ROBUST COLLABORATIVE LEARNING FOR SEQUENCE MODELLING

Francois Buet-Golfouse

University College London
ucahfbu@ucl.uk

Hans Roggeman, Islam Utyagulov

Independent Researchers
London, United Kingdom

ABSTRACT

Current deep learning techniques for RNA classification suffer from over-fitting and lack of reproducibility. We show that by introducing robustness by design in both CNN and RNN algorithms, we are able to achieve standalone state-of-the-art accuracy. By constructing model-agnostic robustness checks and reusing features obtained from both architectures, we build a collaborative framework that improves performance and stability.

***Index Terms*—** Collaborative Learning, Neural Networks, RNA, Genomics, Genomic Signal Processing

1. INTRODUCTION

Sequence modelling is ubiquitous in signal and speech processing [1], while deep learning techniques such as convolutional neural networks ("CNNs") and recurrent neural networks ("RNNs") have become paramount [2]. Throughout this work, our focus lies on genomic signal processing [3, 4] and, in particular, on short non-coding RNA ("ncRNA") sequence classification [5]. However, our approach and the heuristic principles that we derive are broadly applicable.

This paper highlights new types of architectures to ensure the robustness and the generalisation of our findings (while outperforming existing models) by pursuing two avenues. First, in the context of standalone CNNs and RNNs, we introduce robustness by increasing very significantly the amount of noise and dropout injected in the various layers. Second, we check that CNN and RNN architectures learn different feature representations and that leveraging both via *collaborative learning* thus helps improve performance and stability. To do so, we define a *joint learning coefficient* that measures the correlation between the *dynamics* of learning on the training and validation sets.

ncRNA. Within a eukaryotic cell, 80% of the RNA mass is ncRNA (non-coding RNA) as opposed to mRNA (messenger) and pre-mRNA. ncRNA [5] can be classified by functional group: 1) ncRNA involved in the translation of mRNA, 2) Regulatory ncRNA, 3) Intron ncRNA, and, 4) Cis-regulatory RNA. The involvement of ncRNA in all aspects of cell regulation means advancements will impact cancer, immunology and developmental research [6]. New ncRNA is rapidly being discovered, this makes functional identification of ncRNA a problem of practical interest. Here we consider reference data sets of ncRNA sequences and improve on the current state-of-the-art for out-of-sample prediction of ncRNA classes.

Data. Most empirical results are derived on the "**Test13**" data set, which appears in [7] and contains RNA sequences with their labels corresponding to one of the 13 classes.[1] "**Test88**" [5] contains 150,000 sequences from the Rfam database [8], with 88 different classes.

2. COMPONENT MODELS

Our standalone models[2] predict ncRNA classes using the sequence ("seq") and/or secondary structure[3] data ("sec"). Though some of these models improve on the current state-of-the-art (3), we use them as *components* to contribute to a new prediction[4].

2.1. Convolutional Neural Networks

All CNN component models [2] share the same architecture (Figure 1): 5 repeating layers of 1-dimensional convolu-

[1] There are 13 different classes in **Test13** and it contains around 9000 observations, which are split 80:10:10 across train, validation and test sets, after removing duplicates and randomly shuffling observations. **Test13** data can be found at `https://github.com/IcarPA-TBlab/nrc/tree/master/data/ECCB2017`. The original data can be found on the Rfam database [8].

[2] Prior to fitting these models, RNA sequences were encoded into 1000-long arrays using:

- A $\rightarrow [1,0,0,0,0,0,1,0]$
- T/U$\rightarrow [0,0,1,0,1,0,0,0]$
- G $\rightarrow [0,1,0,0,0,0,0,1]$
- C $\rightarrow [0,0,0,1,0,1,0,0]$
- padding/other symbol $\rightarrow [0,0,0,0,0,0,0,0]$.

[5] and [9] show robustness of one-hot encoding as opposed to advanced methods (e.g., k-mers). If the sequence was less than 1000 symbols, zero padding was applied; if the sequence had more than 1000, the sequence tail was dropped.

[3] Secondary structure in RNA refers to the base-pairing abilities of the nucleic acid sequence with other nucleic acids on other nucleic strands or with itself.

[4] All neural networks are coded in TensorFlow and minimise categorical cross-entropy loss using the default Adam settings. The number of epochs is 500 for CNNs and 220 for RNNs.

tion, batch normalization, max pooling, Gaussian noise and dropout followed by a flattening layer that feeds into a two-layer fully connected neural net with a final softmax output layer. CNN architectures for genomics are well-established [7, 10, 5, 11]. We differ from current methodologies in the depth of the chosen architecture (most CNNs for RNA prediction have up to three layers only) and the added robustness of the overall network.

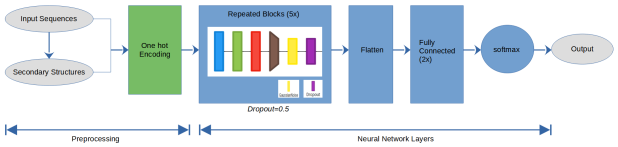

Fig. 1: Model structure for all CNN component models with extensive use of Gaussian noise and Dropout

2.2. Recurrent Neural Networks

All RNN component models share the same architecture (Figure 2) with Gated Recurrent Units layers that are bi-directional [1] (given the absence of a sequence order). These precede an attention mechanism (see [12] and [9])[5], followed by a flattening layer that feeds into a three-layer fully connected neural net with a final softmax output layer.

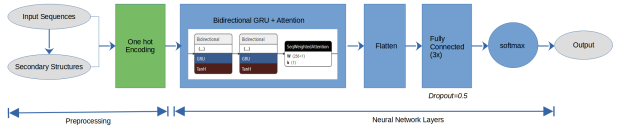

Fig. 2: Model structure of the RNN component model

This architecture is close to that proposed by [12] in the context of precursor microRNAs, except that LSTMs are replaced by BiGRUs (GRUs perform better on small data sets [13]) and regularisation is drastically increased.

3. INTRODUCING ROBUSTNESS

3.1. Importance of robustness in RNA predictions

Robustness focuses on the fact that outputs of a machine learning algorithm should not change much in the presence of (small) corrupting noises in the inputs [14, 15]. Given the importance of RNA applications and the limited understanding of the underpinning biological processes [6], robustness is paramount, but overall unaddressed directly in the ncRNA literature. Indeed, the *categorical* nature of an RNA sequence makes corrupting these directly difficult. *Continuity* does not exist for RNA sequences in that one-letter mutations can have important biological consequences, unlike adversarial learning on images [16] where one can modify individual pixels without creating perceptible changes to the human eye. Thus, instead of perturbing discrete inputs, which lacks a proper biological interpretation and can lead to invalid RNA sequences, we decide on a different route: markedly increasing noise and dropout [17] in the deep neural network's layers thanks to a new heuristic criterion.

3.2. Model-agnostic robustness checks

Our perspective is linked to a practical view of generalisation; learning on the *validation* set should be similar to learning on the *train* set. Learning (here, the progress in decreasing the loss and increasing the accuracy) should move in parallel on both train and validation sets (after a burn-in period).

Fig. 3: Accuracy on validation (y-axis) vs accuracy on train set (x-axis) per epoch, for **(left)** no Gaussian noise and no dropout, **(middle)** Gaussian noise with 1 standard deviation and 50% dropout rate, **(right)** Gaussian noise with 1.8 standard deviation and 90% dropout rate. As regularisation increases, the slope approaches 1.

While training CNN architectures, we have observed a "spiking behaviour" (see Figure 3) in the validation accuracy. Even usual levels of regularisation (via noise and dropout) lead to a sudden increase in performance on the validation set, prior to which the training performance had already reached an extremely high value. There is thus no reason to believe that a sudden increase in the *validation* accuracy would translate into a similar increase in the *test* accuracy, as we may simply be overfitting the validation set too.

Heuristic 1. *A static validation metric, say at the end of training, may be misleading and the dynamics of model fitting should be considered too. One may wish to slow training down to ensure a better correspondence between the dynamics of model fitting on the training and validation sets.*

It is well-known that regularisation and robustness can reduce over-fitting [2], but considering the slope of the relationship between train and validation accuracies is fruitful. Over the T epochs, consider the empirical correlation between the accuracy levels on the train and validation sets (represents a form of joint learning[6]):

$$\textit{Joint learning coefficient}_{t=1,\cdots,T} := \text{corr}_{t=1,\cdots,T}\left(Accuracy_t^{Validation}, Accuracy_t^{Training}\right) \quad (1)$$

[5]Our attention mechanism replaces the final output vector of a standard GRU with a weighted combination of the output vectors at each point in the sequence. This fosters learning long-term dependencies and makes results more interpretable, as the contribution of each period's output is explicitly related to the final output.

[6]Note that a similar analysis is possible on the *changes* in accuracy, but is more delicate given the amount of noise in each epoch due to potentially small batch sizes, etc.

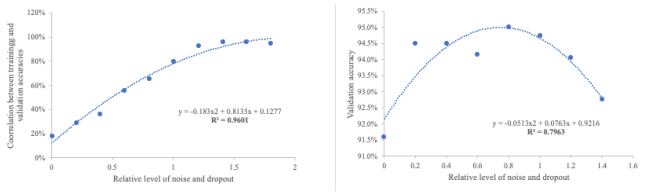

Fig. 4: On Model **A** in Table 3: **(a)** Correlation of training and validation accuracy as function of relative level of noise and dropout **(b)** Validation accuracy as function of relative level of noise and dropout. Standard deviation is λ, whereas the dropout rate is $\lambda \times 50\%$, with λ the relative level.

As seen in Figure 4 **(a)**, when the overall regularisation level increases, so does joint learning. On the other hand, cf. Figure 4**(b)**, there is a well-known trade-off between regularisation and accuracy (which is intuitive, as too much regularisation prevents learning). We argue that one should consider *both* the maximal validation accuracy *and* the point at which the joint learning coefficient stabilises. While the maximal validation accuracy is obtained with a standard deviation of 0.8 and a dropout rate of 40 %, the joint learning coefficient stops increasing once the noise's standard deviation reaches 1.2 and the dropout rate 60 %. We thus choose a dropout rate of 50% and a standard deviation of 1 in Gaussian noise layers, both values being significantly higher than standard recommendations [12, 9].

3.3. Results and interpretation

With this choice of regularisation, we present our results on standalone models in Table 3, with two separate types of architecture (CNN and RNN). For each we trained models based on nucleotide sequence only, on the secondary structure only ("sec") and on a joint encoding of sequence and secondary structure ("seq+sec").

Moniker	Method	Specificity[7]	Precision	Recall	F1-score	Accuracy
A	CNN (256) [8]	**0.997**	**0.963**	**0.961**	**0.962**	**0.962**
B	CNN (128)	0.996	0.960	0.958	0.958	0.958
C	CNN (256) seq+sec[9]	0.996	0.948	0.946	0.946	0.946
D	CNN (256) sec[10]	0.977	0.736	0.730	0.729	0.730
E	RNN	0.996	0.952	0.955	0.954	0.955
F	RNN seq+sec	0.993	0.922	0.917	0.917	0.917
G	RNN sec	0.968	0.652	0.617	0.610	0.617

Table 1: Out-of-sample results on Test13 for various standalone models.

From Table 3, as long as the model is trained on sequence data rather than secondary data only, standalone models perform well (see Section 6. Models **A**, **B** and **E** achieve state-of-the-art accuracy).

4. ARE X-NNS LEARNING THE SAME FEATURES?

We would like to understand if the CNN and RNN architectures learn different features. This of importance to determine whether combined architectures should be explored.

Heuristic 2. *Different types of architecture are likely to learn different feature representations.*

Therefore, we have extracted the features from the last dense layer of the models **A** and **E** and carried out canonical correlation analysis[11] (CCA) [18], and principal component analysis (PCA) on these features. CCA maximises the correlation between both datasets (but may come up with a direction that is irrelevant to the predictive task), whereas PCA maximises each dataset's standalone variance (but may fail to recognise that both datasets are simply orthogonalised differently); both yield complementary insights.

First, given the large number of features and the fact that both architectures yield excellent results, we expect a very high CCA between both feature sets. Indeed, we obtain 95% on the validation set. However, CCA is sensitive to estimation noise, as we show in Figure 5. We perform a Monte Carlo simulation of CCA$(\boldsymbol{X}, \boldsymbol{Y})$, with $\boldsymbol{Y} = \rho\boldsymbol{X} + \sqrt{1-\rho^2}\boldsymbol{Z}$, where $\boldsymbol{X}$ and $\boldsymbol{Z}$ are matrices (with the same dimensions as in our case) of independent standard Gaussian variables and ρ is a "uniform" correlation coefficient varying between 0 and 1. With a 95% empirical CCA coefficient, the corresponding representative correlation ρ is around 85%, which is significantly lower. Second, the correlation between each feature set's first principal component reaches 64%, suggesting that the main directions are different in both feature sets.

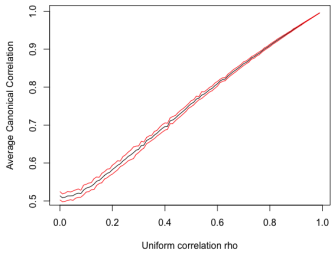

Fig. 5: Average canonical correlation (*black*) plus or minus one standard deviation (*red*), as a function of the uniform correlation ρ, obtained via Monte Carlo simulations with $N = 1,000$ trial runs.

5. COLLABORATIVE LEARNING

The literature on RNA prediction has mostly focused on "standalone" models, i.e., architectures influenced by (mostly) one paradigm, for instance, RNN or CNN. We expect these to be qualitatively different and thus for a combination of both to bring superior results. This is the subject of collaborative learning [19], which presents more generic mechanisms than just ensembling by sharing features learnt in parallel (and not just outcomes). In our case, we combine features created

[11] A canonical correlation analysis performed on two data sets X and Y is seeking to find linear combinations of features of X and Y which have maximum correlation with each other.

by training CNN and RNN models separately. We extract features from either the output layer or the last dense layer in each and build a neural network with those as inputs.

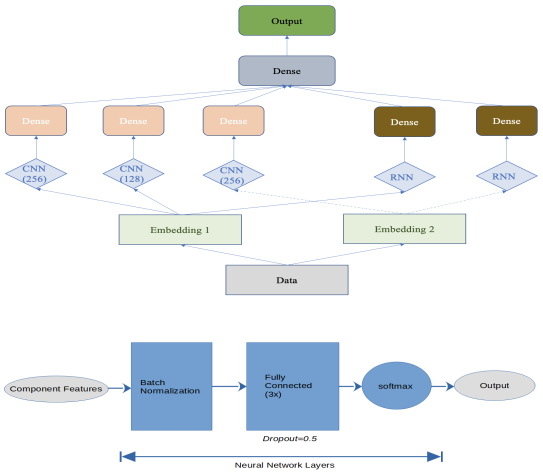

Fig. 6: **(a)** Illustration of the principle of collaborative learning with multiple heads, involving different types of embeddings and/or architectures, before the collaboration phase leading to the output. Based on Figure 1 in [19]. **(b)** Composition of the final dense block in Figure 6 **(a)**.

Heuristic 3. *Combining different architectures should increase both in- and out-of-sample performance. More diverse architectures should lead to a higher gain.*

From results in Table 2, performance always increases when compared to standalone results in Table 3. This is almost guaranteed in training, but could lead to overfitting on the test set. Further, collaborative models with "deeper" features have a slight edge over those trained with outputs only. Last, including features from both CNN and RNN architectures leads to a better performance than collaborative models based on a single architecture.

Combined Layers	Specificity	Precision	Recall	F1-score	Accuracy
O(A)+O(B)+O(D)	0.997	0.967	0.967	0.967	0.967
O(A)+O(E)	0.997	0.966	0.965	0.965	0.965
O(A)+O(B)+O(E)	0.997	0.966	0.965	0.965	0.965
LD(A)+LD(B)+LD(D)	0.997	0.965	0.963	0.963	0.963
LD(A)+LD(B)+LD(E)	**0.998**	**0.975**	**0.974**	**0.974**	**0.974**
LD(A)+LD(E)	0.998	0.973	0.972	0.972	0.972
LD(A)+LD(E)+O(A)+O(E)	0.998	0.973	0.972	0.972	0.972
LD(A)+LD(D)+LD(E)+O(A)+O(D)+O(E)	0.998	0.972	0.972	0.972	0.972

Table 2: Out-of-sample results on **Test13** data set for collaborative models using components from Table 3.[12]

6. ADDITIONAL RESULTS

We present results on **Test13** and **Test88** and compare them against current state-of-the-art ncRNA predictive models. Our proposed approach shows a performance improvement across the board. The exact same architecture (with the exception of the output layer whose dimension increased from 13 to 88) was trained on **Test88**'s train set and evaluated on its test set. We choose LD(A)+LD(E) as the benchmark [13].

Performance results of benchmark models were taken from the respective publications, however our own attempt in replicating some of these results did not show significant discrepancies with the published ones. In addition, it is not always possible to reproduce original embeddings or features, we avoid using our own to avoid skew in the results.

Test13. Our standalone models reach state-of-the-art accuracy on **Test13**, while our collaborative model exceeds it.

Authors	Reference	Method	F1-score	Accuracy
Navarin2017	[20]	Graph kernel	0.650	0.670
Fiannaca2017	[7]	CNN	0.741	0.747
Rossi2019	[21]	Graph CNN	0.863	0.857
Chantsalnyam2020	[10]	CNN	0.876	0.880
Noviello2020	[5]	CNN	0.960	0.960
Asim2021	[11]	CNN	0.954	0.954
Wang2021	[9]	RNN	0.788	0.797
LD(A)+LD(E)	Ours	CNN+RNN	**0.972**	**0.972**

Table 3: Performance on Test13 across existing models in the literature.

Test88. Performance on **Test88** is a test of *generalisation*: the same architecture is used, but the problem's dimension has increased (now including 88 classes).

Moniker	Method	Specificity	Precision	Recall	F1-score	Accuracy
Noviello2020 [5]	CNN	-	-	-	-	0.990
A	CNN (256)	0.999	0.997	0.997	0.997	0.997
E	RNN	0.999	0.918	0.918	0.914	0.918
LD(A)+LD(E)	CNN+RNN	**0.999**	**0.998**	**0.997**	**0.997**	**0.998**

Table 4: Out-of-sample test results for various models on the Test88 data.

7. CONCLUSION AND FUTURE DIRECTIONS

This paper has proposed a robust collaborative learning architecture for sequence modelling, which has obtained high accuracy in some ncRNA classification tasks. The issue of robustness was both important and challenging due to the particular nature of these sequences (in short, there is no straightforward notion of continuity). In doing so, we have shown how looking at the dynamics of training and validation loss functions leads to model-agnostic notions of robustness.

Collaborative learning, beyond usual ensembling techniques, seems to be an under-explored avenue in genomic signal processing. The problem's complexity has spurred a variety of modelling techniques involving CNNs, graph CNNs, RNNs, or even hand-crafted features. Different architectures can yield varied insights by tackling specific aspects of RNA sequences, so that combining them enriches representations.

[12] *O* refers to the features of the softmax output layer, whereas *LD* refers to the features of the last dense layer.

[13] Despite LD(A)+LD(B)+LD(E) (Table 2) having best performance. A & B (Table 3) are both CNN models. We demonstrate the impact of collaborative learning with robustness rather than the incremental value of an additional model.

8. REFERENCES

[1] Alex Graves, Santiago Fernández, and Jürgen Schmidhuber, "Bidirectional lstm networks for improved phoneme classification and recognition," in *Proceedings of the 15th International Conference on Artificial Neural Networks: Formal Models and Their Applications - Volume Part II*, Berlin, Heidelberg, 2005, ICANN'05, p. 799–804, Springer-Verlag.

[2] Ian Goodfellow, Yoshua Bengio, and Aaron Courville, *Deep Learning*, The MIT Press, 2016.

[3] Dimitris Anastassiou, "Genomic signal processing," *IEEE Signal Processing Magazine*, vol. 18, no. 4, pp. 8–20, 2001.

[4] Ilya Shmulevich and Edward R. Dougherty, *Genomic Signal Processing*, Princeton University Press, 2007.

[5] Teresa Maria Rosaria Noviello, Francesco Ceccarelli, Michele Ceccarelli, and Luigi Cerulo, "Deep learning predicts short non-coding rna functions from only raw sequence data," *PLOS Computational Biology*, vol. 16, no. 11, pp. 1–17, 11 2020.

[6] Peijing Zhang, Wenyi Wu, Qi Chen, and Ming Chen, "Non-coding rnas and their integrated networks," *Journal of Integrative Bioinformatics*, vol. 16, no. 3, pp. 20190027, 2019.

[7] Antonino Fiannaca, Massimo La Rosa, Laura La Paglia, Riccardo Rizzo, and Alfonso Urso, "Nrc: Non-coding rna classifier based on structural features," *BioData Mining*, vol. 10, 08 2017.

[8] Ioanna Kalvari, Eric P Nawrocki, Nancy Ontiveros-Palacios, Joanna Argasinska, Kevin Lamkiewicz, Manja Marz, Sam Griffiths-Jones, Claire Toffano-Nioche, Daniel Gautheret, Zasha Weinberg, Elena Rivas, Sean R Eddy, Robert D Finn, Alex Bateman, and Anton I Petrov, "Rfam 14: expanded coverage of metagenomic, viral and microRNA families," *Nucleic Acids Research*, vol. 49, no. D1, pp. D192–D200, 11 2020.

[9] Linyu Wang, Shaoge Zheng, Hao Zhang, Zhiyang Qiu, Xiaodan Zhong, Haiming Liu, and Yuanning Liu, "ncrfp: A novel end-to-end method for non-coding rnas family prediction based on deep learning," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 18, no. 2, pp. 784–789, 2021.

[10] Tuvshinbayar Chantsalnyam, Dae Yeong Lim, Hilal Tayara, and Kil To Chong, "ncrdeep: Non-coding rna classification with convolutional neural network," *Computational Biology and Chemistry*, vol. 88, pp. 107364, 2020.

[11] Muhammad Nabeel Asim, Muhammad Imran Malik, Christoph Zehe, Johan Trygg, Andreas Dengel, and Sheraz Ahmed, "A robust and precise convnet for small non-coding rna classification (rpc-snrc)," *IEEE Access*, vol. 9, pp. 19379–19390, 2021.

[12] Seunghyun Park, Seonwoo Min, Hyun-Soo Choi, and Sungroh Yoon, "Deep recurrent neural network-based identification of precursor micrornas," in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds. 2017, vol. 30, Curran Associates, Inc.

[13] Nicole Gruber and Alfred Jockisch, "Are gru cells more specific and lstm cells more sensitive in motive classification of text?," *Frontiers in Artificial Intelligence*, vol. 3, pp. 40, 2020.

[14] Jerry Zheng Li, *Principled Approaches to Robust Machine Learning and Beyond*, Ph.D. thesis, The Massachusetts Institute of Technology, 2018.

[15] Peter J. Huber and Elvezio M. Ronchetti, *Robust Statistics*, Wiley, second edition, 2009.

[16] Alexey Kurakin, Ian J. Goodfellow, and Samy Bengio, "Adversarial machine learning at scale," 2017.

[17] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, vol. 15, no. 56, pp. 1929–1958, 2014.

[18] Wolfgang Karl Härdle and Léopold Simar, *Applied Multivariate Statistical Analysis*, Springer Series in Statistics. Springer, Berlin, Heidelberg, Berlin, Germany, fourth edition, 2015.

[19] Guocong Song and Wei Chai, "Collaborative learning for deep neural networks," in *Advances in Neural Information Processing Systems*, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, Eds. 2018, vol. 31, Curran Associates, Inc.

[20] Nicolò Navarin and Fabrizio Costa, "An efficient graph kernel method for non-coding RNA functional prediction," *Bioinformatics*, vol. 33, no. 17, pp. 2642–2650, 05 2017.

[21] Emanuele Rossi, Federico Monti, Michael Bronstein, and Pietro Liò, "ncrna classification with graph convolutional networks," in *Proceedings of the 1st International Workshop on Deep Learning on Graphs: Methods and Applications (DLG@KDD)*, 2019.