





Computational Understanding of Pairwise Interactions in ncRNA Data - Supplementary Information

Marco Nicolini^{1, }, Federico Stacchiotti^{1, }, Elena Casiraghi^{1,2,3,4, }, and Giorgio Valentini^{*,1,2, }

¹ AnacletoLab - Dipartimento Informatica, Università degli Studi di Milano, Milan, Italy.

² ELLIS - European Lab for Learning and Intelligent Systems, Milan Unit.

³ Environmental Genomics and Systems Biology Division, Lawrence Berkeley National Laboratory, Berkeley, CA, United States.

⁴ Department of Computer Science, Aalto University, Espoo, Finland.

*corresponding author email: giorgio.valentini@unimi.it

Keywords: ncRNA-ncRNA interaction, Machine Learning, non-coding RNA, Large Language Models.

S1 Dataset

Our dataset comprises a subset of multispecies ncRNA interaction pairs from RNA-KG [1]¹.

Due to the constraints imposed by the ncRNA Language Model (LM) in our pipeline (GenerRNA [2]), we filtered the dataset to retain only sequences that fit within the model’s token limit (That is approximately 4096 nucleotides), considering that its Byte Pair Encoding (BPE) compression typically achieves a 4× reduction in sequence length². After applying this length filter, the dataset contains:

- 99841 interaction pairs (down from an initial 130310 pairs).
- 10644 unique sequences (selected from 19624 potential sequences) belonging to 9 different RNA molecule types: long non-coding RNA (lncRNA), circular RNA (circRNA), microRNA (miRNA), small nuclear RNA (snRNA), small nucleolar RNA (snoRNA), Small Cajal body-specific RNAs (scaRNAs), small cytoplasmic RNAs (scRNA), not (better) classified non coding RNA molecules (ncRNA) and pseudo RNA³.

In the following, we denote the set of length-filtered molecules as

$$\mathcal{S} = \{s_i\}, \quad i = 1, \dots, |\mathcal{S}|,$$

where the type of each molecule $s \in \mathcal{S}$ is given by $\phi(s)$, i.e. $\phi : \mathcal{S} \rightarrow \mathcal{T}$ represents a mapping of a ncRNA sequence $s \in \mathcal{S}$ to its ncRNA type \mathcal{T} , e.g. miRNA, lncRNA or any other ncRNA type.

¹Retrieval of interacting pairs and corresponding sequences was performed using the scripts available from the RNA-KG web site: <https://github.com/AnacletoLAB/RNA-KG>.

²BPE-based tokenization compresses raw nucleotide sequences, allowing longer sequences to fit within the model’s constraints.

³In RNAinter, the term “pseudo” specifically denotes RNA sequences transcribed from pseudogenes. In this context, these are transcripts derived from genes that have lost their protein-coding capability due to accumulated mutations, yet they are still produced as RNA. Similar to other ncRNAs, such pseudogene RNAs can sometimes participate in regulatory networks by, for example, acting as miRNA decoys or sponges, despite not encoding functional proteins.

The identity of an interaction pair is solely determined by its constituent molecules, regardless of order; that is,

$$(s_i, s_j) = (s_j, s_i).$$

The type of an interaction (s_i, s_j) with $s_i \neq s_j$ and $s_i, s_j \in \hat{\mathcal{S}}$ is determined by the types of the ncRNA s_i and s_j themselves, regardless of their order:

$$(\phi(s_i), \phi(s_j)) = (\phi(s_j), \phi(s_i))$$

For instance, possible types of ncRNA interactions are miRNA-lncRNA or miRNA-miRNA. Assuming that interacting ncRNA pairs of different types exhibit distinct specificities that the model should learn, we reasoned that types with negligible sample sizes might introduce noise rather than valuable information. Therefore, the set of interaction pairs used in this work is obtained by further filtering the dataset of interacting pairs to remove interacting pair types represented by fewer than 100 samples, resulting in 10644 unique sequences composing 99841 interacting pairs. Fig S3 shows the distribution of the different types of ncRNA interactions.

S2 *ncRNA-CUPID* Needs bad matches: negative samples generation

In our dataset, only positive non-coding RNA-RNA interactions are explicitly provided, and they occur with varying frequencies.

To effectively train *ncRNA-CUPID*, we generated negative examples for each interaction pair type by matching the frequency distribution of the positive interactions. Specifically, negative examples were generated under the assumption that any pair of ncRNA sequences drawn from the set of unique sequences that is not observed as a positive interaction constitutes a possible negative instance.

Let

$$\mathcal{S} = \{s_1, s_2, \dots, s_N\}$$

be the set of unique ncRNA sequences present in the dataset. Denote by

$$\mathcal{P} = \{(s_i, s_j) \mid \text{a known positive interaction between } s_i \text{ and } s_j\}$$

the set of all positive ncRNA-ncRNA interactions. Then, the set of all possible ncRNA pairs (excluding self-interactions) is given by $\mathcal{S} \times \mathcal{S}$.

The set of *potential negatives* is defined as:

$$\mathcal{N}_{\text{potential}} = \{(s_i, s_j) \in \mathcal{S} \times \mathcal{S} \mid s_i \neq s_j\} \setminus \mathcal{P}.$$

Negative Sampling Procedure. To generate the negative samples for each interacting pair type, we corrupt its tuples. In other words, given a positive pair (s_i, s_j) with type $(\phi(s_i), \phi(s_j))$, we keep the first molecule s_i fixed and sample $s' \in \mathcal{S}$ such that:

$$s' \neq s_i, \quad \phi(s') = \phi(s_j), \quad (s_i, s') \notin \mathcal{P}$$

according to a probability $\mathbb{P}(s') \propto \text{freq}((\phi(s), \phi(s')))$, where $\text{freq}((\phi(s), \phi(s')))$ is the frequency of positive edges of type $(\phi(s), \phi(s'))$. Each valid pair (s_i, s') is then added to our negative set \mathcal{N} . We further added a parameter n the represents the number of negative edges we generate for each positive edge, in order to control the imbalance between positive and negative edges in the testing phase.

Handling Data Augmentation Special Cases Due to our augmentation strategy, a positive instance (s_i, s_j) implies that the following pairs are also considered as positives:

$$(s_j, s_i), \quad (s_i^F, s_j^F), \quad (s_j^F, s_i^F),$$

where s_i^F and s_j^F denote the flipped sequences of s_i and s_j , respectively. Consequently, during negative sampling, we must ensure that pairs such as:

$$(s_i^F, s_j), \quad (s_i, s_j^F), \quad (s_j, s_i^F), \quad (s_j^F, s_i)$$

are not erroneously included as negatives. Checks are implemented during the sampling process to avoid these cases.

Algorithm Description The negative sampling algorithm is detailed in Algorithm 1. In our implementation, we set $n = 20$. Note that, due to parallelization in our code, the effective number of negatives per positive sequence may be approximately 20, rather than exactly 20.

Algorithm 1 Negative Sampling Algorithm

Require: Set of unique ncRNA sequences \mathcal{S} , positive interaction set \mathcal{P} , and negative sampling parameter n

Ensure: Negative sample set \mathcal{N}

```

1: Initialize  $\mathcal{N} \leftarrow \emptyset$ 
2: for each ncRNA sequence  $s \in \mathcal{S}$  that appears in some  $(s, s') \in \mathcal{P}$  do
3:   for  $i = 1$  to  $n$  do
4:     Sample  $s_{\text{neg}} \in \mathcal{S}$  according to  $\mathbb{P}(s_{\text{neg}}) \propto \text{freq}(\phi(s), \phi(s_{\text{neg}}))$ 
5:     if  $(s, s_{\text{neg}}) \notin \mathcal{P}$  and  $(s, s_{\text{neg}})$  is not a data augmentation special case then
6:        $\mathcal{N} \leftarrow \mathcal{N} \cup \{(s, s_{\text{neg}})\}$ 
7:     end if
8:   end for
9: end for
10: return  $\mathcal{N}$ 

```

S3 FFNN architecture and hyperparameter configuration.

The hyper-parameters and configurations used for training the FFNN are reported below. Training and validation loss curves were monitored over epochs to assess model convergence and to avoid potential overfitting by early stopping.

The FFNN network has the following architecture

- **Input Layer Dimension:** 1024 for AVG and Max-pooling embedding strategies, 2048 when the embedding of the input molecule is obtained by concatenating the embeddings obtained by AVG and Max pooling,
- **Hidden Layers:** 4 hidden layers with 1024 neurons each and ReLU activation function,
- **Output Layer:** 1 neuron with sigmoid activation function.

The following hyper-parameters to train the network:

- **Learning Rate:** $\eta = 5 \times 10^{-4}$ with a linear warm-up phase of 4 epochs, followed by cosine decay.

- **Epochs:** 50 epochs with early stopping (patience of 10 epochs). The model with the best validation loss is selected (e.g., if the lowest validation loss is observed at epoch 35, then early stopping is triggered at epoch 45, and the model from epoch 35 is used).
- **Batch Size:** 512.
- **Dropout Rate:** 0.2.
- **Optimizer:** Adam.
- **Loss Function:** Binary Cross-Entropy.
- **Balanced Batch Sampling:** A custom batch-sampler is employed with a proportion of negatives in each batch of 70%, one epoch is defined as an iteration through all the negatives.

S4 Additional Figures

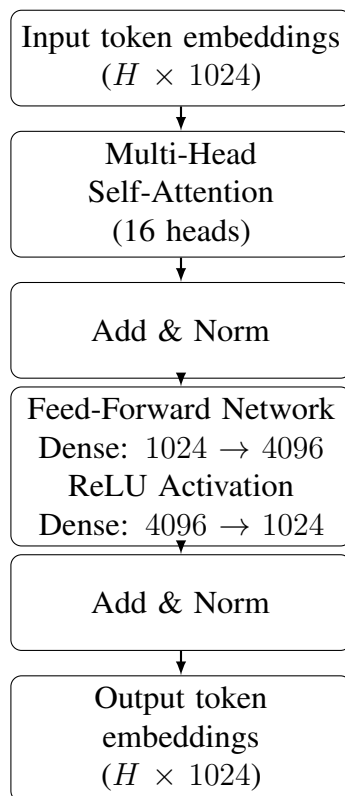


Figure S1: Architecture of the GenerRNA block.

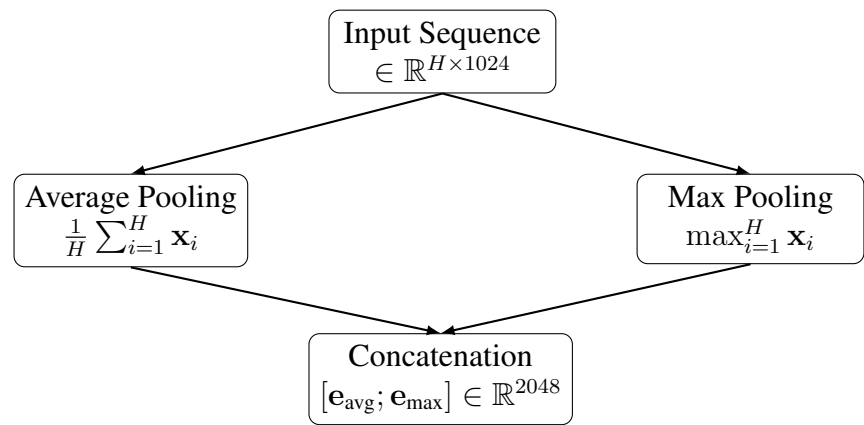


Figure S2: Pooling Embedding Strategy.

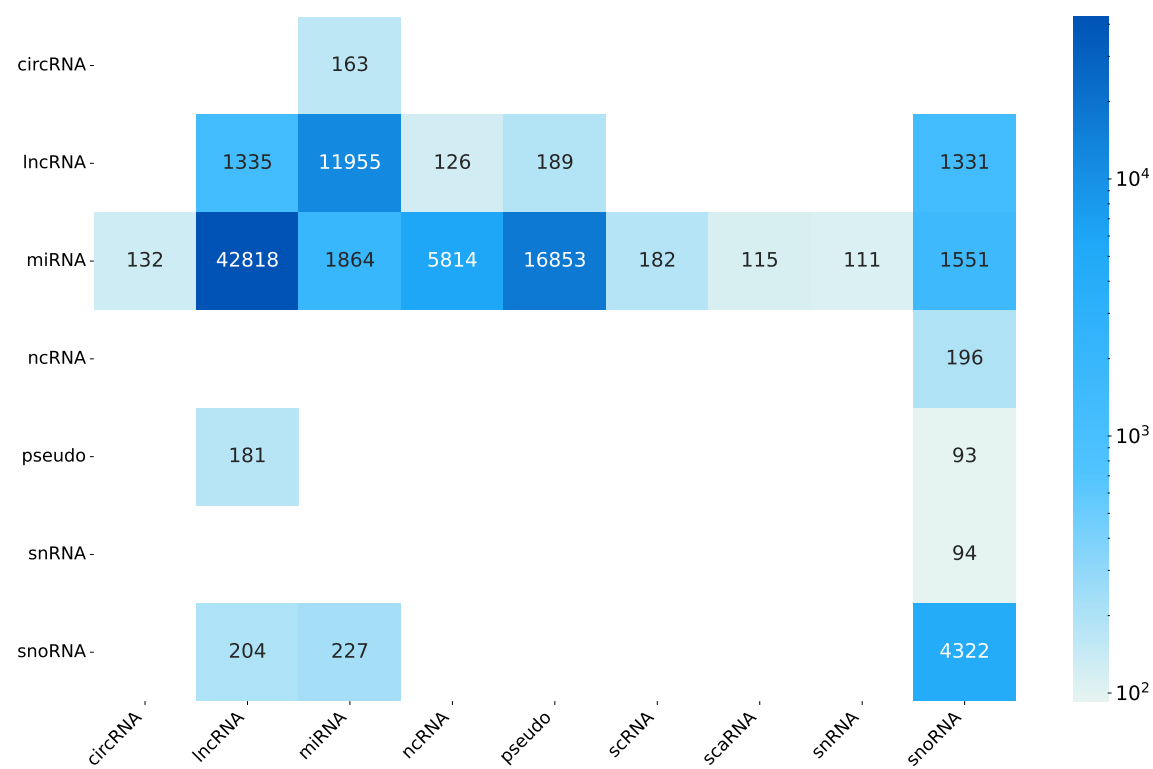


Figure S3: Distribution of ncRNA interactions pairs in the filtered interaction set. Rows: first (left) molecule type; Columns: right molecule type.

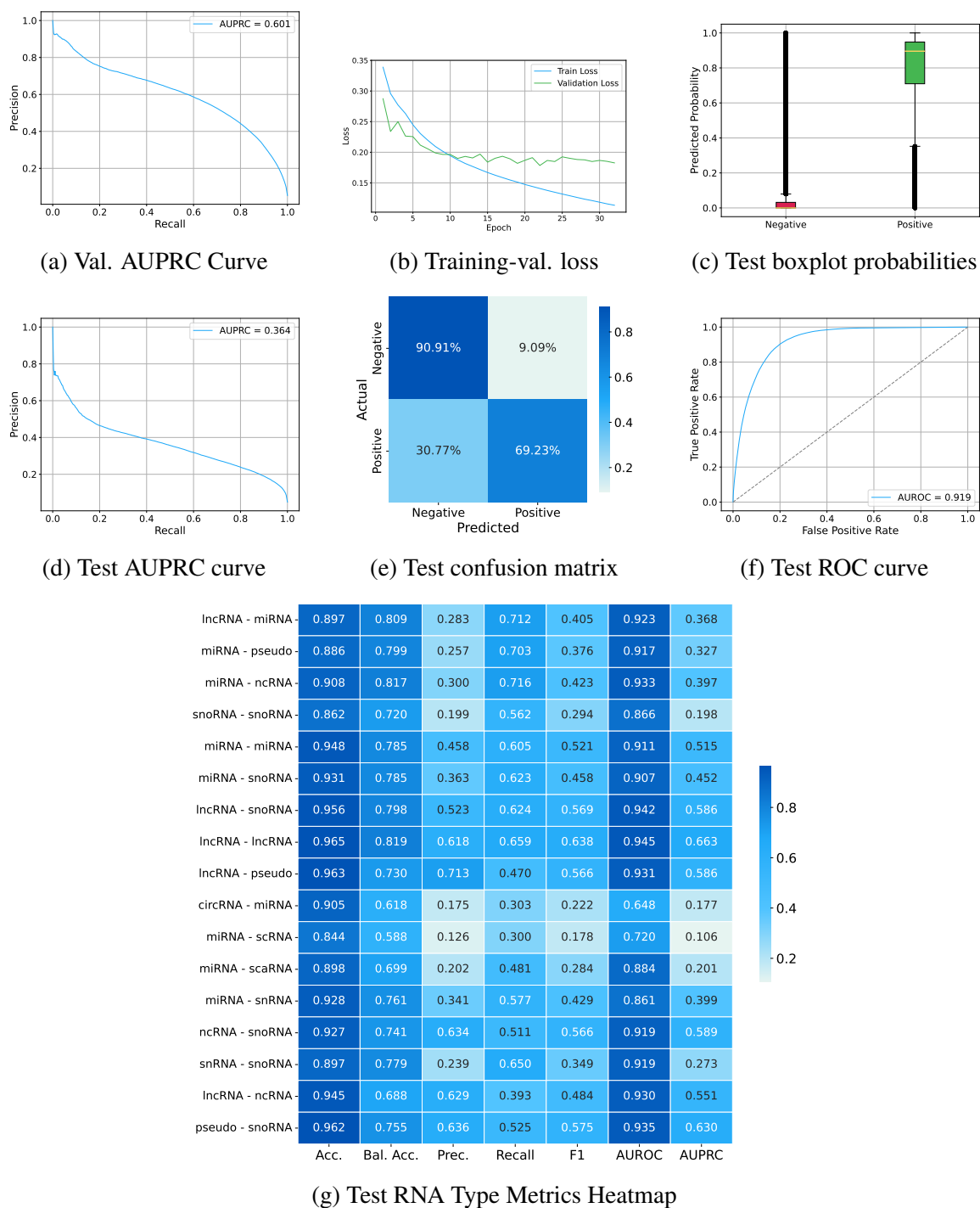


Figure S4: *ncRNA-CUPID* results for Augmented train and test Classification Experiments with types-dependent sampling, and average pooling. (a) Overall precision recall curve on the validation set including all the type of ncRNA interactions; (b) Training and validation loss across epochs; (c) Distribution of the *ncRNA-CUPID* predicted probabilities on negative and positive examples on the test set; (d) Overall precision recall curve on the test set including all the type of ncRNA interactions; (e) Confusion matrix on the test set; (f) ROC curve on the test set including all the type of ncRNA interactions; (g) *ncRNA-CUPID* results on the test set across different types on ncRNA interactions (rows) for different types of metrics (columns).

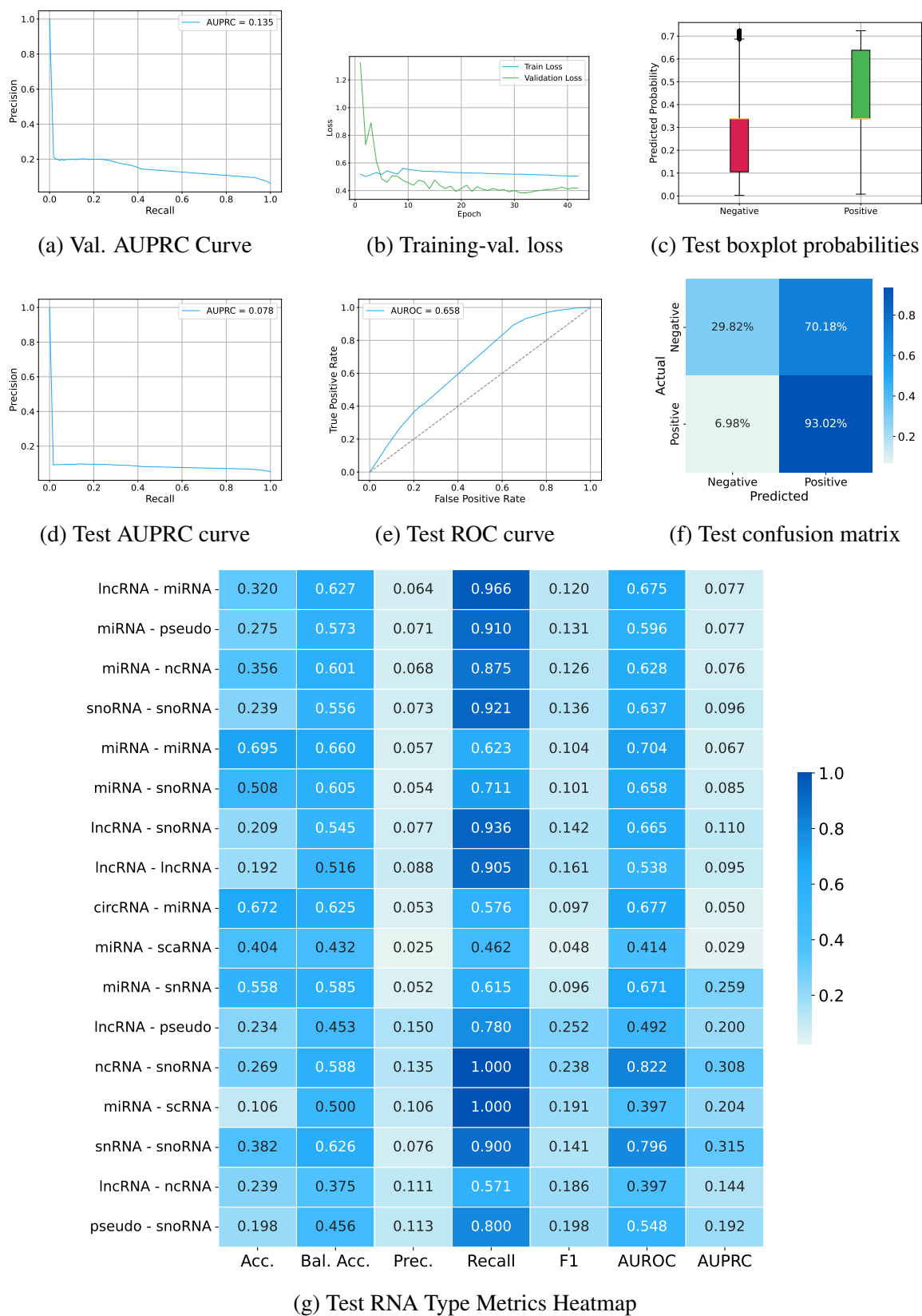


Figure S5: Results for Baseline classification Experiment, without data augmentation and molecule embedding obtained by concatenating AVG and Max embeddings.

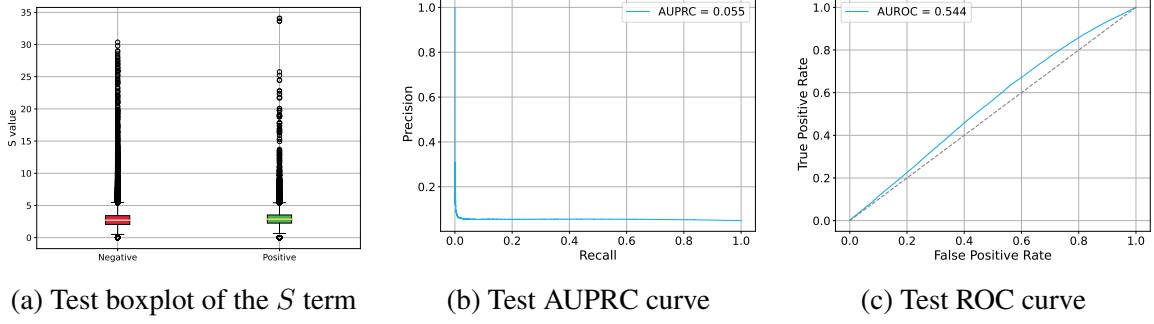


Figure S6: Results for Augmented test Classification Experiment with types-dependent sampling, IntaRNA results

S5 Comparison of AUCPR results to random-guess baseline

The baseline AUPRC, which represents the expected performance of a random classifier, is given by:

$$\text{Baseline AUPRC} = \frac{N_+}{N_+ + N_-}$$

where N_+ is the number of positive samples, and N_- is the number of negative samples. Given the positive:negative ratio we used in our work (1:20), our baseline is

$$\text{Baseline AUPRC} = \frac{1}{1 + 20} = \frac{1}{21} \approx 0.0476.$$

Our best model achieves an AUPRC of 0.364. The improvement factor over the random baseline is computed as:

$$\text{Improvement Factor} = \frac{\text{Model AUPRC}}{\text{Baseline AUPRC}} = \frac{0.364}{0.0476} \approx 7.65.$$

Thus, our model demonstrates a **7.65-fold improvement** over the random baseline.

References

- [1] E. Cavalleri et al. An ontology-based knowledge graph for representing interactions involving rna molecules. *Scientific Data*, 11(1):906, 2024.
- [2] Yichong Zhao, Kenta Oono, Hiroki Takizawa, and Masaaki Kotera. GenerRNA: A generative pre-trained language model for de novo rna design. *PLoS One*, 19(10):e0310814, 2024.