LSTM classification approaches to RNA torsion angle prediction

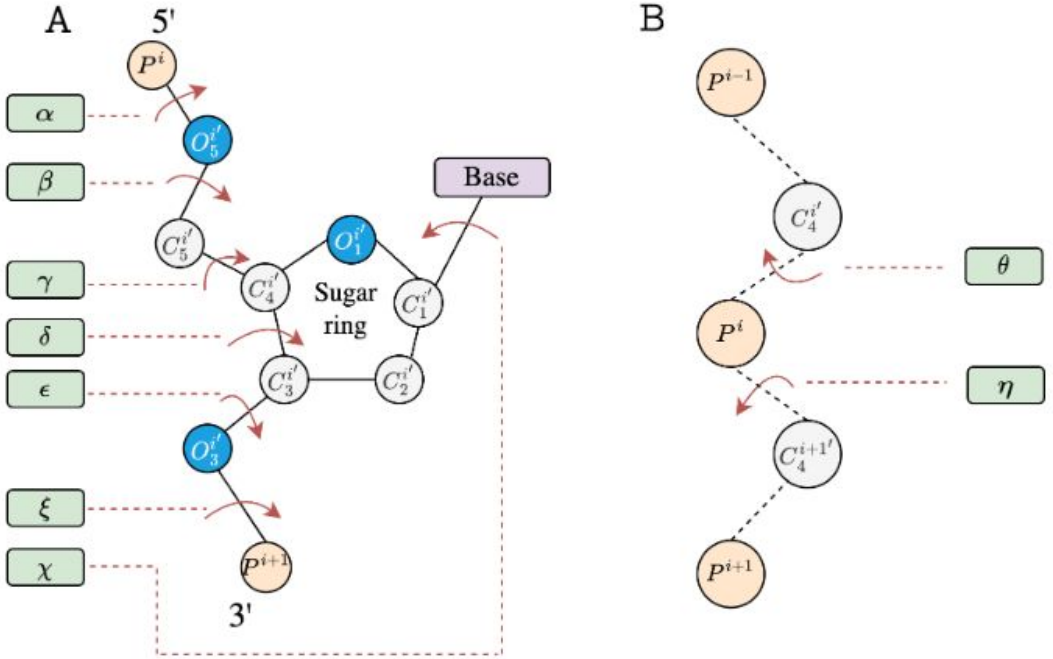
Sharafutdinov Emil
Komlev Savelii
Bogdan Elizaveta
Grigoryev Fedor

M2GENIOMHE 2023-2024

Background & Objectives

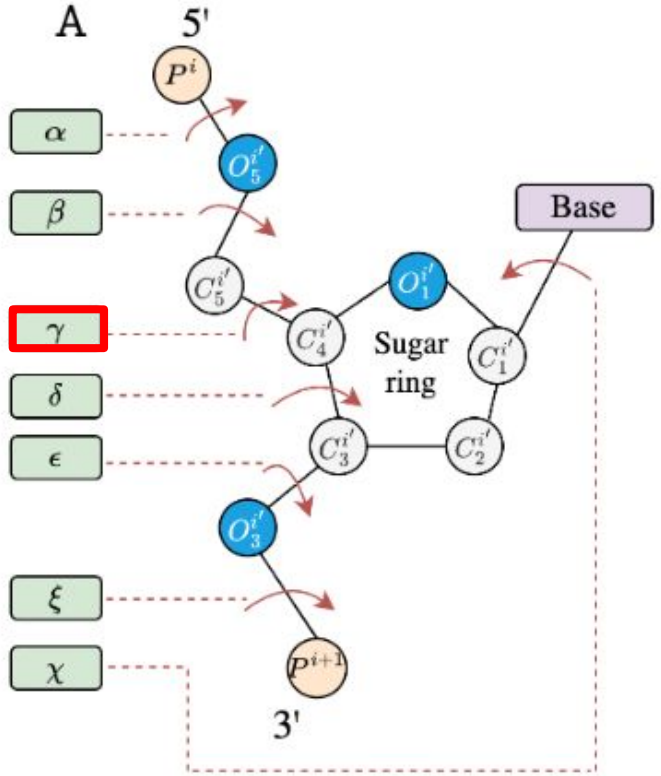
3D RNA structure

- 3D structure of RNA is essential for unraveling its functional and structural aspects
- Nucleotides can be described by 7 torsion angles and 2 pseudo-torsion angles
- Predicting torsion and/or pseudo-torsion angles could help to resolve 3D conformational folding

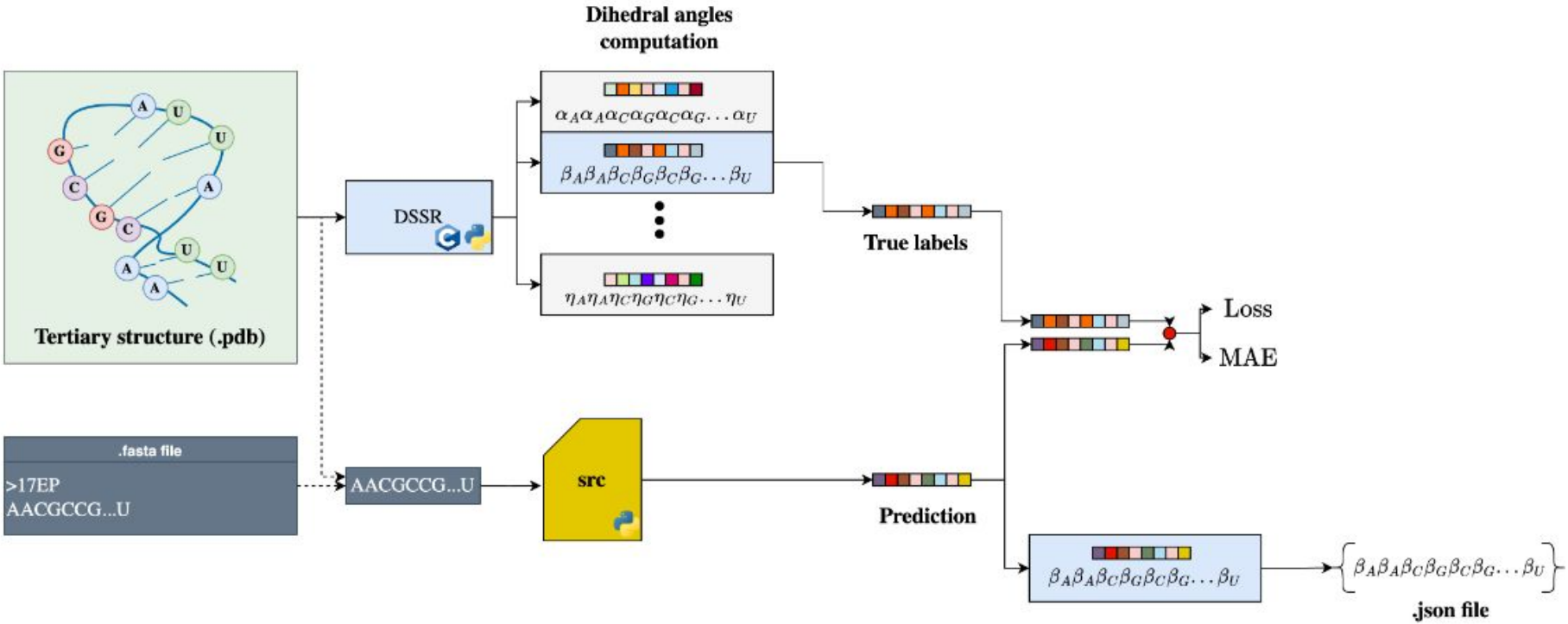


Objectives

- Predict gamma torsion angles per nucleotide
- Create deep learning architecture for binary and multi classification
- Compare results of the classification to SPOT-RNA-1D results



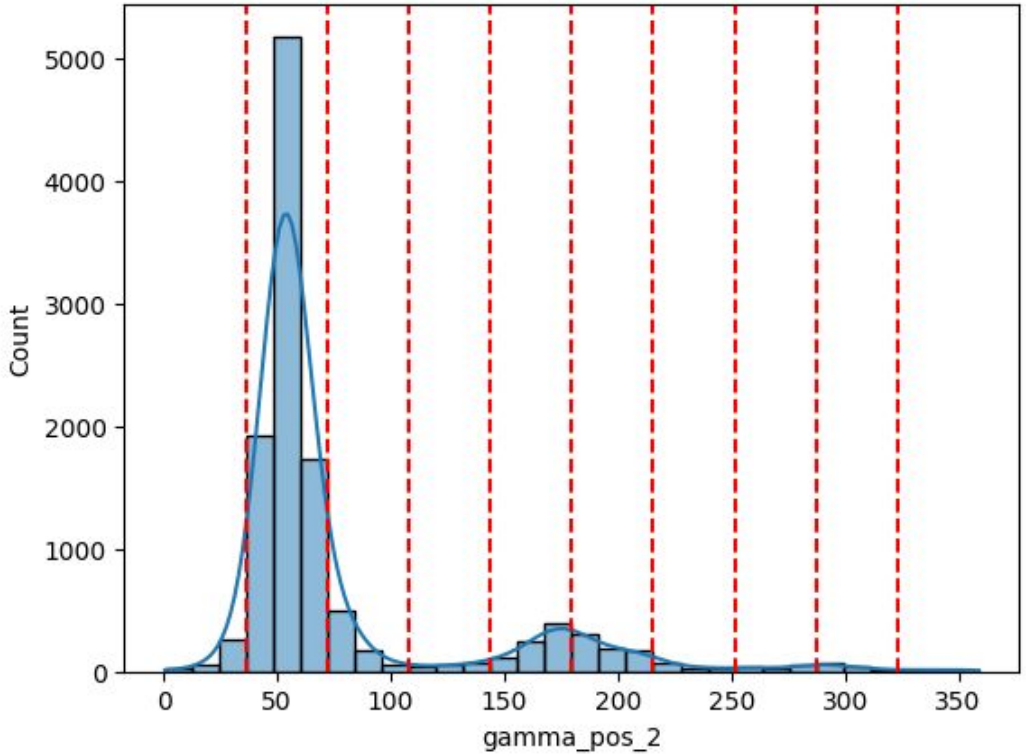
Description of a workflow



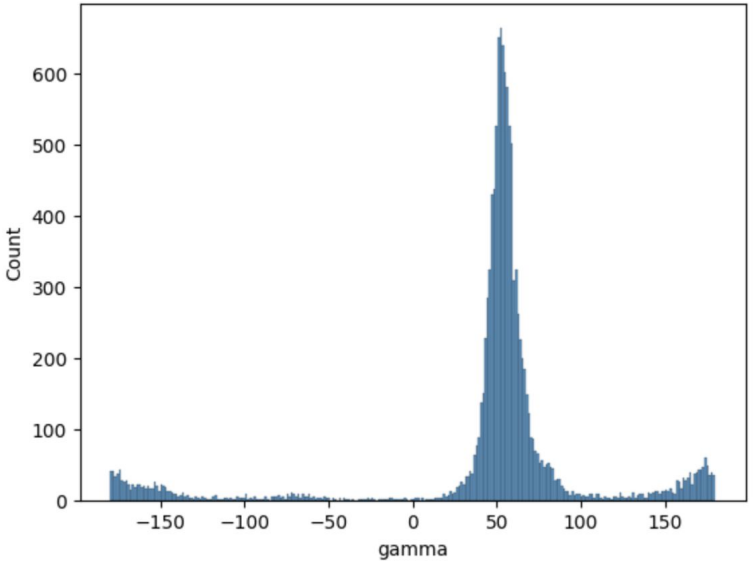
Two Approaches to Angles Discretisation

1) Distribution-Blind Discretization Approach (Uniform)

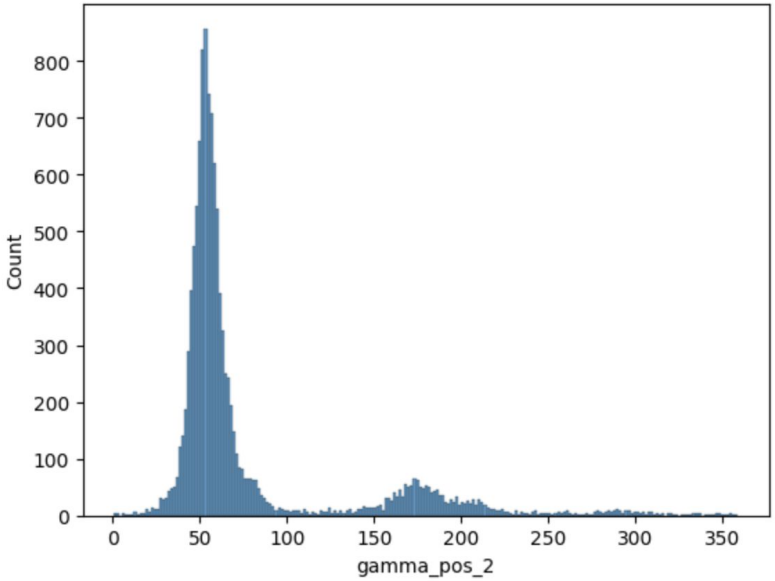
N = 10



2) Distribution-Aware Discretization Approach (GMM-based)

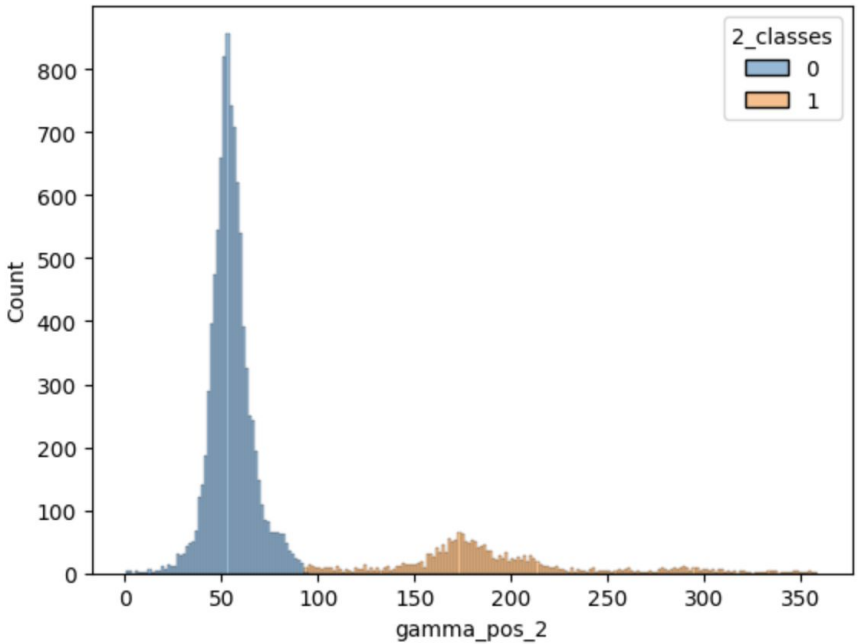


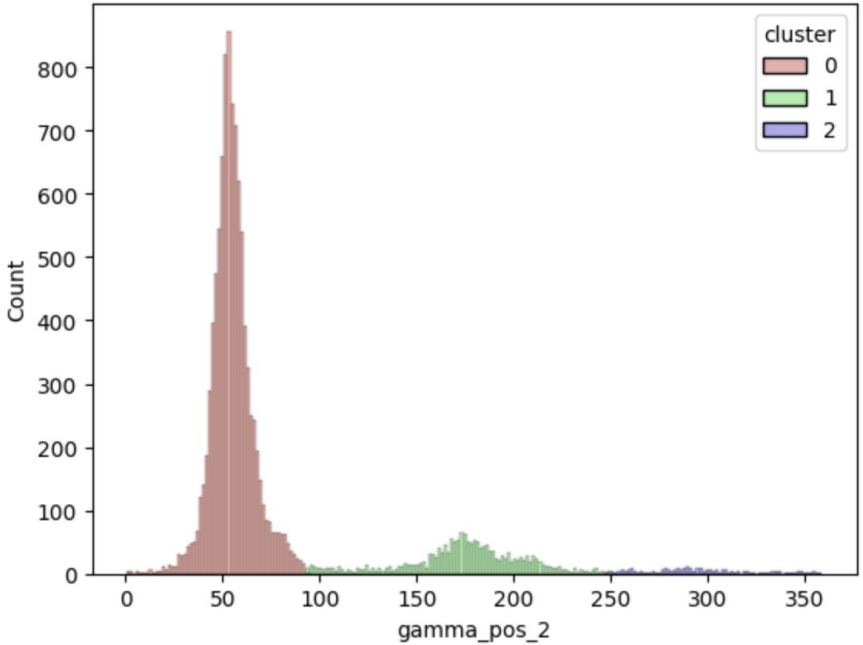
Transform to
(0; 360) space



- **Gamma angles turned out to have bimodal-like distribution**

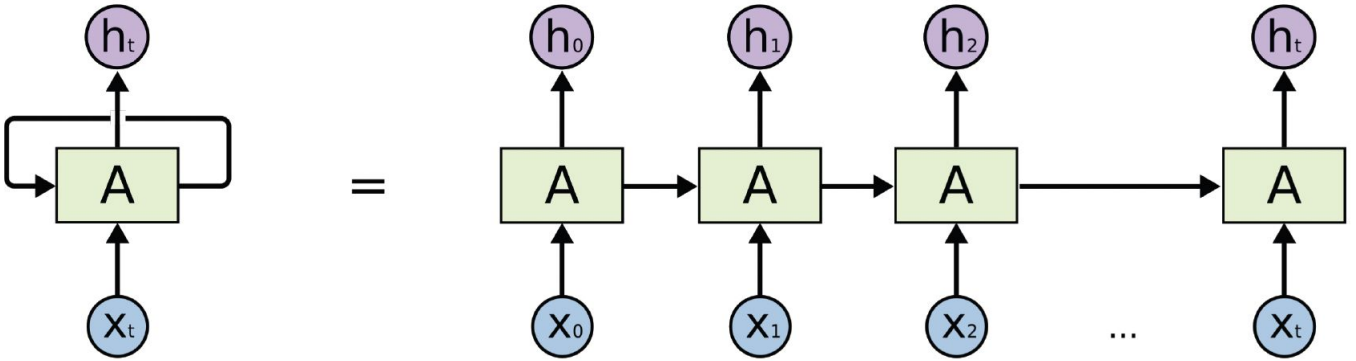
Gaussian Mixture Model (GMM) Confirms Multimodal Distribution





RNN Modeling

RNN

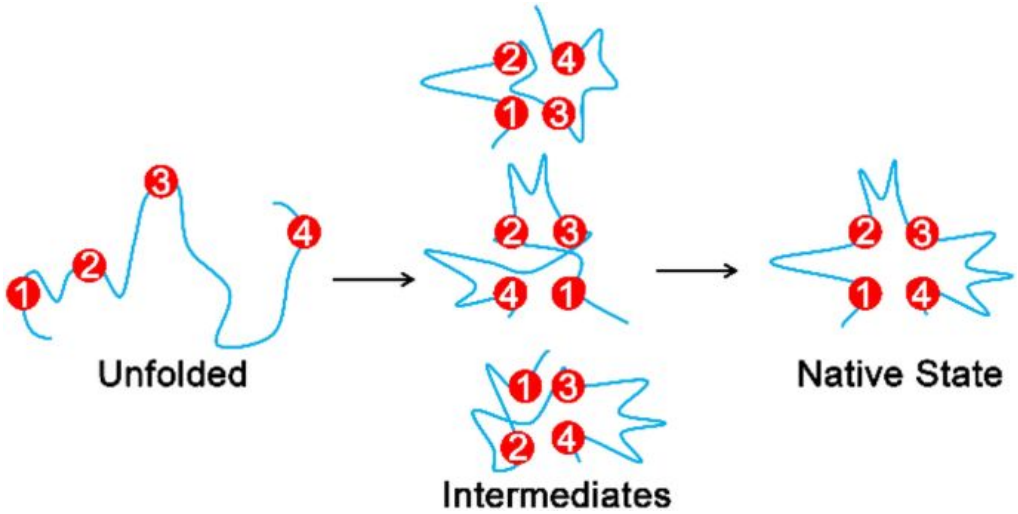


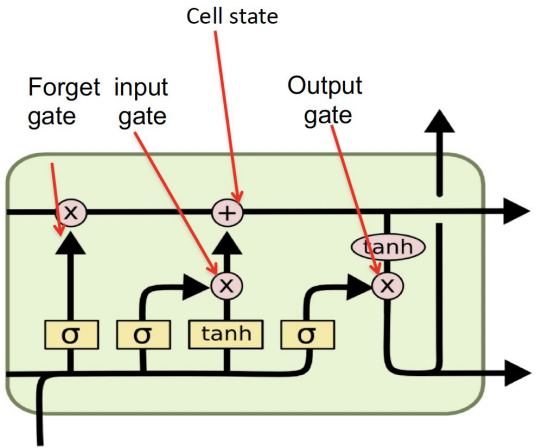
We have a sequence. All tokens in sequence have labels. RNN is a solid choice.

One more benefit of RNN - we could use same architecture for classification and regression. We just need to predict 1 value and not probabilities of different classes.

credits to Farida Zehraoui for the image

LSTM



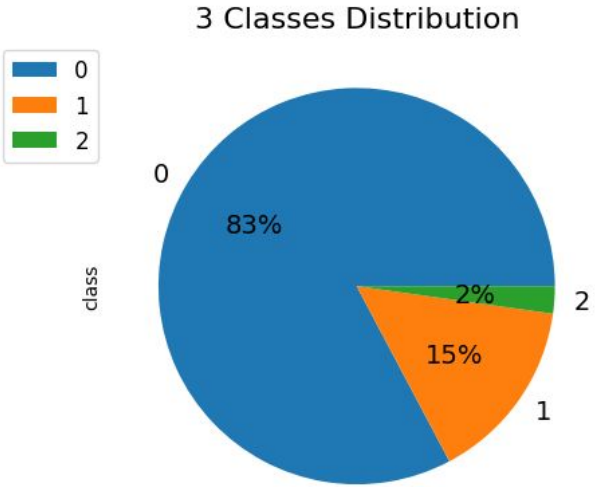


To capture long-distance interaction patterns we chose to preseed with Long Short Term Memory cells.

On the final layer we had MLP to give loggits per number of classes or a regression value.

credits to Farida Zehraoui for the image

Addressing Class Imbalance



$$precision = \frac{TP}{TP + FP}$$

$$recall = \frac{TP}{TP + FN}$$

$$F1 = \frac{2 \times precision \times recall}{precision + recall}$$

$$accuracy = \frac{TP + TN}{TP + FN + TN + FP}$$

$$specificity = \frac{TN}{TN + FP}$$

$$\mathcal{L}_{wBCE} = -\mathbb{E}\left[w_1 \cdot y_{true} \cdot \log(y_{pred}) + w_0 \cdot (1 - y_{true}) \cdot \log(1 - y_{pred})\right]$$

credits to Farida Zehraoui for the image

Classification Scores

$$precision = \frac{TP}{TP + FP}$$

$$recall = \frac{TP}{TP + FN}$$

$$F1 = \frac{2 \times precision \times recall}{precision + recall}$$

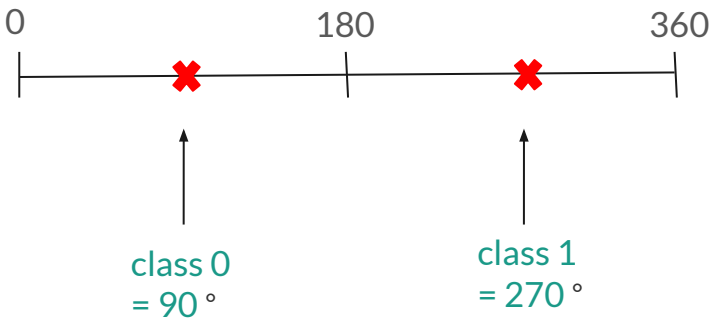
	Binary Uniform	Binary GMM-based	3 Classes GMM-based	20 Classes Uniform	30 Classes Uniform
Precision	0.8712	0.862	0.8542	0.9963	0.9757
Recall	0.3409	0.8304	0.0985	0.0038	0.0207
F1	0.4254	0.7544	0.0421	0.0002	0.005

Going Back from Classes to Angles

Two Approaches to Turn Classes to Angles

For “Uniform” Classification Approach:

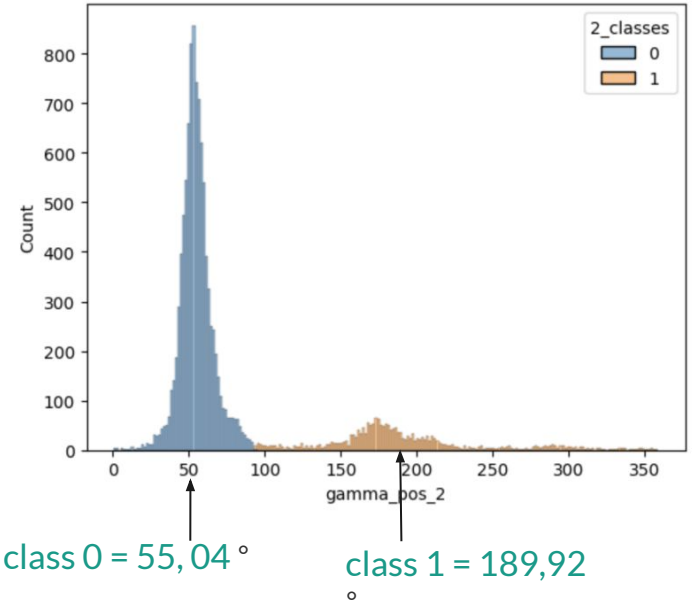
- Predicted angle is center between classes boundaries



Same idea for both binary and multi-class classifiers

For “GMM-based” Classification Approach:

- Predicted angle is the mean of the angle distribution for this class (based on training set)



MAE Scores

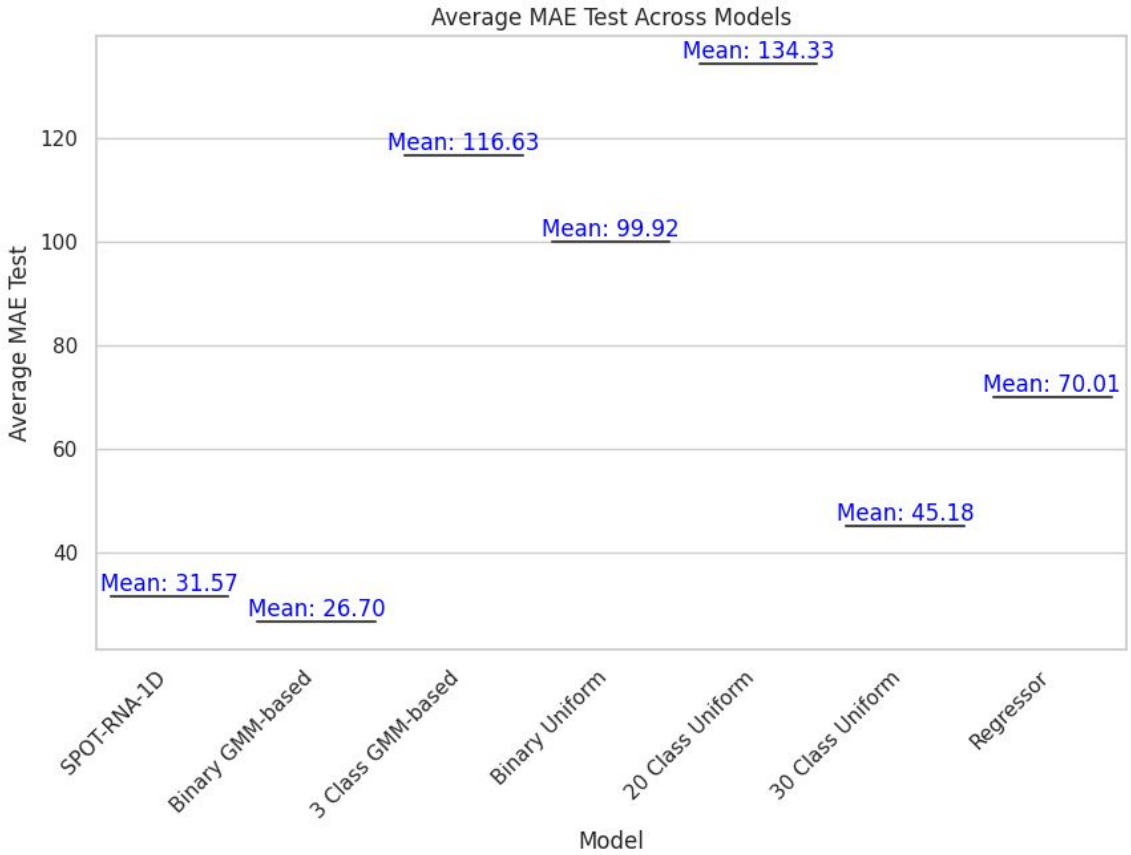
MAE was calculated between models' predictions and real angles (DSSR)

$$MAE = \min(d_i, 360 - d_i)$$

$$d_i = |angle_i^{pred} - angle_i^{true}|$$

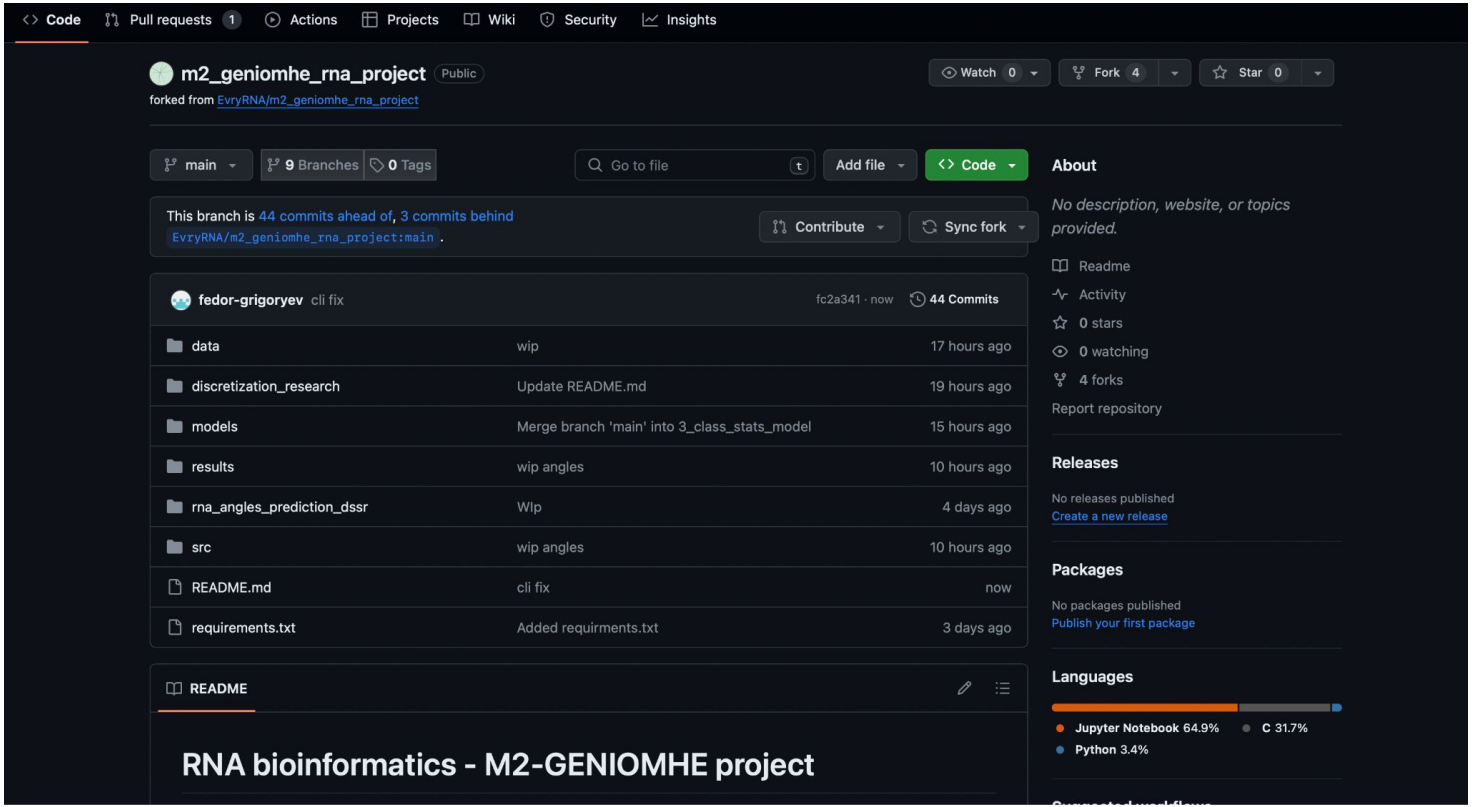
Testing Limitations:

1. Different sequences were tested for different models
2. Test set was ~40 sequences



Software Design

Repo Structure



CLI

Trained models were wrapped into a CLI version for user friendly prediction of torsional angles:

- Takes RNA sequence in a fasta file as an input
- Does automatic preprocessing for the angle prediction
- Predicts gamma torsional angles
- Outputs the results into a JSON file
- Possible to choose the model architecture between:
    - Binary classification model
    - 20-class quantized model
    - 30-class quantized model
    - Regression model
    - GMM based binary classification model

```
python3 angles_helper.py \
    --input_path '../data/sample/example.fasta' \
    --out_path '../results/test_output.json' \
    --model_type 'binary'
```

Limitations & Conclusions

Current limitations

Multiclass classification is being solved with weaker accuracy

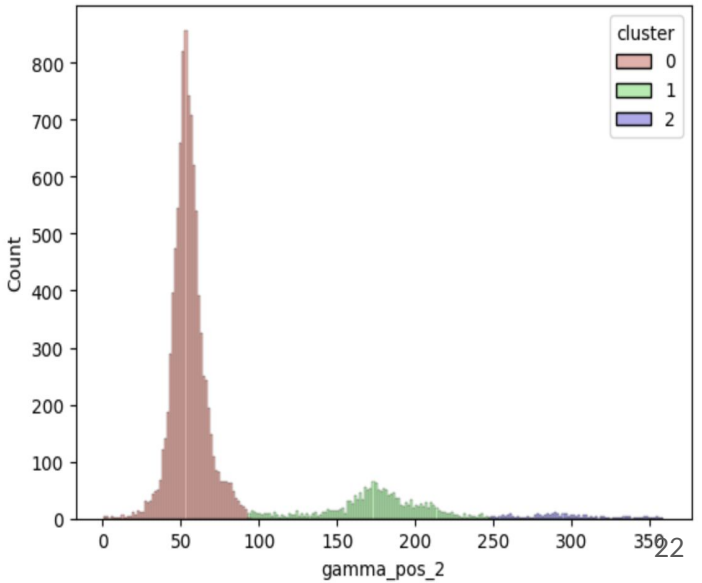
Ordinal embedding of the nucleotides - used as a baseline

GMM idea was currently incorporated only for the binary model, but it is also a good baseline for later development into Bayesian Variational model by:

1) classification into the mixture of distributions;

2) modeling the distribution through predicting its mean and variance

3) sampling the predicted distribution for the output



Conclusions

Explored different discretization strategies

Built and trained several baseline deep learning models to predict gamma torsional angle per nucleotide

Implemented a simple and intuitive CLI design for seamless interaction with the trained models

Proposed a GMM based solution

Compared the results to SPOT-RNA-1D

Thank you for your attention!

Group Contributions

Sharafutdinov Emil:

- RNN models development,
- Addressing class imbalance,
- Major part of code development and organization.

Komlev Savelii:

- Performed discretization research,
- Devised and implemented GMM-based models,
- Presentation structure and visualizations,
- Some code testing.

Grigoryev Fedor:

- CLI development,
- Documentation,
- Participated in GMM-based approach.

Bogdan Elizaveta:

- Data extraction,
- Data Preprocessing,
- Development of visualization.py module.