Viral host prediction with Deep Learning

Florian Mock[1], Adrian Viehweger[1], Emanuel Barth[1], Manja Marz[1,2,3,4]

[1] Bioinformatics/High Throughput Analysis, Faculty of Mathematics and Computer Science, Friedrich Schiller University Jena, Jena, Germany
[2] Leibnitz Institute for Age Research - Fritz Lipmann Institute (FLI), Jena, Germany
[3] German Center for Integrative Biodiversity Research (iDiv), Halle-Jena-Leipzig, Germany
[4] European Virus Bioinformatics Center (EVBC), Jena, Germany

Zoonosis, the natural transmission of infections from animal to human, is a far-reaching global problem. The recent outbreaks of Zika virus and Ebola virus are examples of viral zoonosis, which occur more frequently due to globalization. In case of a virus outbreak, it is helpful to know which host organism was the original carrier of the virus. Once the reservoir or intermediate host is known, it can be isolated to prevent further spreading of the viral infection. Recent approaches aim to predict a viral host based on the viral genome, often in combination with the potential host genome and using arbitrary selected features. This methods have a clear limitation in either the amount of different hosts they can predict or the accuracy of the prediction. Here, we present a fast and accurate deep learning approach for viral host prediction, which is based on the viral genome sequence only. To assure a high prediction accuracy we developed an effective selection approach for the training data, to avoid biases due to a highly unbalanced number of known sequences per virus-host combinations.
We tested our deep neural network on three different virus species (influenza A virus, rabies lyssavirus, rotavirus A) and reached for each virus species a AUC between 0.94 and 0.98, outperforming previous approaches and allowing highly accurate predictions while only using fractions of the viral genome sequences. We show that deep neural networks are suitable to predict the host of a virus, even with a limited amount of sequences and highly unbalanced available data. The deep neural networks trained for this approach build the core of the virus host predicting tool `VIDHOP` (VIrus Deep learning HOst Prediction).

Keywords: virus, host, deep learning, sequence

INTRODUCTION

Zoonosis, more specifically the cross-species transmission of viruses, is a significant threat to human and livestock health, because during a viral breakout it can be difficult to asses from where specific viruses originated[1;2]. However, this information can be crucial for the effective control and eradication of an outbreak, as the virus needs time to fully adapt to the new human or animal host before it can spread within the new host species. Only with this information can the original host be separated from humans and livestock. Such isolation can limit the zoonosis and thus can also limit the intensity of a viral outbreak.

Various different computational tools for predicting the host of a virus by analyzing its DNA or RNA sequence have been developed. These methods can be divided in three general approaches: supervised learning[3–5], probalistic models[6] and similarity rankings[7;8]. All of these

approaches require features with which the input sequence can be classified. The features used for classification are mainly k-mer based with various k sizes between 1-8. In the case of probabilistic models and similarity rankings, not only the viral genomes but also the host genomes have to be analyzed.

Still today, it is largely unknown how viruses adapt to new hosts and which mechanisms are responsible for enabling zoonosis[2;9;10]. Because of this incomplete knowledge it is likely to choose inappropriate features, *i.e.*, features of little or no biological relevance, which is problematic for the accuracy of machine learning approaches. In contrast to classic machine learning approaches, deep neural networks have the ability to learn features, necessary for the solving a specific task, by themselves.

In this study, we present a novel approach, using deep neural networks, to predict viral hosts by analyzing either the whole or only fractions of a given viral genome. We selected three different virus species as individual datasets for training and validation of our deep neural networks. These datasets consist of genomic sequences from influenza A virus, rabies lyssavirus and rotavirus A, together with the information of 49, 19 and 6 different known host species, respectively. These known viral hosts are often phylogentically close related (see Figure 2) and previous prediction approaches have combined single species or even generas to higher taxonomical groups to reduce the classification complexity to the price of prediction precision[4;6]. In contrast, our approach is capable of predicting on host species level, providing much higher accuracy and usefulness of our predictions.

Our training data consists of hundreds of genomic sequences per virus-host combination. The amount of sequences per combination is unbalanced. Meaning that some classes are much more common than others. We provide an approach to handle this problem by generating a new balanced training set at each training circle.
Typically the training of recurrent neural networks on very long sequences is very time consuming and inefficient. TBPTT[11] tries to solve this problem by splitting the sequences. We provide a method to regain prediction accuracy lost through this splitting process, leading to a fast efficient learning of long sequences on recurrent neural networks.
In conclusion, our deep neural network approach is capable of predicting far more complex classification problems then previous approaches[3–6;12]. Meaning, it is more accurate for the same amount of possible hosts and can predict for more hosts with a similar accuracy. Furthermore, our approach does not require any host sequences, which can be helpful due to the limited amount of reference genomes of various species, even ones which are typically known for zoonosis such as ticks and bats[13;14].

METHODS

General workflow

The general workflow of our approach is designed to achieve multiple goals: (**I**) select, preprocess and condense viral sequences with as little information loss as possible (**II**) correctly handle highly unbalanced datasets to avoid bias during the training phase of the deep neural networks (**III**) present the output in a clear user-friendly way while providing as much information as possible.

Our workflow for creating deep neural networks to predict viral hosts consisted of five major steps (see Figure 1). First, we collected all nucleotide sequences of influenza A viruses, rabies lyssaviruses and rotaviruses A with a host label from the European Nucleotide Archive (ENA) database[15]. We curated the host labels using the taxonomic informations provided by the National Center for Biotechnology Information (NCBI), leading to standardized scientific names for the virus taxa and host taxa. Standardization of taxa names enables swift and easy filtering for viruses or hosts on all taxonomic levels.Next, the sequences from the selected hosts and virus are then divided into three sets: the training set, the validation set and the test set. We provide a solution to use all sequences of an unbalanced dataset without biasing the training in terms of sequences per class, while limiting the memory needed to perform this task. Then, the length of the input sequences is equalized to the 0.95 quantile length of the sequences and subsequently further truncated in shorter fragments and parsed into numerical data to facilitate a swift training phase of the deep neural network. After the input preparation the deep neural network predicts the hosts for the subsequences of the originally provided viral sequences. In the final step, the predictions of the subsequences are analyzed and combined to a general prediction for their respective original sequences.

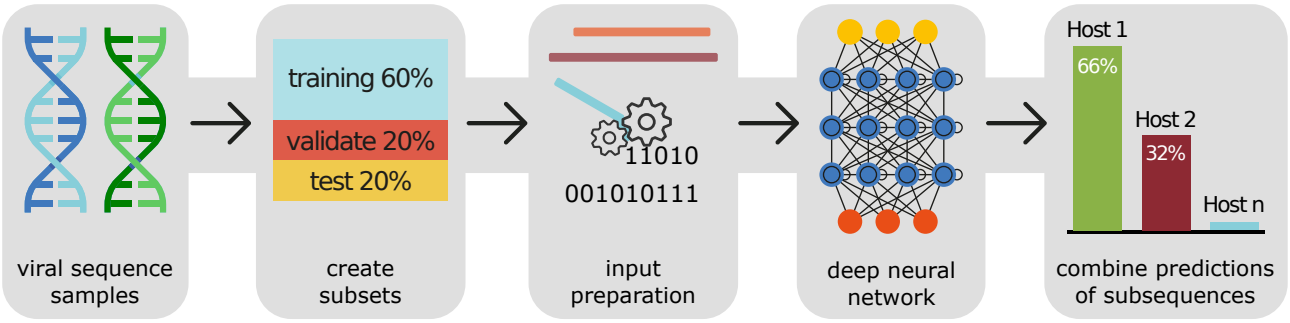

Figure 1. The general workflow consists of several steps. First, suitable viral sequences have to be collected and standardized. Next, these sequences will be distributed into the training set, validation set and test set. The sequences are then adjusted in length and are parsed into numerical data, which is then used to train the deep neural network. The neural network predicts the host for multiple subsequences of the original input. The subsequence predictions are then combined to a final prediction.

Collecting sequences and compiling datasets

Accession numbers of influenza A viruses, rabies lyssaviruses and rotaviruses A were collected from the ViPR database[16] and Influenza Research Database[17] and all nucleotide sequence information were then downloaded from ENA (status 2019-07-12).
From the collected data we created one record per virus species with all known hosts that had at least 100 sequences. All available sequences were used for each of these hosts (see Figure 2).

To train the deep neural network we divided our dataset into three smaller subsets, a training set, a validation set and a test set.
Classically, in neuronal network approaches the data is divided into a ratio of 60 % training set, 20 % validation set and 20 % test set with a balanced number of data points per class. Since in our example nucleotide sequences are the data points and the different hosts are the classes, this would lead to an unbiased training with respect to sequences per host. One major disadvantages is that for huge unbalanced datasets, such as typical viral datasets, the majority of sequences would not be used. This is because the host with the smallest number of sequences would determine the maximum usable amount of sequences per host (see Figure 3a).

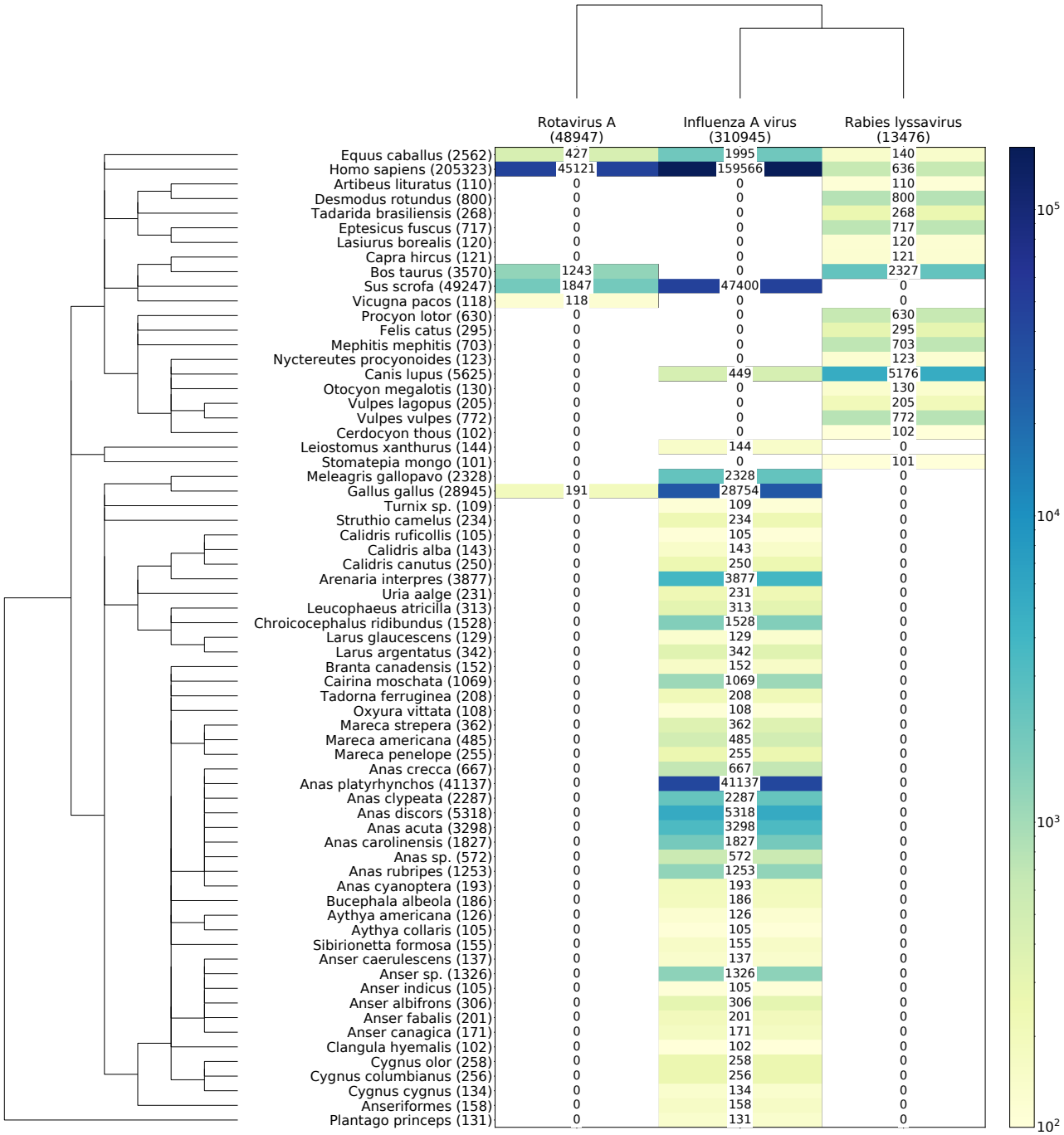

Figure 2. At the top the examined virus species are listed with the total amount of used nucleotide sequences shown in brackets. On the left the potential host species are listed together with the total number of available sequences for this three viruses. The matrix lists the corresponding number of sequences for each virus-host combination used in this study. Dendrograms indicate the phylogenetic relationships of the investigated viruses and host species. As it can be seen, there is a strong imbalance in available viral sequences per virus-host pair.

A more appropriate approach to deal with large unbalanced datasets is to define a fixed validation set and a fixed test set and create variable training sets from the remaining unassigned sequences. In the following we call this the *generator approach*. For each training circle (epoch), a new training set is created by randomly selecting the same number of unassigned sequences per host. The number of selected sequences per host corresponds to the number of unassigned sequences of the smallest class. With this generator approach bias in the training set in terms of sequences per host is avoided while making use of all available sequences. Especially hosts with large quantities of sequences benefit from the generation of many different training sets with random sequence composition.

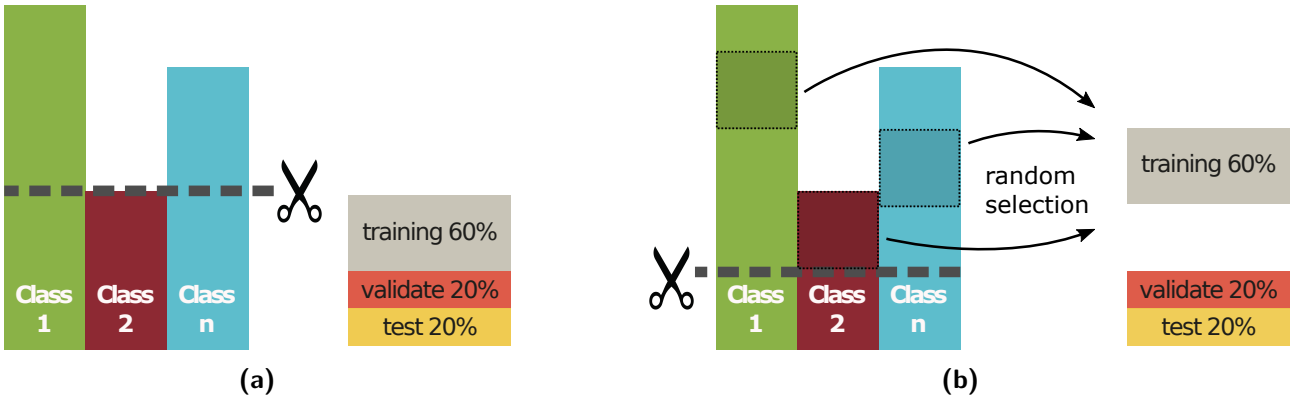

Figure 3. Comparison between the classic approach (**a**) of creating a balanced dataset and the generator approach (**b**). In the classic approach the class with the smallest amount of data points defines the amount of usable data points for all classes. The generator approach creates every epoch a new random composition of training data points from all data points which are not included in any of both fixed validation set and test set. For every epoch, the training set is balanced according to the data points per class. The generator approach can use all available data in a unbalanced dataset, without biasing the training set in terms of data points per class, while limiting the amount of computer memory needed.

Input preparation

The training data needs to fulfill several properties to be utilizable for neural networks. The input (here, nucleotide sequences) has to be of equal length and also has to be numerical. To achieve this, the length of the sequences was limited to the 0.95 quantile of all sequence length by truncating the first positions or in the case of shorter sequences by extension. For sequence extension different strategies were tested and evaluated (see Figure 4, Supplement Table S1):

- *Normal repeat:* repeats the original sequence until the 0.95 quantile of sequence length is reached, all redundant bases are truncated.
- *Normal repeat with gaps:* between one and ten gap symbols are added randomly between each sequence repetition.
- *Random repeat:* appends the original sequence with slices of the original with the same length as the original. For this purpose, the sequence is treated as a circular list, that the end of the sequence is followed by the beginning of the sequence.
- *Random repeat with gaps:* like *random repeat* but repetitions are separated randomly by one to ten gap symbols.

- *Append gaps:* adds gap symbols at the end of the sequence until the necessary length is reached.
- *Online:* uses the idea of *Online Deep Learning*[18] , i.e., slight modifications to the original training data are introduced and learned by the neuronal network, next to the original training dataset. In our case, randomly selected subsequences of the original sequences are provided as training input. Therefore more diverse data is provided to the neural network.
- *Smallest:* all sequences are cut to the length of the shortest sequence of the dataset.

After applying one of the mentioned input preparation approaches, each sequence is divided in multiple non-overlapping subsequences (see Figure 4). The length of these subsequences ranged between 100 and 400 nucleotides, depending on which length results in the least redundant bases. Using subsequences of distinct shorter length than the original sequences is a common approach in machine learning to avoid inefficient learning while training long short-term memory networks on very long sequences (see, *Truncated Backpropagation Through Time* approach[19]). Finally, all subsequences are encoded numerically, using *one hot encoding* to convert the characters A, C, G, T, -, N into a binary representation (*e.g.*, $A = [1, 0, 0, 0, 0]$, $T = [0, 0, 0, 1, 0]$, $- = [0, 0, 0, 0, 0]$). Other characters that may occur in the sequence data were treated as the character N.

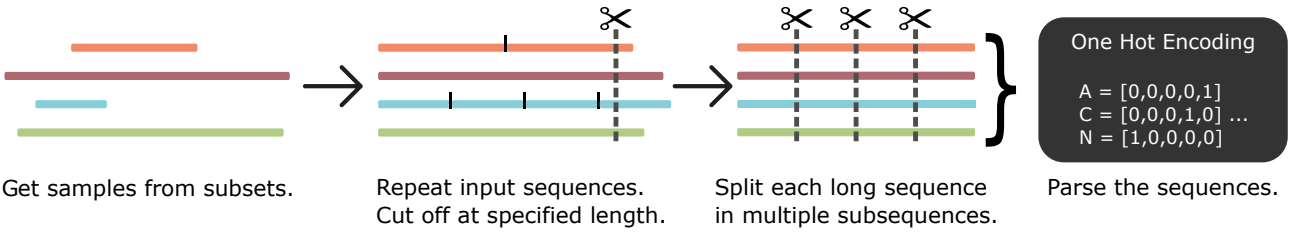

Figure 4. Conversion of given input sequence data into numerical data of equal length on the example of the *normal repeat* method. Each sequence is extended through self-repetition and is then trimmed to the 0.95 quantile length of all sequences. Sequences are then split into multiple non-overlapping subsequences of equal length. Each subsequence is then converted via *one hot encoding* into a list of numerical vectors.

Deep neural network architecture

The performance of the neural network is greatly determined by its underlying architecture. This architecture needs to be complex enough to fully use the available information but at the same time small enough to avoid overfitting effects.
All our models (i.e., the combination of the network architecture and various parameter such as the optimizer or validation metrics) were built with the `Python` (version 3.6.1) package `Keras`[20] (version 2.2.4) using the `Tensorflow`[21] (version 1.7) back-end.
In this study, two different models were built and evaluated to predict viral hosts only given the nucleotide sequence data of the virus (see Figure 5). The architecture of our first model consists of a three bidirectional LSTM layers[22], in the following referred to as *LSTM* architecture (see Figure 5a). This bidirectional LSTM tries to find longterm context in the input sequence data, presented to the model in forward and reverse direction, which helps to identify interesting patterns for data classification. The LSTM layers are followed by two dense layers were the first collects and combines all calculations of the LSTMs and the second generates the output layer. Each layer consists of 150 nodes with exception to the output layer, which has a variable

number of nodes. Each node of the output layer represents a possible host species. Since each tested virus dataset contains different numbers of known virus-host pairs, the number of output nodes varies between the different virus datasets. This architecture is similar to those used in text analysis, but specifically adjusted to handle long sequences, which are typically problematic for deep learning approaches.

The second evaluated architecture uses two layers of *convolutional neural networks* (CNN) nodes, followed by two bidirectional LSTM layers and two dense layers. In the following we will refer to this as the *CNN+LSTM architecture* (see Figure 5b). Similar to the LSTM architecture, each layer consists of 150 nodes with exception to the output layer. The idea behind this architecture is that the CNN identifies important sequence parts (first layer), combines the short sequence features to more complex patterns (second layer), which can then be put into context by the LSTMs, which are able to remember previously seen patterns.

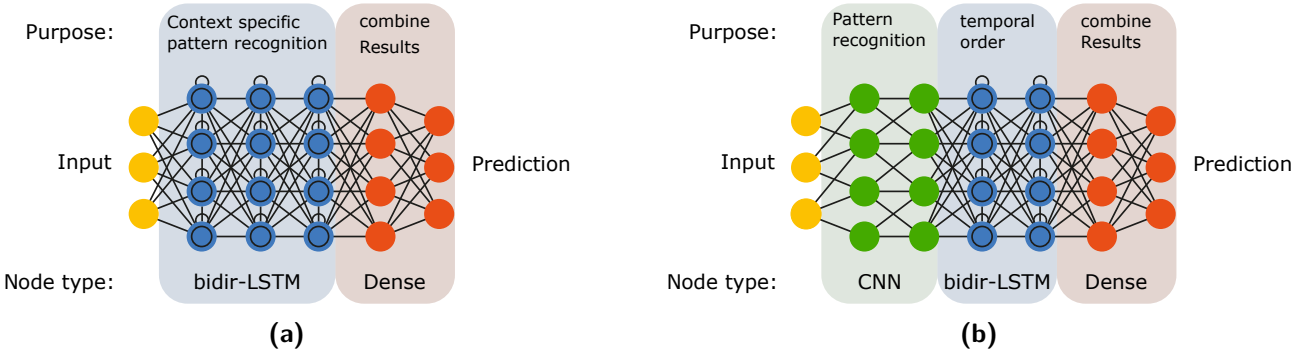

Figure 5. Comparison of the two evaluated architectures. The first architecture (**a**) is similar to neural networks for text analysis. The bidirectional LSTM analyzes the sequence forwards and backwards for important patterns, having an awareness of the context as it can remember previously seen data. This architecture is a classic approach for analyzing sequences with temporal information, like literature text, stocks, weather. The second architecture (**b**) uses CNN nodes, which are common in image recognition, to identify important patterns and combines them into complex features that can then be associated by the bidirectional LSTM layers. This architecture is typically used in either more unordered data, such as images, or data with more noise, such as base-caller output of nanopore sequencing devices.

Deep neural network training

The training was done using the *generator approach* as described above (see Figure 3b), *i.e.,* having a fixed validation set and test set while the training set was newly compiled during every epoch. All classes had an equal amount of sequences during each epoch, leading to an unbiased training in regards to likelihood to observe each class. Both neural networks were trained for 500 epochs during all performed tests. After each epoch, the quality of the model was evaluated by predicting the hosts of the validation set, comparing the prediction with the true known virus-host pairs. As metrics the accuracy and the categorical cross entropy were used. If the current version of the model performed better, *i.e.,* it had a lower validation loss or higher validation accuracy than in previous epochs, the weights of the network were saved. After training the model weights with the lowest validation loss and model weights with the highest validation accuracy were applied for predicting the test set.

Final host prediction from subsequence predictions

Given a viral nucleotide sequence the neural network returns the activation score of the corresponding output nodes of each host. The activation scores of all output nodes add up

to 1.0 and can therefore be treated as probabilities. Thus, the activation score of each output node represents the likelihood of the corresponding species to serve has a host of the given virus sequence. Due to the splitting of the long sequence into multiple subsequences (see Figure 4), the neural network predicts potential hosts for every subsequence. The predictions of the subsequences are then combined to the final prediction of original sequence. Several approaches to combine the subsequence predictions into a final sequence preditction were evaluated:

- *Standard:* shows the original accuracy for each subsequence.
- *Vote:* uses a majority voting over all subsequences to determine the prediction.
- *Mean:* calculates the mean activation score per class over all subsequences and predicts the class with the highest mean activation.
- *Standard deviation:* calculates the standard deviation of each subsequence prediction and uses them as a weight for the corresponding subsequence to calculate their mean activation score.

Table 1. Comparison of host prediction results in regards to the different approaches that can be used to merge prediction scores of subsequences. In this example, a viral nucleotide sequence was split into five subsequences and each of them were used to predict the corresponding host. Depending on the subsequence activation score merging approach, the final host prediction can vary.

subsequences	1	2	3	4	5	Voting	Mean	Std-div
Human	0.4	0.69	0.0	0.15	0.4	-	0.328	0.0575
Swine	0.3	0.3	0.01	0.1	0.22	-	0.186	0.0299
Avian	0.3	0.01	0.99	0.75	0.38	-	0.486	0.1457
std (weight)	0.0471	0.2786	0.4643	0.2953	0.0805	-	-	-
predicted	Human	Human	Avian	Avian	Human	Human	Avian	Avian

After combining the host predictions of a given viral sequence's subsequences the single most likely host can be provided as output. However, this limits the prediction power of the neural network. For example, a virus which is able to survive in two different host species will likely have a high activation score for both hosts. Our tool `VIDHOP` reports all possible hosts that reach a certain user-defined activation score threshold or it can report the n most likely hosts where n is also an user adjustable parameter.

RESULTS & DISCUSSION

To evaluate our deep learning approach, we applied it to three different datasets, each containing a great number of either influenza A virus, rabies lyssavirus or rotavirus A genome sequences and the respectively known host species. We tested two different architectures and six different input sequence preparation methods. For all twelve combinations, a distinct model was trained for 500 epochs and their prediction accuracies were tested, using none or any of the described subsequence prediction approaches.

Rotavirus A dataset

The rotavirus A dataset consists of nearly 50,000 viral sequences, which are associated to one of six phylogentically distinct host species. For the LSTM architecture all sequence input preparation strategies except *append gaps* and *smallest*, perform with an accuracy over 84 % (see Table 2). However, there are significant differences in the host prediction accuracy between the two different architectures of the neural network. Overall, the LSTM architecture achieves an higher accuracy than the CNN+LSTM architecture with 85.83 % and 81.30 %, respectively. The highest accuracy was observed with the combination of the LSTM architecture and the *normal repeat* input preparation. Note that the LSTM architecture has difficulties learning some training sets (see LSTM and *smallest*). This is probably due to the relatively long input sequence, since the LSTM must propagate the error backwards through the entire input sequence and update the weights with the accumulated gradients. The accumulation of gradients over hundreds of nucleotides in the input sequence may cause the values to shrink to zero or result in inflating values[23;24].

The host prediction based on 356 nucleotide long subsequences of the rotavirus A dataset reaches already a high accuracy of 85.83 %. Both the LSTM and the CNN+LSTM architectures are able to identify important features, almost independent from the input preparation. The main differences between the two architectures in the prediction accuracies derive from the applied input preparation strategy. In total, the host prediction quality of rotavirus A sequences achieves an area under the curve (AUC) of 0.98 (see Supplement Figure S1). This is not totally unsuspected, since it is known that rotaviruses A show a distinct adaptation to their respective host[26].

Table 2. Host prediction accuracy in percent on the rotavirus A dataset with different architectures and input preparation strategies. The input preparation strategy with the highest accuracy for each architecture is marked grey. Expected accuracy by chance is ~ 16.67 %.

rotavirus A input vs architecture	normal repeat	normal repeat gaps	random repeat	random repeat gaps	append gaps	online	smallest
LSTM	84.62	85.83	85.46	85.09	43.33	84.58	13.33
CNN+LSTM	70.73	75.34	81.30	77.50	38.41	77.60	75.83

Rabies lyssavirus dataset

The rabies lyssavirus dataset consists of more than 13,000 viral sequences, which are associated to 19 different host species, including closely and more distantly related species. The applied input preparation has a great impact on the prediction accuracy of the rabies lyssavirus dataset, with *random repeat* being the best performing (see Table 3). Despite using only an subsequence length of 100 bases the accuracy of each subsequence prediction is very high. The LSTM architecture reaches a higher accuracy then the CNN+LSTM architecture for the rota lyssavirus dataset with 74.02 % and 71.98 %, respectively. The highest accuracy per subsequence is reached with the combination of the *random repeat* input preparation and the LSTM architecture. Compared to the rotavirus A dataset the higher amount of host species and closer relation between them makes the rabies lyssavirus dataset harder to predict. In total, the host prediction quality of rabies lyssavirus sequences achieves an AUC of 0.98 (see Supplement Figure S2).

Table 3. Host prediction accuracy in percent on the rabies lyssavirus dataset with different architectures and input preparation strategies. The input preparation strategy with the highest accuracy for each architecture is marked grey. Expected accuracy by chance is $\sim 5.26\%$.

rabies lyssavirus input vs architecture	normal repeat	normal repeat gaps	random repeat	random repeat gaps	append gaps	online	smallest
LSTM	58.21	68.75	74.02	72.37	18.3	64.36	65.00
CNN+LSTM	53.93	68.43	71.98	71.69	18.27	37.02	72.10

Influenza A virus dataset

The influenza A virus dataset is with more than 310,000 viral sequences and 49 associated possible host species (around 40 of them are closely related avian species) the most complex of all evaluated datasets. Like in the rotavirus A and rabies lyssavirus dataset, the LSTM architecture outperforms the CNN+LSTM architecture with 44.20 % and 43.54 % host prediction accuracy (see Table 4). The predictions based on 386 nucleotide long influenza A virus subsequences reached comparable accuracies for all input preparation methods except *append gaps* and *smallest*, were it performed worst. The overall best performing variant is a combination of the LSTM architecture with the *random repeat* input preparation. The deep neural network achieved and AUC of 0.94 (see supplement Figure S3). Despite the close evolutionary distance between the given host species, the trained neuronal network was able to identify potential hosts accurately. We assume that some of the influenza viruses which are part of the investigated dataset are capable of infecting not only one but several host species, *i.e.*, a single viral sequence can occur in more than one host. However, since we consider only one distinct host species for every single tested viral sequence within the test set, the measured accuracy is most likely underestimated.

Table 4. Host prediction accuracy in percent on the influenza virus A dataset with different architectures and input preparation strategies. The input preparation strategy with the highest accuracy for each architecture is marked in grey. Expected accuracy by chance is $\sim 2.04\%$.

influenza A input vs architecture	normal repeat	normal repeat gaps	random repeat	random repeat gaps	append gaps	online	smallest
LSTM	39.27	42.86	44.20	43.54	33.96	41.53	2.04
CNN+LSTM	40.78	43.54	43.11	43.48	33.86	9.96	30.51

General observations

Overall, the host prediction quality for short subsequences for all three datasets is very high, indicating that an accurate prediction of a viral host is possible even if the given viral sequence is only a fraction of the corresponding genome's size. The best performing architecture was the LSTM, achieving the highest accuracy for all three datasets. Nevertheless, for a fast prototyping it makes sense to use CNN+LSTM as it trains around 4 times faster and reaches comparable results. Furthermore the CNN+LSTM architecture showed no difficulty in learning long input sequences (see Supplement Figure S4), while the LSTM architecture frequently remained in a state of random accuracy for a long time during training (see Supplement Figure S5).

We observed *random repeat* to be the most suited input preparation, as it achieved the highest accuracy most of the times. This approach provides the neural network with an almost random selection of the original sequence, since the first subsequences are always the beginning of the original sequence, whereas the last subsequences consist of random selections. A completely random selection as in the *online* approach seems to be too diverse. Non-random approaches such as *normal repeat* seem to lead to a faster overfitting of the training set, thus limiting the ability of the deep neural network to identify general usable features.
Smallest and *append gaps* proved to be unsuitable methods for input preparation. Here *append gaps* leads to a prediction of subsequences without usable information because they consist only of gaps, whereas *smallest* limits the available information so much that a prediction becomes inaccurate.

Combining subsequence host predictions results in higher accuracy

Among the tested subsequence prediction combination approaches, *std-div* was observed to perform best (see Table 5). With the combination of all subsequence predictions the accuracy rises between 1.7 %–6 %, with a mean increase of 4.8 %. This shows that a combination of the host prediction results of all subsequences of a given viral sequence can increase the overall prediction accuracy. Presumably, the prediction combination approaches can compensate for the possible information loss, caused by the sequence splitting process during the input preparation.

Table 5. Evaluation of the subsequence host prediction combination methods on the investigated datasets and the respective best working input preparation. The combination method with the highest accuracy for each combination is marked in grey.

combination method vs training setup	Standard	Voting	Mean	Std-div
LSTM *rotavirus A*, normal repeat gaps	85.83	87.50	86.67	87.50
CNN+LSTM *rotavirus A*, random repeat	81.30	85.00	85.00	85.83
LSTM *rabies lyssavirus*, random repeat	74.02	79.21	80.00	80.00
CNN+LSTM *rabies lyssavirus*, random repeat	71.98	77.11	77.63	77.63
LSTM *influenza A*, random repeat	44.20	47.85	49.18	49.29
CNN+LSTM *influenza A*, normal repeat gaps	43.53	47.35	49.39	49.39
mean accuracy	66.81	70.67	71.31	71.61

Deep learning outperforms other approaches

Besides evaluating our deep learning approach on the three virus datasets, we compared our results with a similar study. Our approach predicts hosts on the species level, whereas most other studies are limited to predicting the host genera[4;6] or even higher taxonomic groups[3;12]. In a relatively comparable study, Le *et al.*[5] also tried to predict potential hosts for influenza A viruses and rabies lyssaviruses. In their study they mainly tested three different approaches, which were mostly combinations of already published methods[4;8;12], including a support vector machine approach and two sequence similarity approaches. They tried to predict hosts on a species level, too.

The rabies lyssavirus dataset from Le *et al.* consisted of 148 viruses and 19 associated bat hosts species. For this dataset the group reached an accuracy of nearly 80 %. Our rabies lyssavirus

dataset consists of 13,476 viruses and has 19 associated hosts species, too, but none of which is a bat species. In comparison, with an accuracy of around 80 % our approach performed very similar , however, in contrast to our analysis, Le *et al.* used an unbalanced dataset, which often leads to an overestimation of the prediction accuracy. Furthermore, the presented accuracy from Le *et al.* is based on an n-fold cross-validation, making more difficult to compare with other studies, since the quasi standard for accuracy determination is a 10-fold cross-validation. When applying a 10-fold cross-validation, their host prediction accuracy for their rabies lyssavirus dataset drops under 65 %.

The influenza A dataset from Le *et al.* consisted of 1,200 viral sequences and six associated hosts species. For this dataset they reached a host prediction accuracy of around 60 % (under 40 % when applying a 10-fold cross-validation). Our influenza A dataset consists of 310,945 viral sequences and has 49 associated hosts species, including the six species from the Le *et al.* dataset. Our deep learning approach reached a host prediction accuracy of 49.39 %, which is in comparison quite good, given that we had to predict for more than eight times the number of potential host species with closer phylogenetic relationships among them.

For the rotavirus A dataset no comparable study could be found.

CONCLUSION

In this study, we investigated the usability of deep learning for the prediction of hosts for distinct viruses, based on the viral nucleotide sequences alone. We established a simple but very capable prediction pipeline, including possible data preparation steps, data training strategies and a suitable deep neural network architecture. In addition, we provide different three neural network models, which are able to predict potential hosts for either influenza A viruses, rotaviruses A or rabies lyssaviruses, respectively. These deep neural networks use genomic fragments shorter than 400 nucleotides to predict potential virus hosts directly on a species level. In contrast to similar approaches this is a more complex task than performing host prediction only on the genera level[4;6] or even higher taxonomic groups[3;12]. Moreover, our approach is able to predict more hosts with an comparable accuracy than previous approaches and more accurate when limited to the same amount of potential host species. Additionally, we addressed multiple problems that arise when using DNA or RNA sequences as input for deep learning, such as unbalanced datasets for training and the problem of inefficient learning of recurrent neural networks (RNN) on long sequences. We evaluated different solutions to solve these problems and observed that splitting of the original virus genome sequence in a nearly random way in combination with merging the prediction results of the generated subsequences leads to a fast and efficient learning on long sequences. Furthermore the use of unbalanced datasets is possible if a new balanced training set is generated by random selection of available sequences for each single epoch during the training phase.

With the use of deep neural networks for host predicting of viruses it is possible to rapidly identify the host, without the use of arbitrary selected learning features, for a large number of host species. This allows to identify the original host of zoonotic events and makes it possible to swiftly limit the intensity of an viral outbreak by separating the original host from humans or livestock.

In future approaches it could be interesting to investigate the use of newly developed deep neural network layers, such as transformer self-attention layers[27]. This layer type has been

shown to perform well with character sequences[28], such as DNA or RNA sequences, potentially allowing for a furthermore increase in the prediction quality.

Author Contributions

Conceptualization: FM, AV and MM . Data curation: FM . Formal analysis: FM . Data interpretation: FM . Methodology: FM, AV and MM . Validation: FM . Visualization: FM . Writing – Original Draft Preparation: FM . Writing – Review & Editing: EB . Project Administration: MM . Supervision: MM . Funding acquisition: MM .

Funding

This work was supported by DFG TRR 124 "FungiNet", INST 275/365-1, B05 (FM,MM); and DFG CRC 1076 "AquaDiva", A06 (MM); and DFG MA 5082/7-1 (MM)

Competing Interests

The authors declare no competing interests.

Data availability

The trained models for the prediction of the host for the viruses influenza A virus, rabies lyssavirus, rotavirus A are implemented in the tool VIDHOP. This tool is freely available under https://github.com/flomock/vidhop.
Supplementary data is available at open science framework at https://osf.io/382pd/.

References

[1] Almudena Marí Saéz, Sabrina Weiss, Kathrin Nowak, Vincent Lapeyre, Fee Zimmermann, Ariane Düx, Hjalmar S Kühl, Moussa Kaba, Sebastien Regnaut, Kevin Merkel, et al. Investigating the zoonotic origin of the west african ebola epidemic. *EMBO molecular medicine*, 7(1):17–23, 2015.

[2] Ben Longdon, Michael A Brockhurst, Colin A Russell, John J Welch, and Francis M Jiggins. The evolution and genetics of virus host shifts. *PLoS pathogens*, 10(11):e1004395, 2014.

[3] Christine L P Eng, Joo Chuan Tong, and Tin Wee Tan. Predicting host tropism of influenza a virus proteins using random forest. *BMC Med Genomics*, 7 Suppl 3:S1, 2014.

[4] Mengge Zhang, Lianping Yang, Jie Ren, Nathan A Ahlgren, Jed A Fuhrman, and Fengzhu Sun. Prediction of virus-host infectious association by supervised learning methods. *BMC Bioinf*, 18:60, March 2017.

[5] Han Li and Fengzhu Sun. Comparative studies of alignment, alignment-free and svm based approaches for predicting the hosts of viruses based on viral sequences. *Sci Rep*, 8(1):10032, 2018.

[6] Clovis Galiez, Matthias Siebert, François Enault, Jonathan Vincent, and Johannes Söding. WIsH: who is the host? predicting prokaryotic hosts from metagenomic phage contigs. *Bioinformatics*, 33(19):3113–3114, 2017.

[7] Robert A Edwards, Katelyn McNair, Karoline Faust, Jeroen Raes, and Bas E Dutilh. Computational approaches to predict bacteriophage-host relationships. *FEMS Microbiol Rev*, 40:258–272, March 2016.

[8] Nathan A Ahlgren, Jie Ren, Yang Young Lu, Jed A Fuhrman, and Fengzhu Sun. Alignment-free d_2^* oligonucleotide frequency dissimilarity measure improves prediction of hosts from metagenomically-derived viral sequences. *Nucleic Acids Res*, 45:39–53, January 2017.

[9] Jeffery K Taubenberger and John C Kash. Influenza virus evolution, host adaptation, and pandemic formation. *Cell Host Microbe*, 7(6):440–451, 2010.

[10] Sergio M Villordo, Claudia V Filomatori, Irma Shez-Vargas, Carol D Blair, and Andrea V Gamarnik. Dengue virus rna structure specialization facilitates host adaptation. *PLoS Pathog*, 11:e1004604, January 2015.

[11] GV Puskorius and LA Feldkamp. Truncated backpropagation through time and kalman filter training for neurocontrol. In *Proceedings of 1994 IEEE International Conference on Neural Networks (ICNN'94)*, volume 4, pages 2488–2493. IEEE, 1994.

[12] A Kapoor, P Simmonds, W I Lipkin, S Zaidi, and E Delwart. Use of nucleotide composition analysis to infer hosts for three novel picorna-like viruses. *Journal of virology*, 84:10322–10328, October 2010.

[13] Emma C Teeling, Sonja C Vernes, Liliana M Dávalos, David A Ray, M Thomas P Gilbert, Eugene Myers, and Bat1K Consortium. Bat biology, genomes, and the bat1k project: To generate chromosome-level genomes for all living bat species. *Annual review of animal biosciences*, 6:23–46, 2018.

[14] J Pagel Van Zee, NS Geraci, FD Guerrero, SK Wikel, JJ Stuart, VM Nene, and CA Hill. Tick genomics: the ixodes genome project and beyond. *International journal for parasitology*, 37(12):1297–1305, 2007.

[15] European Molecular Biology Laboratory. European nucleotide archive `ebi.ac.uk/ena`, 2017. [Online; Stand 18. Oktober 2017].

[16] J. Craig Venter Institute Northrop Grumman Health IT and Vecna Technologies. Virus pathogen resource `viprbrc.org/`, 2017. [Online; Stand 18. Oktober 2017].

[17] National Center for Biotechnology Information. Influenza virus database `ncbi.nlm.nih.gov/genomes/FLU/`, 2017. [Online; Stand 18. Oktober 2017].

[18] Doyen Sahoo, Quang Pham, Jing Lu, and Steven C. H. Hoi. Online deep learning: Learning deep neural networks on the fly. *CoRR*, abs/1711.03705, 2017.

[19] Ilya Sutskever. Training recurrent neural networks. *University of Toronto, Toronto, Ont., Canada*, 2013.

[20] François Chollet et al. Keras. `https://keras.io`, 2015.

[21] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[22] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[23] Paul J Werbos et al. Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*, 78(10):1550–1560, 1990.

[24] Corentin Tallec and Yann Ollivier. Unbiasing truncated backpropagation through time. *arXiv preprint arXiv:1705.08209*, 2017.

[25] Manuel P Cuéllar, Miguel Delgado, and MC Pegalajar. An application of non-linear programming to train recurrent neural networks in time series prediction problems. In *Enterprise Information Systems VII*, pages 95–102. Springer, 2007.

[26] V Martella, Krisztián Bányai, Jelle Matthijnssens, Canio Buonavoglia, and Max Ciarlet. Zoonotic aspects of rotaviruses. *Veterinary microbiology*, 140(3-4):246–255, 2010.

[27] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008, 2017.

[28] Rami Al-Rfou, Dokook Choe, Noah Constant, Mandy Guo, and Llion Jones. Character-level language modeling with deeper self-attention. *arXiv preprint arXiv:1808.04444*, 2018.

SUPPLEMENTARY DATA

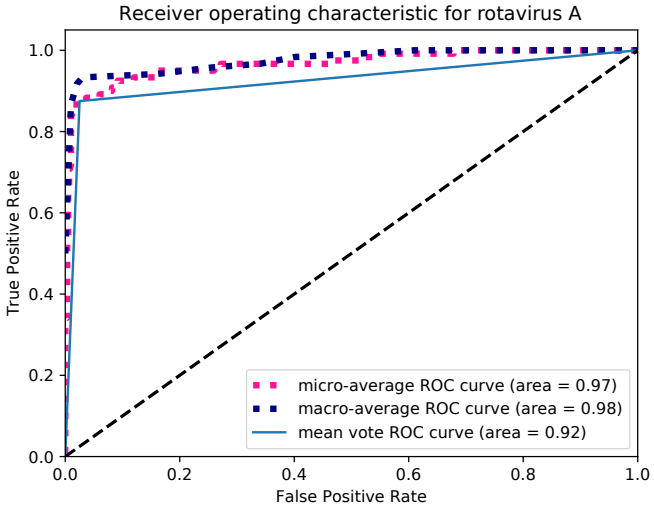

Figure S1. ROC curve for the rota dataset, calculated on the test set. The AUC of the micro-average ROC curve is 0.98, for the macro-average 0.98 and when using mean vote to predict to most likely host the AUC is 0.92.

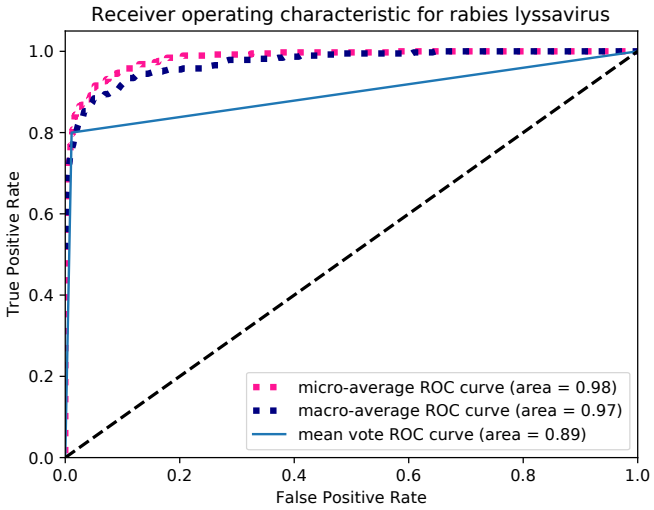

Figure S2. ROC curve for the rabies dataset, calculated on the test set. The AUC of the micro-average ROC curve is 0.98, for the macro-average 0.98 and when using mean vote to predict to most likely host the AUC is 0.89.

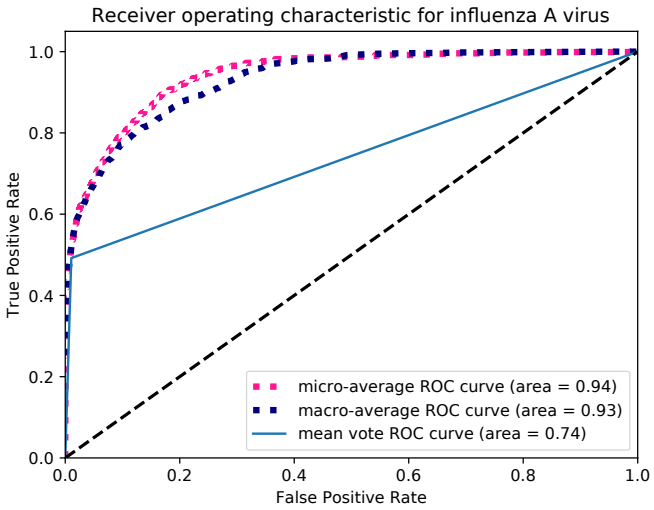

Figure S3. ROC curve for the influenza dataset, calculated on the test set. The AUC of the micro-average ROC curve is 0.94, for the macro-average 0.93 and when using mean vote to predict to most likely host the AUC is 0.74.

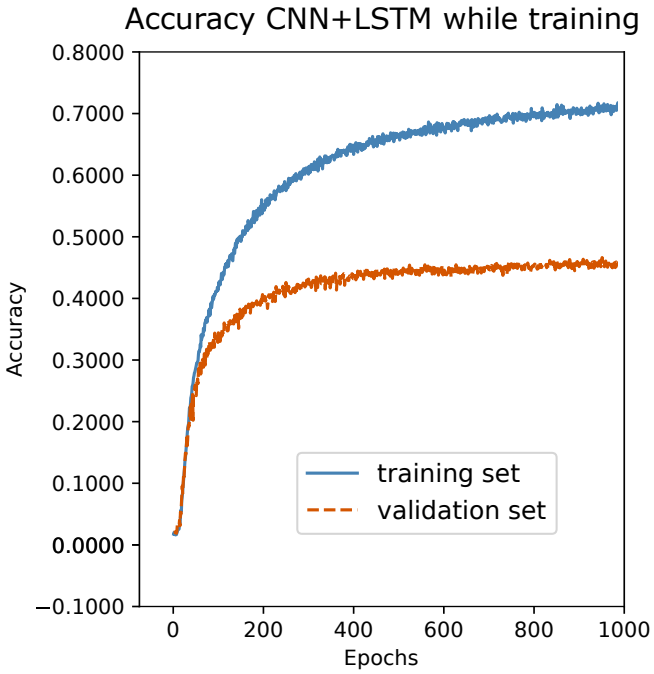

Figure S4. Training accuracy and validation accuracy over 1000 epochs on the influenza A dataset. With the CNN+LSTM architecture the deep neural network shows no difficulties to learn.

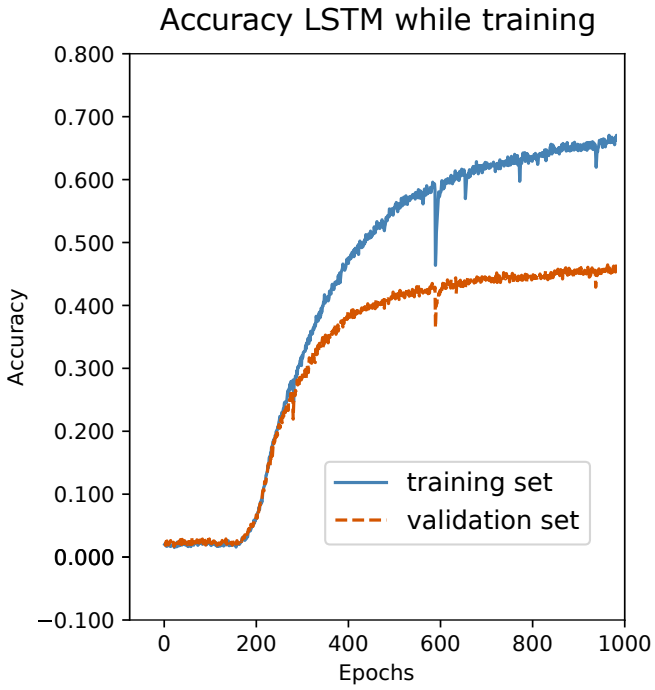

Figure S5. Training accuracy and validation accuracy over 1000 epochs on the influenza A dataset. With the LSTM architecture the deep neural network remained in a state of random accuracy for ca. 200 epochs.

Table S1. Comparison of the functional principle of the input expansion. Note that for real data, the raw data sequence would be hundreds of bases long. Furthermore, in this example ACT is the shortest sequence of our dataset, but still longer than the input length expected by the neural network.

input sequence	normal repeat	normal repeat gaps	random repeat	random repeat gaps	append gaps	online	smallest
ACT	ACTACTAC	ACT-ACT-	ACTCTATA	ACT-CTA-	ACT——	CT	ACT