Heidelberg University
Institute of Computer Science

Project report for the lecture Advanced Machine Learning

Prediction of the next SARS-CoV-2 variants

https://github.com/nilskre/AML-covid-project

Team Member: Felix Hausberger, 3661293,
Applied Computer Science
eb260@stud.uni-heidelberg.de

Team Member: Nils Krehl, 3664130,
Applied Computer Science
pu268@stud.uni-heidelberg.de

Plagiarism statement

We certify that this report is our own work, based on our personal study and/or research and that we have acknowledged all material and sources used in its preparation, whether they be books, articles, reports, lecture notes, and any other kind of document, electronic or personal communication.

We also certify that this report has not previously been submitted for assessment in any other unit, except where specific permission has been granted from all unit coordinators involved, or at any other time in this unit, and that we have not copied in part or whole or otherwise plagiarized the work of other students and/or persons.

Member contributions

Nils Krehl

Felix Hausberger

Contents

List of Abbreviations

GAN	Generative Adversarial Network
LSTM	Long Short-Term Memory
RNN	Recurrent Neural Network

0 Project Setup

For a detailed description of how to set up the project, please have a look at `https://github.com/nilskre/bomberman_rl/blob/master/README.md`.

1 Introduction

2 Fundamentals and Related Work

2.1 From Language Models to modeling Evolution Theory

Using such a seq2seq model a connection to the domain of neural machine translation can be drawn that changes the meaning of sentences but preserves a specific grammar (cf. preserve infectivity and evolutionary fitness but avoid the detection by the human immune system).

2.2 GISAID EpiFlu

2.3 Domain-Specific Methodologies to create Evolutionary Datasets for Mutation Prediction

2.4 Domain-Specific Methodologies to create Evolutionary Datasets for Mutation Prediction

2.5 Previous work on Mutation Prediction

Even before the rise of Covid-19 there had been studies trying to predict mutations of RNA viruses. In the collection of [**Wu2007**, **Yan2007**, **Wu2008**] the authors predict the mutation positions in hemagglutinins from influenza A virus using logistic regression and plain neural networks and then use the resulting amino acid mutating probabilities to derive possible mutated amnio acids. The same approach was further used for H5N1 neuraminidase proteins.

[**Salama2016**] proved that nucleotides in an RNA sequence can change based on their local neighborhood. Neural networks are used to predict new strains of the Newcastle virus and subsequently a rough set theory based algorithm is introduced to extract the according point mutation patterns.

[**Mohamed2021**] uses a more modern seq2seq LSTM neural network approach to learn nucleotide mutations between time-series species of H1N1 Influenza virus and the Newcastle virus as mutations can also be influenced by long-distance relations of amino acids. Therefore one hot-encoded RNA sequences of a parent generation preprocessed to words is given as input and the output is the predicted offspring generation evaluated by accuracy to the true offspring generation. The achieved accuracy in this paper is questionably high with 98.9% on the H1N1 Influenza virus and 96.9% on the Newcastle virus, possibly because of overfitting to the few 4.609 samples for H1N1 Influenza virus and only 83 for the Newcastle virus. Our approach therefore tries to increase the number of samples available for training when building the dataset.

Our approach will neither use any of the just mentioned architectures, but uses a Transformer based architecture coupled with a GAN-style training architecture. Nevertheless we would like to give a short introduction into sequence to sequence models and the underlying long short-term memory components to better point out our architectural decisions .

2.6 Sequence2Sequence Models based on Long Short-Term Memory

The original Long Short-Term Memory (LSTM) unit was introduced in [**Hochreiter1997**] and can be used for language modeling instead of using plain Recurrent Neural Networks (RNNs) to prevent running into vanishing gradient problems [**Sundermeyer2012**].

[**bla**] introduced sequence to sequence learning by using one LSTM to learn a large fixed-dimensional vector representation of the input that is provided one timestamp at the time and another LSTM to map the so-called context vector to a corresponding output sequence whose length does not need to match with the length of the input sequence. The output sequence is therefore given by the equation

$$p(y_1, ..., y_{T'}|x_1, ..., x_T) = \Pi_{t=1}^{T'} p(y_t|v, y_1, ..., y_{t-1}) \tag{1}$$

where each $p(y_t|v, y_1, ..., y_{t-1})$ is given by the softmax over all words in the vocabulary. Using an LSTM is prefered over a normal RNN as it is used to capture the long range temporal dependencies of the input data, each LSTM uses four layers with 1000 cells each. One finding was also that reversing the input sequence introduces many short term dependencies making optimization easier. The sequence to sequence model approach was evaluated for neural machine translation and reached a 34.81 BLEU score with an output vocabulary of 80k words (160k words input vocabulary, 1000 dimensional word embeddings, 8000 real numbers to represent a sentence).

—

- LSTM: `https://www.researchgate.net/publication/13853244_Long_Short-term_Memory`
- LSTM: `https://www.isca-speech.org/archive/archive_papers/interspeech_2012/i12_0194.pdf`
- Seq2Seq: `https://arxiv.org/abs/1409.3215`

2.7 Applying Generative Adversarial Networks

- Covid-Paper: `https://arxiv.org/pdf/2008.11790.pdf`

2.8 Transformer and Attention Mechanism

- Improvement: `https://arxiv.org/abs/1706.03762`

2.9 Other Techniques

- NNs/SVMs: `https://bsb-eurasipjournals.springeropen.com/articles/10.1186/s13637-016-0042-0`
- BiLSTM: `https://science.sciencemag.org/content/371/6526/284`

3 Approach

3.1 Dataset Creation

3.2 Data Preprocessing

- DNA Sequencing
- DNA Sequence Tokenization for Amino Acid Dictionary
- DNA Sequence Padding

3.3 Model Architecture

3.4 Training Process

4 Experimental results

5 Conclusion