# KAGGLE PROJECT – OpenVaccine: COVID-19 mRNA vaccine degradation prediction

***Document made with LaTeX***

**REPORTER:** Ms. GALOCHKINA T., M^r GELLY J-C. and M^r GUYON F.

**BEL Alexis, BELAKTIB Anas, OUSSAREN Mohamed, ROUAUD Lucas**

**Master 2 BI**: Université Paris Cité

**Abstract**

**Background** Kaggle is a worldwide competition of machine learning, where scientist are attempting to answer to a complex question. Here, the problem is on the prediction of the stability of mRNA for COVID-19 vaccine. Here, we tried to answer this question by using three different embedding on two different neural networks with the aim to compare these different methods.

**Results** We test two simple neural networks, which gave approximately the same results. The only differences is that the Google inception module converge quickly than the CNN one. In addition, the different input embedding give various results.

**Dispobility** The program, with the original dataset, are available at this Github page: https://github.com/FilouPlains/KAGGLE_OPENVACCINE_MRNA.

**ABBREVIATIONS: *mRNA* = messenger ribonucleic acid; *CNN* = convolutional neural network.**

## 1. Introduction

### 1.1. Context

The COVID-19 is a mondial scale epidemics who paralyzed world's economics for almost 2 years [1–3]. To treat this disease, a quick vaccine has to be done. This is where mRNA vaccine are involved. They are quick to design and permits to immunize against a given a patient.

However, even if the mRNA vaccines are the fastest vaccine candidates for COVID-19, they still have a major issue: their lack of stability. Researchers have observed that RNA molecules tend to spontaneously degrade. Vaccine with degraded mRNA become deprecated to the population's vaccination [5, 6]. Currently, little is known on the details of where in the backbone of a given RNA is most prone to being affected. Without this knowledge, current vaccines against COVID-19 must be prepared and shipped under intense refrigeration, and are un-

Tab. 1: Lifespan of RNA vaccines at different temperature [4].

| Manufacturer | Frozen State | 2 – 8 °C | Room temperature |
|---|---|---|---|
| Moderna | 6 months (-20 °C) | 30 days | 12 h |
| Pfizer | 6 months (-80 °C) | 5 days | 2 h |
| CureVac | ∼3 months (-60 °C) | 3 months | 24 h |

likely to reach more than a tiny fraction of human beings on the planet unless they can be stabilized. The **[tab. 1]** contains the lifespan of different RNA vaccines against COVID-19. Each vaccine as a different temperature of conservation (frozen state) before they are use. When they are defrozen they must be immediately use before the vaccine start to degrade. The vaccination of the population must be organized to ensure that the vaccines are use efficiently for the two injections.

This project, from a Kaggle competition sponsor by Das Lab at Stanford Biochemistry, aims to use machine learning to a have better understanding of the regions which are most likely to be degraded by using the local characteristic of the sequence. The competition finish in 2020 and an article was published on it [7]. To do that, the degradation rates at various locations along RNA sequence is predicted. Two major problems are notable. The first one is the various input size of the sequences. Indeed, they are either size of 107 or 130. The second one is the various output size, which will be either 68 or 91 (depending on the input size sequence). In order to deal with these problems, mask are going to be design.

### 1.2. Data

Two datasets are available at Kaggle website: the first one contains 2,400 mRNA of 107 bases described by three descriptive variables that originate from Eterna [8]. These descriptive variables correspond to the RNA sequence, the structure and the predicted loop types assigned by bpRNA from Vienna RNAfold 2 structure. These variables describe the degradation rate in five values :

**reactivity:** These numbers are reactivity value and used to determine the likely secondary structure of the RNA sample.

**deg_pH10:** These numbers are reactivity value and used to determine the likelihood of degradation at the base/linkage after incubating without magnesium at pH 10.

**deg_Mg_pH10:** These numbers are reactivity value and used to determine the likelihood of degradation at the base/linkage after incubating with magnesium at pH 10.

**deg_50C:** These numbers are reactivity value and used to determine the likelihood of degradation at the base/linkage after incubating without magnesium at 50 °C.

**deg_Mg_50C:** These numbers are reactivity value and used to determine the likelihood of degradation at the base/linkage after incubating with magnesium at 50 °C.

The second one contains 3,634 molecules among which there are 629 sequences of size 107 and 3,005 of size 130 were synthesized and characterized at the Stanford University.

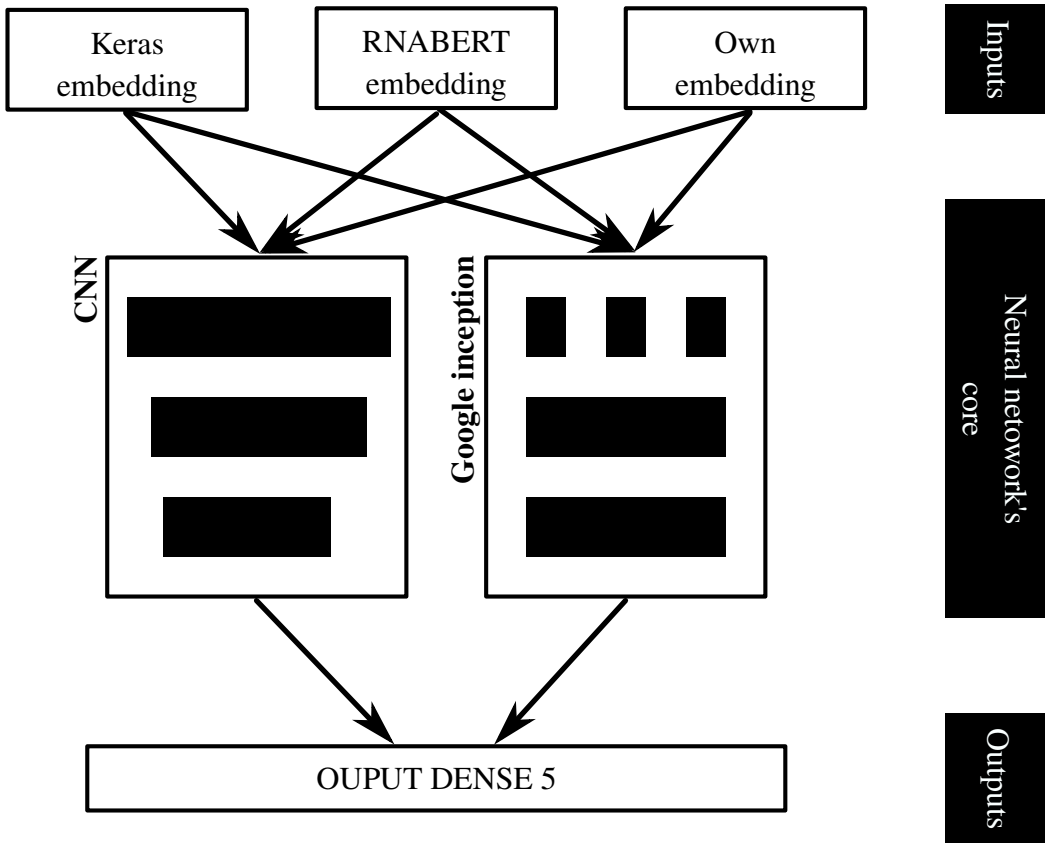


**Fig. 1: Diagram of the neural network created for this Kaggle project.** It's divided in three part: the inputs, which are embedding with dimensionnal adjusment; the neural network's core, consisting of two differents modules; the ouputs. All combinaison (6 in total) have been tested.

## 2. Materials and Methods

To answer to the problematics, we decide to design a program which transform the input into three type of embedding and two module for the "core" **[fig. 1]**.

### 2.1. Program implementation

The program was implemented in `python 3.10.6` using the module `keras 2.7.0` and `tensorflow 2.7.0`. The module `numpy 1.23.3` was also used to perform vectorial calculation.

### 2.2. Hot encoding

The one hot encoding consists in encoding the string sequences in vectors of different dimensions. Each string will be represented by a numpy array. The hot encoding is necessary before the embedding.

### 2.3. Embedding

Before feeding the data to the networks, we will use a three-way to embed the data. For two first methods, we used the sequences of RNA sequences, the sequences of the structure of RNA and the sequences of loop types of the RNA. For the third method, we only used the RNAs sequences.

#### 2.3.1. "Homade embedding"

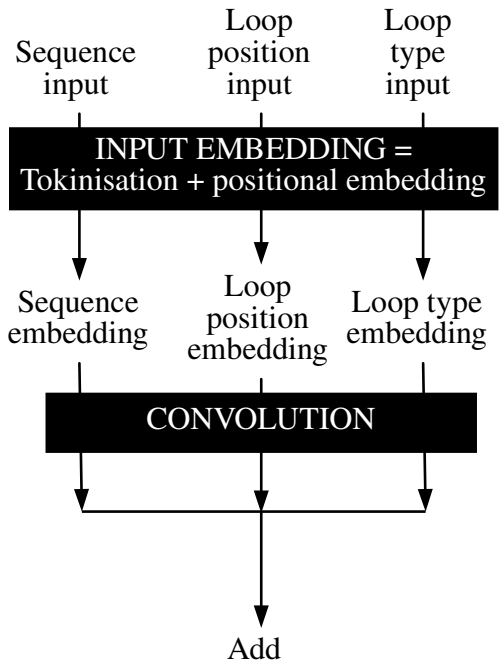


**Fig. 2: Diagram showing how the "homemade embedding" are modified to be input into the neural network.** The convolution is 1D.

The first way of embedding was with our own methods. First, we hot encode the three sequences. The RNA sequence will give a sequence of vector of 4. The structure sequence will give a sequence of vector 3. The loop type sequence will give a sequence of vector 7.

Then we used a positional embedding by taking the odd position of the sequence and the pair position:

- For the odd position, we use the formula:

$$\mathrm{E}_{\mathrm{p(pos,\ 2\times i)}} = \sin\left(\frac{\mathrm{pos}}{1 \cdot 10,000^{\frac{2 \times \mathrm{i}}{\mathrm{d}}}}\right)$$

- For the pair position, we use the formula:

$$\mathrm{E}_{\mathrm{p(pos,\ 2\times i+1)}} = \cos\left(\frac{\mathrm{pos}}{1 \cdot 10,000^{\frac{2 \times \mathrm{i}}{\mathrm{d}}}}\right)$$

**For the above equation** $E_p$ = positional embedding; *pos* = the index of the embedding or, here, the index of the sequence; $i$ = the index of the positional embedding or, here, the index of the base. With these methods, three embedded sequences of vector 4, 3 and 7 were obtained **[fig. 2]**.

#### 2.3.2. Keras embedding

For the second way of embedding show in **[fig. 3]**, the function of the module Keras "embedding" was used. We hot encoded the three sequences, then we gave the sequences to the Keras embedding. The Keras embedding gave us three sequences of vectors.

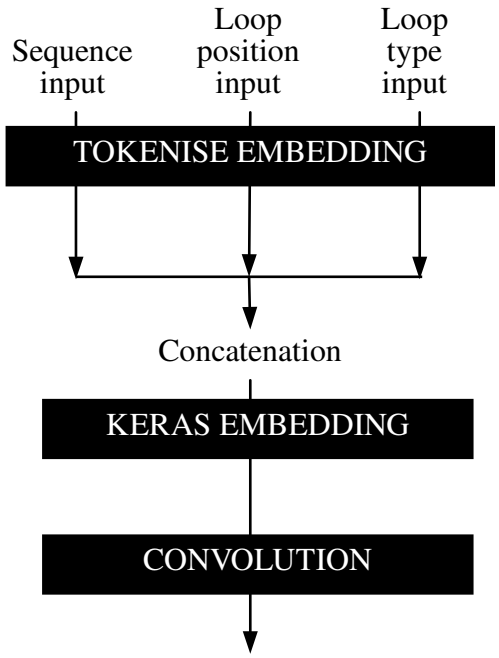


**Fig. 3: Diagram showing the embedding by keras.** The convolution is 1D.

#### 2.3.3. RNABERT embedding

For the third way of embedding, we used come from the article "Informative RNA base embedding for RNA structural alignment and clustering by deep representation" [9]. We used the neural network that was pre-trained on the database Rfam 14.3. This neural network will embed the RNA sequence in sequences of vector of 120 values. This embedding encode the characteristics of each RNA family and structures.

We give an RNA sequences to the network that will generate a 120 vector that encode for A, C, G and T for each base. Then the position embedding will generate a vector of 120 that encode the position of each base. Finally, the two vector is sum and given to the transformer. The transformer will return a sequence of length l of vector 120 that is informative base.

### 2.4. Neural network

During the study, we use two neural network modules: a convolutional neural network and an inception module.

#### 2.4.1. CNN

The first deep learning approach was a convolutional neural network (CNN). This CNN has 4 layers with a kernel initializer: he_uniform, and an activation: Relu with a 0,2 dropout. The model was then compiled using the Adam optimizer **[fig. 4]**.

$$\mathrm{MCRMSE} = \frac{1}{\mathrm{m}} \times \sum_{\mathrm{j}=1}^{\mathrm{m}} \mathrm{RMSE_j}$$

$$\mathrm{RMSE_j} = \sqrt{\frac{\sum_{\mathrm{i}=1}^{\mathrm{n}} \left(\mathrm{y_{ij}} - \hat{\mathrm{y}}_{\mathrm{ij}}\right)^2}{\mathrm{n}}}$$

**For the above equation** *MCRMSE* = Mean Column-wise Root Mean Squared Error; $m$ = number of predicted variables; $n$ = number of test samples; $y_{ij}$ = $i^{th}$ actual/real value of $j^{th}$ variable; $\hat{y}_{ij}$ = $i^{th}$ predicted value of $j^{th}$ variable.

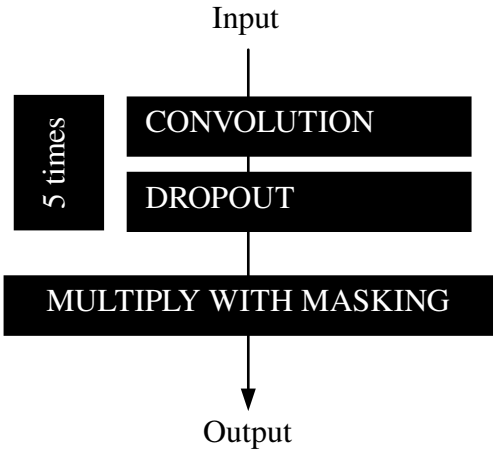


**Fig. 4: Diagram showing the model CNN.** The convolution is 1D.

### 2.5. Inception

The second deep learning approach was an inception model [10]. This model is divided into two parts **[fig. 5]**. The first part is a set of convolutional layers with a dropout that will be concatened and injected in the second part. The second part is composed of dense layers of decreasing filters size. The model was then compiled using the Adam optimizer.

#### 2.5.1. Non Cross-validation

To establish the best model, the number of epoch was established at 500 and the batch size at 50. The training data was divided by 20 % to estimate the fitting of the model. The best model was selected by looking at the validation loss between each epoch. When this loss decreases the model was safe, but if the loss didn't show any change for 10 epochs by a value of

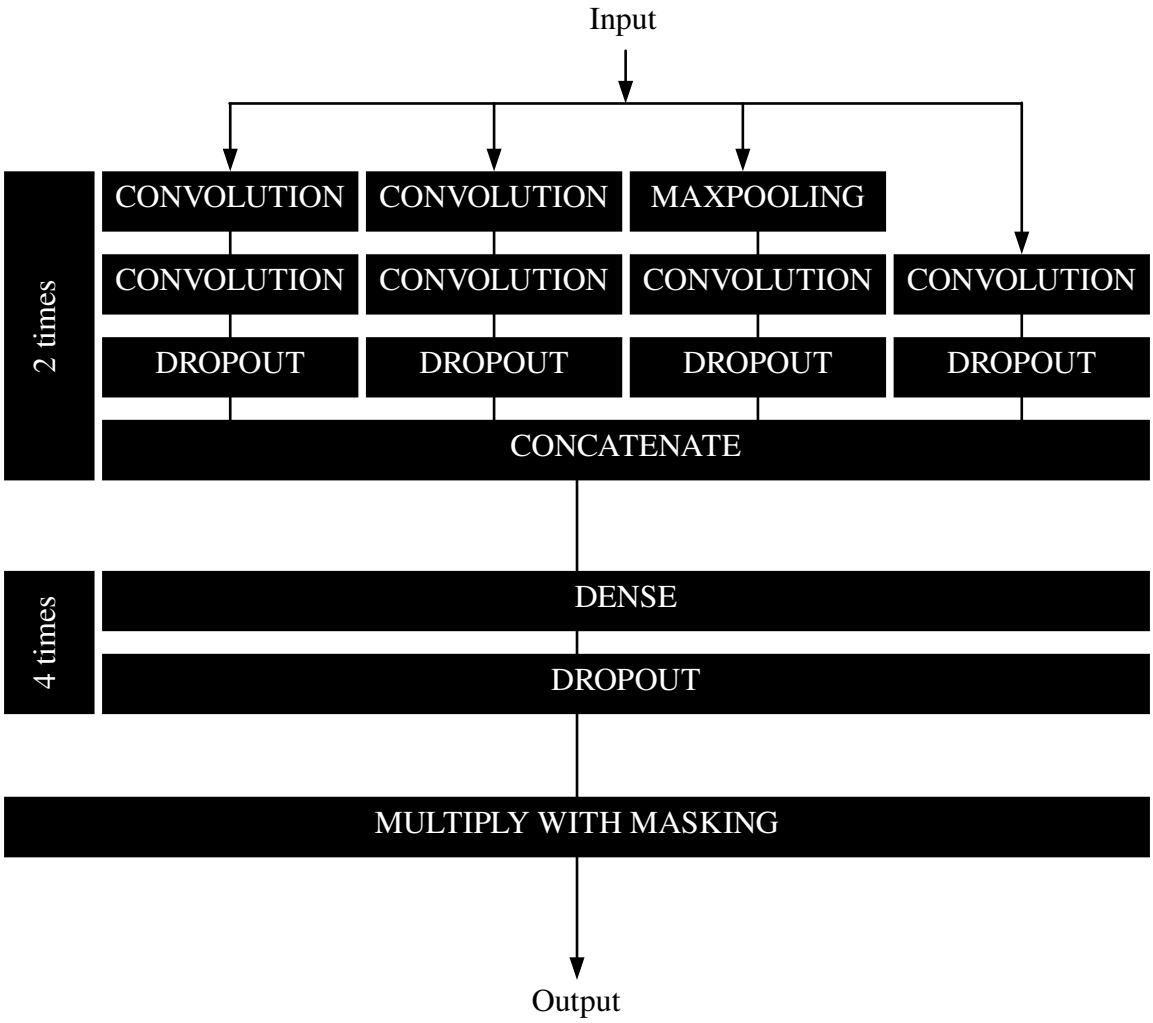


**Fig. 5: Diagram showing the model inception.** The convolution and the maxpooling are 1D.

0.001 the fitting is stopped.

## 3. Result

The first experiment was to build a pre-CNN to perform pre-processing on the given inputs. The three types of data are not structured in the same way. From this data, we define 2 neural networks of different architecture. This CNN will be compared in order to evaluate their performance. The proposed approach has been evaluated on a remote Cluster server having the characteristics of being hyper-threaded (numerous cores) to decrease the execution time.

### 3.1. CNN

Preliminary results from the CNN help to show the effectiveness and predictive ability. The results below were performed on the training datasets.

According to the figure **[fig. 6]**, the three fitted models have different epochs: the model keras stopped at the epoch 154; the model own at the epochs 59 and the model re at the epoch 137.

Globally, the CNN shows good performances according to the provided em-

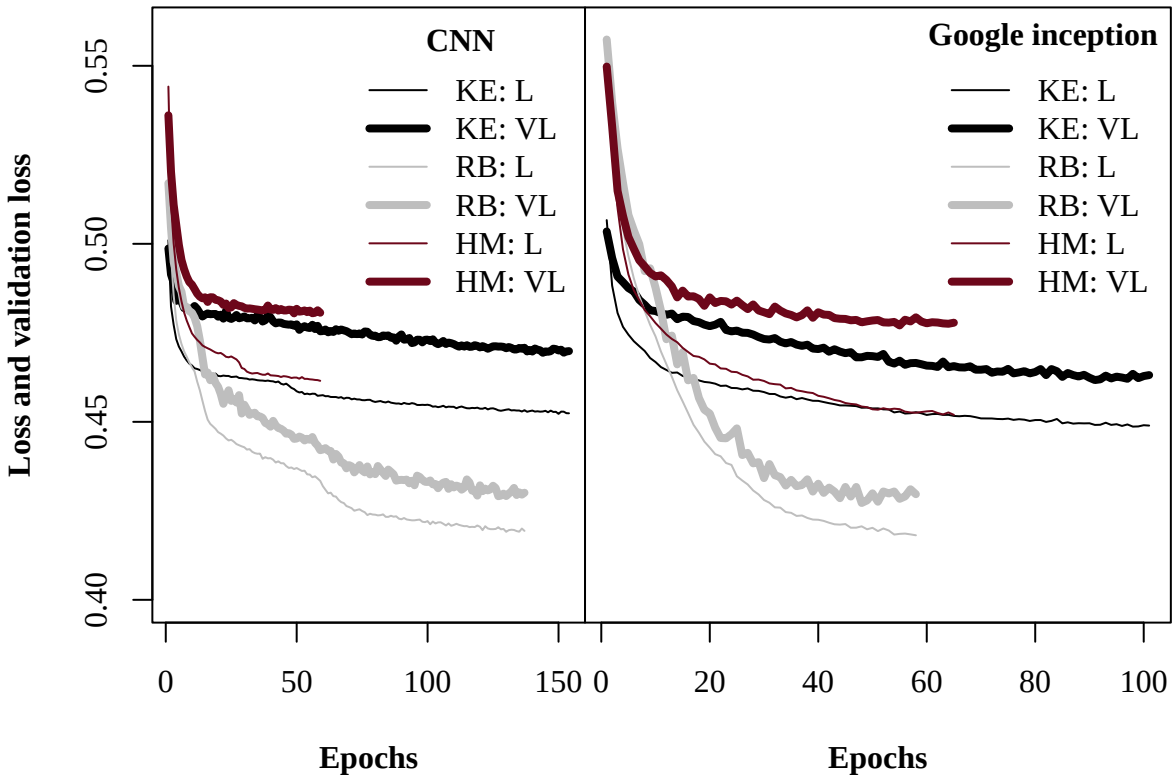


**Fig. 6: Plot of the different tested neural network.** *KE* = Keras' embedding; *RB* = RNABERT's embedding; *HM* = "homemade" embedding; *L* = loss evolution; *VL* = validation losss evolution. Some curve does not do all Epochs, as far as the program keep only the best model.

bedding types. Indeed, as illustrated on the figure above, the model keras and the model own seems to have similar values of loss but with a different number of epochs. The model owe obtain the values faster than the model keras. The models re has the lower values of loss and validation loss than the others. The model seems the most efficient. We can also remark that the model has a better capacity of generalization, with values of loss and validation loss close.

### 3.2. Inception

By applying a neuronal network following the model of Google Inception slightly modified, we have three fitted models with differents epochs: the model keras stopped at the epoch 101; the model own at the epochs 65 and the model re at the epoch 58.

The figure **[fig. 6]** represent the loss and the validation loss of the three models. The model own seems to be the most over-fitted. The keras model has his values of loss and validation parallel that indicates a good fitting. The best model is yet again the model re with the lowest value of loss and validation loss and with the smaller difference between the two values. This model is the most efficient and the most generalize.

### 3.3. The optimal neuronal network

By comparing the loss and the validation loss of the 6 models, the best model can be estimated. The criteria of selection are

**Tab. 2: Result publish on Kaggle [7]. The random result come from a submission containing random number.**

| Model | Public score | Private score |
|---|---|---|
| Best model from KAGGLE | 0.22614 | 0.34152 |
| This paper best model | 0.51887 | 0.58209 |
| Random | 0.58349 | 0.64742 |

the generalization (over or under fitting) and the value of loss. As we can see, the curve of the model RNABERT are close in value and shape, indicating a better generalization than the others. Also, it has the lowest value of loss and validation loss. The result of the model give better result that a model tha a random model **[tab. 2]**.

## Conclusion

For the three models, we can see that the different way of embedding have an effect on the prediction. The most efficient model is the one with the embedding from RNABERT on the model inception.

## References


[1] Ciotti, M. *et al.* The COVID-19 pandemic. *Critical Reviews in Clinical Laboratory Sciences* **57**, 365–388 (2020).

[2] Bartik, A. W. *et al.* The impact of COVID-19 on small business outcomes and expectations. *Proceedings of the National Academy of Sciences* **117**, 17656–17666 (2020).

[3] Long, E. *et al.* COVID-19 pandemic and its impact on social relationships and health. *Journal of Epidemiology and Community Health* **76**, 128–132 (2022).

[4] Crommelin, D. J., Anchordoquy, T. J., Volkin, D. B., Jiskoot, W. & Mastrobattista, E. Addressing the Cold Reality of mRNA Vaccine Stability. *Journal of Pharmaceutical Sciences* **110**, 997–1001 (2021).

[5] Leppek, K. *et al.* Combinatorial optimization of mRNA structure, stability, and translation for RNA-based therapeutics. *Nature Communications* **13**, 1536 (2022).

[6] Verbeke, R., Lentacker, I., De Smedt, S. C. & Dewitte, H. Three decades of messenger RNA vaccine development. *Nano Today* **28**, 100766 (2019).

[7] Muneer, A., Fati, S. M., Arifin Akbar, N., Agustriawan, D. & Tri Wahyudi, S. iVaccine-Deep: Prediction of COVID-19 mRNA vaccine degradation using deep learning. *Journal of King Saud University - Computer and Information Sciences* **34**, 7419–7432 (2022).

[8] Lee, J. *et al.* RNA design rules from a massive open laboratory. *Proceedings of the National Academy of Sciences* **111**, 2122–2127 (2014).

[9] Akiyama, M. & Sakakibara, Y. Informative RNA base embedding for RNA structural alignment and clustering by deep representation learning. *NAR Genomics and Bioinformatics* **4**, lqac012 (2022).

[10] Szegedy, C. *et al.* Going Deeper with Convolutions (2014).