from pydca.plmdca import plmdca
from pydca.meanfield_dca import meanfield_dca
from pydca.sequence_backmapper import sequence_backmapper
from pydca.msa_trimmer import msa_trimmer
```

Listing 1: Example importing pydca into Python source code.

`msa_trimmer` modules are imported. This makes the packages and their modules accessible within the source code.

Next I show how to compute DCA scores using the pseudo-likelihood maximization algorithm in Listing 2 where an instance of PlmDCA class named `plmdca_inst` is cre-

```
from pydca.plmdca import plmdca

# creat PlmDCA instance

plmdca_inst = plmdca.PlmDCA(
'alignment.fa',
'rna',
seqid = 0.8,
lambda_h = 1.0,
lambda_J = 20.0,
num_threads = 10,
max_iterations = 500
)

# compute DCA scores summarized by direct information

di_scores = plmdca_inst.compute_sorted_DI_APC()
```

Listing 2: Example of using pydca's pseudo-likelihood maximization algorithm into Python source code.

ated. The constructor of `PlmDCA` class takes some positional and keyword arguments. The positional arguments `'alignment.fa'` and `'rna'` refer to the path to a FASTA formatted input file containing MSA data and the type of biomolecule the sequences represent,

respectively. The keyword arguments are: `seqid` – the sequence identity cut-off value; `lambda_h` – value of regularization parameter for the fields; `lambda_J` – value of regularization parameter for the couplings; `num_threads` – number of threads for parallel execution and `max_iterations` – the maximum number of gradient descent steps. Finally we used `compute_sorted_DI_APC()` method on the `plmdca_inst` instance. This action computes and returns a sorted list of tuples of site-pairs and DCA scores. The DCA scores represent direct-information scores. Besides, they are average product corrected (APC) results. The corresponding method for computing DCA scores summarized by Frobenius norm of the couplings is `compute_sorted_FN_APC()`.

The keyword arguments take a range of valid values. The sequence identity (`seqid`) takes values greater than zero and less than or equal to one. The regularization parameters (`lambda_h` and `lambda_J`) can take any non-negative values. The number of threads for parallel execution (`num_threads`) takes a non-negative integer. Setting multiple threads works if OpenMP is supported in the machine where *pydca* is installed. The number of gradient descent iterations (`max_iterations`) takes a non-negative integer. This value determines the maximum number of gradient decent iterations. It is possible that the number of gradient descent iterations taken during computation can be less than the required number if convergence is reached in advance. Setting the value `max_iterations` very large helps to allow gradient descent to be performed until convergence.

The mean-field DCA algorithm of *pydca* offers a similar usage into Python source codes (see Listing 3). An instance of `MeanFieldDCA` class can be constructed, and DCA scores computed using the appropriate method. The constructor of `MeanFieldDCA` class takes

```
from pydca.meanfield_dca import meanfield_dca

# create an instance of MeanFieldDCA class

mfdca_inst = meanfield_dca.MeanFieldDCA(
        'alignment.fa',
        'rna',
        seqid=0.8,
        pseudocount=0.5,
)

# compute DCA scores quantified by the Frobenius
# norm of the couplings

FN_scores = mfdca_inst.compute_sorted_FN_APC()
```

Listing 3: Example of using pydca's mean-filed algorithm within Python source code.

two position and two keyword arguments. Like the constructor of `PlmDCA` class, the first positional arguments refer to the path to the MSA file and the biomolecule type the sequence represent. The new keyword argument `pseudocount` refers to the relative value of the pseudocount to regularize frequency counts. That is `pseudocount` refers to the value θ in Eq. 4-3 and Eq. 4-4. Setting the value of θ to 0.5 is equivalent to using $\lambda = M_{eff}$. Finally the DCA scores quantified by the Frobenius norm of the couplings is computed by the action of `compute_sorted_FN_APC()` on the `mfdca_inst` object. These scores are average product corrected. One can also use the `compute_sorted_DI_APC()` method in order to compute DCA scores summarized by direct-information.

Before DCA computation its possible to trim the MSA to reduce the noise that can adversely affect contact prediction accuracy. An example of MSA trimming by reference sequence is shown in Listing 4 where an instance of `MSATrimmer` class is created. `MSATrimmer`

```
from pydca.msa_trimmer import msa_trimmer

# create a trimmer instance

trimmer = msa_trimmer.MSATrimmer(
'alignment.fa',
biomolecule='rna',
refseq_file='refseq.fa'
)

# remove all gaps that are in the best matching sequence
# with the reference sequence.

trimmed_data = trimmer.get_msa_trimmed_by_refseq(
remove_all_gaps=True
)
```

Listing 4: Example of using pydca's MSATrimmer to trim by reference sequence.

constructor takes one positional and two keyword arguments. The positional argument refers to FASTA formatted file containing the MSA data. The keyword argument `biomolecule` takes values `'rna'` or `'protein'` in case insensitive form. The keyword argument `refseq_file` takes a path to FASTA formatted file containing a reference sequence. The MSA data is trimmed by removing all gaps that correspond to the best matching sequence with the reference. This is accomplished by the action of `get_msa_trimmed_by_refseq(remove_all_gaps =True)` method on the `MSATrimmer` instance.

Using pydca from the command line

The usage of *pydca* is not limited to within Python source codes. It can separately be used from the command line. It provides some commands and sub-commands to execute a variety of independent tasks at a time. In the current version, there are three main commands: `mfdca` – for the mean-field DCA algorithm; `plmdca` – for the pseudo-likelihood maximization algorithm and `pydca` – for contact map and true positive rate visualization, as well as MSA, trimming algorithms. For instance, one can compute DCA scores using the mean-field DCA algorithm from the command line as:

```
mfdca compute_di <biomolecule>  <msa_file> --verbose
```

where `compute_di` is the sub-command to compute DCA scores summarized by Frobenius norm of the couplings. This command takes two positional arguments: (1) `<biomolecule>` which refers to the type of biomolecule the sequences in MSA data represent. This argument can be either `rna` or `protein` and is case insensitive. While providing wrong information about the biomolecule is not algorithmically a problem for the *pydca* software, doing so results in a wrong DCA computation outcome. Thus the correct biomolecule type should be provided. (2) The positional argument `<msa_file>` takes a FASTA formatted MSA file containing the input data. Also, we have an optional parameter `--verbose`. This parameter triggers logging messages to be displayed on the screen as DCA computation progresses.

```
INFO 2020-02-06 13:58:59,096 meanfield_dca get_single_site_freqs
       Computing single site frequencies
INFO 2020-02-06 13:58:59,362 meanfield_dca get_reg_single_site_freqs
       Regularizing single site frequencies
INFO 2020-02-06 13:58:59,449 meanfield_dca get_pair_site_freqs
       Computing pair site frequencies
INFO 2020-02-06 13:59:01,633 meanfield_dca get_reg_pair_site_freqs
       Regularizing pair site frequencies
INFO 2020-02-06 13:59:01,767 meanfield_dca construct_corr_mat
       Constructing the correlation matrix
INFO 2020-02-06 13:59:01,942 meanfield_dca compute_couplings
       Computing couplings
```

Figure **4-2**.: Figure showing a snapshot of logging messages while executing *pydca* from the command line. Each logging message begins with a logging level (e.g., INFO). Then follows the date and time that message was displayed, the name of the module where the logging message originated, and the function (method) that emitted the logging message. Beneath this line is the logging message, e.g., the message "computing single site frequencies."

pydca provides three types of logging messages: (1) `INFO` – that indicates information about the computation steps being carried out. (2) `WARNING` – that displays warning message when *pydca* encounters ambiguities and (3) `ERROR` – that signals when an error occurs. When

an error occurs, e.g., due to a wrong parameter input chosen by the user, *pydca* displays the error message and halts the computation. The logging feature of *pydca* helps keep track of DCA computation and easily locate stack traces when errors occurred.

4.11. Summary and conclusion

This chapter presents an open-source software called *pydca* I developed for direct coupling analysis of RNA and protein sequences. The software is implemented using Python, C++, and C programming languages and provides high-level application programming interfaces (APIs) using Python. It implements two popular unsupervised machine learning contact prediction algorithms—namely mean-field and pseudo-likelihood maximization—based on a global probability function derived from the principle of entropy maximization with a minimal set of constraints. Besides, the software provides utilities for data preprocessing and prediction accuracy visualization.

The mean-field algorithm is very efficient and entirely implemented using Python and its numerical packages such as Scipy and Numpy. To speed it up further *pydca* uses a just in time (JIT) compiler called Numba. However, the pseudo-likely maximization algorithm can be computationally costly, particularly for large proteins. As a consequence, this algorithm is implemented using a C and C++ backend. Furthermore, if OpenMP is supported, it can be executed using multiple threads.

In addition to the two statistical algorithms, *pydca* provides utilities for preprocessing multiple sequence alignment data and for visualization of contact maps and positive predictive values. Data preprocessing is done by trimming aligned sequence columns that correspond to a significant number of gaps or trimming sites that do not correspond to a target's sequence residues/nucleotide. Contact map and positive predictive value visualization APIs allow to quickly evaluate the accuracy of predicted contacts for protein/RNA sequences whose PDB structures exist.

With the rapid growth of protein and RNA sequences, this software will provide standalone and easy-to-use computational utilities using direct coupling analysis for scientists from a wide range of backgrounds. The software is freely available under the MIT license and can be extended to add other functionalities in a modular manner.

5. Comparing DCA Algorithms for RNA Contact Prediction

5.1. Introduction

The ability of direct coupling analysis (DCA) to disentangle correlations that arise as a result of direct and indirect effects has revolutionized protein contact prediction from multiple sequence alignments of homologous proteins. Although direct sampling of the partition function from entropy maximization with a minimal set of constraints, e.g., consistency condition on single and pair-site frequency counts obtained from multiple sequence alignment, is a computationally prohibitive task, some approximate algorithms have been developed and employed since the inception of DCA. In the context of protein contact prediction, the approximate DCA algorithm accuracy has been systematically improved.

The first approximate DCA algorithm employed for protein contact prediction is based on a message-passing algorithm[103]. This method has been applied to infer contact of protein-protein interactions and predict residue-residue contacts more accurately compared to methods based on covariance-based methods such as mutual information. Nevertheless, this algorithm is still computationally costly, particularly for proteins that have considerable sequence length. Later, a more efficient DCA algorithm was introduced[66]. It is based on an inversion of covariance matrix obtained from protein multiple sequence alignments and computationally as efficient as matrix inversion. Soon after, a method based on the least absolute shrinkage and selection operator (LASSO) was introduced and shown to improve protein contact prediction compared to mutual information. Next, a more accurate DCA algorithm based on pseudo-likelihood maximization is introduced[27]. It approximates the likelihood function obtained from MSA data by a pseudo-likelihood function and improves protein contact prediction.

However, DCA was not applied to contact prediction from RNA sequences at the same time it was done for proteins. The first applications of DCA to RNA multiple sequence alignments are tested in[23, 104]. In these two seminal works of applying DCA to RNA, the objective was mainly to use one of the DCA algorithms—so far applied to proteins—for RNA contact prediction and integrate predicted nucleotide pairs as constraints/restraints in three-dimensional molecular modeling software. These works have shown that DCA-predicted contacts improve RNA three-dimensional structure prediction more accurately than contacts obtained from mutual information.

Recently assessment of DCA-based methods on RNA contact prediction is carried out in [21]. In this work, direct sampling of the global probability distribution without approximation is applied for RNA contact prediction. This algorithm —also known as Boltzmann learning—is compared with mean-field and pseudo-likelihood approximation DCA algorithms. A total of 17 RNAs have been included in this study. In addition to proposing the new Boltzmann learning algorithm for RNA contact prediction, the study evaluates mutual information performances, R-scape [82], mean-field and pseudo-likelihood DCA methods on the 17 RNAs.

The objective here is to expand RNA families' dataset by a significant number and evaluate DCA-based algorithms for RNA contact prediction. To this end, we have collected about 60 RNAs that have a variety of an effective number of sequences (see Table **A-1**). There is various software that implements DCA algorithms for proteins, and most of them use pseudo-likelihood DCA algorithm[89, 36, 9, 106]. However, we pick one software per DCA algorithms to focus on the algorithm's performances instead of various implementations. As a consequence, I choose EVCouplings for plmDCA[36], pydca for mean-field DCA[106], Boltzmann learning implemented in[21] and the graphical LASSO algorithm in PSICOV[**?**].

The chapter is organized as follows. First I introduce the RNA dataset going to be used in Section 5.2. Then, I evaluate the effect of input data—in the form of multiple sequence alignments—on contact prediction in Section5.3. Next, I evaluate contact prediction accuracy by nucleotide-nucleotide pair in Section 5.4. This is followed by comparison of various DCA algorithms on contact prediction accuracy including tertiary contact prediction accuracy in Sections 5.5 and 5.6. Finally, I summarize and provide outlook in Section 5.7

5.2. Dataset formation

In this section, we introduce the RNA dataset we used for comparing DCA algorithms of contact prediction. First, the criteria used to chose the RNAs in the dataset are presented. Then, an overview of the dataset composition by nucleotide pair is summarized. Finally, we define nucleotide pair contact in RNA three-dimensional structure and analyze the average number of contacts—all contact types and tertiary ones—in the dataset as a function of distance in RNA structures.

RNA data collection

The first step in our study involved the collection of RNA that have known PDB structures. We collected about 60 RNAs (see Table **A-1** for complete description of the dataset). The RNA dataset—we also call it dataset $\mathcal{D}$—is formed by surveying the PDB database and selecting RNAs that satisfy the following criteria:

- RNA structures were not in complex with protein or DNA.

- Only monomeric RNA structures are considered.
- The length of the RNA sequence has to be more than 40 nucleic bases.
- If multiple RNAs of sequence similarity beyond 50% exist, the one with the highest PDB structure resolution is selected.
- Only structures resolved via X-ray crystallography were taken into account with resolution below 3.6 Å.

The entire dataset is divided into two groups based on the number of effective sequences (M_{eff}) the RNA has in its family. One group contains RNAs with $M_{eff} > 70.0$—dataset $\mathcal{D}^H$ —while the other group contains the remaining RNAs, i.e., RNAs with $M_{eff} \leq 70.0$—dataset $\mathcal{D}^L$. This grouping is carried out in order to evaluate the effect of effective number of sequences on contact prediction accuracy.

Overview of dataset composition

The composition by nucleotide base pair percentage of the dataset is displayed in Fig.**5-1**. C-G and A-G are the most abundant among the base pairs, each constituting 15% of the dataset. G-U and A-C pairs are the second most abundant, each taking 12% of the entire dataset base pairs. The third most abundant pairs—A-U and G-G base pairs—each make up a tenth of the dataset base pairs. The C-U pair takes a slightly less portion of the dataset compared to the A-U and G-G pairs. Nucleotide pairs A-A and C-C together account for the same proportion as for pairs A-C or G-U. The least abundant nucleotide pair in the dataset is U-U making up the only 4%.

Nucleotide pair contact definition

Two nucleotides in the dataset RNAs are classified as contacts or non-contacts based on the distance between their heavy atoms, i.e., atoms other than the hydrogen atom. In particular, two nucleotides are considered as contacts if they have heavy atoms that are less than 10Å in the three-dimensional structure of the RNA. Also, contacts are classified as tertiary contacts or not based on their proximity to RNA secondary structure pairs. If a pair (i, j) forms secondary structure pair then all pairs in $\{(i \pm 0, 1, 2, 3, ..., k; j \pm 0, 1, 2, 3, ..., k)\}$ are removed and not considered to be tertiary contacts. Typically, we chose the value $k = 2$. It excludes 25—including the secondary structure pair itself—nucleotide pairs in the vicinity of a pair.

The average number of contacts NC_{PDB} in dataset $\mathcal{D}$ as a function of contact distance cut-off r_c measured in Ångstrom is displayed in Fig. **5-2**. As expected, the number of contacts increases as the cut-off distance is enlarged. At $r_c = 10\mathring{A}$ there are about 500 contacts per RNA on on average when no distinction is made whether a contact is tertiary or not (top figure in Fig. **5-2**). However, when tertiary only contacts are considered by

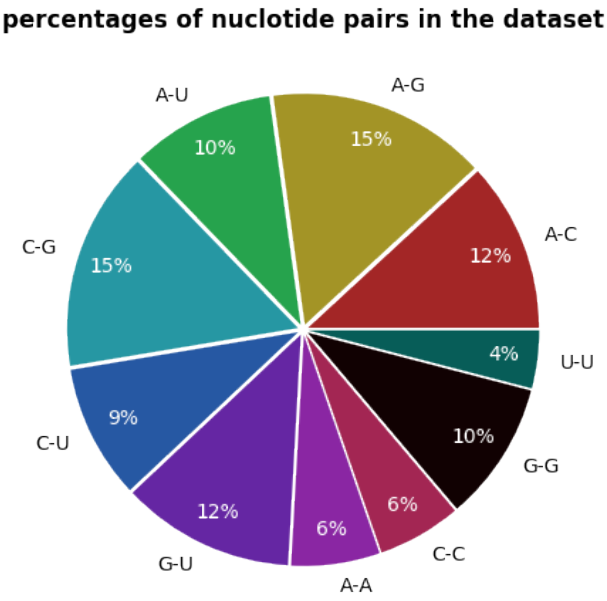


Figure **5-1**.: Figure showing the percentage of nucleotide pairs in the dataset. C-G and A-G pairs are the most prevalent pairs, each representing 15% of the entire nucleotide pairs. The least abundant pair is U-U representing only 4% of the nucleotide pairs in the dataset.

excluding 25 pairs that include the secondary structure pairs and their neighbors, about half the average number of contacts exists than before. That is there are about 250 tertiary contacts per RNA on average (bottom figure in Fig. **5-2**). In the next section, we evaluate DCA algorithms' performance using the RNA dataset described in this section. Specifically, we evaluate the effect of MSA data on contact prediction accuracy, compare the various DCA-based algorithm, and finally, provide a summary and outlook regarding RNA contact prediction using these algorithms.

5.3. Effect of multiple sequence alignment data

The contact prediction capability of DCA-based methods, just like any other statistical/machine learning algorithms, can be affected by the quality of input data they operate. The primary input data for DCA algorithms is multiple sequence alignment (MSA) of homologous protein/RNA sequences. The extent to which a set of sequences are properly aligned can be affected by a few factors, such as the MSA algorithm used to align these sequences and the similarity of sequences in the protein/RNA families. It, in turn, can significantly affect the performance of contact prediction algorithms. In this section we evaluate contact prediction accuracy using mean-field DCA when multiple sequence alignment is carried out using four alignment methods, namely, Infernal [72], CLUSTAL [95], MUSCLE [26], and

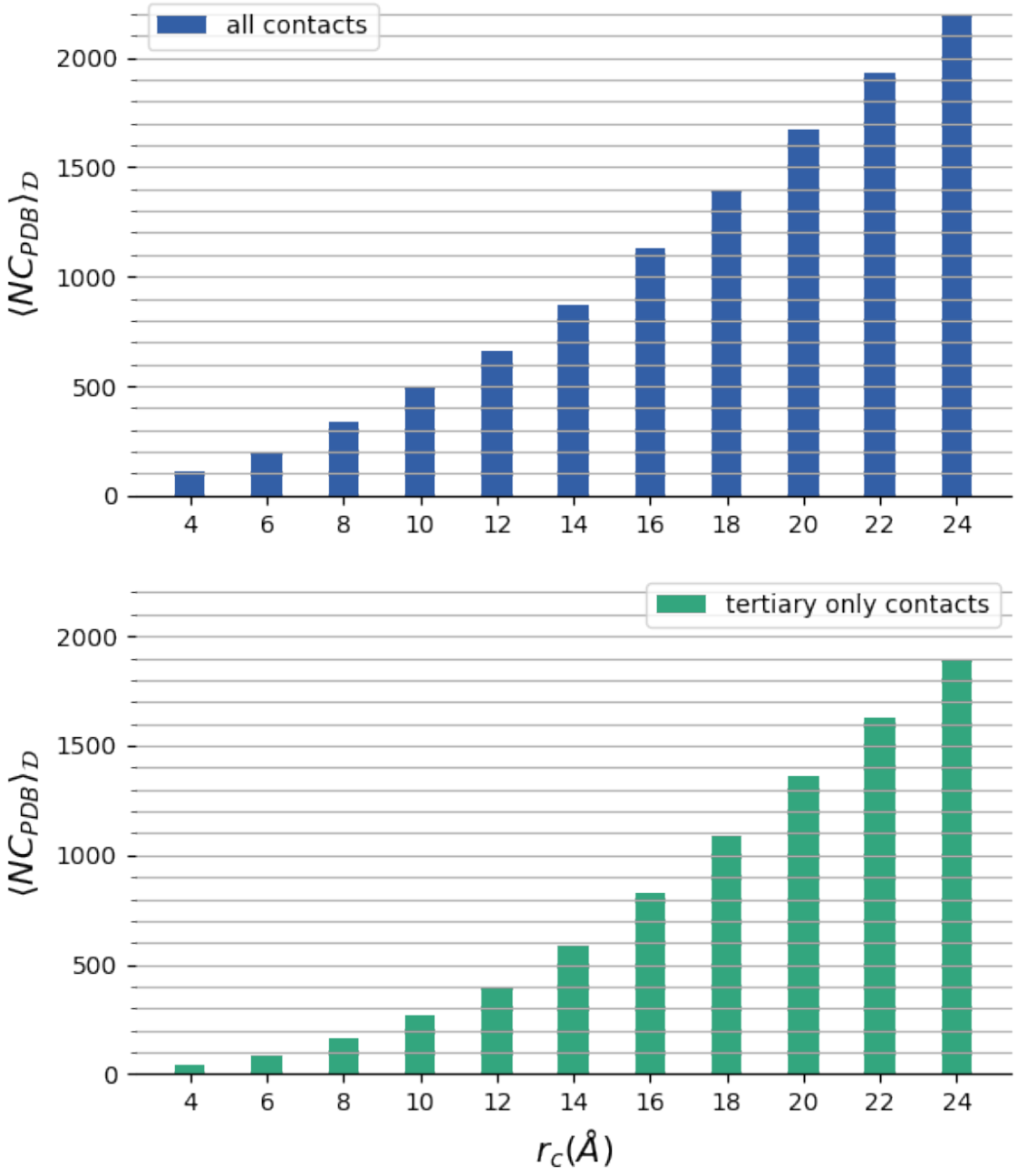


Figure **5-2**.: Figure showing the average number of contacts in the RNA dataset PDB structure as a function of contacts' distance. The top represents all contact types, i.e., tertiary contacts or otherwise, and the bottom represents only tertiary contacts. At $r_c = 10$ Å there are on average about half tertiary contacts compared to the entire contacts in the dataset.

MAFFT [43].

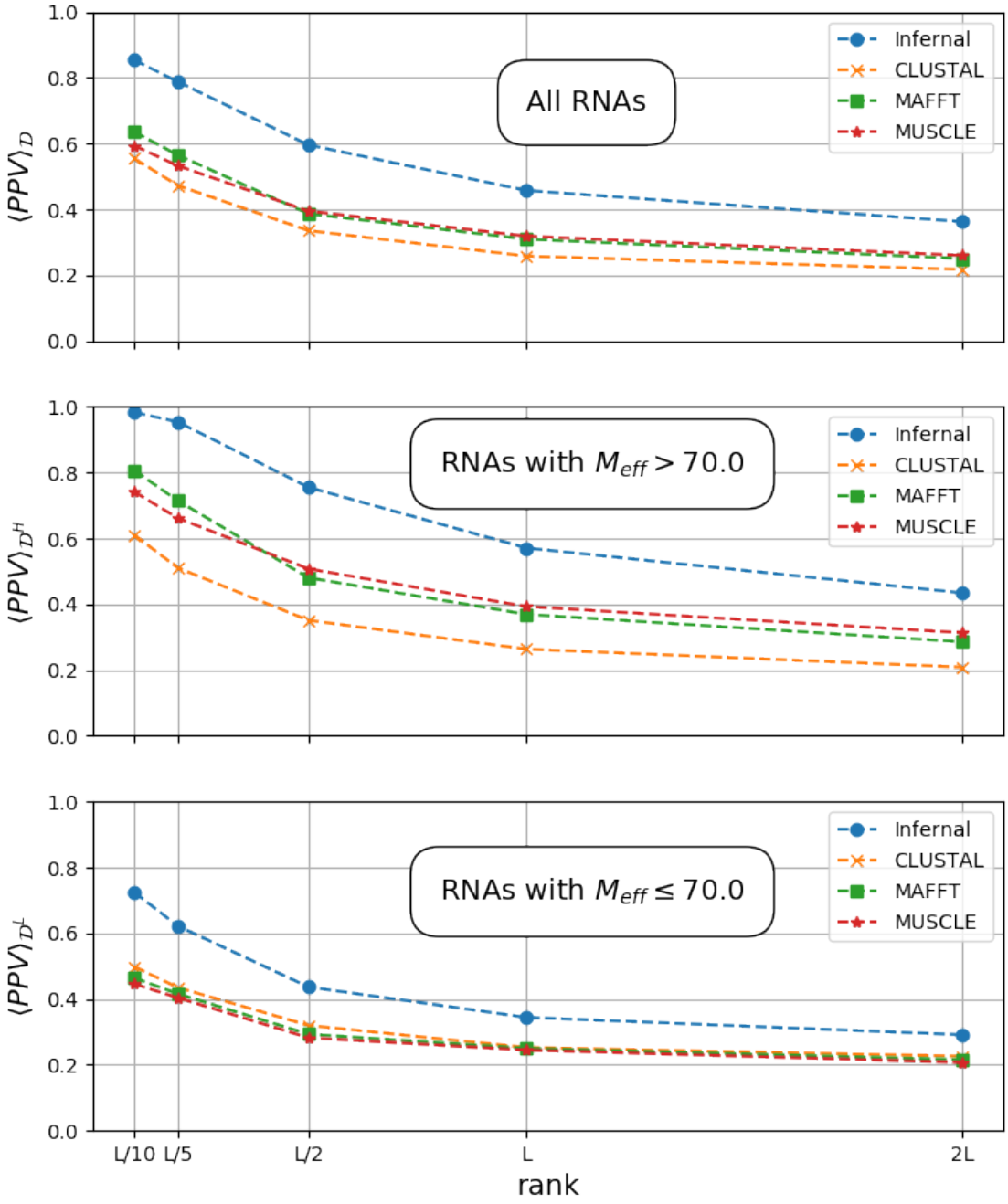


Figure **5-3**.: Figure comparing MSA methods performance on mean-field DCA. The vertical axes represent average PPV values, and the horizontal axes represent ranks at which the PPVs are taken. In the topmost, middle and bottommost figures the averages are performed over datasets $\mathcal{D}$—all RNA; $\mathcal{D}^H$—RNA with $M_{eff} > 70.0$; and $\mathcal{D}^L$—RNA with $M_{eff} \leq 70.0$, respectively. In all dataset categories, MSA obtained using Infernal results in best PPV across all ranks.

Fig.**5-3** shows the average positive predictive values $\langle PPV \rangle$ obtained using mean-field DCA operating on MSA input data generated by using the four sequence alignment methods. The averages are performed over three dataset categories, i.e., all the 57 RNAs ($\mathcal{D}$); RNAs whose families contain effective number sequences are more than 70.0 ($\mathcal{D}^H$); and RNAs whose effective number of sequences are less than or equal to 70.0 ($\mathcal{D}^L$). In all three dataset categories, the average PPV obtained from Infernal alignments results in the highest PPV values than alignments obtained using the other three methods. It may not be surprising—Infernal uses structural information in its seed alignments where parameters of covariance models are computed. Two of the alignment methods—MAFFT and MUSCLE— provide comparable average PPV when the effective number of sequences is high—$M_{eff} > 70.0$. Three of the alignment methods—CLUSTAL, MUSCLE, and MAFFT—result in low and comparable average PPV when there are no sufficient homologous sequences in an RNA family (see bottommost subfigure of Fig. **5-3**). Likewise, Infernal alignments result in low but better PPV than the other three alignment methods.

The average PPV depends on the rank it is taken and generally decreases as the rank gets lower and lower, indicating that high-ranked nucleotide pairs are more likely to be in proximity within RNA 3D structure than low-ranked nucleotide pairs. Taking rank L—length of RNA sequences—the average PPVs in dataset $\mathcal{D}$ are: 45.8% for Infernal; 31.9% for MUSCLE; 31.0% for MAFFT; and 25.8% for CLUSTAL. In dataset $\mathcal{D}^H$ the respective average PPVs at the same rank are: 57.1%, 39.3%, 36.9%, and 26.4% for Infernal, MUSCLE, MAFFT, and CLUSTAL. Interestingly, the average PPV obtained from alignments performed by MUSCLE, MAFFT, and CLUSTAL are nearly the same across all ranks, down to $2L$. However, Infernal provides much better predictions than the other three alignment methods even in the absence of a sufficient number of sequences—e.g., when $M_{eff} \leq 70.0$ in the Fig.**5-3**.

Generally, many effective sequences in an RNA family can result in more accurate predictions than a smaller number of effective sequences. However, this may not be necessarily always true—in addition to the effective number of sequences, the quality of the alignment can significantly affect contact prediction from MSA data. To evaluate the effect of MSA quality on average PPV, we plotted the average PPV versus M_{eff}, and BIT score of each RNA in the dataset (see Fig. **5-4**). In the Figure, we see that some RNA families have high M_{eff} but low average PPV than those with low M_{eff}; some RNAs have low M_{eff} but high average PPV (Fig. **5-4** A). The BIT score may provide additional information about such controversy. In Fig. **5-4** B, we see that there are RNAs with low M_{eff} (colored in red) that have high average PPV when the BIT score is also high—greater than 50 in this case.

To sum up, a high number of effective sequences in RNA generally result in good contact prediction. On the contrary, properly aligned sequences can result in good contact prediction even when the effective number of sequences is not large enough. Theoretically, one expects the number of effective sequences to be significantly larger than $\mathcal{O}(5^2)$ to observe sufficient information in nucleotide pair coevolution from RNA multiple sequence alignments. Never-

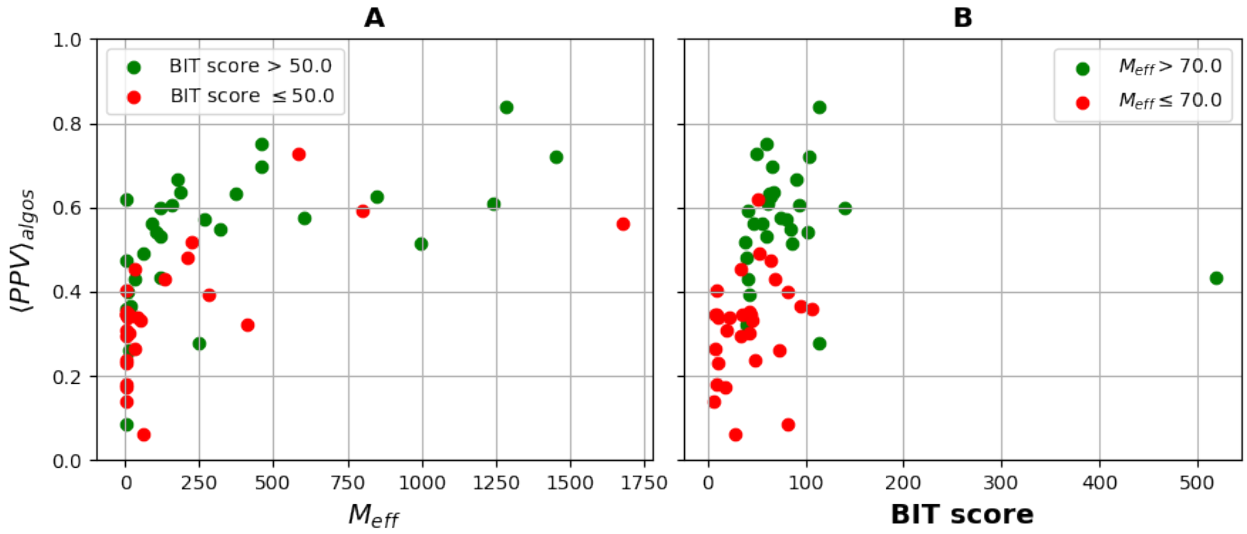


Figure **5-4**.: Dependence of nucleotide contact positive predictive value $\langle PPV \rangle$ on effective number of sequences (M_{eff}) and BIT score. The averages are taken through the four DCA-based algorithms implemented in pydca, mfDCA; EVCouplings, plmDCA; Boltzmann learning; and PSICOV, graphical LASSO. A) $\langle PPV \rangle_{algos}$ vs M_{eff}:green RNAs whose BIT score is– greater than 50.0, and red less than or equal to 50.0. B) $\langle PPV \rangle_{algos}$ vs BIT score:green RNAs whose M_{eff} is– greater than 70.0, and red less than or equal to 70.0.

theless, a small number of sequences in the RNA family can be sufficient to carry out contact prediction if an RNA is evolutionary closely related to the sequences in the family. That is, the BIT score resulting from matching an RNA family for a reference/target sequence is large enough. In the next section, we compare the four DCA algorithms' ability to predict specific RNA nucleotide pairs.

5.4. Contact prediction by nucleotide pair

Here we analyze RNA contact prediction accuracy by nucleotide pair. It provides insight into pairs of RNA nucleotides that show a great deal of coevolution due to evolutionary pressure applied to them during evolution. First, we compare the percentages of nucleotide pairs correctly predicted by the four DCA algorithms, namely, mfDCA of pydca; graphical LASSO of PSICOV; plmDCA EVCouplings; and Boltzmann learning. Then we look at the average PPV—by nucleotide pair—of each algorithm at fixed ranks.

Percentage of correct predictions

I evaluate which nucleotide pairs are more likely to coevolve by computing the relative percentage of correctly predicted nucleotide pairs using the four DCA algorithms (see Fig.**5-5**). Results in the Figure are obtained by taking top L nucleotide pairs from each RNA family DCA prediction in the dataset. Secondary structure base pairs dominate the percentage of correctly predicted nucleotide pairs—in all of the four DCA algorithms, canonical secondary structure base pairs account for about half of the total correctly predicted base pairs.

Looking at each DCA algorithm, the canonical secondary structure base pairs C-G account for 34% in mean-field pydca and Boltzmann learning, 37% in PSICOV, 33% in EVCouplings of the total correctly predicted top L ranked pairs in our dataset. The other canonical base pairs, A-U, represent the second-highest correctly predicted base pairs—accounting 19% for mean-field pydca, EVCouplings, and Boltzmann learning, and 20% for PSICOV. The non-canonical secondary base pairs G-U are the third most base pairs of top L ranked correctly predicted base pairs—it accounts for 11% of the total correctly predicted base pairs. Both canonical and non-canonical secondary structure base pairs make up about two-thirds of the total correct predictions at rank L in the dataset:64% for mean-field pydca, and Boltzmann learning; 67% for PSICOV; and 63% for EVCouplings.

Nucleotide pairs that are not commonly involved in RNA secondary structure formation makeup only a third of correctly predicted pairs at rank L in the RNA dataset. A-G pair makes up the most significant portion of correctly predicted pairs in this category, representing 9% in mean-field pydca, EVCouplings, Boltzmann learning, and 8% in PSICOV. Nucleotide pairs A-C and G-G are the second most prevalent correctly predicted pairs. A-C pair makes up 6% in both mean-field pydca and PSICOV; 8% and 7% in EVCouplings, and Boltzmann learning, respectively. G-G pair comprises of 6% top L correct predictions in

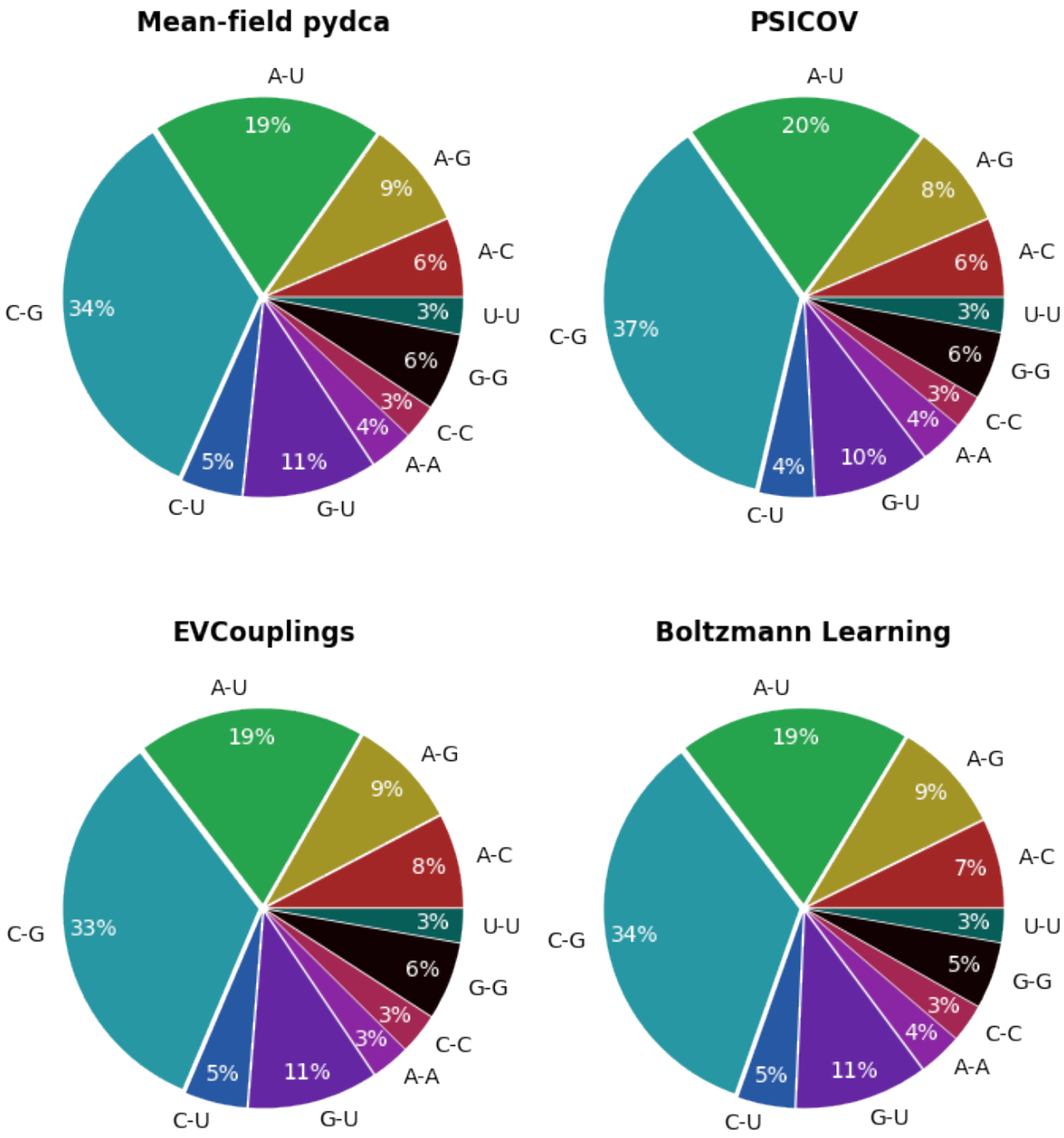


Figure **5-5**.: The relative percentage of correctly predicted nucleotide pairs computed at rank L of each RNA family. In all of the four DCA algorithms, secondary structure base pairs dominate the fraction of correctly predicted base pairs–canonical base pairs accounting for about half of the dataset's total predictions.

mean-field pydca, PSICOV, and EVCouplings, whereas Boltzmann learning constitutes 5% the same nucleotide pair. C-U pairs account for 5% of the total correct predictions in all but PSICOV, where the nucleotide pair makes up 4%. Similarly, the A-A pair makes up 4% of the nucleotide pairs in all of the four DCA algorithms except EVCouplings, which comprises 3% top L ranked correct predictions. Finally, the C-C pair represents 3% in all of the first L ranked correct predictions in all of the four DCA algorithms.

Comparing percentage of nucleotide pairs in the dataset (Fig. **5-1**) and those of correct predictions at rank L (Fig. **5-5**) abundance of a nucleotide pair in the dataset does not necessarily mean high rate of prediction. For instance, the A-G pair makes up 15% in the dataset—in equal proportion with C-G pairs and more than A-U pairs. Nevertheless, the A-G pair makes up only 9% or 8% of top L ranked correct predictions, whereas C-G and A-U pairs have way higher rates of predictions in all of the four DCA algorithms. It indicates that nucleotide pairs with high DCA scores result from coevolution as opposed to occurring randomly.

In summary, the coevolution signal of nucleotide pairs in our RNA dataset shows that secondary structure base pairs of standard nucleotide—both canonical and wobble pairs—show a strong trace of coevolution during evolution. The application of DCA-based algorithms for RNA contact prediction from multiple sequence alignments of RNA families indicates this RNA secondary structure base-pair coevolution pattern dominating other nucleotide pairs. Although the DCA-based methods have various algorithmic implementations, all of them show similar performance patterns to detect coevolution based on nucleotide pair type. In all of the four DCA algorithms used in this dataset, about two-third of correctly predicted nucleotide pairs down to rank L are composed of secondary structure nucleotide pairs. In the next section, I evaluate each of the four DCA algorithms' performance across several ranks for both overall and tertiary contact prediction.

Positive predictive values

In the above subsection, we have looked at the percentages of nucleotide pair types correctly predicted by DCA algorithms. The analysis provides quantified information about which nucleotide pair types are more likely to be true positives instead of incorrect predictions. Here we analyze the average positive predictive value of each nucleotide pair type using the four DCA algorithms. The averages are done on RNA families that a particular DCA algorithm provides at least a pair of nucleotide within a given rank— it is possible that a pair x-y where x and y are elements of {A, C, G, U} may not exist above a certain rank of an RNA nucleotide pair prediction. In this instance, that RNA is not included in the averaging.

The average positive predictive values of each specific nucleotide pair are displayed in Fig. **5-6** at ranks L (top) and $2L$ (bottom). As expected, the average PPVs are lower at rank $2L$ than L for each nucleotide pair and every DCA algorithm used here. Roughly speaking, the

PPVs range is 40–70% at rank L and 30–45% at rank $2L$. Also, secondary structure base pairs A-U and C-G show superior average PPV than the rest of the base pairs. Nevertheless, each particular DCA algorithm performs slightly differently in many nucleotide base pairs at both ranks.

Considering rank L (top Figure in Fig. **5-6**), EVCouplings performs better than other algorithms for pairs A-C, A-U, and U-U, whereas mean-field pydca shows better predictions for pairs C-U and G-U. Boltzmann learning and mean-field pydca show comparable performance for pairs A-C, A-U, C-G, and U-U. Except for PSICOV, all other three algorithms show no significant difference for nucleotide pair C-G. PSICOV shows the least accuracy for all pairs except C-C—where it performs in a tie with EVCouplings—and the two pairs C-U and A-A—where PSICOV and Boltzmann learning perform closely the same.

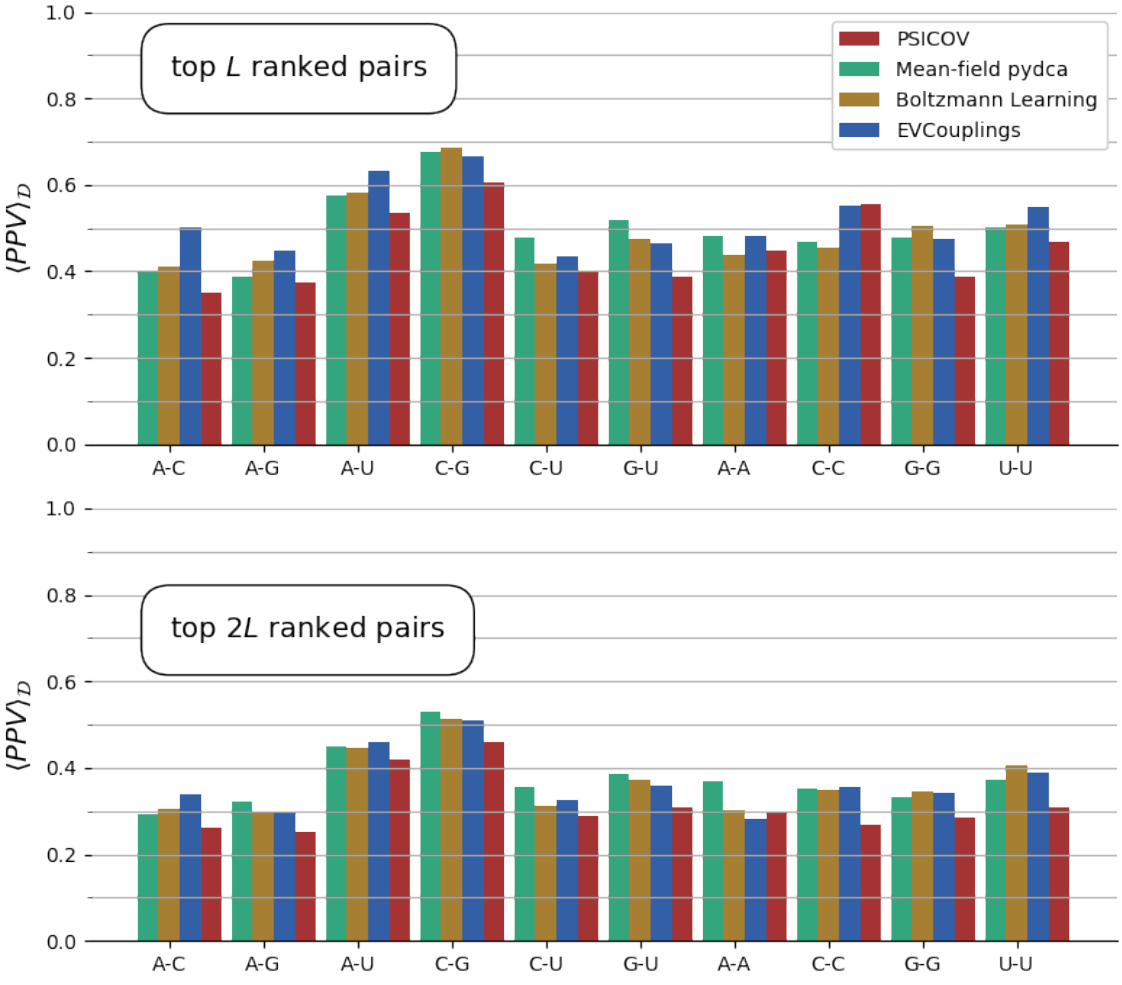


Figure **5-6**.: Average positive predictive values by nucleotide pair type computed using the four DCA algorithms. Top—the PPV is computed at rank L, and bottom—the PPV is computed at $2L$. In both cases, secondary structure pairs A-U and G-C are more accurately predicted by the algorithms than other nucleotide pairs.

At rank $2L$ (bottom Figure in Fig. **5-6**), the performance differences of EVCouplings, Boltzmann learning, and mean-field pydca are not significant for most nucleotide pairs. However, EVcouplings still show a slightly more accurate prediction for pairs A-C and A-G. Boltzmann learning is more accurate for pair U-U than the rest. Interestingly, mean-field pydca is slightly more accurate for a few nucleotide pairs, namely, C-G, C-U, G-U, and A-A. PSICOV is the least performing algorithm for all nucleotide pairs except A-A, which performs slightly better than EVCouplings and in a tie with Boltzmann learning.

In conclusion, nucleotide pairs A-U and G-C are more accurately predicted—as measured by average positive predictive values—than others by all four DCA algorithms. Although performance differences may exist at higher ranks, the three DCA algorithms, namely, mean-field pydca, EVCouplings, and Boltzmann learning, show comparable accuracy. However, PSICOV is the least accurate of the four DCA algorithms in our test.

5.5. Overall contact prediction accuracy

Here, I compare the performance of four DCA algorithms on the RNA dataset described in Section 5.2 using average positive predictive value as a metric. The comparison is made at various ranks of nucleotide base pairs sorted by DCA score in descending order, i.e., pairs with significant scores appear at the top. Each algorithm's prediction capability will be done on overall contact prediction accuracy—no distinction based on nucleotide pair or type of contact such as secondary structure base pair or otherwise. Like in the previous section, we will keep the dataset classification into three categories based on the number of effective sequences an RNA has in its family: all RNA data, dataset $\mathcal{D}$; RNAs whose M_{eff} is greater than 70.0, dataset $\mathcal{D}^H$; and RNAs whose M_{eff} is less than or equal to 70.0, dataset $\mathcal{D}^L$. I also compare the mean-field pydca algorithm with the other DCA algorithms' performance for each of the RNA families in the dataset $\mathcal{D}$. Finally, I study how each DCA algorithm's contact prediction accuracy is influenced by the value of nucleotide contact cut-off distance.

Average positive predictive value

Average positive predictive values, $\langle PPV \rangle$, as a function of rank expressed in terms of RNA sequence length L is plotted in Fig. **5-7** for the three dataset categories. The $\langle PPV \rangle$ monotonically decreases for the four DCA algorithms. Nevertheless, there are slight differences between each algorithm's performance (see Table **5-1**). For instance, the $\langle PPV \rangle$ at rank $L/10$ in dataset $\mathcal{D}$ are: 83.9%, mean-field pydca; 83.1%, EVCouplings; 81.6%, PSICOV; and 80.9%, Boltzmann learning. Interestingly, mean-field DCA shows slightly better performance at this rank than the rest of DCA algorithms. At rank, $L/5$ EVCouplings performs better than the rest of the algorithms with an average PPV of 78.3% while its closest competitor at the same rank is mean-field pydca with a PPV of 77.4%. In contrast, PSICOV and Boltzmann learning, respectively, indicate average PPVs 76.3% and 75.3%. At ranks,

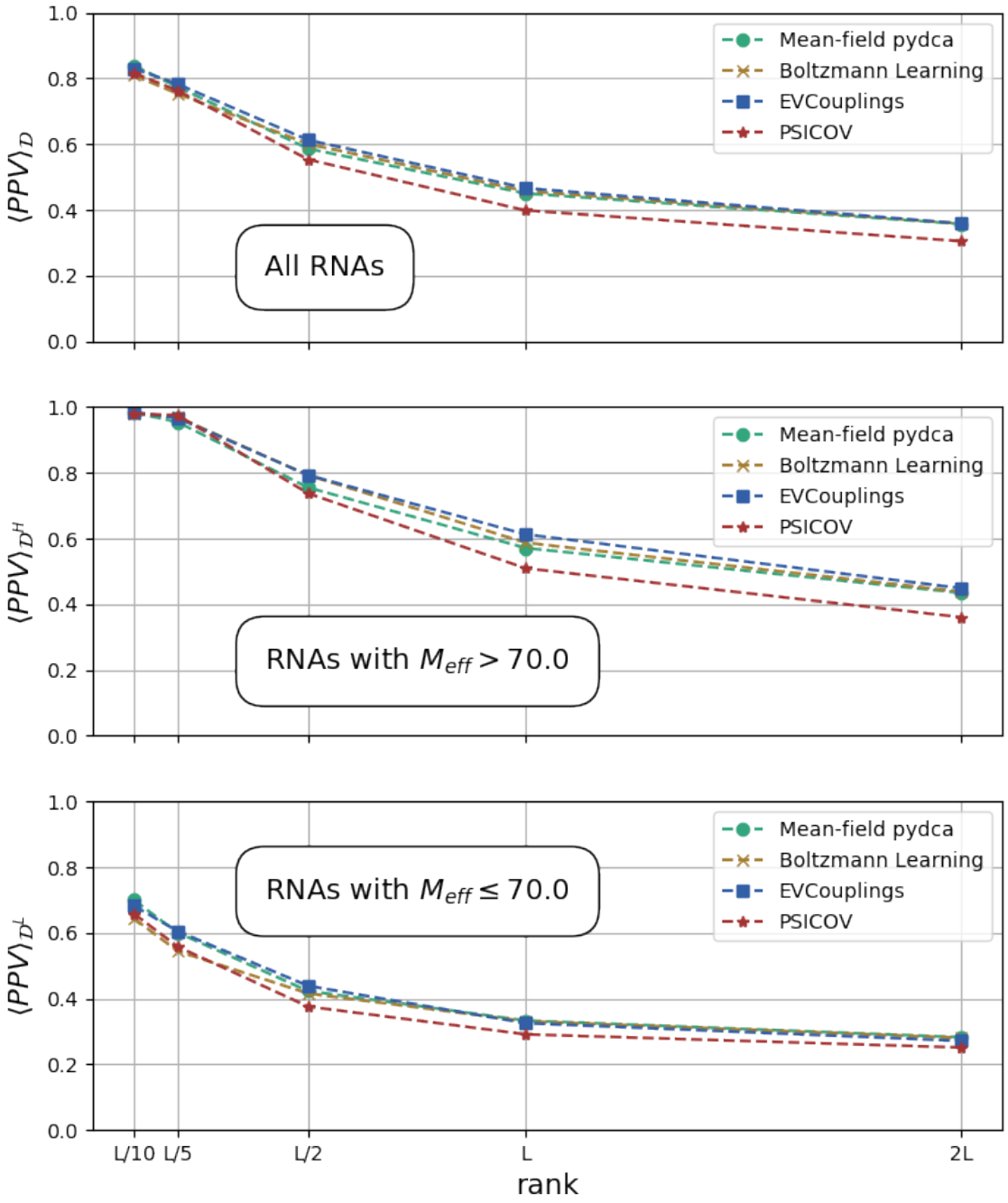


Figure **5-7**.: Average PPV computed using the four DCA algorithms as a function of RNA sequence length L. In the topmost Figure, PPVs are averaged over all RNAs considered in this study; the middle Figure is only for RNAs whose effective number of sequences greater than 70.0; and in the bottom Figure are RNAs whose effective number of sequences less than or equal to 70.0. There is a negligible difference between mean-field pydca, EVCouplings, and Boltzmann learning DCA algorithms, while PSICOV performs slightly less accurate than the other three.

$L/2$, L, and $2L$ EVCouplings perform slightly better than other algorithms. Indeed at rank $2L$, all of the DCA algorithms show similar average PPV except that of PSICOV, where it is about 5% less accurate than others.

Considering the dataset $\mathcal{D}^H$, which contains RNAs whose effective number of sequences is greater than 70.0, we see a significant improvement of average PPVs across all ranks. For instance, at rank, $L/10$, all of DCA algorithms show an average PPV of about 98%—more than by about 14% compared to the best performing algorithm in dataset $\mathcal{D}$ at the same rank. It shows the importance of an RNA family to have a large number of effective homologous sequences. At rank L the average PPVs are:57.1%, mean-field pydca; 61.3%, EVCouplings; 58.7%, Boltzmann learning; and 50.9%, PSICOV. Among the DCA algorithm, PSICOV is the least accurate method performing about 10% less than EVCouplings.

Table **5-1**.: Table showing the numerical value of average positive predictive values $\langle PPV \rangle$ as a function of rank expressed in terms of RNA sequence length L. The averages are done over the three categories of datasets—$\mathcal{D}$, $\mathcal{D}^H$, and $\mathcal{D}^L$—that are grouped based on the number of effective sequences M_{eff}. Across all ranks, all of the four DCA algorithms perform better in dataset D^H. At rank, L mean-field pydca, EVCouplings, and Boltzmann learning show comparable performance. PSICOV is the least accurate method by up to about 5% than the other three.

All RNA—Dataset $\mathcal{D}$					
Algorithm	$L/10$	$L/5$	$L/2$	L	$2L$
mean-field pydca	83.9	77.4	58.6	45.0	35.7
EVCouplings	83.1	78.3	61.3	46.6	35.9
Boltzmann learning	80.9	75.3	60.1	45.7	35.8
PSICOV	81.6	76.3	55.4	39.9	30.5
RNA with $M_{eff} > 70.0$—Dataset $\mathcal{D}^H$					
mean-field pydca	98.3	95.4	75.5	57.1	43.5
EVCouplings	98.5	96.7	79.3	61.3	44.9
Boltzmann learning	98.0	97.0	79.3	58.7	43.8
PSICOV	98.0	97.5	73.7	50.9	36.1
RNA with $M_{eff} \leq 70.0$—Dataset $\mathcal{D}^L$					
mean-field pydca	70.0	60.1	42.3	33.3	28.3
EVCouplings	68.2	60.5	43.9	32.5	27.1
Boltzmann learning	64.4	54.4	41.6	33.1	28.1
PSICOV	65.8	55.8	37.6	29.1	25.1

When there is a lack of sequences in RNA families as in dataset $\mathcal{D}^L$ the prediction capability of DCA algorithms is negatively affected, i.e., the average PPVs are smaller across all

ranks. For instance, at rank $L/10$, the average PPVs are about 98% in dataset $\mathcal{D}$ computed using the four DCA algorithms. The best average PPV at the same rank in dataset $\mathcal{D}^L$ is the only 70%. At rank L the average PPVs in dataset $\mathcal{D}^L$ are: 33.3%, mean-field pydca; 32.5%, EVcouplings; 33.1%, Boltzmann learning; and 29.3%, PSICOV— less than 10–12% compared to the corresponding average PPVs in dataset $\mathcal{D}^H$.

The cut-off distance between two nucleotides to be considered contacts is taken to be 10Å for a pair of heavy atoms (see Section 5.2) in PDB structures. Nevertheless, we investigated how average positive predictive values are influenced by this distance (see Fig. **5-8**). The figure shows APPV at rank L of the four DCA algorithms—the averages are taken over the total RNAs in the dataset—as a function of contact distance r_c. For $r_c = 4, 6, 8,$, and 10Å EVCouplings is slightly more accurate than mean-field pydca and Boltzmann learning (see Table **5-1** for the exact PPV for $r_c = 10$Å). In all cases, PSICOV is the least accurate of all. Above 10Å mean-field pydca and EVCouplings perform close to each other while Boltzmann learning is slightly lesser than the two. However, in practice, these distances are too large to be taken as contact distance cut-off values.

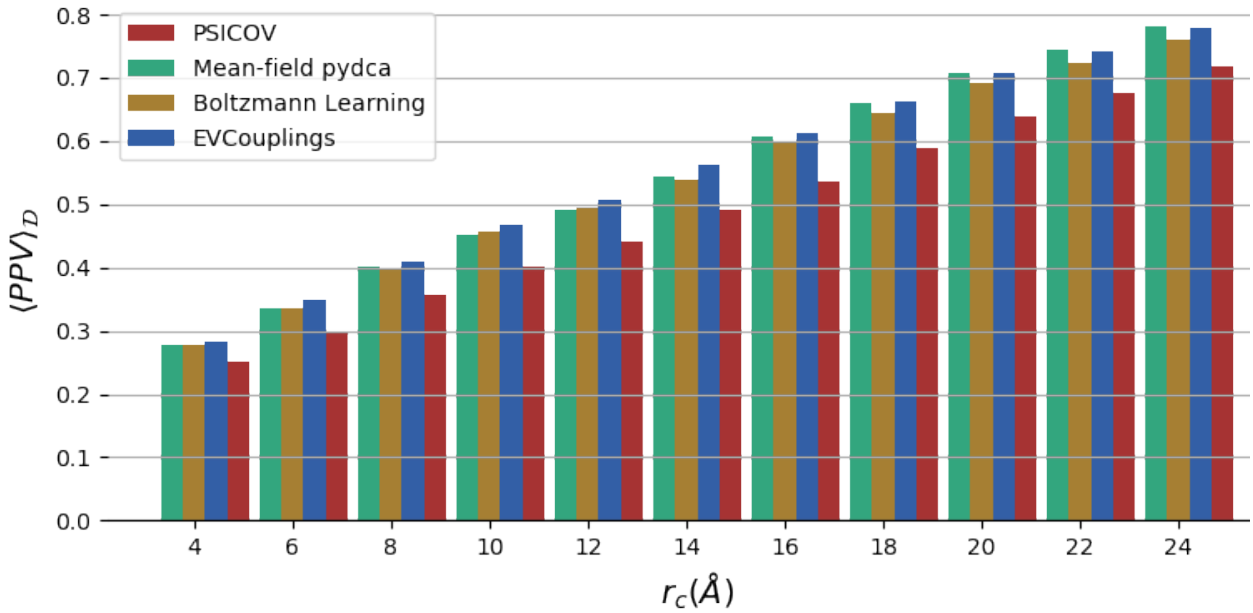


Figure **5-8**.: Average positive predictive values $\langle PPV \rangle$ at rank L of the DCA algorithms as a function of contact distance r_c measured in Angstrom. The averages are taken over all RNAs in the dataset $\mathcal{D}$. All four algorithms show they improve $\langle PPV \rangle$ as r_c is increased. Below 10Å the algorithms except PSICOV show comparable performances, although EVCouplings is slightly more accurate than mean-field pydca and Boltzmann learning.

Individual RNAs positive predictive values

Here I look at the performance of DCA algorithms on individual RNAs. Instead of computing the average positive predictive values over the RNA families, we compare each RNA's positive predictive value. I also analyze how each RNA computed PPV using mean-field pydca deviates when compared with PPVs obtained using other DCA-based algorithms.

The PPV at rank L—where L is RNA sequence length—of each RNA in dataset $\mathcal{D}$ is compared in Fig. **5-9**. The horizontal lines show PPV of RNAs when computed using mean-field DCA. The vertical lines are obtained using EVCouplings, Fig.**5-9** (A); Boltzmann learning Fig. **5-9**(B); PSICOV, Fig. **5-9**(C); and pseudo-likelihood maximization DCA implemented in pydca Fig. **5-9**(D). The diagonal lines show values whereby the vertical and horizontal axes are equal. The blue dots are PPVs of RNAs. If PPVs of RNAs computed using mean-field pydca and other algorithms are equal, they must lie diagonally. Deviations from the diagonal lines indicate that an algorithm is more accurate than the other. Specifically, PPVs that lie below diagonal lines imply that mean-field pydca is more accurate than the algorithms.

The DCA algorithms are compared with each other by truncating the PPVs to three significant digits. If the difference between PPVs is more than 0.001, they are considered different, otherwise the same. Mean-field pydca and EVCouplings algorithms estimate the same PPV for three RNAs, whereas PPVs computed using EVCouplings are superior to mean-field DCA for 36 RNAs. Similarly, the PPV of 18 RNAs is larger for mean-field pydca than that of EVCouplings. Comparing mean-filed pydca with Boltzmann learning, the PPV of three RNAs is identical, while 24 RNAs PPVs are better predicted by mean-field pydca and that of 30 RNAs by Boltzmann learning. Like mean-field pydca and EVCouplings or Boltzmann learning, PSICOV and mean-field pydca provide equivalent PPVs for three RNAs. Mean-field pydca provides better PPVs for 38 RNAs, whereas PSICOV more accurately predicts 16 RNA's PPV than mean-field pydca.

In addition, we compared (see Fig. **5-9**(D)) the PPVs computed using pseudo-likelihood maximization and mean-field algorithms implemented in pydca. This comparison shows that the two DCA algorithms provide similar PPVs for 5 RNAs. However, mean-field pydca provides better PPVs for 29 RNAs, whereas pseudo-likelihood maximization results in more PPVs for 23 RNAs than mean-field.

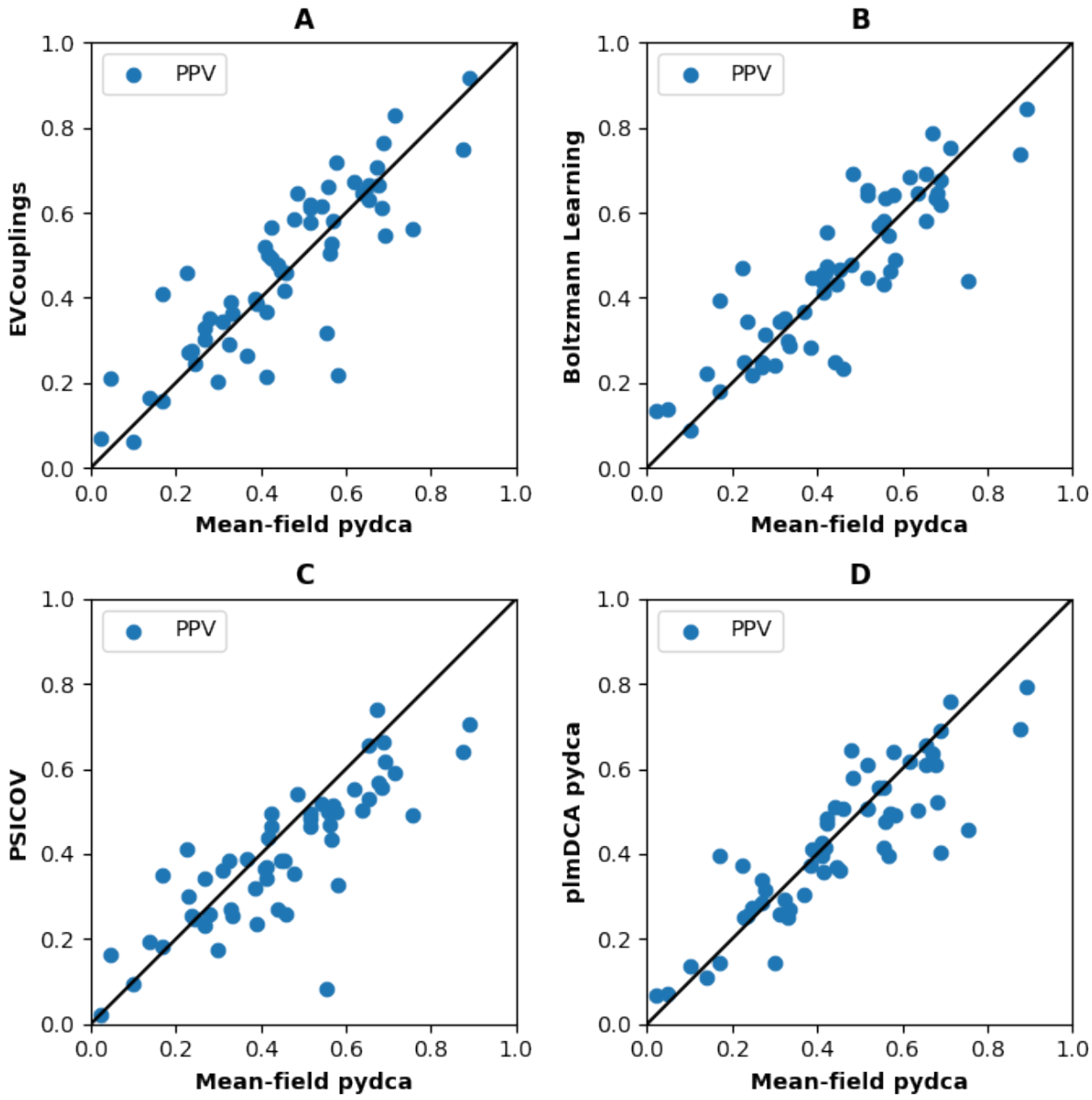


Figure **5-9**.: Contact prediction accuracy comparison of mean-field DCA with other algorithms. The PPVs are computed at rank L, where L is the RNA sequence length of each RNA in dataset $\mathcal{D}$. The horizontal axes are values of mean-field pydca PPV, whereas the vertical axes those of EVCouplings (A); Boltzmann learning (B); PSICOV (C); and pseudo-likelihood DCA of pydca (D). The diagonal lines indicate the values that the horizontal and vertical axes are equal. When there is no difference in PPV between mean-field pydca and other DCA algorithms, the PPVs lie on the diagonals. Note that, in this comparison, contacts are taken without making a distinction between tertiary contacts or otherwise.

5.6. Tertiary contact prediction accuracy

In this section, I evaluate the performances of DCA algorithms based on tertiary contact prediction. Recall that in Section 5.5 we evaluated DCA algorithms' performances without making a distinction between the nature of nucleotide pair contacts. As we described in Section 5.2, tertiary contacts are nucleotide pairs that are not in proximity in the sequence (primary structure) of an RNA. Also, tertiary contacts do not involve pairs that form secondary structures and are not in the vicinity of secondary structure pairs. Like in Section 5.5, first, I look at average positive predictive values of the DCA algorithms. Then, I compare mean-field pydca with other DCA algorithms based on their ability to detect individual RNAs' tertiary contacts. Finally, we look at how the average PPVs behave when the contact cut-off distance is changed.

Average positive predictive values

The average positive predictive values for tertiary contacts as a function of rank measured in terms of sequence length are compared in Fig. **5-10** and the corresponding numerical values are displayed in Table **5-2**. Considering all RNAs, i.e., dataset $\mathcal{D}$, Boltzmann learning results in best accuracy at high ranks, e.g., 39.8% at rank $L/10$. EVCouplings is the closest to Boltzmann learning at that rank with an average PPV of 36.2%. When the ranks are lowered, the performance difference among the algorithms diminishes. For instance, at rank, L[1] mean-field pydca, EVCouplings and Boltzmann learning perform almost the same, whereas PSICOV is slightly less accurate but very close to the other three.

Tertiary contact prediction accuracy of DCA algorithms is improved when the number of effective sequences in the alignment is large. For example, the average positive predictive values at rank $L/10$ are up by between about 11% to 15% when the averages are performed over dataset $\mathcal{D}^H$ compared to dataset $\mathcal{D}$. Among the four DCA algorithms, Boltzmann learning and EVCouplings perform better than others at high ranks. Nevertheless, except for PSICOV, all the DCA algorithm's performances are indistinguishable from one another at rank L, each scoring an average PPV of about 22%.

For dataset $\mathcal{D}^L$, i.e., $M_{eff} \leq 70.0$, Boltzmann learning performs slightly better than mean-field pydca and EVCouplings (20% vs 23%) at rank $L/10$. PSICOV is slightly less than the rest of the algorithms at that rank (about 17%). Like in dataset $\mathcal{D}^H$, the performance differences among mean-field pydca, EVCouplings, and Boltzmann learning diminishes at rank L. Although very close to other algorithms, PSICOV results in less average PPV. The variation of tertiary contacts average positive predictive values—the averages being performed over dataset $\mathcal{D}$—at rank L of the DCA algorithms as a function of contact distance (r_c) is shown in Fig. **5-11**. We see that the average PPV measured by mean-field pydca, EVCouplings, and Boltzmann learning are comparable for $r_c = 4, 6, 8$, and 10Å. For r_c

[1] At this rank, some RNAs have less number of tertiary contacts in their PDB structures, see Table **A-2**.

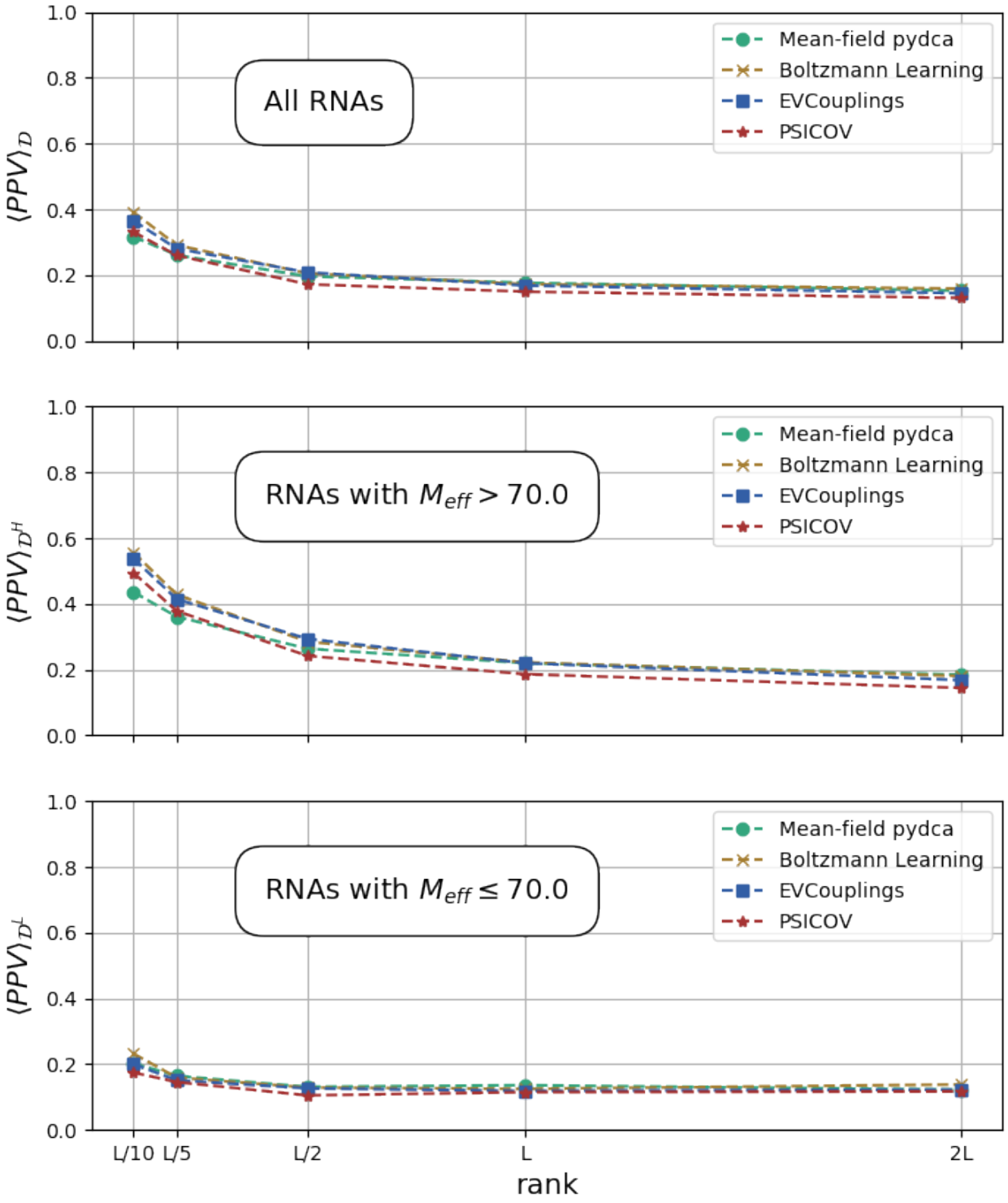


Figure **5-10**.: Tertiary contacts average positive predictive values $\langle PPV \rangle$ as a function of rank measured in terms of sequence's length L. Topmost, middle and bottom plots are for datasets $\mathcal{D}$, $\mathcal{D}^H$, and $\mathcal{D}^L$, respectively. At higher ranks, EVCouplings and Boltzmann learning perform better than mean-field pydca and PSICOV, particularly for dataset $\mathcal{D}^H$.

beyond 10Å mean-field pydca has the highest average PPVs; however, these distances are too large for nucleotide pairs to be considered contacts.

Table **5-2**.: Table showing the average positive predictive values $\langle PPV \rangle$ as a function of rank expressed in terms of RNA sequence length L. The averages are done over the three categories of datasets—$\mathcal{D}$, $\mathcal{D}^H$, and $\mathcal{D}^L$—that are grouped based on the number of effective sequences M_{eff}. Across all ranks, all of the four DCA algorithms perform better in dataset D^H. At rank, L mean-field pydca, EVCouplings, and Boltzmann learning show comparable performance. PSICOV is the least accurate method by up to about 5% than the other three.

All RNA—Dataset $\mathcal{D}$					
Algorithm	$L/10$	$L/5$	$L/2$	L	$2L$
mean-field pydca	31.7	26.1	19.6	17.7	15.3
EVCouplings	36.6	28.0	20.9	16.8	14.5
Boltzmann learning	39.2	29.1	20.6	17.3	15.9
PSICOV	33.2	26.0	17.2	15.0	13.1
RNA with $M_{eff} > 70.0$—Dataset $\mathcal{D}^H$					
mean-field pydca	43.6	36.1	26.4	22.0	18.4
EVCouplings	53.8	41.4	29.4	22.0	16.8
Boltzmann learning	55.7	42.8	28.5	22.2	18.1
PSICOV	49.5	37.8	24.2	18.6	14.4
RNA with $M_{eff} \leq 70.0$—Dataset $\mathcal{D}^L$					
mean-field pydca	20.2	16.4	13.1	13.6	12.2
EVCouplings	20.0	15.1	12.6	11.9	12.3
Boltzmann learning	23.2	15.9	12.9	12.5	13.8
PSICOV	17.5	14.5	10.5	11.5	11.7

Individual RNA positive predictive values

Now I look at how each DCA algorithm performs on tertiary contact prediction for individual RNAs in dataset $\mathcal{D}$. Specifically, we look at how each RNA's PPV at rank L deviates with respect to mean-field pydca. The comparisons are shown in Fig.**5-12** where the horizontal lines are PPVs as computed by mean-field pydca and the vertical lines are by: EVCouplings, Fig.**5-12**(A); Boltzmann learning, Fig.**5-12**(B); PSICOV, Fig.**5-12**(C); and plmDCA pydca, Fig.**5-12**(D). The diagonal lines are points where mean-field pydca's PPV are equal to the corresponding DCA algorithm's PPV.

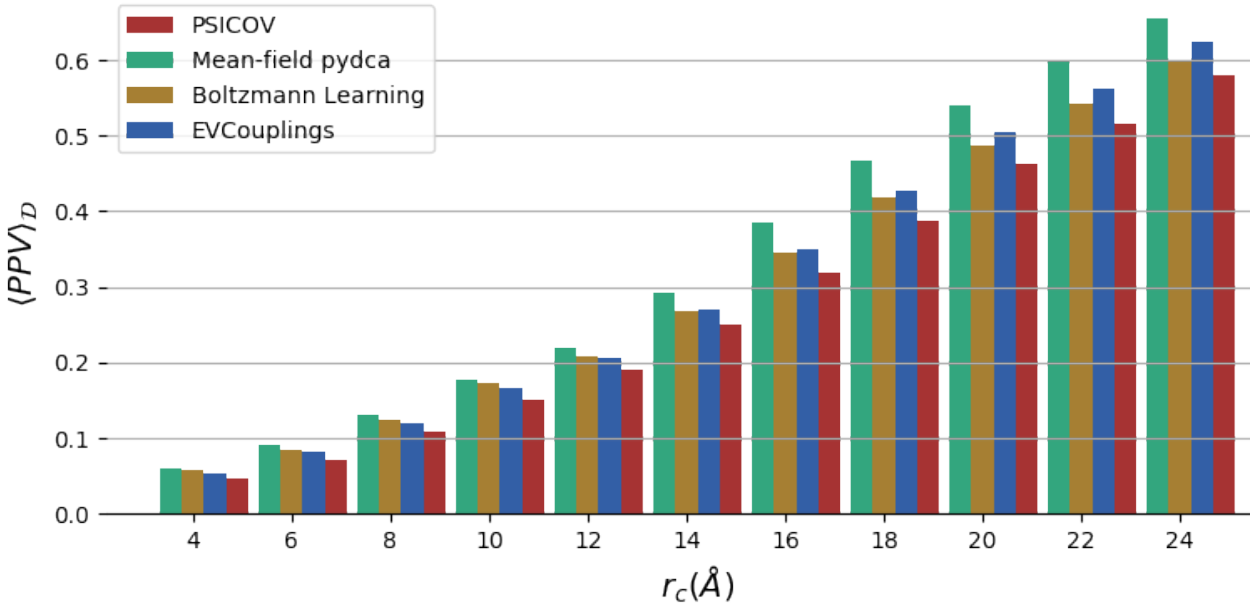


Figure **5-11**.: Tertiary contacts average positive predictive values at rank L as a function of contact distance cut-off value r_c. At $r_c = 4, 6, 8$ and 10Å, mean-field pydca, EVCouplings, and Boltzmann learning perform very closely, whereas PSICOV Is slightly less accurate. Above 10Å there are significant differences between the algorithm's performance; however, these values are too large to be considered contact distances.

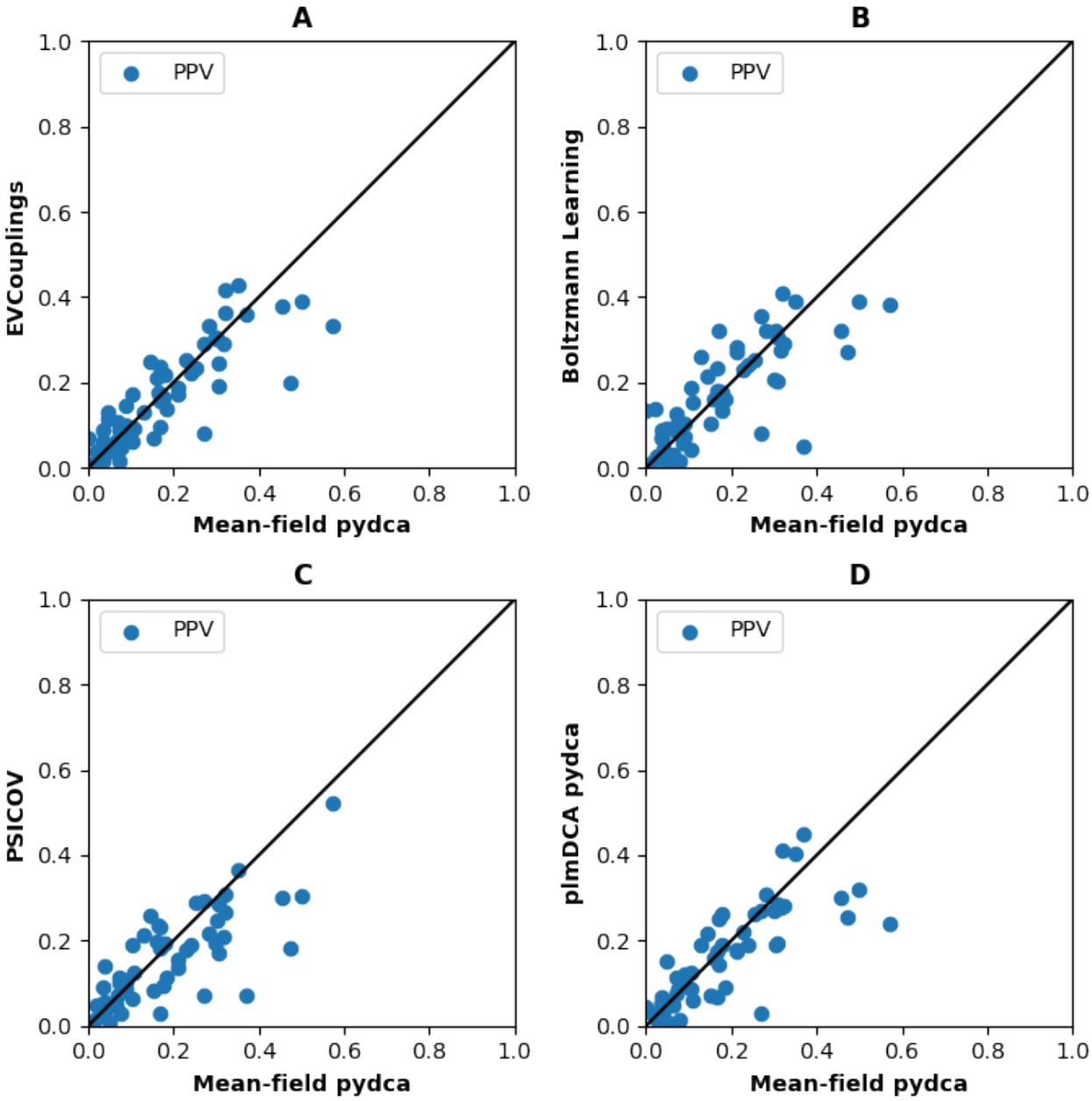


Figure **5-12**.: Contact prediction accuracy comparison of mean-field DCA with other algorithms for tertiary contacts. The PPVs are computed at rank L, where L is the RNA sequence length of the RNAs. The results show that the algorithms can have significant PPV variation for individual RNAs.

The RNAs whose PPVs are above/below the diagonal are counted and compared. PPVs are considered to be on the diagonal line, i.e., the mean-field pydca and other DCA algorithms perform the same if the difference between the PPVs is not more than 0.001. Mean-field pydca and EVCouplings predict equal PPVs for seven RNAs while EVCouplings than mean-field better predicts 26 RNA's PPVs, and other 24 RNAs PPVs are better predicted by mean-field pydca rather than EVCouplings. Likewise, comparing mean-field pydca and Boltzmann learning, three RNA's PPVs are the same for both algorithms, whereas Boltzmann learning predicted PPVs are better than mean-field pydca's for 26 RNA. However, the remaining 22 RNAs mean-field pydca's PPVs are better than Boltzmann learning's. Between PSICOV and mean-field pydca, both algorithms results in the same PPVs for nine RNAs while 26 RNA's PPVs predicted by mean-field pydca are better than that of PSICOV's, but PSICOV results in better PPVs for 22 RNAs compared to mean-field pydca.

Besides, I compared individual RNA's PPVs computed by pseudo-likelihood DCA of pydca and mean-field pydca. Although plmDCA in pydca and EVCouplings are the same algorithms, there are implementation differences. For instance, pydca's plmDCA is implemented using single precision while EVCouplings were compiled using a double-precision option. For individual RNA's tertiary contact prediction, 9 RNA's PPVs are the same when computed using mean-field pydca or plmDCA pydca, whereas 26 RNA's PPVs are better when using mean-field pydca than plmDCA pydca. However, plmDCA pydca results in more PPVs for 22 RNAs than mean-field pydca.

5.7. Summary and conclusion

This chapter has compared the performance of various DCA-based contact prediction algorithms/software for RNA sequences. The algorithms include mean-field DCA implemented in pydca[106], pseudo-likelihood maximization DCA implemented in EVCouplings[**?**], Boltzmann learning implemented [21] and graphical LASSO algorithm of PSICOV[40]. To this end, we collected about 60 RNAs that belong to unique families in Rfam (see Section 5.2). Then, in Section 5.3 we evaluated how the input data —in the form of multiple sequence alignments—influences contact prediction algorithms. In the remaining sections, we assessed how DCA-based algorithm's performances compare with each other using positive predictive value as a metric. Specifically, in Section 5.4 we investigated contact prediction accuracy by nucleotide pair. Next, overall contact prediction ability of DCA algorithms was studied in Section 5.5 and then we did similar analysis for tertiary contacts only in Section 5.6.

The analysis shows that multiple sequence alignments obtained using Infernal [72] result in the four DCA-based algorithms' best positive predictive values. This might be because Infernal uses seed alignments that covariance models are built on have structural information about related RNAs in a family. Indeed top L ranked DCA predictions indicate that secondary structure pairs A-U and G-C make up about half of the correctly predicted nucleotide pairs. C-G makes up 33–37% of correct prediction, whereas A-U makes up 19–20%. Also,

these nucleotide pairs result in higher positive predictive values than others. In addition to the alignment method/algorithm used, DCA analysis of nucleotide contacts can significantly affect the quality and quantity of alignment data. RNAs that have a large effective number of sequences have mostly high positive predictive values. However, the extent to which the sequences are properly aligned also matters. In particular, we saw that RNAs with high BIT scores tend to results in more accurate contact predictions.

Looking at overall contact prediction—without classifying contacts as tertiary or otherwise—EVCouplings is slightly more accurate than mean-field pydca, and Boltzmann learning is at rank L. However, the difference in the performance of these three algorithms vanishes at lower ranks. For instance, at rank $2L$, they perform comparably, but PSICOV remains slightly less accurate than the other three. Also, the three algorithms' performances are similar at rank L for contact distance thresholds 4, 6, 8Å as they are at 10Å. Furthermore, the algorithms perform differently for each RNA family. Taking mean-field pydca as a reference algorithm and comparing its performance with the other DCA-based algorithms, the vast majority of the RNA's PPV is higher/lower instead of being close to each other. Only three to five RNAs result in the same PPV computed from mean-field pydca and other algorithms.

For RNA three-dimensional structure prediction, contacts that are far apart in the sequence often play a crucial role when used as constraints/restraints. Therefore, the importance of correctly predicting these tertiary contacts cannot be overstated. Among the algorithms, Boltzmann learning and EVCouplings result in high average positive predictive values at high ranks, particularly when the effective number of sequences is high. Nevertheless, the three algorithms–mean-field pydca, EVCouplings, and Boltzmann learning perform almost the same at rank L for contact cut-off values 4, 6, 8, and 10Å. In addition, there the algorithms perform differently to predict three-dimensional contacts for each RNA. Comparing the average PPVs at rank L of mean-field pydca and others shows that the same result can be obtained for a maximum of about a sixth of the RNAs.

To sum up, the performance of DCA-based algorithms contact predictions from homologous RNA sequences depends on multiple sequence alignment tools used and the quality and number of sequences in RNA families. Overall performance of three algorithms—mean-field, pseudo-likelihood maximization, and Boltzmann learning—is, on average, very comparable at rank L. However, pseudo-likelihood and Boltzmann learning tend to be more accurate at higher ranks than mean-field DCA, especially when the number of effective sequences is sufficiently large.

6. Enhancing RNA Contact Prediction by Convolutional Neural Networks

6.1. Introduction

In Chapter 5, I have compared various flavors of direct couplings analysis (DCA) algorithms for RNA contact prediction. In particular, four DCA-based algorithms have been applied on an RNA dataset that contains approximately 60 RNAs where nearly half of them have a relatively sufficient number of effective sequences and the other half do not.

A rigorous comparison of these unsupervised algorithms indicates that the performances measured by positive predictive values (PPV) have no significant differences for RNA contact prediction. However, there are slight variations in performance at higher ranks of predicted nucleotide pairs. The pseudo-likelihood maximization and Boltzmann learning show slightly better accuracy at higher ranks than mean-field and graphical LASSO—acronym for least absolute square and shrinkage operator—algorithms. While these two DCA algorithms show slightly better accuracy for about ten highest-ranking pairs for RNAs with a relatively sufficient number of effective sequences, the accuracy difference among all the four but graphical LASSO algorithms is invisible when assessed for the RNAs in the entire dataset.

This chapter combines an unsupervised machine learning algorithm based on mean-field DCA with supervised machine learning techniques based on convolutional neural networks to enhance RNA contact prediction from multiple sequence alignments. The new RNA contact prediction algorithm dubbed as **CoCoNet** is shown to significantly enhance RNA contact prediction as evaluated by various classification metrics.

The chapter is organized as follows. In Section 6.2 coevolutional structural patterns obtained from the dataset are analyzed. Next, in Section 6.3 I present the network architecture of the model before explaining how the model parameters are trained in Section 6.4. Then, the new algorithm results are presented and compared with state-of-the-art DCA algorithms in Section 6.5. Finally, I provide summary and conclusions in Section 6.6.

6.2. Coevolutional structural features

Contact prediction algorithms from multiple sequence alignments presume that mutations in sequences through the course of evolution occur in tandem to preserve the structure and

function of an RNA or protein. Algorithms such as DCA are more accurate in disentangling direct contacts from indirect contacts. Here, the DCA contact maps of the RNAs in the dataset are analyzed to study the relation between DCA scores and structural features in the experimental (PDB) structures of the RNAs.

The average mean-field DCA scores computed using a 7×7 window for various heavy atom distances between nucleotide pairs in PDB are shown in Fig. **6-1**. The strength of the average coevolutionary scores varies based on the distance between nucleotide pairs. Nucleotide pairs that are very close to each other in the PDB structure result in strong coevolutionary signals. On the other hand, distant nucleotide pairs in the three-dimensional structure show no specific patterns on their average DCA scores.

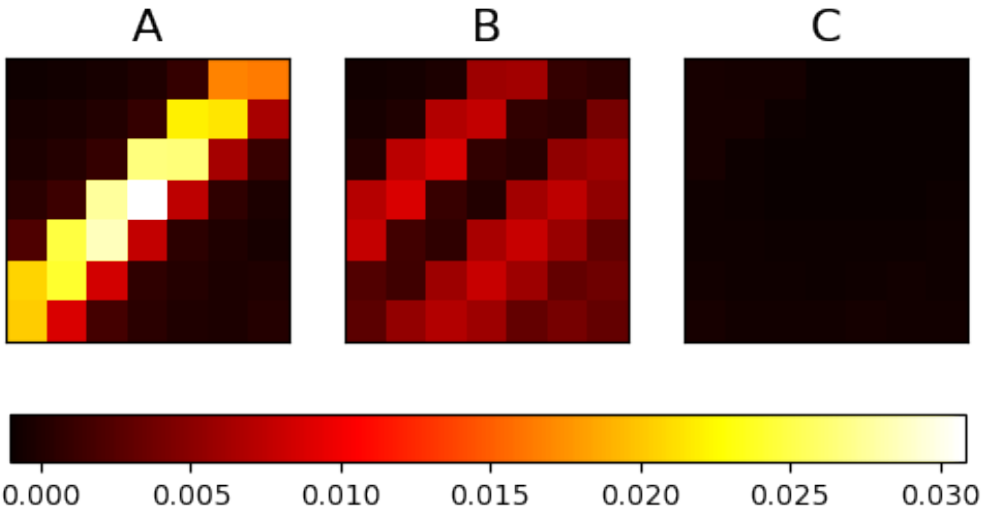


Figure **6-1**.: Patterns of average DCA scores in a 7×7 window for a range of distances, r. (A) $r \leq 4.0$ Å; (B) $4.0 < r \leq 10.0$ Å; (C) $r > 10.0$ Å. Brighter patterns show strong DCA scores and darker ones show weak DCA scores.

For small distances ($r \leq 4.0$ Å), the average DCA scores are strong across the diagonal and weaker outside. It indicates that RNA secondary structure nucleotide pairs are strongly coevolving as RNA sequences mutate during evolution. For intermediate distances ($4.0 < r \leq 10.0$ Å, Fig. **6-1** B), the average DCA scores are relatively strong surrounding the diagonal. However, for large distances ($r > 10.0$ Å, Fig. **6-1** C), there is no relative strength of DCA scores.

This analysis shows that DCA can detect coevolutionary signals that occur due to mutations arising on spatially adjacent nucleotide-nucleotide pairs. Based on this observation of coevolutionary signal patterns in the two-dimensional map of nucleotide-nucleotide pairs, adding a convolution layer to the network can enhance contact prediction accuracy. In the next section, I describe the new model that enhances contact prediction.

6.3. Network architecture

Deep convolutional neural networks are becoming very popular for learning patterns in images. However, two main drawbacks can limit their utilization. First, as convolutional neural networks get deeper and deeper, the training time also grows larger and larger and requires costly computational resources. Second, the number of parameters to be learned also grows fast. Learning a massive number of parameters requires a great deal of data to overcome overfitting.

In this study, deep neural networks are ruled out since the RNA dataset available for training them is very limited. Consequently, the convolutional layer is kept shallow and contains only a single layer (see Fig. **6-2**). The entire network contains the following layers:

- **Input layer**: This layer consists of an RNA sequence multiply aligned with its homolog RNA sequences from the RFAM database. The MSA data should be in FASTA format.
- **Coevolution layer**: The second layer is the coevolution layer. It involves the construction of a two-dimensional contact map from nucleotide-nucleotide pair interaction scores obtained from mean-field DCA.
- **Convolution layer**: The third layer is the convolution layer. Here, the convolution operation is performed on the coevolution layer using a $d \times d$ filter matrix. The coevolutional layer is padded to preserve the dimension of the output layer resulting from convolution operation.
- **Output layer**: The output layer is a two-dimensional contact map of nucleotide-nucleotide contacts annotated with interaction scores. The interaction scores are the result of doing convolution on the coevolution layer using filter matrices.

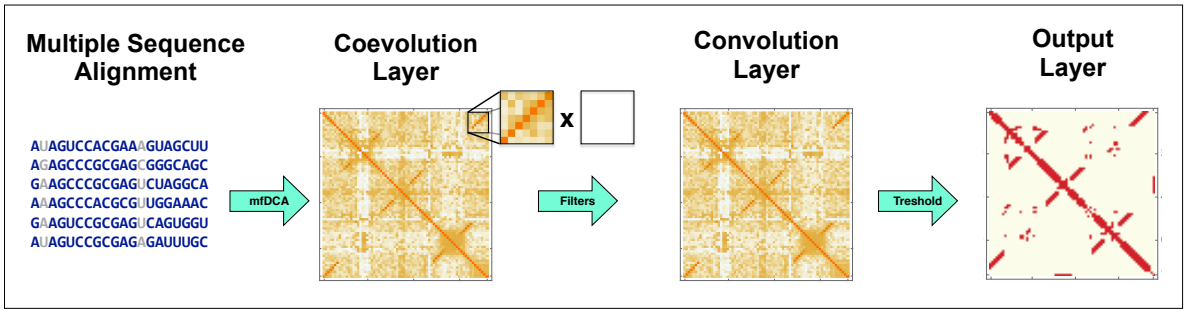


Figure **6-2**.: Network architecture of **CoCoNet**. There are four main layers of the network: i) the input layer containing input MSA data; ii) the coevolution layer involving DCA contact maps; iii) the convolution layer that results from doing convolution on the coevolution layer; iv) the output layer that contains contact maps of nucleotide pairs with high scores.

6.4. Learning parameters of filter matrices

The convolution layer parameters, i.e., filter matrices elements, are learned using a standard optimization procedure. First, nucleotide-nucleotide interaction scores are computed from MSA data of an RNA using the mean-field DCA algorithm implemented in the pydca software package [106]. From these interaction scores, a two-dimensional map of nucleotide-nucleotide contacts is formed. Then convolution operations are performed on this map by sliding the filter matrices over it. The resulting map is compared with the PDB contact map of the RNA, which is constructed by assigning a value of one for contacting nucleotide pairs and zero otherwise. Two nucleotides are considered contacts if they have at least a pair of heavy atoms less than 10.0 Å apart in the PDB structure.

The cost (objective) function associated with a particular RNA denoted by R in the training dataset is computed using

$$\mathcal{F}_{ij}^{R} = \left(\mathcal{WD}_{ij}^{R} - \delta(\mathcal{C}_{ij}^{R})\right)^2. \tag{6-1}$$

The indices i and j represent nucleotide pairs' positions in the target RNA and $\delta(\mathcal{C}_{ij})$ is one when the pairs are contacts, otherwise zero. The total cost function for a set of RNA data in the training set is obtained by summing the contributions in equation 6-1 over all pairs and RNA in the set, i.e.,

$$\mathcal{F} = \sum_{R} \sum_{j>i+4} \mathcal{F}_{ij}^{R}. \tag{6-2}$$

The summation over nucleotide pairs is performed only for those that are farther than four nucleotides apart in the RNA sequence since we are interested in long-range contacts. The optimization is performed using limited-memory BFGS algorithm [57, 74] implemented in the standard SciPy library [100]. Next, the results of contact prediction using the **CoCoNet** algorithm are presented. In addition, the performance of **CoCoNet** is compared with state-of-the-art DCA-based algorithms.

6.5. Results

In this section, the performance of **CoCoNet** is evaluated using two metrics: positive predictive value and Mathews correlation coefficient (MCC). Besides, **CoCoNet**'s performance is compared with DCA-based algorithms, including mean-field, pseudo-likelihood, graphical LASSO, and Boltzmann learning DCA algorithms.

Contact prediction evaluated by positive predictive value

Here, I show that the network improves RNA contact prediction significantly. Fig. **6-3** shows the average positive predictive value ($\langle PPV \rangle$) as a function of the number of contacts of

ranked nucleotide pairs. The Figure compares the $\langle PPV \rangle$ of mean-field DCA and **CoCoNet** with the theoretical value for all contact types, i.e, tertiary or otherwise.

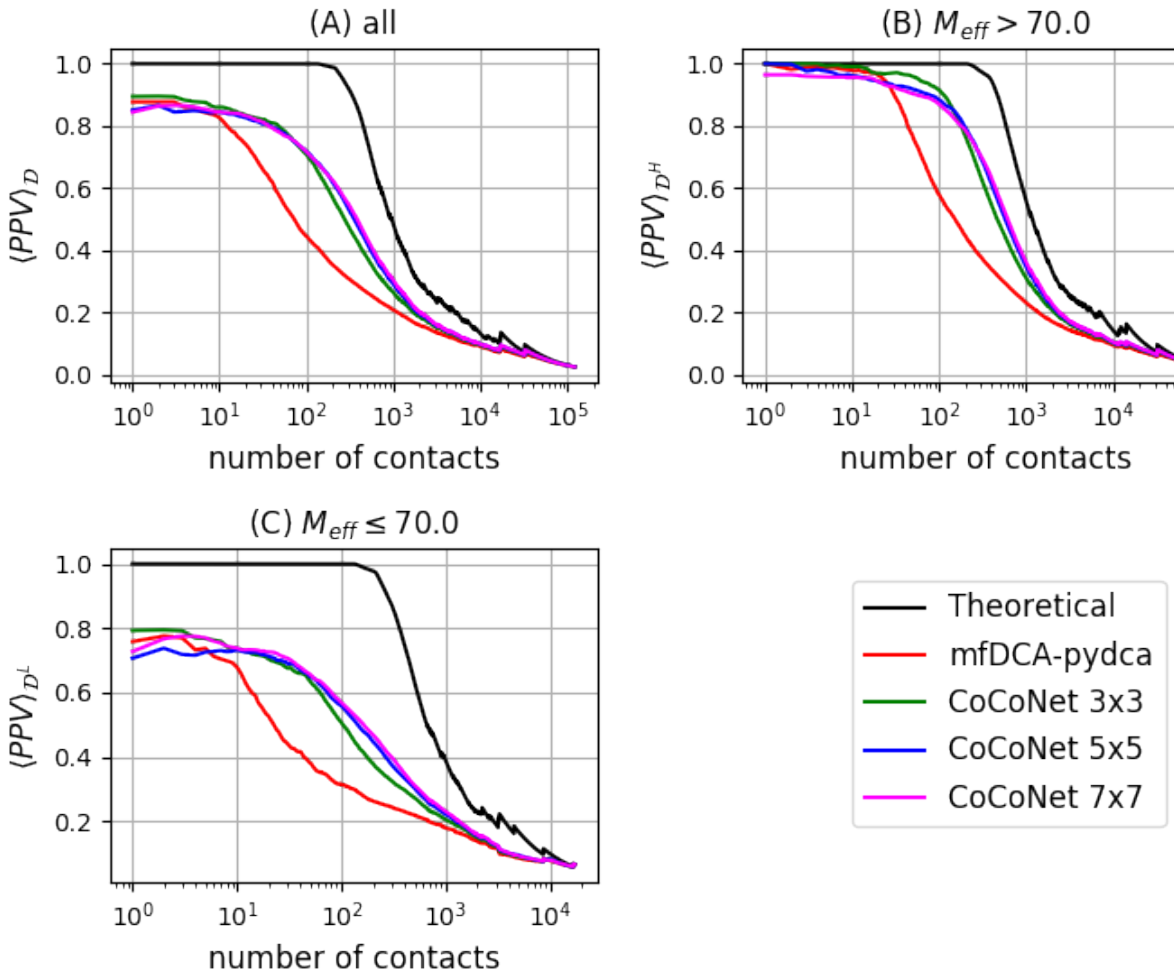


Figure **6-3**.: Average positive predictive values, $\langle PPV \rangle$, as a function of the number of contacts for (A) all the 57 RNAs; (B) RNAs with $M_{eff} > 70.0$; and (C) RNAs with $M_{eff} \leq 70.0$. In all of the three dataset categories, **CoCoNet** significantly outperforms mean-field DCA as the average PPV curves of **CoCoNet** are closer to the theoretical curve than mean-field DCA's cures are.

For dataset $\mathcal{D}$, which contains all RNA regardless of the effective number of sequences (Fig. **6-3** (A)), both mean-field DCA and **CoCoNet** perform comparably down to rank ten, i.e., both algorithms show comparable average PPV for top ten ranked pairs. Below rank ten performance variations are significant, with **CoCoNet** showing superiority. For instance, for 100 top-ranked nucleotide pairs, mean-field DCA's average PPV is slightly greater than 40%, whereas **CoCoNet**'s is about 70%. Not surprisingly, the average PPVs of both mean-field DCA and **CoCoNet** significantly vary based on the number of effective sequences (M_{eff}). It is evident from Fig. **6-3** (B) and (C), where M_{eff} is greater than

70 and less than 70, respectively. For $M_{eff} > 70$, mean-field DCA's average PPV for 100 top-ranked pairs is about 60% and, that of **CoCoNet**'s is about 90%. For $M_{eff} < 70$, mean-field DCA's average PPV is about 30%, whereas **CoCoNet**'s is more than 50% for 100 top-ranked nucleotide-pairs.

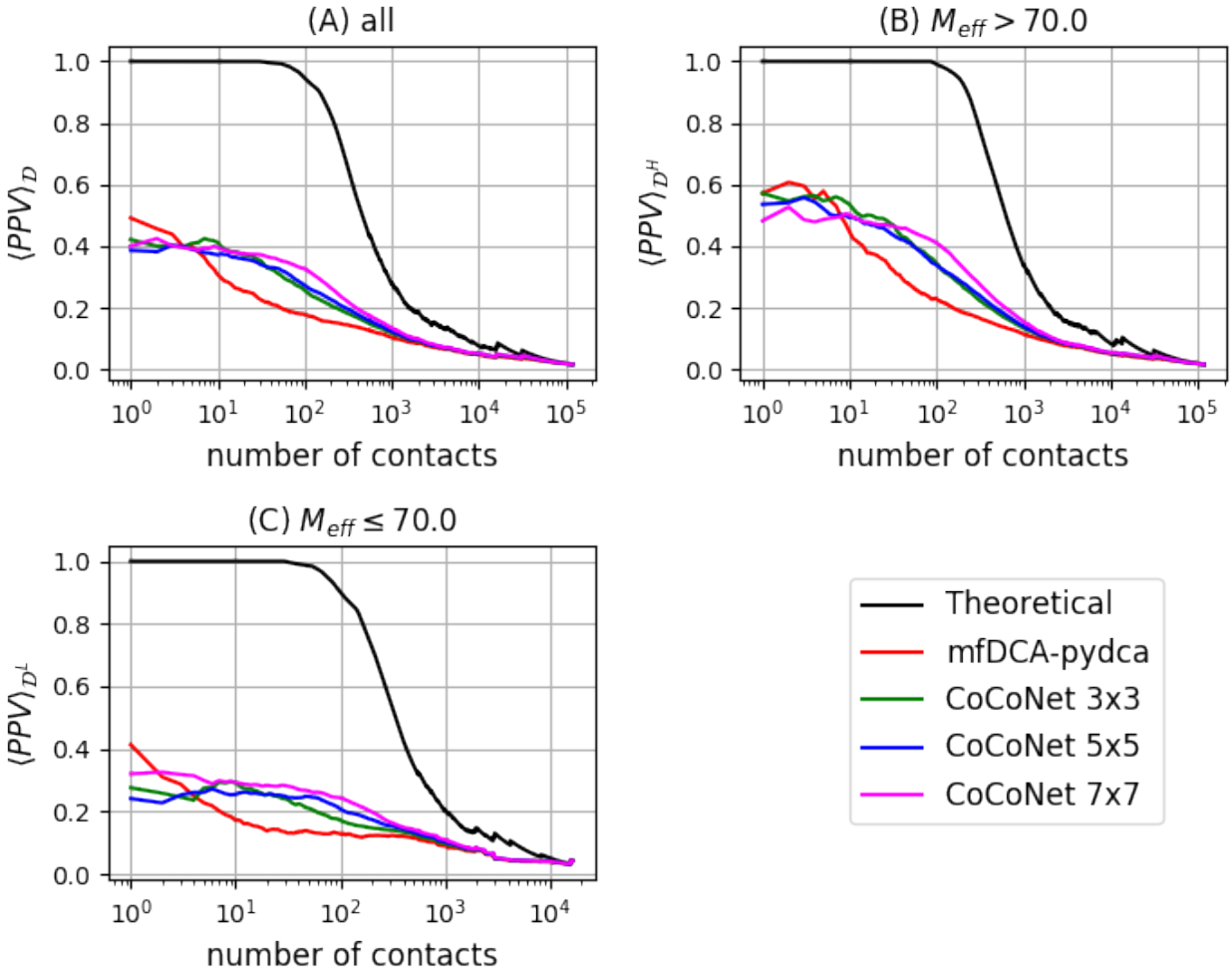


Figure **6-4**.: Tertiary contacts average positive predictive values, $\langle PPV \rangle$, as a function of the number of contacts (rank) for (A) all the 57 RNAs; (B) RNAs with $M_{eff} > 70.0$; and (C) RNAs with $M_{eff} \leq 70.0$. For the highest-ranked nucleotide pairs, mean-field DCA shows slightly better performance than **CoCoNet**; however, **CoCoNet** overtakes mean-field DCA below rank 10.

For three-dimensional structure prediction, tertiary constraints are crucial[23]. Interestingly, despite the simplicity of the model, **CoCoNet** enhances the average PPV for tertiary contacts. Here, tertiary contacts are pairs of nucleotides that are not in the secondary structure (without pseudoknots) and not within a 5×5 window around a secondary structure pair in the contact map. Also, these pairs must have at least a pair of heavy atoms less than $10\mathring{A}$ apart. Fig. **6-4** compares mean-field DCA's and **CoCoNet**'s average PPV for tertiary

contact at full rank. At the highest ranks, mean-field DCA shows slightly better performance than **CoCoNet**. For 100 top-ranked nucleotide pairs, **CoCoNet** shows superiority in predicting correct tertiary contacts. The 7×7 matrix performs better than the 3×3 and 5×5 matrices at the same rank, i.e., 100 top-ranked nucleotide pairs. Like in all contact prediction (Fig. **6-3**), average PPV prediction for tertiary contacts is also dependent on the number of effective sequences (M_{eff}). Like before, for RNAs whose M_{eff} is greater than 70, both mean-field DCA and **CoCoNet** show better contact prediction than when M_{eff} is lower or equal to 70.

The number of putative contacts an RNA molecule has depends on its sequence length, i.e., the number of nucleotides. Instead of measuring average PPV at a fixed rank, say 100, a rank that scales with the number of putative contacts is chosen. On this ground, the rank is set to be at L for each RNA, and the average positive predictive values at that rank are computed and displayed in Table **6-1** and Table **6-2**.

Table **6-1** shows the average positive predictive values, $\langle PPV \rangle$, at rank L for all of the RNA in the dataset $\mathcal{D}$, i.e., the dataset containing all of the 57 RNAs. Looking at the Table, **CoCoNet** significantly boosts contact prediction, not only for all contact types but also tertiary contacts. Indeed, the new algorithm outperforms state-of-the-art DCA algorithms implemented in various software suites such as EVcouplings plmDCA, pydca's mfDCA, and plmDCA, Boltzmann learning, and PSICOV's graphical LASSO algorithms. The best performing DCA algorithm for all contact types is the pseudo-likelihood maximization of EVCouplings (plmDCA-EVC), showing an average PPV of 46.6%. At the same rank, **CoCoNet**'s average PPV is about 77% when using two 5×5 or two 7×7 matrices that contain 50 and 98 free parameters, respectively. Interestingly, the improvement in average PPV is observed for tertiary contacts as well. While the DCA algorithms can produce average PPV between 15% to 18%, **CoCoNet** yields between 26% to 35%, indicating that **CoCoNet** predicts almost twice true contacts compared to state-of-the-art DCA algorithms.

To study the impact of effective number of sequences on PPV, the average PPVs at rank L are computed and displayed in Table **6-2** for the two dataset subsets $\mathcal{D}^H$ and $\mathcal{D}^L$. As expected, all of the DCA algorithms show improved PPVs in dataset $\mathcal{D}^H$. This improvement is manifested in **CoCoNet**'s PPVs performing as high as 90% for all contact types when a single 3×3 matrix is used (see Table **6-2** column three). It is an outstanding performance given that the number of free parameters to be learned is minimal. The corresponding best DCA result stands at about 61% when plmDCA in EVCouplings is used. For the dataset, $\mathcal{D}^L$, the overall average PPVs of DCA are lower than dataset $\mathcal{D}^H$. The best DCA average PPV for $\mathcal{D}^L$ is about 33%, whereas that of **CoCoNet** is 67% resulted when two 7×7 filter matrices are used. Again **CoCoNet** significantly improves true contacts. In addition, the dependence of **CoCoNet**'s performance on the number of effective sequences is clear—just like DCA algorithms, a large number of effective sequences are likely to result in better true contact predictions.

Enhancement of PPVs by **CoCoNet** are also reflected in tertiary (3D) contacts. The

Table **6-1**.: Average positive predicted value, $\langle PPV \rangle$, at rank L for all RNAs in the dataset $\mathcal{D}$. The first two columns indicate the number and size of filter matrices used, respectively. The third column corresponds to the number of free parameters to learn. The fourth and last columns show $\langle PPV \rangle$ at rank L for all and tertiary contacts, respectively. The bottom five rows represent the $\langle PPV \rangle$ of DCA algorithms including, (i) mean-field DCA implemented in pydca, mfDCA-pydca; (ii) pseudo-likelihood maximization of EVCouplings, plmDCA-EVC; (iii) pseudo-likelihood maximization of pydca, plmDCA-pydca; (iv) Boltzmann learning and (v) the graphical LASSO algorithm implemented in PSICOV. Columns four and five represent $\langle PPV \rangle$s for all contacts and tertiary only contacts, respectively. The $\langle PPV \rangle$s show that CoCoNet significantly outperforms state-of-the-art DCA algorithms.

CoCoNet				
#Filters	Filter Size	#Free Parameters	$\langle \text{PPV} \rangle_{\text{ALL}}$ (top L)	$\langle \text{PPV} \rangle_{\text{3D}}$ (top L)
1	3x3	9	74.6	27.1
1	5x5	25	74.6	29.2
1	7x7	49	74.4	33.6
2	3x3	18	76.5	26.6
2	5x5	50	77.7	27.1
2	7x7	98	77.3	35.0
mfDCA-pydca			45.0	17.7
plmDCA-EVC			46.6	16.8
plmDCA-pydca			45.0	16.2
Boltzmann learning			45.7	17.3
PSICOV			39.9	15.0

best PPV delivered by DCA is about 22% for dataset $\mathcal{D}^H$. The corresponding best result by **CoCoNet** is about 40%—almost doubling the PPV obtained by DCA. The improvement on PPV is also seen on dataset $\mathcal{D}^L$ where DCA shows about 13% whereas **CoCoNet**'s is about 30% when two 7×7 filter matrices are used. Positive predictive value is one of several metrics to evaluate the performance of classification algorithms. Although PPV is an excellent metric to evaluate contact prediction, I also study the new algorithm's performance by another metric, namely, the Mathews correlation coefficient, in the next section.

Table **6-2**.: Average positive predictive value, $\langle PPV \rangle$, at rank L when the dataset is split into two categories—$\mathcal{D}^H$ and $\mathcal{D}^L$ based on the effective number of sequences. The first two columns indicate the number and size of filter matrices used, respectively. The third and fifth columns represent $\langle PPV \rangle$s for all contact types, whereas the fourth and sixth columns represent tertiary only contacts $\langle PPV \rangle$s. The bottom five rows represent the $\langle PPV \rangle$ of DCA algorithms including, (i) mean-field DCA implemented in pydca, mfDCA-pydca; (ii) pseudo-likelihood maximization of EVCouplings, plmDCA-EVC; (iii) pseudo-likelihood maximization of pydca, plmDCA-pydca; (iv) Boltzmann learning and (v) the graphical LASSO algorithm implemented in PSICOV. The $\langle PPV \rangle$s show that CoCoNet significantly outperforms state-of-the-art DCA algorithms.

CoCoNet					
#Filters	Filter Size	$\langle \text{PPV} \rangle_{\text{ALL}}$ $\mathcal{D}^H$	$\langle \text{PPV} \rangle_{\text{3D}}$ $\mathcal{D}^H$	$\langle \text{PPV} \rangle_{\text{ALL}}$ $\mathcal{D}^L$	$\langle \text{PPV} \rangle_{\text{3D}}$ $\mathcal{D}^L$
1	3x3	90.3	35.0	59.4	19.5
1	5x5	87.4	35.0	62.3	23.8
1	7x7	86.3	40.3	62.8	27.1
2	3x3	91.7	34.7	61.8	18.8
2	5x5	89.6	32.0	66.1	22.4
2	7x7	87.7	40.3	67.2	29.8
mfDCA-pydca		57.1	22.0	33.3	13.6
plmDCA-EVC		61.3	22.0	32.5	11.9
plmDCA-pydca		57.6	22.0	33.0	10.6
Boltzmann learning		58.7	22.2	33.1	12.5
PSICOV		50.9	18.6	29.1	11.5

Contact prediction evaluated by Mathews correlation coefficient

In this section, the new RNA contact prediction algorithm's performance is evaluated by using the Mathews correlation coefficient (MCC). For binary classification, the positive predictive value, PPV, considers the numbers of true and false positives. For data that has a class imbalance, PPV may not be an accurate metric. However, for contact prediction, the number of contacts is a fraction of the total number of nucleotide pairs, making PPV an excellent measure to evaluate contact prediction algorithms.

MCC is also an excellent measure to evaluate classification algorithms' performance. It incorporates the numbers of true and false positives of the confusion matrix together with

true and false negatives[18]. Mathematically, MCC is written given by

$$MCC = \frac{(TP)(TN) - (FP)(FN)}{\sqrt{((TP+FP)(TP+FN)(TN+FP)(TN+FN)}}, \tag{6-3}$$

where TP and TN represent the numbers of true positive and negatives, respectively, whereas FP and FN represent the numbers of false positives and negatives, respectively.

For quantifying the MCC, the true/false positives/negatives are defined as follows. A fixed rank L is taken, and nucleotide pairs up to this rank from above are considered positive contacts. Thus if nucleotide pairs are present in PDB contacts, they are counted as true positive, if not false positives. All those nucleotide pairs below rank L are true/false negatives. If a pair is not a contact in PDB structure, it is counted as a true negative and false negative, otherwise. The MCC values of **CoCoNet** are compared with those of state-of-the-art DCA algorithms for all contact types in Fig. **6-5**, and tertiary only contacts in Fig. **6-6**. In both scenarios, **CoCoNet** significantly performs all of state-of-the-art DCA algorithms. Indeed, **CoCoNet**'s MCC can be up to twice the MCC of DCA-based algorithms.

For all contact types, PSICOV results in the least MCC among DCA-based algorithms while the rest of the algorithms show more or less similar MCC. For the dataset, $\mathcal{D}^H$ the plmDCA in EVCouplings has a slight edge on MCC compared to the other DCA-based algorithms. However, still the **CoCoNet** MCC are significantly higher than all of the state-of-the-art DCA algorithms' MCC. Among the various flavors of **CoCoNet**, the 3×3 matrix has a slightly better MCC. However, for dataset $\mathcal{D}^L$, the 5×5 and 7×7 filter matrices result in a slightly better MCC than the 3×3 filter matrix, indicating that larger filter matrices have a better capability of averaging noisy coevolutionary signal resulting from insufficient MSA data.

The trend that **CoCoNet** outperforms state-of-the-art DCA algorithms as evaluated by MCC is also observed when considering tertiary contacts only. Recall that tertiary contacts are defined as nucleotide pairs that are not secondary structure pairs and not in the vicinity of a 5×5 window around these secondary pairs in the contact map. Like for all contact types, all DCA algorithms show similar MCC except PSICOV that results in the lowest MCC for dataset $\mathcal{D}^H$. For dataset $\mathcal{D}^L$, the plmDCA of pydca results in the lowest MCC. This is not surprising as pydca's plmDCA uses large values for penalizing fields and couplings. Among the **CoCoNet** flavors, the 7×7 filter matrix results in best MCC than the other. This is due to the fact that in RNA contact prediction, the coevolutionary signal is dominated by secondary structure pars, mainly when the alignment is performed by Infernal [72]. Tertiary pairs are noisy, and the larger filter matrices do better averaging than, the smaller ones. This averaging effect is more pronounced when there are insufficient MSA data where the 7×7 filter matrix outperforms the others significantly.

To summarize, while state-of-the-art DCA algorithms result in similar MCC values at rank L for the RNA dataset studied in this thesis, **CoCoNet** results in way better MCC than the DCA algorithms. In addition to PPV as a performance metric, the MCC shows

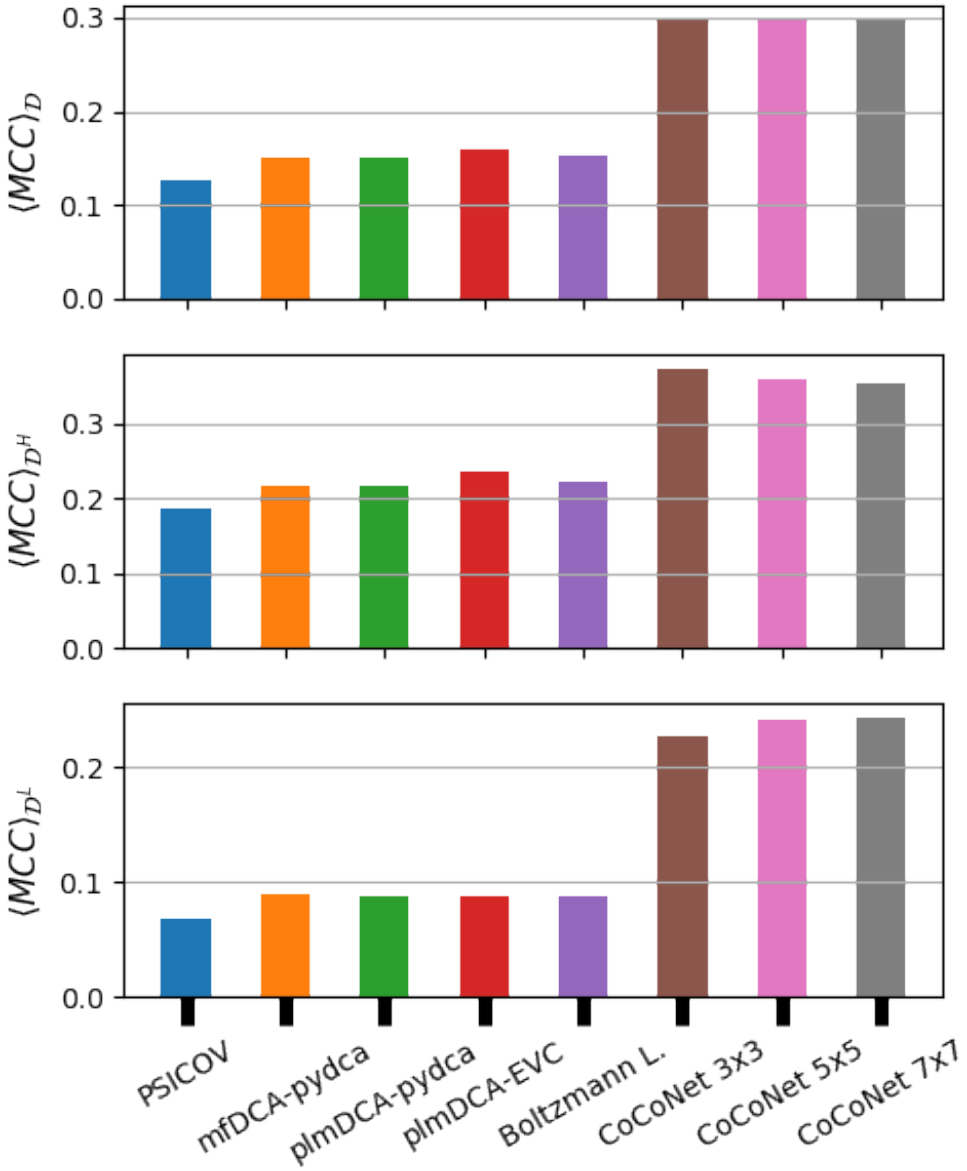


Figure **6-5**.: Average MCC at rank L for all contacts types. The averages are performed over datasets (i) all RNAs, $\mathcal{D}$, top Figure (ii) RNAs whose $M_{eff} > 70.0$, $\mathcal{D}^H$, middle Figure and (iii) RNAs whose $M_{eff} \leq 70.0$, $\mathcal{D}^L$, bottom Figure. The MCC values of **CoCoNet** are significantly higher than those of state-of-the-art DCA algorithms, showing twice as much MCC as DCA for dataset $\mathcal{D}$.

CoCoNet's ability to enhance RNA contact prediction even though it has only a few free parameters to be learned from the limited available data.

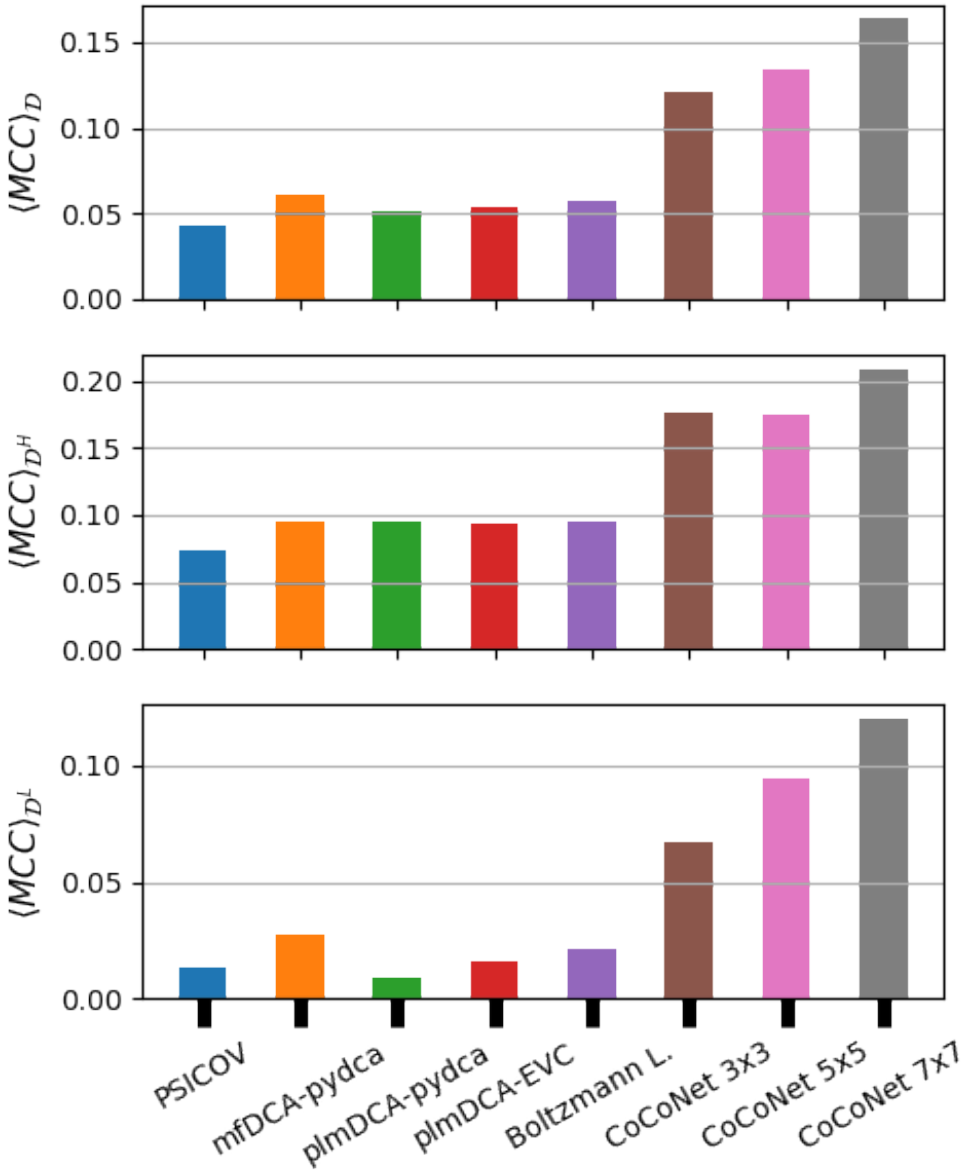


Figure **6-6**.: Average MCC at rank L for tertiary only contacts. The averages are performed over datasets (i) all RNAs, $\mathcal{D}$, top Figure (ii)RNAs whose $M_{eff} > 70.0$,$\mathcal{D}^H$, middle Figure and (iii) RNAs whose $M_{eff} \leq 70.0$, $\mathcal{D}^L$, bottom Figure. The MCC values of **CoCoNet** are significantly higher than those of state-of-the-art DCA algorithms, showing twice as much MCC as DCA for dataset $\mathcal{D}$.

6.6. Summary and conclusion

This chapter presents a new RNA contact prediction algorithm that can significantly outperform state-of-the-art DCA-based algorithms known to predict contacts accurately. Due to their ability to identify correlations resulting from direct and indirect contacts in the three-

dimensional protein and RNA structures, DCA-based algorithms have been the standard to predict contacts from MSA. The new algorithm dubbed as **CoCoNet** is based on a shallow convolutional neural network as the RNA data available to train the model is significantly limited.

The model is based on a single convolutional layer added on top of the coevolution layer obtained using the classical mean-field DCA algorithm. The convolutional layer has a minimal number of free parameters that are learned from labeled data. Indeed the number of parameters is only nine when a single 3×3 filter matrix is used and 98 when two 7×7 filter matrices are used. The small number of parameters makes sure that the parameters are not over-fitted during training which ensures generality.

The evaluation of the model is carried out using positive predictive value (PPV)—a metric regularly used in protein and RNA contact prediction—and the Mathews correlation coefficient (MCC). By both metrics, **CoCoNet** significantly improves RNA contact prediction when compared to state-of-the-art DCA algorithms that are known to accurately predict contacts from multiple sequence alignments of protein or RNA.

The **CoCoNet** algorithm is written using the popular open-source Python programming language and requires pydca[106] as a dependency. Both of **CoCoNet** and pydca software implementation are freely available under the MIT license at https://github.com/KIT-MBS/coconet and https://github.com/KIT-MBS/pydca, respectively.

7. RNA 3D Structure Prediction

7.1. Introduction

Computer simulation methods such as Monte-Carlo and molecular dynamics can complement biomolecular structure prediction in general and RNA structure prediction in particular. However, predicting biomolecular structures starting from sequences can be challenging[30, 87]. In protein simulations, supplying data obtained from experiments and statistical analysis can aid the simulation process[78, 102]. Furthermore, using co-evolutionary information in the form of pairs of putative contacts for RNA structure prediction has shown promising results[23, 104].

In this chapter, co-evolutionary information extracted using the CoCoNet algorithm is assessed using six selected RNAs from the dataset in Chapter 5]. In particular, RNA structure prediction done using mfDCA and CoCoNet putaive tertiary contacts are compared to see if the improvements seen contact prediction could translate to RNA 3D structure prediction. The selected RNAs are (i)the thymine phyrophosphate (TPP) sensing riboswitch; PDB ID 3d2g[96]; (i) the S-adenosylmethionine (SAM-1) riboswitch, PDB ID 3gx5[64]; (iii) the vibrio vulnificus adenine riboswitch, PDB ID 4tzx[107]; (iv) the glycine riboswitch, PDB ID 3ox0[38]; (v) the 3',3'-cGAMP Sensing Riboswitch, PDB ID 4yaz[80]; and (vi) the fluoride riboswitch, PDB ID 4enc[79]. A detailed information such as about sequence length, number of effective sequences in Rfam, etc is found in Table**A-1** and Table**A-2**.

The chapter is organized as follows. First, the material and methods used for the RNA 3D structure prediction are described. Then, the results are analyzed and compared. Finally, a summary and conclusion are provided.

7.2. Materials and methods

The RNA 3D structure prediction is carried out using co-evolutionary information guided replica-exchange Monte-Carlo simulation using the SimRNA simulation software[11]. Co-evolutionary information is extracted using the CoCoNet algorithm. The 3D structure simulation general workflow is outlined in Fig.**7-1**.

Input parameters

For the 3D structure prediction simulation, the sequence, consensus secondary structure, and tertiary restraints are required. The consensus secondary structures are obtained from the Rfam database[42, 41]. Tertiary constraints are obtained—as is done throughout this thesis–by removing all secondary structure pairs and their neighbors in a 5×5 window in the contact map from all pairs ranked by the CoCoNet algorithm. For each of the RNAs top L, where L is the length of the sequence, contacts are taken.

The simulations are set up and performed in the SimRNA replica-exchange algorithm protocol. The parameters are set to take the values recommended in the SimRNA software manual. Doing so the simulation is configuration takes the values as follows: (i) initial temperature 1.35; (ii) final temperature 0.90; (iii) bond weight 1.0; (iv) angles weight 1.0; (v) torsional angles weight 0.0; (vi) $\eta - \theta$ weight 0.40. All other parameters are used with their default value. The details about these parameters is found in[11].

The tertiary restraints are used in the form of a linear potential with a weight value 0.25[11]. The distance between nucleotide pairs is set to be using a pseudo middle-atom representing the centers of the nucleotides of the pairs. The minimum and maximum distances between the centers are set to be 5.5 and 10.0, respectively. When the distance between the pseudo atoms is not in between the minimum and maximum distance, a linear potential penalty is applied. The penalty value is the product of the constraint weight (0.25) and the amount of violated distance. Ten replicas are used for each simulation, each simulation running for a total number of 1.6×10^7 Monte-Carlo steps. The trajectories are recorded at every 1.6×10^4 steps. For each case, a total of 4000 decoys are generated. The simulations are performed on an Intel Core-i7 8700 CPU with a clock speed of 3.20GHz and ten processors desktop computer.

Structure ranking and selection

For each case, the 4000 decoys are ranked by energy, with the lowest energy structure taking the highest position. Then, the top 10% are taken from all structures. The structures are clustered using a root-mean-square deviation (RMSD) of 3.0Å. The first 20 clusters are considered, and each cluster's first structure is taken as a predicted structure.

Each of the 20 structures is aligned with the corresponding PDB structures. The RMSD is computed using the Biopython software[19]. While aligning the structure, no atom is rejected. The average RMSD of the twenty clusters and the minimum RMSD among them are tabulated in Table**7-1**.

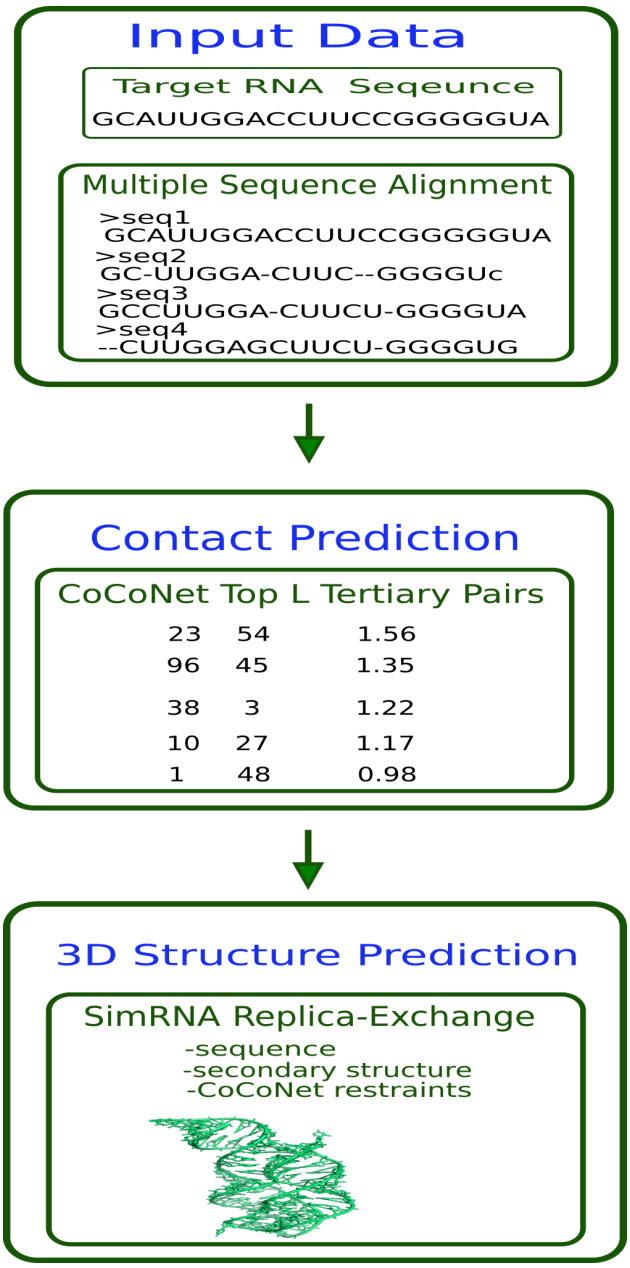


Figure **7-1**.: Figure showing the workflow diagram for RNA 3D structure prediction using co-evolutionary information obtained by the CoCoNet algorithm and the 3D modeling done using the SimRNA RNA modeling software. In this workflow, nucleotide pair ranking is done by computing the co-evolutionary interaction scores using the CoCoNet algorithm that uses multiple sequence alignment of a target sequence with homologous sequences in the Rfam database. Then, top L tertiary putative contacts computed relative to a secondary structure are used as restraints in the SimRNA software.

7.3. Results

In this section, simulation results of RNA 3D structure prediction are presented and discussed. First, the effect of constraints on RNA structure prediction using the SimRNA software is analyzed. Then, tertiary contacts' influence on the predicted structure accuracy measured by RMSD is compared with respect to mean-field DCA and CoCoNet putative tertiary contacts.

Effect of constraints on 3D structure prediction

Here I discuss the difference between simulating RNA structure using only the sequence and when information from constraints/restraints are used on top of the sequence. Constraints/restraints in this context are pairs of nucleotides that are putative contacts—they are assumed to be spatially adjacent within an RNA's 3D structure.

Putative contacts are classified into secondary structure pairs or tertiary structure contacts. In this work, secondary structure putative contacts are obtained from the consensus secondary structure of an RNA from its family in the Rfam database[41, 42]. Tertiary putative contacts are taken from nucleotide pairs that are not secondary structure putative contacts and not in the vicinity of a 5×5 window around a putative secondary structure contact in the contact map.

Table **7-1** displays the average/best RMSD values obtained for the six RNA structures. The first and second columns represent the PDB ID in the PDB database and the family name in the Rfam database of the RNAs, respectively. The other columns (from the third column to last) display the average and lowest RMSD values obtained from 20 clusters. When the structures are simulated without secondary or tertiary structure putative contacts, the RMSD of each RNA is larger than when secondary or tertiary structure information is included. The only exception is the RNA with PDB code 4enc (belonging to RF01734). For this RNA, the RMSD obtained using the sequence alone is comparable to when secondary or tertiary information is supplied to SimRNA. Although the 3oX0 (RF00504) has the minimum RMSD (10.7Å) comparable when secondary or tertiary information is supplied, the average RMSD is still higher. These results indicate that including secondary or tertiary structure information mainly results in better structures evaluated by RMSD.

Mean-field DCA versus CoCoNet

The last two columns in Table**7-1** contain average/minimum RMSD values obtained from 20 structures when putative tertiary contacts are used from mean-field DCA, and CoCoNet respectively. Both columns indicate that using putative tertiary contacts in general results in improved structure prediction.

The average RMSD obtained using CoCoNet's putative tertiary contacts is lower than that of mfDCA's for three of the RNAs —namely 3d2g, 3gx5, and 4yaz. The RMSD im-

Table **7-1**.: The root-mean-squared deviation (RMSD) of simulated RNA 3D structures obtaining by aligning the simulated structures with their experimental structures taken from the PDB database. The first and second columns label PDB IDs and family names of the RNAs, respectively. The third, fourth, fifth, and sixth columns contain the average/minimum RMSD values when simulations are done using the SimRNA modeling software using (i) the sequence (ii) sequence plus secondary structure (labeled on the table as Secondary struc.) (iii) sequence plus secondary structure plus mfDCA tertiary contacts and (iv) sequence plus secondary structure plus CoCoNet 3×3 tertiary contacts. The number of tertiary contacts used is L, where L is the RNA length.

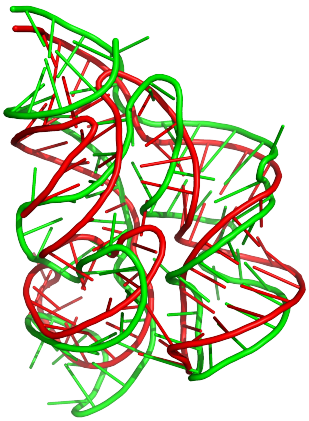
PDB ID	Family Name	Sequence	Secondary Struc.	mfDCA	CoCoNet 3x3
3d2g	RF00059	22.4/17.3	19.6/13.8	14.4/12.3	10.4/7.3
3gx5	RF00162	24.5/15.0	18.1/13.8	17.2/15.5	11.6/9.8
4tzx	RF00167	24.9/23.4	9.1/7.3	8.3/6.8	8.7/6.8
3ox0	RF00504	20.1/10.7	16.5/12.0	12.7/11.3	13.5/9.2
4yaz	RF01051	22.4/19.2	21.2/18.7	19.0/16.9	15.0/13.2
4enc	RF01734	11.8/9.1	14.1/9.6	10.3/8.9	9.6/8.2

provements with respect to mfDCA are up to about 5Å. In the other three RNAs, the average RMSD of mfDCA and CoCoNet are comparable. For instance, for the 4enc RNA, mfDCA's average RMSD is 10.3Å, and that of CoCoNet's is 9.6Å. However, a comparable result, 11.8Å, can be obtained without using secondary structure or putative tertiary contacts for the same RNA. Also, for the 4tzx RNA, both algorithm's putative contacts perform almost the same(8.3Å vs.8.7Å). However, a comparable result of 9.1Å is obtained from the consensus secondary structure information alone.

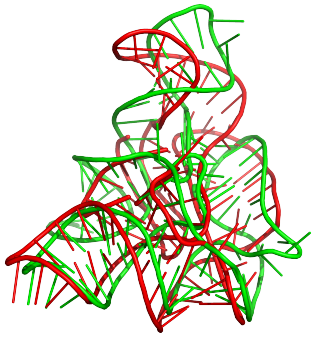
Among the minimum RMSD values, like the average RMSD values, CoCoNet performs better for the three RNAs. For the 3d2g RNA, the minimum RMSD for CoCoNet and mfDCA is 7.3Å and 12.3Å, showing an improvement by 5Å when CoCoNet putative tertiary contacts are used. CoCoCoNet also shows an improvement close to 6Å for the 3gx5 RNA and about 3.5Å for the 4yaz RNA. For the 4enc RNA, the minimum RMSDs are comparable with or without secondary structure information or putative tertiary contacts.

7.4. Summary and conclusion

This chapter presented a workflow to guide RNA 3D structure prediction using co-evolutionary information. In particular, six RNAs were selected to test the new RNA contact prediction



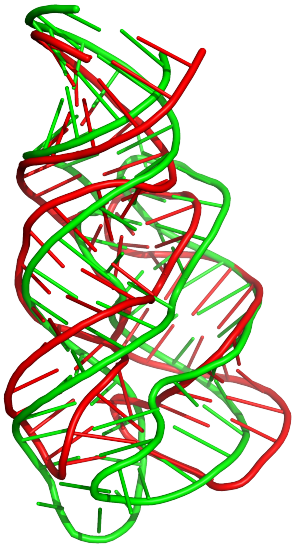
(A) 3d2g, chain A



(B) 3gx5, chain A

(C) 4enc, chain A

(D) 3oX0, chain A



(E) 4tzx, chain X

(F) 4yaz, chain A

Figure **7-2**.: Figure showing the alignment of predicted structures(in red) with the corresponding structures obtained from the PDB database(in green). The predicted structures are obtained using top L tertiary restraints of CoCoNet 3×3 algorithm plus consensus secondary structure from the Rfam database. The structures are rendered using the open-source version of PyMol[86]. No atom is rejected while aligning the structures.

algorithm CoCoCoNet. To this end, 4000 decoys of RNAs were generated using the SimRNA software[11], which is a coarse-grained replica-exchange Monte-Carlo molecular modeling software. The simulations were performed using (i) the RNA sequence alone (ii) the RNA sequence and the consensus secondary structure from Rfam[42] (iii) the RNA sequence, consensus secondary structure, and top L putative tertiary contacts obtained using mean-field DCA and (iv) the RNA sequence, consensus secondary structure, and top L putative tertiary

contacts obtained using CoCoNet.

Among the 4000 decoys, 10% of the lowest energy structures were taken for further analysis. These decoys were clustered using a root-mean-square deviation (RMSD) cut-off value 3Å. The lowest energy structure from twenty clusters is taken, and the RMSD with respect to the RNA structures from the PDB database is computed.

On average, CoCoNet 3×3 putative tertiary contacts result in lower RMSD than mean-field DCA's for three of the six RNAs. For the other three RNAs, both of these algorithms' putative tertiary contacts result in comparable RMSD. However, in these cases, comparable RMSD is also obtained using either the sequence alone or sequence plus consensus secondary structure.

The average RMSD is still high, particularly compared to in the corresponding field of protein structure prediction. Co-evolutionary information in the form of putative contacts has been successful, resulting in protein structure predictions at experimental resolution. For the workflow presented in this thesis, several parameters can be tuned and optimized, such as tuning restraints potential parameters or combining different potentials within the SimRNA software. Also, using an accurately predicted secondary instead of taking the consensus secondary structure could result in lower RMSD.

8. Summary and Outlook

One of the primary goals of molecular biology is to understand the structure and function of biomolecules. Ribonucleic acids (RNA) are one of the key players for cellar activities. Often, their structure is related to their function. However, resolving RNAs' structure is an arduous task—the RNA molecule is exceptionally flexible. Although many RNAs are sequenced, the vast majority lack the corresponding three-dimensional (3D) structure.

In the last decade, statistical models based on global probability models, collectively known as direct coupling analysis (DCA), helped to accurately predict protein structure, reaching experimental resolutions[62, 88]. Recently, DCA has been started to be applied for RNA 3D structure prediction[23, 104]. However, the success of DCA for predicting spatially adjacent nucleotide pairs with the ultimate goal of using them for 3D structure predictions remains limited compared to the corresponding field of protein 3D structure prediction[77]. This chapter highlights my contribution to RNA 3D structure prediction and provides an outlook for future improvements.

8.1. Summary

One of my contributions in this work is to provide a light-weight, easy-to-use, and stand-alone open-source software to the field of protein and RNA contact prediction. I have implemented two popular DCA algorithms into computer software and made them freely accessible under the MIT License. The software implements mean-field (mfDCA) and pseudo-likelihood maximization (plmDCA) DCA algorithms. The mfDCA algorithm is very efficient and is fast compared to the plmDCA algorithm. The plmDCA can result in relatively better accuracy for protein contact prediction. The software is implemented mainly in the Python programming language. However, the computationally costly algorithm in the plmDCA are optimized and implemented using the C and C++ programming languages. Nevertheless, the user interface is entirely accessible using Python programming. It allows the seamless integration of the software with other machine learning algorithms available as Python libraries.

The second step in this work was to assess DCA-based algorithms' performance for RNA contact prediction. To this end, a set of about sixty RNAs that have a high-resolution 3D structure in the PDB database were gathered. The RNA contact prediction is carried out using several DCA-based algorithms—mean-field DCA (mfDCA) and pseudo-likelihood

maximization DCA (plmDCA), Boltzmann learning, and PSICOV which is a sparse penalized maximum likelihood estimator. For overall contact prediction, all but PSICOV perform on average, almost the same for any number of putative contact nucleotide pairs. Only PSICOV's positive predictive value (PPV) slightly less than the rest of the algorithms. For tertiary contact prediction, plmDCA and Boltzmann learning algorithms tend to be more accurate for the first ten putative contacts. However, the number of putative contacts in RNAs depends on the size of the RNA molecule. Generally, when the length of the RNA sequence is considerable, the number of contacts becomes significant. Consequently, long RNAs tend to lead to a large PPV value for a fixed number of contacts than shorter ones. The standard number of contacts is taken to scale with the length (L) of the RNA. It avoids over-estimation in PPV for large RNAs when a fixed number of contacts are taken. Taking L predicted contacts, the algorithms show no significant variation in PPV for both all contact types and tertiary only contacts.

A new method based on a combination of DCA and convolutional neural networks (CNN) is developed to predict RNA contacts more accurately. To avoid over-fitting due to limited RNA training data and enable generality of the model, the convolution layer is restricted to a single layer to keep the number of parameters low. Despite this restriction, the new model dubbed as CoCoNet significantly improves the number of true positives leading to an increment of PPV by up to 70% compared to the classic mfDCA algorithm. PPV is not a biased measure for contacts prediction—the number of possible pairs of an RNA scales with its length as L^2, but the actual number of contacts is a fraction of this number. However, the performance of CoCoNet is also compared with state-of-the-art DCA algorithms using Mathews correlation coefficient (MCC) as a metric. CoCoNet significantly outperforms the DCA algorithms when compared by MCC too.

The other task performed in this thesis is to evaluate CoCoNet's performance for RNA 3D contact prediction, i.e., using CoCoNet predicted putative contacts as inputs to molecular modeling software. To this end, the SimRNA[11] is based on a coarse-grained replica-exchange Monte-Carlo algorithm, is used to predict the structure of six RNAs. Root mean-square deviation (RMSD) is used as a metric to evaluate 3D RNA structure prediction accuracy. The simulations were carried out using consensus secondary structure obtained from Rfam[42] plus top L tertiary putative contacts obtained either from CoCoNet or mfDCA. For comparing the results, simulations using the sequence and sequence plus consensus secondary structure were also performed. In three of the six RNAs, CoCoNet predicted putative tertiary contacts result in, on average least RMSD with respect to the RNAs' experimental PDB structure. On the other three RNAs, the performance of mfDCA and CoCoNet is comparable; nevertheless, a comparable performance for these three RNAs can also be obtained using either the sequence alone or the sequence plus the consensus secondary structure. The RMSD is still large, particularly when compared to the corresponding field of protein structure prediction.

8.2. Outlook

Here, I would like to point out future works that need focus to improve RNA contact prediction with the ultimate goal of reaching high-resolution 3D RNA structure through molecular modeling.

First, the application of state-of-the-art DCA algorithms indicates that while the algorithms can accurately predict pairs of nucleotides from multiple sequence alignments (MSA), pairs that show strong co-evolution tend to be secondary structure base-pairs. However, RNA 3D structure is usually stabilized by long-range tertiary contacts. The tertiary contacts, however, show weaker co-evolutionary signals compared to the secondary structure pairs. The problem may be solved by developing MSA algorithms that enhance tertiary structure nucleotide pairs.

The other area that can be explored to improve RNA contact prediction is applying machine learning methods to enhance tertiary contact prediction. For instance, although CoCoNet shows significant improvement in this area compared to state-of-the-art DCA algorithms, parts of the model can further be improved to increase its ability to learn complex contact patterns without increasing the number of free parameters. For instance, systematic investigation adding non-linearities to the convolutional layer; using only 3×3 filter matrices, but increasing the depth of the network; or using other machine learning methods to classify RNA contact maps of secondary structure and tertiary structure patterns; may further improve contact prediction accuracy and approach the theoretical limit.

Some suggestions can be listed to improve the 3D structure prediction guided by co-evolving nucleotide pairs. First, the parameters associated with molecular modeling tools need to be tuned and rigorously investigated to understand the RNA folding further. Second, a combination of putative contacts from both DCA and CoCoNet could be used instead of only using either the algorithms' putative contacts. DCA has its strengths, such as predicting relatively uniformly distributed contacts across the contact map, but it is less accurate than CoCoNet. On the other hand, CoCoNet predicted contacts tend to cluster at a specific place on the contact map, which is less accurate when using only a fraction of top L putative tertiary contacts. A systematic combination of putative contacts from both DCA and CoCoNet that includes both algorithms' strengths might help improve RNA 3D structure folding.

With the rapid growth of sequence data, the advent of sophisticated machine learning algorithms, and the power of high-performance computing, the field of RNA structure prediction has a promising future. It can lead to practical applications beyond basic research, such as to design drugs. A case in point, some of the most successful Covid-19 virus vaccines are RNA-based.

A. RNA Dataset Description

Table **A-1**.: Table showing the dataset of RNA used in our analysis. The second column is the RNA type. The third column contains the family in Rfam the RNA belongs. The fourth column refers to the PDB code in the PDB database. The fifth is the resolution (in Å) of the PDB structure. The sixth is column contains the sequence length of the RNA. The seventh is the BIT score for multiple sequence alignment, and the last column is the number of effective sequences in the family.

No.	**RNA Type**	**Family**	**PDB Code**	**Resolution (Å)**	**Length**	**BIT Score**	M_{eff}
1	7SK RNA	RF00100	5lys	2.32	57	47.300	1679.745
2	Lysine riboswitch	RF00168	3dil	1.9	174	103.100	1455.511
3	Ribonuclease P	RF00010	1u9s	2.9	155	114.400	1285.282
4	TPP riboswitch	RF00059	3d2g	2.25	77	61.000	1241.931
5	ydao riboswitch	RF00379	4qln	2.65	117	85.500	998.938
6	Domain II of glycine riboswitch	RF00504	3ox0	3.05	87	64.600	847.594
7	P4-P6 RNA Ribozyme domain	RF00028	1gid	2.5	158	40.800	799.302
8	GlmS ribozyme	RF00234	2h0s	2.35	125	74.800	603.709
9	3',3'-cGAMP riboswitch	RF01051	4yaz	2	84	49.300	583.357
10	Phe-tRNA	RF00005	1ehz	1.93	76	65.600	461.009
11	Adenine Riboswitch	RF00167	4tzx	2	71	59.400	458.874
12	YkoY riboswitch	RF00080	6cb3	1.89	101	39.000	409.471
13	c-di-GMP-II riboswitch	RF01786	3q3z	2.51	75	63.300	372.130
14	SAM-I riboswitch	RF00162	3gx5	2.4	94	85.300	322.101
15	7S.S SRP	RF00169	1z43	2.6	101	42.300	281.462
16	Sec-tRNA	RF01852	3rg5	2	86	80.500	268.101

17	group II intron, domain 1	RF02001	4y1o	2.95	258	114.400	248.274
18	Fluoride riboswitch	RF01734	4enc	2.27	52	37.700	222.076
19	SAM-I/IV variant riboswitch aptamer	RF01725	4l81	2.95	96	40.400	210.969
20	THF riboswitch	RF01831	4lvv	2.1	89	67.500	187.739
21	SRP Alu domain	RF01854	4wfl	2.49	107	90.400	176.994
22	Ribonuclease P	RF00011	1nbs	3.15	120	94.000	157.580
23	ZMP riboswitch	RF01750	4xwf	1.8	64	40.800	130.655
24	23S Ribosomal RNA	RF02540	1ffz	3.2	497	518.800	120.116
25	M-box Riboswitch	RF00380	3pdr	1.85	161	139.600	119.645
26	NiCo transition-metal riboswitch	RF02683	4rum	2.65	93	60.200	117.035
27	FMN Riboswitch	RF00050	3f2q	2.95	108	101.700	104.843
28	YrlA effector-binding module	RF02553	6cu1	3	80	55.000	91.370
29	Cobalamin riboswitch aptamer domain	RF01689	4frg	2.95	84	53.200	62.800
30	Hammerhead Ribozyme	RF00163	3zp8	1.5	43	27.600	59.045
31	ai5g group II Self-splicing intron	RF00029	1kxk	3	70	44.900	50.960
32	HDV ribozyme	RF02682	3nkb	1.9	64	22.300	40.976
33	ykkC riboswitch	RF00442	5u3g	2.3	85	68.900	33.886
34	Native pistol ribozyme	RF02679	5k7d	2.68	47	33.800	32.373
35	Myotonic Dystrophy Type 2	RF02695	4k27	2.35	55	7.100	32.000
36	mRNA for the coat protein	RF00233	4p5j	1.99	86	94.200	16.221
37	preQ1 riboswitch	RF01054	4jf2	2.28	77	35.000	14.558
38	Ribosomal Binding Domain of the IRES RNA	RF00458	2il9	3.1	135	72.700	14.000
39	SAM-III Riboswitch	RF01767	3e5c	2.25	53	42.400	13.621

40	Distal Stem I region of the glyQS T box leader RNA	RF02447	4jrc	2.67	57	9.900	6.771
41	2'- Deoxyguanosine riboswitch	RF01510	3slq	2.5	68	81.200	6.643
42	Guanidine III riboswitch	RF01763	5nwq	1.91	41	50.900	5.059
43	Xrn1-resistant	RF01415	4pqv	2.46	68	48.300	4.862
44	RES pseudoknot domain	RF00061	3t4b	3.55	84	42.500	3.609
45	s2m RNA	RF00164	1xjr	2.7	47	65.100	3.293
46	Lariat capping ribozyme	RF01807	4p95	2.5	189	81.200	3.164
47	metY SAM-V riboswitch	RF01826	6fz0	2.5	48	44.400	2.806
48	PreQ1-III Riboswitch	RF02680	4rzd	2.75	99	106.800	2.602
49	Ligase Ribozyme	RF03017	2oiu	2.6	71	9.700	2.004
50	Ribozyme	RF02927	4r4v	3.07	186	11.300	2.000
51	ASH1 mRNA	RF00606	5m0h	2.65	42	6.800	2.000
52	OMe substituted twister ribozyme	RF00921	5dun	2.64	54	9.600	2.000
53	Prohead RNA	RF00044	3r4f	3.5	66	34.100	2.000
54	Twister Ribozyme	RF02681	5t5a	2	62	18.800	2.000
55	Core ENE hairpin and A-rich tract from MALAT1	RF02266	4plx	3.1	76	18.300	2.000
56	iSpinach aptamer	RF01300	5ob3	2	69	8.800	2.000
57	5-hydroxytryptophan aptamer	RF01982	5kpy	2	71	8.000	2.000

Table **A-2**.: Table showing the RNA dataset with the number of PDB contacts annotated for contact distance cut-off value 10Å.

№	Family Name	PDB ID	M_{eff}	Length	№All Contacts	№Tertiary Contacts
1	RF00100	5lys	1679.745	57	254	88
2	RF00168	3dil	1455.511	174	1036	556
3	RF00010	1u9s	1285.282	155	854	466

4	RF00059	3d2g	1241.931	77	417	243
5	RF00379	4qln	998.938	117	549	300
6	RF00504	3ox0	847.594	87	442	209
7	RF00028	1gid	799.302	158	951	513
8	RF00234	2h0s	603.709	125	687	484
9	RF01051	4yaz	583.357	84	429	215
10	RF00005	1ehz	461.009	76	370	190
11	RF00167	4tzx	458.874	71	382	199
12	RF00080	6cb3	409.471	101	509	243
13	RF01786	3q3z	372.130	75	363	222
14	RF00162	3gx5	322.101	94	509	273
15	RF00169	1z43	281.462	101	512	248
16	RF01852	3rg5	268.101	86	398	176
17	RF02001	4y1o	248.274	258	1516	878
18	RF01734	4enc	222.076	52	209	128
19	RF01725	4l81	210.969	96	483	277
20	RF01831	4lvv	187.739	89	486	269
21	RF01854	4wfl	176.994	107	562	323
22	RF00011	1nbs	157.580	120	662	414
23	RF01750	4xwf	130.655	64	232	85
24	RF02540	1ffz	120.116	497	3139	1937
25	RF00380	3pdr	119.645	161	1088	663
26	RF02683	4rum	117.035	93	406	155
27	RF00050	3f2q	104.843	108	650	405
28	RF02553	6cu1	91.370	80	393	180
29	RF01689	4frg	62.800	84	438	246
30	RF00163	3zp8	59.045	43	131	88
31	RF00029	1kxk	50.960	70	252	66
32	RF02682	3nkb	40.976	64	295	159
33	RF00442	5u3g	33.886	85	429	225
34	RF02679	5k7d	32.373	47	203	150
35	RF02695	4k27	32.000	55	210	54
36	RF00233	4p5j	16.221	86	463	275
37	RF01054	4jf2	14.558	77	391	233
38	RF00458	2il9	14.000	135	521	315
39	RF01767	3e5c	13.621	53	212	84
40	RF02447	4jrc	6.771	57	283	143
41	RF01510	3slq	6.643	68	363	178

42	RF01763	5nwq	5.059	41	212	146
43	RF01415	4pqv	4.862	68	296	142
44	RF00061	3t4b	3.609	84	341	139
45	RF00164	1xjr	3.293	47	221	75
46	RF01807	4p95	3.164	189	1145	702
47	RF01826	6fz0	2.806	48	218	156
48	RF02680	4rzd	2.602	99	415	210
49	RF03017	2oiu	2.004	71	224	64
50	RF02927	4r4v	2.000	186	825	326
51	RF00606	5m0h	2.000	42	140	29
52	RF00921	5dun	2.000	54	313	200
53	RF00044	3r4f	2.000	66	213	54
54	RF02681	5t5a	2.000	62	322	150
55	RF02266	4plx	2.000	76	412	287
56	RF01300	5ob3	2.000	69	319	138
57	RF01982	5kpy	2.000	71	346	211

B. Mathematical Details Related to DCA

Recall that the marginal probability for single-site pairs is given by

$$p_i(a_i) = \sum_{\{a_k | k \neq i\}} p(a_1, a_2, ..., a_L) \tag{B-1}$$

where $p(a_1, a_2, ..., a_L)$ is given by

$$p(a_1, ..., a_L) = \frac{1}{Z} \exp\left(\sum_{i<j} J_{ij}(a_i, a_j) + \sum_i h_i(a_i) \right). \tag{B-2}$$

The partition function Z is written as

$$Z = \sum_{\{a_k\}} \exp\left(\sum_{i<j} J_{ij}(a_i, a_j) + \sum_i h_i(a_i) \right). \tag{B-3}$$

Computing $\frac{\partial Z}{\partial h_i(a_i)}$

$$\begin{aligned}
\frac{\partial Z}{\partial h_l(b_l)} &= \sum_{\{a_k\}} \exp\left(\sum_{i<j} J_{ij}(a_i, a_j) + \sum_i h_i(a_i) \right) \sum_i \frac{\partial h_i(a_i)}{\partial h_l(b_l)} \\
&= \sum_{\{a_k\}} \exp\left(\sum_{i<j} J_{ij}(a_i, a_j) + \sum_i h_i(a_i) \right) \sum_i \delta_{l,i} \delta_{b_l, a_i} \\
&= \sum_{\{a_k\}} \sum_i \exp\left(\sum_{i<j} J_{ij}(a_i, a_j) + \sum_i h_i(a_i) \right) \delta_{l,i} \delta_{b_l, a_i} \\
&= \sum_{\{a_k\}} \exp\left(\sum_{l<j} J_{lj}(a_l, a_j) + \sum_l h_l(a_l) \right) \delta_{b_l, a_l} \\
&= \sum_{\{a_k | k \neq l\}} \sum_{a_l} \exp\left(\sum_{l<j} J_{lj}(a_l, a_j) + \sum_l h_l(a_l) \right) \delta_{b_l, a_l} \\
&= \sum_{\{a_k | k \neq l\}} \exp\left(\sum_{l<j} J_{lj}(b_l, a_j) + \sum_l h_l(b_l) \right) \\
&= Z p_l(b_l).
\end{aligned} \tag{B-4}$$

Computing $\frac{\partial^2 Z}{\partial h_i(a_i)\partial h_j(a_j)}$

Starting from the result of $\frac{\partial Z}{\partial h_i(a_i)}$

$$\begin{aligned}
\frac{\partial^2 Z}{\partial h_l(b_l)\partial h_m(c_m)} &= \frac{\partial}{\partial h_m(c_m)} \sum_{\{a_k|k\neq l\}} \exp\left(\sum_{l<j} J_{lj}(b_l,a_j) + \sum_l h_l(b_l)\right) \\
&= \sum_{\{a_k|k\neq l\}} \exp\left(\sum_{l<j} J_{lj}(b_l,a_j) + \sum_l h_l(b_l)\right) \sum_l \frac{\partial h_l(b_l)}{\partial h_m(c_m)} \\
&= \sum_{\{a_k|k\neq l\}} \exp\left(\sum_{l<j} J_{lj}(b_l,a_j) + \sum_l h_l(b_l)\right) \sum_l \delta_{l,m}\delta_{c_m,b_l} \\
&= \sum_{\{a_k|k\neq l\}} \sum_l \exp\left(\sum_{l<j} J_{lj}(b_l,a_j) + \sum_l h_l(b_l)\right) \delta_{l,m}\delta_{c_m,b_l} \\
&= \sum_{\{a_k|k\neq l\}} \exp\left(\sum_{m<j} J_{mj}(b_m,a_j) + \sum_m h_m(b_m)\right) \delta_{c_m,b_m} \\
&= \sum_{\{a_k|k\neq l,m\}} \sum_{c_m} \exp\left(\sum_{m<j} J_{mj}(b_m,a_j) + \sum_m h_m(b_m)\right) \delta_{c_m,b_m} \\
&= \sum_{\{a_k|k\neq l,m\}} \exp\left(\sum_{m<j} J_{mj}(c_m,a_j) + \sum_m h_m(c_m)\right) \\
&= Z p_{lm}(b_l,c_m)
\end{aligned} \tag{B-5}$$

Computing $\frac{\partial^2 \ln Z}{\partial h_i(a_i)\partial h_j(a_j)}$

$$\begin{aligned}
\frac{\partial^2 \ln Z}{\partial h_i(a_i)\partial h_j(a_j)} &= \frac{\partial}{\partial h_i(a_i)}\left(\frac{1}{Z}\frac{\partial Z}{\partial h_j(a_j)}\right) \\
&= -\frac{1}{Z^2}\frac{\partial Z}{\partial h_i(a_i)}\frac{\partial Z}{\partial h_j(a_j)} + \frac{1}{Z}\frac{\partial^2 Z}{\partial h_i(a_i)\partial h_j(a_j)} \\
&= p_{ij}(a_i,a_j) - p_i(a_i)p_j(a_j)
\end{aligned} \tag{B-6}$$

Legendre transformation

Now we rewrite the global probability model as

$$p(a_1,...,a_L) = \frac{1}{Z}\exp\left(\alpha \sum_{i<j} J_{i,j}(a_i,a_j) + \sum_i h_i(a_i)\right), \tag{B-7}$$

with a new hamiltonian that depends on the paramter α. The free energy for this hamiltonian is

$$\mathcal{F} = -\ln Z_\alpha, \tag{B-8}$$

where

$$Z_\alpha = \sum_{\{a_i\}} \exp\left(\alpha \sum_{i<j} J_{i,j}(a_i, a_j) + \sum_i h_i(a_i)\right). \tag{B-9}$$

The free energy $\mathcal{F}$ depends on couplings and fields as well as the parameter α. We want to eliminate the dependency on the fields. Using Legendre transformation, we write

$$\mathcal{G} = \mathcal{F} - \sum_i \sum_{a_i=1}^{q-1} h_i(a_i) \frac{\partial \mathcal{F}}{\partial h_i(a_i)}. \tag{B-10}$$

The summation $\sum_{a_i=1}^{q-1}$ runs until $q-1$ to indicates that gap states are excluded. The partial derivate of $\mathcal{F}$ with respect to fields is given by

$$\begin{aligned} \frac{\partial F}{\partial h_l(b_l)} &= \frac{\partial(-\ln Z)}{\partial h_l(b_l)} \\ &= -\frac{1}{Z}\frac{\partial Z}{\partial h_l(b_l)} \\ &= -p_l(b_l). \end{aligned} \tag{B-11}$$

In the last step of equation B-11 we have used the result of equation B-4. Now the Gibbs free energy takes the form

$$\mathcal{G} = \mathcal{F} + \sum_i \sum_{a_i=1}^{q-1} h_i(a_i) p_i(a_i). \tag{B-12}$$

As a result of Legendre transformation the Gibbs free energy depends on single-site probabilities instead of fields in additon to the couplings and the parameter α. Considering $\mathcal{G} = \mathcal{G}(\{J_{ij}(a_i, a_j), p_i(a_i)\})$, the total derivative of $\mathcal{G}$ is

$$d\mathcal{G} = \sum_{i<j} \sum_{a_i, a_j} \frac{\partial \mathcal{G}}{\partial J_{ij}(a_i, a_j)} dJ_{ij}(a_i, a_j) + \sum_i \sum_{a_i} \frac{\partial \mathcal{G}}{\partial p_i(a_i)} dp_i(a_i). \tag{B-13}$$

The corresponding expression from equation B-12 is

$$\begin{aligned} d\mathcal{G} &= d\mathcal{F} + \sum_i \sum_{a_i=1}^{q-1} [h_i(a_i)dp_i(a_i) + p_i(a_i)dh_i(a_i)] \\ &= \sum_{i<j} \sum_{a_i,a_j} \frac{\partial \mathcal{F}}{\partial J_{ij}(a_i,a_j)} dJ_{ij}(a_i,a_j) + \sum_i \sum_{a_i} \frac{\partial \mathcal{F}}{\partial h_i(a_i)} dh_i(a_i) + \sum_i \sum_{a_i=1}^{q-1} [h_i(a_i)dp_i(a_i) + p_i(a_i)dh_i(a_i)] \\ &= \sum_{i<j} \sum_{a_i,a_j} \frac{\partial \mathcal{F}}{\partial J_{ij}(a_i,a_j)} dJ_{ij}(a_i,a_j) - \sum_i \sum_{a_i} p_i(a_i)dh_i(a_i) + \sum_i \sum_{a_i=1}^{q-1} [h_i(a_i)dp_i(a_i) + p_i(a_i)dh_i(a_i)] \\ &= \sum_{i<j} \sum_{a_i,a_j} \frac{\partial \mathcal{F}}{\partial J_{ij}(a_i,a_j)} dJ_{ij}(a_i,a_j) + \sum_i \sum_{a_i=1}^{q-1} h_i(a_i)dp_i(a_i). \end{aligned} \tag{B-14}$$

Comparting equations B-13 and B-14 we see that

$$h_i(a_i) = \frac{\partial \mathcal{G}}{\partial p_i(a_i)} \tag{B-15}$$

Small coupling expansion and mean-field approximation

Expanding $\mathcal{G}$ in powers of α upto linear order we write,

$$\mathcal{G}(\alpha) = \mathcal{G}(0) + \frac{\partial \mathcal{G}}{\partial \alpha}\Big|_{\alpha=0} \alpha + \mathcal{O}(\alpha^2). \tag{B-16}$$

At $\alpha = 0$ there is no site-pair interaction. In addition, the Legendre transformation has removed the contribution of the fields. This implies that $\mathcal{G}(0)$ has contribution from the (negative) entropy which can be expressed in terms of single-site probabilities as

$$\begin{aligned} \mathcal{G}(0) &= \sum_i \sum_{a_i=1}^{q} p_i(a_i) \ln p_i(a_i) \\ &= \sum_i \sum_{a_i=1}^{q-1} p_i(a_i) \ln p_i(a_i) + \sum_i p_i(q) \ln p_i(q) \\ &= \sum_i \sum_{a_i=1}^{q-1} p_i(a_i) \ln p_i(a_i) + \sum_i [1 - \sum_{a_i=1}^{q-1} p_i(a_i)] \ln[1 - \sum_{a_i=1}^{q-1} p_i(a_i)]. \end{aligned} \tag{B-17}$$

In the last line of equation B-17 we have used the normalization condition for single-site probabilities $\sum_{a_i=1}^{q} p_i(a_i) = 1$. The partial derivative of $\mathcal{G}$ with respect to α is

$$\begin{aligned} \frac{\partial \mathcal{G}}{\partial \alpha} &= \frac{\partial \mathcal{F}_\alpha}{\partial \alpha} + \sum_i \sum_{a_i} p_i(a_i) \frac{\partial h_i(a_i)}{\partial \alpha} \\ &= -\frac{1}{Z_\alpha} \frac{\partial Z_\alpha}{\partial \alpha} + \sum_i \sum_{a_i} p_i(a_i) \frac{\partial h_i(a_i)}{\partial \alpha}. \end{aligned} \tag{B-18}$$

Let us compute $\frac{\partial Z_\alpha}{\partial \alpha}$ for equation B-18

$$\begin{aligned}\frac{\partial Z_\alpha}{\partial \alpha} &= \frac{\partial}{\partial \alpha}\sum_{\{a_i\}} \exp[\alpha \sum_{i<j} J_{ij}(a_i,a_j) + \sum_i h_i(a_i)] \\ &= \sum_{\{a_i\}} \left(\exp[\alpha \sum_{i<j} J_{ij}(a_i,a_j) + \sum_i h_i(a_i)][\sum_{i<j} J_{ij}(a_i,a_j)] \right)\end{aligned} \tag{B-19}$$

Substituting equation B-19 in to equation B-18 we get

$$\frac{\partial \mathcal{G}}{\partial \alpha} = -\left\langle \sum_{i<j} J_{ij}(a_i,a_j) \right\rangle_\alpha , \tag{B-20}$$

i.e., the partial derivative of $\mathcal{G}$ with respect to α is proportional to the average of the couplings over the distribution whose partition function is Z_α. When $\alpha = 0$ the averaging can be done using the independent two-site joint probabilities as

$$\frac{\partial G}{\partial \alpha}\Big|_{\alpha=0} = -\sum_{i<j}\sum_{a_i=1}^{q-1}\sum_{a_j=1}^{q-1} J_{ij}(a_i,a_j) p_i(a_i) p_j(a_j). \tag{B-21}$$

Note that the summations in equation B-21 run up to $q-1$ since the couplings involving q are zero due to gauge fixation. Next we compute the partial derivatives of the Gibbs free energy with respect to single site probabilities.

Computing $\frac{\partial \mathcal{G}}{\partial p_i(a_i)}$

Lets compute $\frac{\partial \mathcal{G}(0)}{\partial p_i(a_i)}$ first term by term.

First term

$$\begin{aligned}\frac{\partial}{\partial p_k(b_k)} \sum_i \sum_{a_i=1}^{q-1} p_i(a_i)\ln p_i(a_i) &= \sum_i \sum_{a_i=1}^{q-1} [\ln p_i(a_i) + 1]\delta_{i,k}\delta_{a_i,b_k} \\ &= \sum_i [\ln p_i(b_k) + 1]\delta_{i,k} \\ &= \ln p_k(b_k) + 1\end{aligned} \tag{B-22}$$

Second term

$$\begin{aligned}\frac{\partial}{\partial p_k(b_k)} \sum_i [1 - \sum_{a_i=1}^{q-1} p_i(a_i)] \ln[1 - \sum_{a_i=1}^{q-1} p_i(a_i)] &= -\sum_i \sum_{a_i=1}^{q-1} [\ln[1 - \sum_{a_i=1}^{q-1} p_i(a_i)] + 1]\delta_{i,k}\delta_{a_i,b_k} \\ &= -\sum_k [\ln[1 - \sum_{b_k=1}^{q-1} p_i(b_k)] + 1]\delta_{i,k} \\ &= -\ln\left(1 - \sum_{b_k=1}^{q-1} p_k(b_k)\right) - 1 = -\ln p_k(q) - 1\end{aligned} \tag{B-23}$$

Next we compute the partial derivative of $\frac{\partial \mathcal{G}}{\partial \alpha}\Big|_{\alpha=0}$ with respect to single-site probabilities.

$$\begin{aligned}
\frac{\partial}{\partial p_k(b_k)}\left(\frac{\partial \mathcal{G}}{\partial \alpha}\Big|_{\alpha=0}\right) &= -\frac{\partial}{\partial p_k(b_k)}\left(\sum_{i<j}\sum_{a_i=1}^{q-1}\sum_{a_j=1}^{q-1} J_{ij}(a_i,a_j)p_i(a_i)p_j(a_j)\right) \\
&= -\sum_{i<j}\sum_{a_i=1}^{q-1}\sum_{a_j=1}^{q-1} J_{ij}(a_i,a_j)[p_j(a_j)\delta_{i,k}\delta_{a_i,b_k} + p_i(a_i)\delta_{j,k}\delta_{a_j,b_k}] \\
&= -\sum_{i<j}\left(\sum_{a_j=1}^{q-1} J_{ij}(b_k,a_j)p_j(a_j)\delta_{i,k} + \sum_{a_i=1}^{q-1} J_{ij}(a_i,b_k)p_i(a_i)\delta_{j,k}\right) \\
&= -\frac{1}{2}\sum_{i}\sum_{j\neq i}\left(\sum_{a_j=1}^{q-1} J_{ij}(b_k,a_j)p_j(a_j)\delta_{i,k} + \sum_{a_i=1}^{q-1} J_{ij}(a_i,b_k)p_i(a_i)\delta_{j,k}\right) \\
&= -\frac{1}{2}\left(\sum_{j\neq k}\sum_{a_j=1}^{q-1} J_{kj}(b_k,a_j)p_j(a_j) + \sum_{i\neq k}\sum_{a_i=1}^{q-1} J_{ik}(a_i,b_k)p_i(a_i)\right) \\
&= -\frac{1}{2}\left(\sum_{j\neq k}\sum_{a_j=1}^{q-1} J_{kj}(b_k,a_j)p_j(a_j) + \sum_{i\neq k}\sum_{a_i=1}^{q-1} J_{ki}(b_k,a_i)p_i(a_i)\right) \\
&= -\frac{1}{2}\left(\sum_{j\neq k}\sum_{a_j=1}^{q-1} J_{kj}(b_k,a_j)p_j(a_j) + \sum_{j\neq k}\sum_{a_j=1}^{q-1} J_{kj}(b_k,a_j)p_i(a_j)\right) \\
&= -\sum_{j\neq k}\sum_{a_j=1}^{q-1} J_{kj}(b_k,a_j)p_j(a_j).
\end{aligned} \tag{B-24}$$

In equation B-24 we have used the fact that $J_{ij}(a_i,a_j) = J_{ji}(a_j,a_i)$. Combining the results of equations B-22, B-23 and B-24 we have

$$\begin{aligned}
h_k(b_k) = \frac{\partial \mathcal{G}}{\partial p_k(b_k)} &= \ln p_k(b_k) - \ln p_k(q) - \sum_{j\neq k}\sum_{a_j=1}^{q-1} J_{kj}(b_k,a_j)p_j(a_j) \\
&= \ln \frac{p_k(b_k)}{p_k(q)} - \sum_{j\neq k}\sum_{a_j=1}^{q-1} J_{kj}(b_k,a_j)p_j(a_j).
\end{aligned} \tag{B-25}$$

Solving for single site-probabilities, a self-consistent equation is obtained as

$$\frac{p_k(b_k)}{p_k(q)} = \exp\left(h_k(b_k) + \sum_{j\neq k}\sum_{a_j=1}^{q-1} J_{kj}(b_k,a_j)p_j(a_j)\right) \tag{B-26}$$

Computing $\frac{\partial^2 \mathcal{G}}{\partial p_i(a_i)\partial p_j(a_j)}$

Using the result of $h_{a_i} = \frac{\partial \mathcal{G}}{\partial p_i(a_i)}$,

$$\begin{aligned}\frac{\partial^2 \mathcal{G}}{\partial p_k(b_k)\partial p_l(c_l)} &= \frac{\partial}{\partial p_l(c_l)}\left(\ln p_k(b_k) + \ln[1 - \sum_{b_k=1}^{q-1} p_k(b_k)] - \sum_{j\neq k}\sum_{a_j=1}^{q-1} J_{kj}(b_k, a_j)p_j(a_j)\right) \\ &= \frac{\delta_{k,l}\delta_{b_k,c_l}}{p_k(b_k)} - \sum_{b_k=1}^{q-1}\frac{\delta_{k,l}\delta_{b_k,c_l}}{p_k(q)} - \sum_{j\neq k}\sum_{a_j=1}^{q-1} J_{kj}(b_k, a_j)\delta_{j,l}\delta_{a_j,c_l} \\ &= \frac{\delta_{k,l}\delta_{b_k,c_l}}{p_k(b_k)} - \frac{\delta_{k,l}}{p_k(q)} - \sum_{j\neq k} J_{kj}(b_k, c_l)\delta_{j,l} \\ &= \frac{\delta_{k,l}\delta_{b_k,c_l}}{p_k(b_k)} - \frac{\delta_{k,l}}{p_k(q)} - J_{kl}(b_k, c_l)\end{aligned} \tag{B-27}$$

The correlation is related to the Gibbs free energy as

$$\frac{\partial h_i(a_i)}{\partial p_i(a_i)} = \frac{\partial^2 \mathcal{G}}{\partial p_i(a_i)\partial p_j(a_j)} = C_{ij}^{-1}(a_i, a_j) \tag{B-28}$$

From which we get

$$C_{ij}^{-1}(a_i, a_j) = \frac{\delta_{i,j}\delta_{a_i,a_j}}{p_i(a_i)} - \frac{\delta_{i,j}}{p_i(q)} - J_{ij}(a_i, a_j) \tag{B-29}$$

For $i = j$ the couplings are zero by definition (no self interaction). The correlation then becomes

$$C_{ii}^{-1}(a_i, b_i) = \frac{\delta_{a_i,b_i}}{p_i(a_i)} - \frac{1}{p_i(q)} \tag{B-30}$$

For $j \neq i$ the correlation is

$$C_{ij}^{-1}(a_i, a_j) = -J_{ij}(a_i, a_j) \tag{B-31}$$

Pseudo-likelihood maximization

Gradients of the objective function

Consider the objective function give by

$$\log l(\{h_i(a^m)\}, \{J_{ij}(a^m, a_j^m)\}) = \sum_{m=1}^{M}\sum_{i=1}^{L}\left(h_i(a^m) + \sum_{i=1|i\neq j}^{L} J_{ij}(a^m, a_j^m) - \ln \mathcal{Z}_i^m\right). \tag{B-32}$$

$$\begin{aligned}\frac{\partial \log l(\{h_i(a^m)\},\{J_{ij}(a^m,a_j^m)\})}{\partial h_k(b^n)} &= \sum_{m=1}^{M}\sum_{i=1}^{L}\left(\delta_{ik}\delta_{a^m,b^n} - \frac{\partial \ln \mathcal{Z}_i^m}{\partial h_k(b^n)}\right) \\ &= \sum_{m=1}^{M}\delta_{a^m,b} - \sum_{m=1}^{M}\sum_{i=1}^{L}\frac{\partial \ln \mathcal{Z}_i^m}{\partial h_k(b^n)} \\ &= \sum_{m=1}^{M}\delta_{a^m,b} - \sum_{m=1}^{M}\sum_{i=1}^{L}\frac{1}{\mathcal{Z}_i^m}\frac{\partial \mathcal{Z}_i^m}{\partial h_k(b^n)}\end{aligned} \tag{B-33}$$

Lets compute $\frac{\partial \mathcal{Z}_i^m}{\partial h_k(b)}$

$$\begin{aligned}\frac{\partial \mathcal{Z}_i^m}{\partial h_k(b)} &= \frac{\partial}{\partial h_k(b)}\sum_{a^m=1}^{L}\exp\left(h_i(a^m) + \sum_{j=1|j\neq i}^{L} J_{ij}(a^m,a_j^m)\right) \\ &= \sum_{a^m=1}^{L}\exp\left(h_i(a^m) + \sum_{j=1|j\neq i}^{L} J_{ij}(a^m,a_j^m)\right)\delta_{ik}\delta_{a^m,b} \\ &= \exp\left(h_i(b) + \sum_{j=1|j\neq i}^{L} J_{ij}(b,a_j^m)\right)\delta_{ik}\end{aligned} \tag{B-34}$$

It follows that

$$\begin{aligned}\frac{\partial \log l(\{h_i(a^m)\},\{J_{ij}(a^m,a_j^m)\})}{\partial h_k(b)} &= \sum_{m=1}^{M}\delta_{a^m,b} - \sum_{m=1}^{M}\sum_{i=1}^{L}\frac{1}{\mathcal{Z}_i^m}\exp\left(h_i(b) + \sum_{j=1|j\neq i}^{L} J_{ij}(b,a_j^m)\right)\delta_{ik} \\ &\sum_{m=1}^{M}\delta_{a^m,b} - \sum_{m=1}^{M}\frac{1}{\mathcal{Z}_k^m}\exp\left(h_k(b) + \sum_{j=1|j\neq k}^{L} J_{kj}(b,a_j^m)\right) \\ &= \sum_{m=1}^{M}(\delta_{a^m,b} - p_k^m(b))\end{aligned} \tag{B-35}$$

Now we compute the gradient of $\log l(\{h_i(a^m)\},\{J_{ij}(a^m,a_j^m)\})$ with respect to couplings.

$$\begin{aligned}\frac{\partial \log l(\{h_i(a^m)\},\{J_{ij}(a^m,a_j^m)\})}{\partial J_{kl}(a,b)} &= \sum_{m=1}^{M}\sum_{i=1}^{L}\left(\sum_{j=1|j\neq i}^{L}\delta_{i,k\delta_{j,l}}\delta_{a^m,a}\delta_{a_j^m,b} - \frac{\partial \ln \mathcal{Z}_i^m}{\partial J_{kl}(a,b)}\right) \\ &= \sum_{m=1}^{M}\sum_{i=1|i\neq l}^{L}\delta_{i,k}\delta_{a^m,a}\delta_{a_l^m,b} - \sum_{m=1}^{m}\sum_{i=1}^{L}\frac{1}{\mathcal{Z}_i^m}\frac{\partial \mathcal{Z}_i^m}{\partial J_{kl}(a,b)}\end{aligned} \tag{B-36}$$

Let us compute $\frac{\partial \mathcal{Z}_i^m}{\partial J_{kl}(a,b)}$.

$$\begin{aligned}\frac{\partial \mathcal{Z}_i^m}{\partial J_{kl}(a,b)} &= \frac{\partial}{\partial J_{kl}(a,b)} \sum_{a^m=1}^{q} \exp\left(h_i(a^m) + \sum_{j=1|j\neq i} J_{ij}(a^m, a_j^m)\right) \\ &= \sum_{a^m=1}^{q} \exp\left(h_i(a^m) + \sum_{j=1|j\neq i} J_{ij}(a^m, a_j^m)\right) \sum_{j=1|j\neq i} \delta_{ik}\delta_{jl}\delta_{a^m,a}\delta_{a_j^m,b} \qquad \text{(B-37)} \\ &= \exp\left(h_i(a) + \sum_{j=1|j\neq i} J_{ij}(a, a_j^m)\right) \delta_{ik}\delta_{a_l^m,b}\end{aligned}$$

It follows that

$$\begin{aligned}\frac{\partial \log l(\{h_i(a^m)\}, \{J_{ij}(a^m, a_j^m)\})}{\partial J_{kl}(a,b)} &= \sum_{m=1}^{M} \delta_{a^m,a}\delta_{a_l^m,b} - \\ \sum_{m=1}^{M}\sum_{i=1}^{L} \frac{1}{\mathcal{Z}_i^m} \exp\left(h_i(a) + \sum_{j=1|j\neq i} J_{ij}(a, a_j^m)\right) \delta_{ik}\delta_{a_l^m,b} \\ = \sum_{m=1}^{M} \delta_{a^m,a}\delta_{a_l^m,b} - \sum_{m=1}^{M} \frac{1}{\mathcal{Z}_k^m} \exp\left(h_k(a) + \sum_{j=1|j\neq k} J_{kj}(a, a_j^m)\right) \delta_{a_l^m,b} \qquad \text{(B-38)} \\ = \sum_{m=1}^{M} \left(\delta_{a^m,a}\delta_{a_l^m,b} - p_k^m(a)\delta_{a_l^m,b}\right) \\ = \sum_{m=1}^{M} (\delta_{a^m,a} - p_k^m(a))\delta_{a_l^m,b}\end{aligned}$$

Bibliography

[1] Abadi, Martín ; Agarwal, Ashish ; Barham, Paul ; Brevdo, Eugene ; Chen, Zhifeng ; Citro, Craig ; Corrado, Greg S. ; Davis, Andy ; Dean, Jeffrey ; Devin, Matthieu ; Ghemawat, Sanjay ; Goodfellow, Ian ; Harp, Andrew ; Irving, Geoffrey ; Isard, Michael ; Jia, Yangqing ; Jozefowicz, Rafal ; Kaiser, Lukasz ; Kudlur, Manjunath ; Levenberg, Josh ; Mané, Dandelion ; Monga, Rajat ; Moore, Sherry ; Murray, Derek ; Olah, Chris ; Schuster, Mike ; Shlens, Jonathon ; Steiner, Benoit ; Sutskever, Ilya ; Talwar, Kunal ; Tucker, Paul ; Vanhoucke, Vincent ; Vasudevan, Vijay ; Viégas, Fernanda ; Vinyals, Oriol ; Warden, Pete ; Wattenberg, Martin ; Wicke, Martin ; Yu, Yuan ; Zheng, Xiaoqiang: *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. 2015. – Software available from tensorflow.org

[2] Adhikari, Badri ; Hou, Jie ; Cheng, Jianlin: Protein contact prediction by integrating deep multiple sequence alignments, coevolution and machine learning. In: *Proteins: Structure, Function, and Bioinformatics* 86 (2018), Nr. S1, S. 84–96

[3] Al-Hashimi, Hashim M. ; Walter, Nils G.: RNA dynamics: it is about time. In: *Current Opinion in Structural Biology* 18 (2008), Nr. 3, S. 321–329. – Nucleic acids / Sequences and topology. – ISSN 0959–440X

[4] Altschuh, D. ; Lesk, A.M. ; Bloomer, A.C. ; Klug, A.: Correlation of co-ordinated amino acid substitutions with function in viruses related to tobacco mosaic virus. In: *Journal of Molecular Biology* 193 (1987), Nr. 4, S. 693 – 707. – ISSN 0022–2836

[5] Altschuh, D. ; Vernet, T. ; Berti, P. ; Moras, D. ; Nagai, K.: Coordinated amino acid changes in homologous protein families*. In: *Protein Engineering, Design and Selection* 2 (1988), 09, Nr. 3, S. 193–199. – ISSN 1741–0126

[6] Amidi, Shervine: *Convolutional neural nets*. 3 2021. – [Online; accessed 2021-03-25]

[7] Anderson, Kelly M. ; Anderson, Douglas M. ; McAnally, John R. ; Shelton, John M. ; Bassel-Duby, Rhonda ; Olson, Eric N.: Transcription of the non-coding RNA upperhand controls Hand2 expression and heart development. In: *Nature* 539 (2016), Nov, Nr. 7629, S. 433–436. – ISSN 1476–4687

[8] Bai, Fang ; Morcos, Faruck ; Cheng, Ryan R. ; Jiang, Hualiang ; Onuchic, José N.: Elucidating the druggable interface of protein-protein interactions using fragment docking and coevolutionary analysis. In: *Proceedings of the National Academy of Sciences* 113 (2016), Nr. 50, S. E8051–E8058. – ISSN 0027–8424

[9] Balakrishnan, Sivaraman ; Kamisetty, Hetunandan ; Carbonell, Jaime G. ; Lee, Su-In ; Langmead, Christopher J.: Learning generative models for protein fold families. In: *Proteins: Structure, Function, and Bioinformatics* 79 (2011), Nr. 4, S. 1061–1078

[10] Baldassi, Carlo ; Zamparo, Marco ; Feinauer, Christoph ; Procaccini, Andrea ; Zecchina, Riccardo ; Weigt, Martin ; Pagnani, Andrea: Fast and Accurate Multivariate Gaussian Modeling of Protein Families: Predicting Residue Contacts and Protein-Interaction Partners. In: *PLOS ONE* 9 (2014), 03, Nr. 3, S. 1–12

[11] Boniecki, Michal J. ; Lach, Grzegorz ; Dawson, Wayne K. ; Tomala, Konrad ; Lukasz, Pawel ; Soltysinski, Tomasz ; Rother, Kristian M. ; Bujnicki, Janusz M.: SimRNA: a coarse-grained method for RNA folding simulations and 3D structure prediction. In: *Nucleic Acids Research* 44 (2015), 12, Nr. 7, S. e63–e63. – ISSN 0305–1048

[12] Boureau, Y. Lan ; Ponce, Jean ; Lecun, Yann: A theoretical analysis of feature pooling in visual recognition. In: *ICML 2010 - Proceedings, 27th International Conference on Machine Learning*, 2010 (ICML 2010 - Proceedings, 27th International Conference on Machine Learning). – 27th International Conference on Machine Learning, ICML 2010 ; Conference date: 21-06-2010 Through 25-06-2010. – ISBN 9781605589077, S. 111–118

[13] Byron, Olwyn ; Gilbert, Robert J.: Neutron scattering: good news for biotechnology. In: *Current Opinion in Biotechnology* 11 (2000), Nr. 1, S. 72–80. – ISSN 0958–1669

[14] Castel, Stephane E. ; Martienssen, Robert A.: RNA interference in the nucleus: roles for small RNAs in transcription, epigenetics and beyond. In: *Nature Reviews Genetics* 14 (2013), Feb, Nr. 2, S. 100–112. – ISSN 1471–0064

[15] Cate, Jamie H. ; Doudna, Jennifer A.: [12] Solving large RNA structures by X-ray crystallography. In: *RNA - Ligand Interactions, Part A* Bd. 317. Academic Press, 2000. – ISSN 0076–6879, S. 169–180

[16] Cheng, R. R. ; Nordesjö, O. ; Hayes, R. L. ; Levine, H. ; Flores, S. C. ; Onuchic, J. N. ; Morcos, F.: Connecting the Sequence-Space of Bacterial Signaling Proteins to Phenotypes Using Coevolutionary Landscapes. In: *Molecular Biology and Evolution* 33 (2016), 09, Nr. 12, S. 3054–3064. – ISSN 0737–4038

[17] Cheng, Ryan R. ; Morcos, Faruck ; Levine, Herbert ; Onuchic, José N.: Toward rationally redesigning bacterial two-component signaling systems using coevolutionary information. In: *Proceedings of the National Academy of Sciences* 111 (2014), Nr. 5, S. E563–E571. – ISSN 0027–8424

[18] Chicco, Davide ; Jurman, Giuseppe: The advantages of the Matthews correlation coefficient (MCC) over F1 score and accuracy in binary classification evaluation. In: *BMC Genomics* 21 (2020), Jan, Nr. 1, S. 6. – ISSN 1471–2164

[19] Cock, Peter J. A. ; Antao, Tiago ; Chang, Jeffrey T. ; Chapman, Brad A. ; Cox, Cymon J. ; Dalke, Andrew ; Friedberg, Iddo ; Hamelryck, Thomas ; Kauff, Frank ; Wilczynski, Bartek ; de Hoon, Michiel J. L.: Biopython: freely available Python tools for computational molecular biology and bioinformatics. In: *Bioinformatics* 25 (2009), 03, Nr. 11, S. 1422–1423. – ISSN 1367–4803

[20] Cover, T. M. ; Thomas, Joy A.: *Elements of information theory*. 2nd ed. Hoboken, N.J : Wiley-Interscience, 2006. – OCLC: ocm59879802. – ISBN 9780471241959

[21] Cuturello, Francesca ; Tiana, Guido ; Bussi, Giovanni: Assessing the accuracy of direct-coupling analysis for RNA contact prediction. In: *RNA* (2020)

[22] Dago, Angel E. ; Schug, Alexander ; Procaccini, Andrea ; Hoch, James A. ; Weigt, Martin ; Szurmant, Hendrik: Structural basis of histidine kinase autophosphorylation deduced by integrating genomics, molecular dynamics, and mutagenesis. In: *Proceedings of the National Academy of Sciences* 109 (2012), Nr. 26, S. E1733–E1742. – ISSN 0027–8424

[23] De Leonardis, Eleonora ; Lutz, Benjamin ; Ratz, Sebastian ; Cocco, Simona ; Monasson, Rémi ; Schug, Alexander ; Weigt, Martin: Direct-Coupling Analysis of nucleotide coevolution facilitates RNA secondary and tertiary structure prediction. In: *Nucleic Acids Research* 43 (2015), 09, Nr. 21, S. 10444–10455. – ISSN 0305–1048

[24] Dhillon, Anamika ; Verma, Gyanendra K.: Convolutional neural network: a review of models, methodologies and applications to object detection. In: *Progress in Artificial Intelligence* 9 (2020), Jun, Nr. 2, S. 85–112. – ISSN 2192–6360

[25] Eddy, Sean R.: Computational Analysis of Conserved RNA Secondary Structure in Transcriptomes and Genomes. In: *Annual Review of Biophysics* 43 (2014), Nr. 1, S. 433–456. – PMID: 24895857

[26] Edgar, Robert C.: MUSCLE: multiple sequence alignment with high accuracy and high throughput. In: *Nucleic Acids Research* 32 (2004), 03, Nr. 5, S. 1792–1797. – ISSN 0305–1048

[27] Ekeberg, Magnus ; Lövkvist, Cecilia ; Lan, Yueheng ; Weigt, Martin ; Aurell, Erik: Improved contact prediction in proteins: Using pseudolikelihoods to infer Potts models. In: *Phys. Rev. E* 87 (2013), Jan, S. 012707

[28] El-Gebali, Sara ; Mistry, Jaina ; Bateman, Alex ; Eddy, Sean R. ; Luciani, Aurélien ; Potter, Simon C. ; Qureshi, Matloob ; Richardson, Lorna J. ; Salazar, Gustavo A. ; Smart, Alfredo ; Sonnhammer, Erik L. ; Hirsh, Layla ; Paladin, Lisanna ; Piovesan, Damiano ; Tosatto, Silvio C. ; Finn, Robert D.: The Pfam protein families database in 2019. In: *Nucleic Acids Research* 47 (2018), 10, Nr. D1, S. D427–D432. – ISSN 0305–1048

[29] Figliuzzi, Matteo ; Jacquier, Hervé ; Schug, Alexander ; Tenaillon, Oliver ; Weigt, Martin: Coevolutionary Landscape Inference and the Context-Dependence of Mutations in Beta-Lactamase TEM-1. In: *Molecular Biology and Evolution* 33 (2015), 10, Nr. 1, S. 268–280. – ISSN 0737–4038

[30] Freddolino, Peter L. ; Harrison, Christopher B. ; Liu, Yanxin ; Schulten, Klaus: Challenges in protein-folding simulations. In: *Nature Physics* 6 (2010), Oct, Nr. 10, S. 751–758. – ISSN 1745–2481

[31] Garneau, Nicole L. ; Wilusz, Jeffrey ; Wilusz, Carol J.: The highways and byways of mRNA decay. In: *Nature Reviews Molecular Cell Biology* 8 (2007), Feb, Nr. 2, S. 113–126. – ISSN 1471–0080

[32] Green, Anna G. ; Elhabashy, Hadeer ; Brock, Kelly P. ; Maddamsetti, Rohan ; Kohlbacher, Oliver ; Marks, Debora S.: Large-scale discovery of protein interactions at residue resolution using co-evolution calculated from genomic sequences. In: *Nature Communications* 12 (2021), Mar, Nr. 1, S. 1396. – ISSN 2041–1723

[33] Hahnloser, Richard H. R. ; Sarpeshkar, Rahul ; Mahowald, Misha A. ; Douglas, Rodney J. ; Seung, H. S.: Digital selection and analogue amplification coexist in a cortex-inspired silicon circuit. In: *Nature* 405 (2000), Jun, Nr. 6789, S. 947–951. – ISSN 1476–4687

[34] Harris, Charles R. ; Millman, K. J. ; van der Walt, Stéfan J. ; Gommers, Ralf ; Virtanen, Pauli ; Cournapeau, David ; Wieser, Eric ; Taylor, Julian ; Berg, Sebastian ; Smith, Nathaniel J. ; Kern, Robert ; Picus, Matti ; Hoyer, Stephan ; van Kerkwijk, Marten H. ; Brett, Matthew ; Haldane, Allan ; Fernández del Río, Jaime ; Wiebe, Mark ; Peterson, Pearu ; Gérard-Marchant, Pierre ; Sheppard, Kevin ; Reddy, Tyler ; Weckesser, Warren ; Abbasi, Hameer ; Gohlke, Christoph ; Oliphant, Travis E.: Array programming with NumPy. In: *Nature* 585 (2020), S. 357–362

[35] Hayat, Sikander ; Sander, Chris ; Marks, Debora S. ; Elofsson, Arne: All-atom 3D structure prediction of transmembrane β-barrel proteins from sequences. In: *Proceedings of the National Academy of Sciences* 112 (2015), Apr, Nr. 17, S. 5413

[36] Hopf, Thomas A. ; Green, Anna G. ; Schubert, Benjamin ; Mersmann, Sophia ; Schärfe, Charlotta P I. ; Ingraham, John B. ; Toth-Petroczy, Agnes ; Brock, Kelly ; Riesselman, Adam J. ; Palmedo, Perry ; Kang, Chan ; Sheridan, Robert ; Draizen, Eli J. ; Dallago, Christian ; Sander, Chris ; Marks, Debora S.: The EVcouplings Python framework for coevolutionary sequence analysis. In: *Bioinformatics* 35 (2018), 10, Nr. 9, S. 1582–1584. – ISSN 1367–4803

[37] Hopf, Thomas A. ; Colwell, Lucy J. ; Sheridan, Robert ; Rost, Burkhard ; Sander, Chris ; Marks, Debora S.: Three-Dimensional Structures of Membrane Proteins from Genomic Sequencing. In: *Cell* 149 (2012), Nr. 7, S. 1607–1621. – ISSN 0092–8674

[38] Huang, Lili ; Serganov, Alexander ; Patel, Dinshaw J.: Structural Insights into Ligand Recognition by a Sensing Domain of the Cooperative Glycine Riboswitch. In: *Molecular Cell* 40 (2010), Dezember, Nr. 5, S. 774–786. – ISSN 10972765

[39] Jackson, Lisa A. ; Anderson, Evan J. ; Rouphael, Nadine G. ; Roberts, Paul C. ; Makhene, Mamodikoe ; Coler, Rhea N. ; McCullough, Michele P. ; Chappell, James D. ; Denison, Mark R. ; Stevens, Laura J. ; Pruijssers, Andrea J. ; McDermott, Adrian ; Flach, Britta ; Doria-Rose, Nicole A. ; Corbett, Kizzmekia S. ; Morabito, Kaitlyn M. ; O'Dell, Sijy ; Schmidt, Stephen D. ; Swanson, Phillip A. ; Padilla, Marcelino ; Mascola, John R. ; Neuzil, Kathleen M. ; Bennett, Hamilton ; Sun, Wellington ; Peters, Etza ; Makowski, Mat ; Albert, Jim ; Cross, Kaitlyn ; Buchanan, Wendy ; Pikaart-Tautges, Rhonda ; Ledgerwood, Julie E. ; Graham, Barney S. ; Beigel, John H.: An mRNA Vaccine against SARS-CoV-2 — Preliminary Report. In: *New England Journal of Medicine* 383 (2020), Nr. 20, S. 1920–1931

[40] Jones, David T. ; Buchan, Daniel W. A. ; Cozzetto, Domenico ; Pontil, Massimiliano: PSICOV: precise structural contact prediction using sparse inverse covariance estimation on large multiple sequence alignments. In: *Bioinformatics* 28 (2011), 11, Nr. 2, S. 184–190. – ISSN 1367–4803

[41] Kalvari, Ioanna ; Argasinska, Joanna ; Quinones-Olvera, Natalia ; Nawrocki, Eric P. ; Rivas, Elena ; Eddy, Sean R. ; Bateman, Alex ; Finn, Robert D. ; Petrov, Anton I.: Rfam 13.0: shifting to a genome-centric resource for non-coding RNA families. In: *Nucleic Acids Research* 46 (2017), 11, Nr. D1, S. D335–D342. – ISSN 0305–1048

[42] Kalvari, Ioanna ; Nawrocki, Eric P. ; Ontiveros-Palacios, Nancy ; Argasinska, Joanna ; Lamkiewicz, Kevin ; Marz, Manja ; Griffiths-Jones, Sam ; Toffano-Nioche, Claire ; Gautheret, Daniel ; Weinberg, Zasha ; Rivas, Elena ; Eddy, Sean R. ; Finn, Robert D ; Bateman, Alex ; Petrov, Anton I.: Rfam 14: expanded coverage of metagenomic, viral and microRNA families. In: *Nucleic Acids Research* 49 (2020), 11, Nr. D1, S. D192–D200. – ISSN 0305–1048

[43] Katoh, Kazutaka ; Standley, Daron M.: MAFFT Multiple Sequence Alignment Software Version 7: Improvements in Performance and Usability. In: *Molecular Biology and Evolution* 30 (2013), 01, Nr. 4, S. 772–780. – ISSN 0737–4038

[44] Kerpedjiev, Peter ; Hammer, Stefan ; Hofacker, Ivo L.: Forna (force-directed RNA): Simple and effective online RNA secondary structure diagrams. In: *Bioinformatics* 31 (2015), 06, Nr. 20, S. 3377–3379. – ISSN 1367–4803

[45] Krizhevsky, Alex ; Sutskever, Ilya ; Hinton, Geoffrey E.: ImageNet Classification with Deep Convolutional Neural Networks. In: Pereira, F. (Hrsg.) ; Burges, C. J. C. (Hrsg.) ; Bottou, L. (Hrsg.) ; Weinberger, K. Q. (Hrsg.): *Advances in Neural Information Processing Systems* Bd. 25, Curran Associates, Inc., 2012

[46] Lam, Siu K. ; Pitrou, Antoine ; Seibert, Stanley: Numba: A LLVM-Based Python JIT Compiler. In: *Proceedings of the Second Workshop on the LLVM Compiler Infrastructure in HPC*. New York, NY, USA : Association for Computing Machinery, 2015 (LLVM '15). – ISBN 9781450340052

[47] LE CUN, Y.: Convolutional networks for images, speech, and time series. In: *The handbook of brain theory and neural networks* (1995), S. 255–258

[48] Le Cun, Y. ; Boser, B. ; Denker, J. S. ; Henderson, D. ; Howard, R. E. ; Hubbard, W. ; Jackel, L. D.: Handwritten Digit Recognition with a Back-Propagation Network. In: *Proceedings of the 2nd International Conference on Neural Information Processing Systems*. Cambridge, MA, USA : MIT Press, 1989 (NIPS'89), S. 396–404

[49] LeCun, Y. ; Boser, B. ; Denker, J. S. ; Henderson, D. ; Howard, R. E. ; Hubbard, W. ; Jackel, L. D.: Backpropagation Applied to Handwritten Zip Code Recognition. In: *Neural Computation* 1 (1989), Nr. 4, S. 541–551

[50] Lecun, Y. ; Bottou, L. ; Bengio, Y. ; Haffner, P.: Gradient-based learning applied to document recognition. In: *Proceedings of the IEEE* 86 (1998), Nr. 11, S. 2278–2324

[51] LeCun, Y. ; Kavukcuoglu, K. ; Farabet, C.: Convolutional networks and applications in vision. In: *Proceedings of 2010 IEEE International Symposium on Circuits and Systems*, 2010, S. 253–256

[52] LeCun, Yann ; Bengio, Yoshua ; Hinton, Geoffrey: Deep learning. In: *Nature* 521 (2015), May, Nr. 7553, S. 436–444. – ISSN 1476–4687

[53] LeCun, Yann ; Bottou, Léon ; Orr, Genevieve B. ; Müller, Klaus-Robert: Efficient BackProp. In: *Neural Networks: Tricks of the Trade*. 1996, S. 9–50

[54] LEONTIS, NEOCLES B. ; WESTHOF, ERIC: Geometric nomenclature and classification of RNA base pairs. In: *RNA* 7 (2001), Nr. 4, S. 499–512

[55] Li, Bing ; Cao, Yang ; Westhof, Eric ; Miao, Zhichao: Advances in RNA 3D Structure Modeling Using Experimental Data. In: *Frontiers in Genetics* 11 (2020), S. 1147. – ISSN 1664–8021

[56] Li, Bing ; Cao, Yang ; Westhof, Eric ; Miao, Zhichao: Advances in RNA 3D Structure Modeling Using Experimental Data. In: *Frontiers in Genetics* 11 (2020), S. 1147. – ISSN 1664–8021

[57] Liu, Dong C. ; Nocedal, Jorge: On the limited memory BFGS method for large scale optimization. In: *Mathematical Programming* 45 (1989), Aug, Nr. 1, S. 503–528. – ISSN 1436–4646

[58] Long, Yicheng ; Wang, Xueyin ; Youmans, Daniel T. ; Cech, Thomas R.: How do lncRNAs regulate transcription? In: *Science Advances* 3 (2017), Nr. 9

[59] Luco, Reini F. ; Misteli, Tom: More than a splicing code: integrating the role of RNA, chromatin and non-coding RNA in alternative splicing regulation. In: *Current Opinion in Genetics & Development* 21 (2011), Nr. 4, S. 366–372. – Differentiation and gene regulation. – ISSN 0959–437X

[60] MacKay, David J. C.: *Information Theory, Inference and Learning Algorithms*. USA : Cambridge University Press, 2002. – ISBN 0521642981

[61] Marks, Debora S. ; Colwell, Lucy J. ; Sheridan, Robert ; Hopf, Thomas A. ; Pagnani, Andrea ; Zecchina, Riccardo ; Sander, Chris: Protein 3D Structure Computed from Evolutionary Sequence Variation. In: *PLOS ONE*

[62] Marks, Debora S. ; Hopf, Thomas A. ; Sander, Chris: Protein structure prediction from sequence variation. In: *Nature Biotechnology* 30 (2012), Nov, Nr. 11, S. 1072–1080. – ISSN 1546–1696

[63] Mattick, John S. ; Makunin, Igor V.: Non-coding RNA. In: *Human Molecular Genetics* 15 (2006), 04, Nr. suppl_1, S. R17–R29. – ISSN 0964–6906

[64] Montange, Rebecca K. ; Mondragón, Estefanía ; van Tyne, Daria ; Garst, Andrew D. ; Ceres, Pablo ; Batey, Robert T.: Discrimination between Closely Related Cellular Metabolites by the SAM-I Riboswitch. In: *Journal of Molecular Biology* 396 (2010), Nr. 3, S. 761–772. – ISSN 0022–2836

[65] Morcos, Faruck ; Jana, Biman ; Hwa, Terence ; Onuchic, José N.: Coevolutionary signals across protein lineages help capture multiple protein conformations. In: *Proceedings of the National Academy of Sciences* 110 (2013), Nr. 51, S. 20533–20538. – ISSN 0027–8424

[66] Morcos, Faruck ; Pagnani, Andrea ; Lunt, Bryan ; Bertolino, Arianna ; Marks, Debora S. ; Sander, Chris ; Zecchina, Riccardo ; Onuchic, José N. ; Hwa, Terence ; Weigt, Martin: Direct-coupling analysis of residue coevolution captures native contacts across many protein families. In: *Proceedings of the National Academy of Sciences* 108 (2011), Nr. 49, S. E1293–E1301. – ISSN 0027–8424

[67] Morcos, Faruck ; Schafer, Nicholas P. ; Cheng, Ryan R. ; Onuchic, José N. ; Wolynes, Peter G.: Coevolutionary information, protein folding landscapes, and the thermodynamics of natural selection. In: *Proceedings of the National Academy of Sciences* 111 (2014), Nr. 34, S. 12408–12413. – ISSN 0027–8424

[68] Morris, Kevin V. ; Mattick, John S.: The rise of regulatory RNA. In: *Nature Reviews Genetics* 15 (2014), Jun, Nr. 6, S. 423–437. – ISSN 1471–0064

[69] Mulligan, Mark J. ; Lyke, Kirsten E. ; Kitchin, Nicholas ; Absalon, Judith ; Gurtman, Alejandra ; Lockhart, Stephen ; Neuzil, Kathleen ; Raabe, Vanessa ; Bailey, Ruth ; Swanson, Kena A. ; Li, Ping ; Koury, Kenneth ; Kalina, Warren ; Cooper, David ; Fontes-Garfias, Camila ; Shi, Pei-Yong ; Türeci, Özlem ; Tompkins, Kristin R. ; Walsh, Edward E. ; Frenck, Robert ; Falsey, Ann R. ; Dormitzer, Philip R. ; Gruber, William C. ; Şahin, Uğur ; Jansen, Kathrin U.: Phase I/II study of COVID-19 RNA vaccine BNT162b1 in adults. In: *Nature* 586 (2020), Oct, Nr. 7830, S. 589–593. – ISSN 1476–4687

[70] Mustoe, Anthony M. ; Busan, Steven ; Rice, Greggory M. ; Hajdin, Christine E. ; Peterson, Brant K. ; Ruda, Vera M. ; Kubica, Neil ; Nutiu, Razvan ; Baryza, Jeremy L. ; Weeks, Kevin M.: Pervasive Regulatory Functions of mRNA Structure Revealed by High-Resolution SHAPE Probing. In: *Cell* 173 (2018), Nr. 1, S. 181–195.e18. – ISSN 0092–8674

[71] Nair, Vinod ; Hinton, Geoffrey E.: Rectified Linear Units Improve Restricted Boltzmann Machines. In: *Proceedings of the 27th International Conference on International Conference on Machine Learning*. Madison, WI, USA : Omnipress, 2010 (ICML'10). – ISBN 9781605589077, S. 807–814

[72] Nawrocki, Eric P. ; Eddy, Sean R.: Infernal 1.1: 100-fold faster RNA homology searches. In: *Bioinformatics* 29 (2013), 09, Nr. 22, S. 2933–2935. – ISSN 1367–4803

[73] Necsulea, Anamaria ; Soumillon, Magali ; Warnefors, Maria ; Liechti, Angélica ; Daish, Tasman ; Zeller, Ulrich ; Baker, Julie C. ; Grützner, Frank ; Kaessmann, Henrik: The evolution of lncRNA repertoires and expression patterns in tetrapods. In: *Nature* 505 (2014), Jan, Nr. 7485, S. 635–640. – ISSN 1476–4687

[74] Nocedal, Jorge: Updating Quasi-Newton Matrices with Limited Storage. In: *Mathematics of Computation* 35 (1980), Nr. 151, S. 773–782. – ISSN 00255718, 10886842

[75] Patterson, Josh ; Gibson, Adam: *Deep Learning: A Practitioner's Approach*. Kindle Edition. O'Reilly Media, 7 2017

[76] Peng, Martin ; Maier, Manfred ; Esch, Jan ; Schug, Alexander ; Rabe, Kersten S.: Direct coupling analysis improves the identification of beneficial amino acid mutations for the functional thermostabilization of a delicate decarboxylase. In: *Biological Chemistry* 400 (2019), Nr. 11, S. 1519–1527

[77] Pucci, Fabrizio ; Schug, Alexander: Shedding light on the dark matter of the biomolecular structural universe: Progress in RNA 3D structure prediction. In: *Methods* 162-163 (2019), S. 68–73. – Experimental and Computational Techniques for Studying Structural Dynamics and Function of RNA. – ISSN 1046–2023

[78] Reinartz, Ines ; Sinner, Claude ; Nettels, Daniel ; Stucki-Buchli, Brigitte ; Stockmar, Florian ; Panek, Pawel T. ; Jacob, Christoph R. ; Nienhaus, Gerd U. ; Schuler, Benjamin ; Schug, Alexander: Simulation of FRET dyes allows quantitative comparison against experimental data. In: *The Journal of Chemical Physics* 148 (2018), Nr. 12, S. 123321

[79] Ren, Aiming ; Rajashankar, Kanagalaghatta R. ; Patel, Dinshaw J.: Fluoride ion encapsulation by Mg2+ ions and phosphates in a fluoride riboswitch. In: *Nature* 486 (2012), Jun, Nr. 7401, S. 85–89. – ISSN 1476–4687

[80] Ren, Aiming ; Wang, Xin C. ; Kellenberger, Colleen A. ; Rajashankar, Kanagalaghatta R. ; Jones, Roger A. ; Hammond, Ming C. ; Patel, Dinshaw J.: Structural Basis for Molecular Discrimination by a 3′,3′-cGAMP Sensing Riboswitch. In: *Cell Reports* 11 (2015), April, Nr. 1, S. 1–12. – ISSN 22111247

[81] Rinn, John L. ; Chang, Howard Y.: Genome Regulation by Long Noncoding RNAs. In: *Annual Review of Biochemistry* 81 (2012), Nr. 1, S. 145–166. – PMID: 22663078

[82] Rivas, Elena ; Clements, Jody ; Eddy, Sean R.: A statistical test for conserved RNA structure shows lack of evidence for structure in lncRNAs. In: *Nature Methods* 14 (2017), Jan, Nr. 1, S. 45–48. – ISSN 1548–7105

[83] Rumelhart, David E. ; Hinton, Geoffrey E. ; Williams, Ronald J.: Learning Internal Representations by Error Propagation / CALIFORNIA UNIV SAN DIEGO LA JOLLA INST FOR COGNITIVE SCIENCE. 1985. – Forschungsbericht

[84] Rumelhart, David E. ; Hinton, Geoffrey E. ; Williams, Ronald J.: Learning representations by back-propagating errors. In: *Nature* 323 (1986), Oct, Nr. 6088, S. 533–536. – ISSN 1476–4687

[85] dos Santos, Ricardo N. ; Morcos, Faruck ; Jana, Biman ; Andricopulo, Adriano D. ; Onuchic, José N.: Dimeric interactions and complex formation using direct coevolutionary couplings. In: *Scientific Reports* 5 (2015), Sep, Nr. 1, S. 13652. – ISSN 2045–2322

[86] Schrödinger, LLC. *The PyMOL Molecular Graphics System, Version 1.8.* November 2015

[87] Schroeder, Susan J.: Challenges and approaches to predicting RNA with multiple functional structures. In: *RNA (New York, N.Y.)* 24 (2018), Dec, Nr. 12, S. 1615–1624. – 30143552[pmid]. – ISSN 1469–9001

[88] Schug, Alexander ; Weigt, Martin ; Onuchic, José N. ; Hwa, Terence ; Szurmant, Hendrik: High-resolution protein complexes from integrating genomic information with molecular simulation. In: *Proceedings of the National Academy of Sciences* 106 (2009), Nr. 52, S. 22124–22129. – ISSN 0027–8424

[89] Seemayer, Stefan ; Gruber, Markus ; Söding, Johannes: CCMpred—fast and precise prediction of protein residue–residue contacts from correlated mutations. In: *Bioinformatics* 30 (2014), 07, Nr. 21, S. 3128–3130. – ISSN 1367–4803

[90] Sermanet, Pierre ; Eigen, David ; Zhang, Xiang ; Mathieu, Michael ; Fergus, Rob ; LeCun, Yann. *OverFeat: Integrated Recognition, Localization and Detection using Convolutional Networks.* 2014

[91] Sharp, Phillip A.: The Centrality of RNA. In: *Cell* 136 (2009), Nr. 4, S. 577–580. – ISSN 0092–8674

[92] Soderberg, Tim: *Introduction to nucleic acid (DNA and RNA) structure.* 3 2021. – [Online; accessed 2021-03-21]

[93] Strobel, Eric J. ; Watters, Kyle E. ; Loughrey, David ; Lucks, Julius B.: RNA systems biology: uniting functional discoveries and structural tools to understand global roles of RNAs. In: *Current Opinion in Biotechnology* 39 (2016), S. 182–191. – Systems biology • Nanobiotechnology. – ISSN 0958–1669

[94] Sułkowska, Joanna I. ; Morcos, Faruck ; Weigt, Martin ; Hwa, Terence ; Onuchic, José N.: Genomics-aided structure prediction. In: *Proceedings of the National Academy of Sciences* 109 (2012), Nr. 26, S. 10340–10345. – ISSN 0027–8424

[95] Thompson, Julie D. ; Higgins, Desmond G. ; Gibson, Toby J.: CLUSTAL W: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice. In: *Nucleic Acids Research* 22 (1994), 11, Nr. 22, S. 4673–4680. – ISSN 0305–1048

[96] Thore, Stéphane ; Frick, Christian ; Ban, Nenad: Structural Basis of Thiamine Pyrophosphate Analogues Binding to the Eukaryotic Riboswitch. In: *Journal of the American Chemical Society* 130 (2008), Jul, Nr. 26, S. 8116–8117. – ISSN 0002–7863

[97] Toth-Petroczy, Agnes ; Palmedo, Perry ; Ingraham, John ; Hopf, Thomas A. ; Berger, Bonnie ; Sander, Chris ; Marks, Debora S.: Structured States of Disordered Proteins from Genomic Sequences. In: *Cell* 167 (2016), Nr. 1, S. 158–170.e12. – ISSN 0092–8674

[98] Tuschl, T ; Gohlke, C ; Jovin, TM ; Westhof, E ; Eckstein, F: A three-dimensional model for the hammerhead ribozyme based on fluorescence measurements. In: *Science* 266 (1994), Nr. 5186, S. 785–789. – ISSN 0036–8075

[99] Uguzzoni, Guido ; John Lovis, Shalini ; Oteri, Francesco ; Schug, Alexander ; Szurmant, Hendrik ; Weigt, Martin: Large-scale identification of coevolution signals across homo-oligomeric protein interfaces by direct coupling analysis. In: *Proceedings of the National Academy of Sciences* 114 (2017), Nr. 13, S. E2662–E2671. – ISSN 0027–8424

[100] Virtanen, Pauli ; Gommers, Ralf ; Oliphant, Travis E. ; Haberland, Matt ; Reddy, Tyler ; Cournapeau, David ; Burovski, Evgeni ; Peterson, Pearu ; Weckesser, Warren ; Bright, Jonathan ; van der Walt, Stéfan J. ; Brett, Matthew ; Wilson, Joshua ; Millman, K. J. ; Mayorov, Nikolay ; Nelson, Andrew R. J. ; Jones, Eric ; Kern, Robert ; Larson, Eric ; Carey, C J. ; Polat, İlhan ; Feng, Yu ; Moore, Eric W. ; VanderPlas, Jake ; Laxalde, Denis ; Perktold, Josef ; Cimrman, Robert ; Henriksen, Ian ; Quintero, E. A. ; Harris, Charles R. ; Archibald, Anne M. ; Ribeiro, Antônio H. ; Pedregosa, Fabian ; van Mulbregt, Paul ; SciPy 1.0 Contributors: SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. In: *Nature Methods* 17 (2020), S. 261–272

[101] Voronin, Arthur ; Weiel, Marie ; Schug, Alexander: Including residual contact information into replica-exchange MD simulations significantly enriches native-like conformations. In: *PLOS ONE* 15 (2020), 11, Nr. 11, S. 1–24

[102] Weiel, Marie ; Reinartz, Ines ; Schug, Alexander: Rapid interpretation of small-angle X-ray scattering data. In: *PLOS Computational Biology* 15 (2019), 03, Nr. 3, S. 1–27

[103] Weigt, Martin ; White, Robert A. ; Szurmant, Hendrik ; Hoch, James A. ; Hwa, Terence: Identification of direct residue contacts in protein–protein interaction by message passing. In: *Proceedings of the National Academy of Sciences* 106 (2009), Nr. 1, S. 67–72. – ISSN 0027–8424

[104] Weinreb, Caleb ; Riesselman, Adam J. ; Ingraham, John B. ; Gross, Torsten ; Sander, Chris ; Marks, Debora S.: 3D RNA and Functional Interactions from Evolutionary Couplings. In: *Cell* 165 (2016), Nr. 4, S. 963 – 975. – ISSN 0092–8674

[105] Zeiler, Matthew D. ; Fergus, Rob. *Visualizing and Understanding Convolutional Networks*. 2013

[106] Zerihun, Mehari B. ; Pucci, Fabrizio ; Peter, Emanuel K. ; Schug, Alexander: pydca v1.0: a comprehensive software for direct coupling analysis of RNA and protein sequences. In: *Bioinformatics* (2019), 11. – btz892. – ISSN 1367–4803

[107] Zhang, Jinwei ; Ferré-D'Amaré, Adrian R.: Dramatic Improvement of Crystals of Large RNAs by Cation Replacement and Dehydration. In: *Structure* 22 (2014), September, Nr. 9, S. 1363–1371. – ISSN 09692126

[108] Zhang, Xiang ; Zhao, Junbo ; LeCun, Yann. *Character-level Convolutional Networks for Text Classification*. 2016

The curriculum vitae is not included in the online version for data protection reasons.

Declarations required for the submission of the thesis

Declaration:

In accordance with § 6 (para. 2, clause g) of the Regulations Governing the Doctoral Proceedings of the Faculty of Biology for awarding the doctoral degree Dr. rer. nat., I hereby declare that I represent the field to which the topic "*RNA Structure Prediction Guided by Co-evolutionary Information—Method Development and Applications"* is assigned in research and teaching and that I support the application of Mehari Bayou Zerihun.

Essen, date 05.04.2021 Alexander Schug _____

Name of the scientific supervisor/member of the University of Duisburg-Essen

Signature of the supervisor/ member of the University of Duisburg-Essen

Declaration:

In accordance with § 7 (para. 2, clause d and f) of the Regulations Governing the Doctoral Proceedings of the Faculty of Biology for awarding the doctoral degree Dr. rer. nat., I hereby declare that I have written the herewith submitted dissertation independently using only the materials listed, and have cited all sources taken over verbatim or in content as such.

Essen, date 05.04.2021 _____

Signature of the doctoral candidate

Declaration:

In accordance with § 7 (para. 2, clause e and g) of the Regulations Governing the Doctoral Proceedings of the Faculty of Biology for awarding the doctoral degree Dr. rer. nat., I hereby declare that I have undertaken no previous attempts to attain a doctoral degree, that the current work has not been rejected by any other faculty, and that I am submitting the dissertation only in this procedure.

Essen, date 05.04.2021 _____

Signature of the doctoral candidate