IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received May 18, 2019, accepted June 9, 2019, date of publication June 19, 2019, date of current version July 3, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2923687

Viral Genome Deep Classifier

ANNA FABIJAŃSKA AND SZYMON GRABOWSKI
Institute of Applied Computer Science, Lodz University of Technology, 90-924 Lodz, Poland

Corresponding author: Anna Fabijańska (anna.fabijanska@p.lodz.pl)

This work was supported by the Faculty of Electrical, Electronic, Computer, and Control Engineering, Lodz University of Technology, as a statutory activity.

ABSTRACT The task of virus classification into subtypes is an important concern in many categorization studies, e.g., in virology or epidemiology. Therefore, the problem of virus subtyping has been a subject of considerable interest in the last decade. Although there exist several virus subtyping tools, they are often dedicated to a specific family of viruses. Even specialized methods, however, often fail to correctly subtype viruses, such as HIV or influenza. To address these shortcomings, we present a viral genome deep classifier (VGDC)—a tool for an automatic virus subtyping, which employs a deep convolutional neural network (CNN). The method is universal and can be applied for subtyping any virus, as confirmed by experiments on dengue, hepatitis B and C, HIV-1, and influenza A datasets. For all considered virus types, the obtained classification rates are very high with the corresponding values of the F1-score ranging from about 0.85 to 1.00 depending on the virus type and the considered number of subtypes. For HIV-1 and influenza A, the VGDC significantly outperforms the leading competitors, including CASTOR and COMET. The VGDC source code is freely available to facilitate easy usage and comparison with future approaches.

INDEX TERMS Genome, virus, subtyping, classification, convolutional neural network.

I. INTRODUCTION

Genomic sequence classification aims at assigning a given sequence into a group of already known sequences which share similar characteristics. This task is of crucial importance in many categorization studies, especially in virology and epidemiology where virus subtypes may relate to the rates of disease progression or susceptibility to drug treatments. Therefore, the problem of virus subtyping has been a subject of considerable interest in the last decade.

As a result, a number of approaches to automatic classification of viral strains into groups representing virus subtypes have been proposed. These approaches can be roughly divided into three categories, namely alignment-based, feature-based, and model-based methods [1].

The most popular representatives of the alignment-based methods for the automatic virus subtyping are SCUEAL [2], USEARCH [3] and REGA [4], [5]. They all are phylogenetic methods which rely on an initial alignment between the virus sequence being classified and the reference set. To improve classification, especially in the case of recombinants, SCUEAL utilizes the phylogenetic likelihood of mosaic structures, while REGA applies a bootstrap supported sliding window.

However, for these methods, there may still exist several possible initial alignments and selection of a particular one may adversely affect the classification result. Additionally, the application of both SCUEAL and REGA is limited mainly to HIV virus strains. Alignment-based methods are rather expensive computationally and their performance, depending on heuristically chosen parameters, may be unstable for highly variable regions of the genome.

The feature-based methods transform genomic sequences into feature vectors which are then classified into subtypes using traditional machine learning algorithms. As a result, they can be applied to any virus type. The CASTOR web platform [1] utilizes the signatures of restriction fragments length polymorphism (RFLP). Particularly, it builds feature vectors based upon the distribution of the restriction site patterns and then refines them using relevant feature selection methods. Finally, the feature vectors are inputs to popular classifiers (including, i.a., SVM and AdaBoost) which need to be tested to select the best one. Additionally, tests need to be performed on balanced datasets, i.e., having more or less the same number of instances of each subtype. A similar approach was also used in KAMERIS [6], where feature vectors expressing the respective k-mer frequencies of virus

The associate editor coordinating the review of this manuscript and approving it for publication was Xiaochun Cheng.

sequences were passed to popular supervised classifiers. The input genomic sequences need however to be preprocessed by removal of any ambiguous nucleotide codes. The frequencies of k-mer nucleotide strings were also utilized by [7], where the authors trained a simple feed-forward neural network to predict Influenza virus antigenic types and hosts. The method, however, was tested only on a limited number of virus subtypes (namely: H1, H3, and H5).

Finally, a representative of the model-based methods is COMET (COntext-based Modeling for Expeditious Typing) [8]. The method makes use of the Prediction by Partial Matching (PPM) compression scheme. In particular, it builds a variable-order Markov model for each reference sequence. Then, for all symbols of the query sequence, the likelihoods of their occurrences in each of the reference types are derived, and the final classification is obtained from a simple decision tree. Although the model performs well for pure virus types, it requires the recombinants to be treated in a special way. Additionally, it is required to adjust the best model order, window size and the threshold for recombination detection.

In parallel, the deep learning (DL) approach has demonstrated outstanding performance in the field of artificial intelligence, with prominent applications in machine vision, natural language processing, and audio signal recognition (see, e.g., [9] and references therein). Particularly, in various visual recognition tasks (including object classification, localization, and detection from digital images) convolutional architectures like LeNet-5 [10], AlexNet [11], VGG [12], ResNet [13] or GoogLeNet [14] present the accuracy nearing or even exceeding the human performance.

Starting with early applications in the '90s [15], neural networks and recently deep learning have also been gradually revolutionizing genomics [16]–[19] (see also [20], [21] for broader overviews of deep learning applications in computational biology). The existing DL research in the area of genomics mostly concentrates on two major problems. These are (*i*) genomic sequencing and gene expression analysis [22]–[25], and (*ii*) protein structure prediction [23], [26]–[28]. There also exist some attempts to classify DNA sequences with DL [29], [30]. Sample applications in this area include chromatin structure classification [31], polyadenylation site prediction [32] or classification of G protein-coupled receptors [33]. However, to the best of our knowledge, no attempts to perform virus subtyping with the application of deep learning have been reported. Therefore, this study presents the Viral Genome Deep Classifier (VGDC), the first convolutional neural network (CNN) based approach for automatic classification of viral genomes into subtypes. Our approach makes use of the fact that a genomic sequence can be perceived as a one-dimensional signal. Particularly, it accounts for positional relationships between sequence signals. Like in the case of images where the CNNs are able to detect specific combinations of pixels (i.e., patterns) that allow distinguishing between objects, in the case of genomes the CNNs are able to detect specific combinations of nucleotides that allow distinguishing between particular virus subtypes.

The proposed method is dedicated to the classification of full-length genomes. Additionally, it is universal and thus not limited to a specific family of viruses, which is not always the case among existing solutions.

II. MATERIALS AND METHODS

A. INPUT DATA

In this study virus genomic sequences retrieved from publicly available databases were used. In particular, five datasets, each representing one virus type, were considered. These datasets represent the viruses of Dengue, Hepatitis B, Hepatitis C, HIV-1, and Influenza A.

The Dengue virus sequences were downloaded from the National Center for Biotechnology Information (NCBI) virus database.[1] The same database was the source of Influenza genomes.[2] The genomic sequences of Hepatitis B were downloaded from The Hepatitis B Virus Database (HBVdb)[3] while Hepatitis C genomic sequences came from the database of Los Alamos National Laboratory (LANL).[4] The latter database was also the source of HIV-1 genomes.[5] The query options used for the sequences retrieval are summarized in Table 1.

TABLE 1. Query options used for genomic sequences retrieval.

Virus type	Database	Query options
Dengue	NCBI	**sequence type**: nucleotide, full-length sequences only, collapse identical sequences, **other options**: default
Hepatitis B	HBVdb	none
Hepatitis C	LANL	**genomic region**: complete genome, exclude recombinants, exclude problematic, exclude no genotype, **other options**: default
HIV-1	LANL	**virus**: HIV-1, **genomic region**: complete genome, **subtype**: any subtype, excluding problematic, **other options**: default
Influenza A	NCBI	**sequence type**: nucleotide, **type**: A, full-length only, collapse identical sequences, **other options**: default

Prior to the experiment, virus subtypes represented by less than 10 samples were removed from each of the considered datasets. This ensured a sufficient representation of each virus subtype in the training/testing subsets used for five-fold cross-validation used to assess the performance of our method. Additionally, in the case of Influenza A, only pure NH subtypes were considered. Mixed subtypes were disregarded and thus removed from the datasets.

[1] https://www.ncbi.nlm.nih.gov/genomes/VirusVariation/Database/nph-select.cgi?taxid=12637

[2] https://www.ncbi.nlm.nih.gov/genomes/FLU/Database/nph-select.cgi#mainform

[3] https://hbvdb.ibcp.fr/HBVdb/HBVdbDataset?seqtype=0

[4] https://hcv.lanl.gov/components/sequence/HCV/search/searchi.html

[5] https://www.hiv.lanl.gov/components/sequence/HIV/search/search.html

Additionally, for Hepatitis B virus two subsets were prepared. The first one contained only genomes representing the main subtypes (from A to H). The second subset contained also the most frequent recombinants.

In the case of the HIV-1 virus also two subsets were considered. The first one contained the 12 most frequent virus subtypes. In the second subset, all data (remaining after filtering as described above) were used. A similar procedure was also applied to Influenza A genomic sequences with the difference that the first subset contained 56 most frequent subtypes.

The properties of the resulting datasets are summarized in Table 2.

TABLE 2. The datasets summary.

Virus	No. subtypes	No. genomes	Min. genome length	Max. genome length
Dengue	4	5079	10161	11195
Hepatitis B (1)	8	6138	3182	3257
Hepatitis B (2)	13	6824	3182	3257
Hepatitis C	9	2068	24751	24751
HIV-1 (1)	12	6540	19685	24307
HIV-1 (2)	37	7194	19685	24307
Influenza A (1)	56	313782	173	2867
Influenza A (2)	113	317728	173	2867

B. DATA PREPROCESSING

The proposed VGDC approach does not require a sophisticated preprocessing of the input genomes. However, two important steps were performed prior to network training and prediction (classification).

First, symbols (letters) representing nucleotides were replaced by the corresponding ASCII codes. To this end, *A*, *C*, *G*, *T* were replaced by integers 65, 67, 71 and 84, respectively. A similar operation was also applied to other symbols occurring in the genome sequences (e.g., *N* or -). The ASCII codes were used instead of popular one-hot encoding for efficiency and flexibility reasons. Particularly, for the proposed ASCII-based encoding one value per nucleotide is required compared to six (or more) elements vector as in the case of one-hot encoding. Additionally, this encoding can easily handle both capital and uppercase letters used for nucleotide representation.

Second, the length of each genome was extended to the length of the longest genome among the genomes of a particular virus. The extension was performed by appending numeric zeros to the sequence of ASCII codes representing nucleotides. This step was required by CNN since it accepts only fixed-size inputs.

C. CNN ARCHITECTURE

Popular convolutional architectures for visual recognition (eg., LeNet [10], AlexNet [11], GoogLeNet [14], ResNet [13] or VGG [12]) cannot be straightforwardly applied in the genomics. It is because of the difference in data dimensionality (one dimension in the case of genomes vs. two dimensions in the case of images). Therefore, a new architecture was proposed for the considered virus subtyping problem.

The general architecture of the convolutional neural network behind the proposed VGDC approach for genome classification is presented in Figure 1. This is a convolutional encoder model whose inputs are the preprocessed genomes (see Sect. II-B). The length n of CNN inputs is a parameter of the model and thus can vary depending on the virus type. However, for a particular dataset, it is equal to the length of the longest genome.

The model outputs a vector $\mathbf{P}$ of size N where each element $P_i \in \mathbf{P}$ $(1 \leq i \leq N)$ corresponds to the probability that the genome belongs to the i-th class, where $i < N$, and N is the total number of classes (i.e., viral subtypes) in the given problem.

In the model, there is a total of 30 layers divided into the convolutional part dedicated for feature extraction (layers 1–19) and the classifier which aims at predicting the genome subtype based on the features determined by the convolutional part.

Our model (see Fig. 1) starts with five repeated 1D convolution layers, each followed by the ReLU activation. The convolution layers convolve an input signal (representing a genome) with a set of learnable filters which are slid across a genome. These filters are used to detect the specific patterns in the input genomes. The filters' coefficients are learned through network training. In the proposed solution, convolutions are performed with the filters of size w, which is also a parameter of the proposed model and can be adjusted depending on the genome length. The number of filters in convolution layers increases by a factor of two from eight in the first convolution layer to 128 in the fifth one.

A batch normalization follows each convolution layer and precedes the ReLU (Rectified Linear Unit) activation layer. This is a standard procedure in convolutional neural networks used to improve training performance [34].

The ReLU activation layers that follow next apply an element-wise operation $max(0, x)$ to the resulting feature maps. This operation introduces non-linearity but also reduces overfitting. The ReLU activation was used since in the case of convolutional neural networks it has proven to perform better than sigmoid or tanh activation functions. Particularly, ReLu reduces the vanishing gradient problem, avoids backpropagation errors, and is much faster, especially when compared to sigmoid.

The ReLU is nowadays the most popular activation function, applied in almost all existing convolutional neural networks or deep learning.

The ReLU activation layers are next followed by the 1D pooling layers which perform a max-pool operation with the pool of size 2 and stride of size 2. Particularly, from the sliding window of size 2 and moved by the step (stride) of 2 the maximum value is taken. Thus pooling reduces the genome length but also extracts characteristic genomic features and

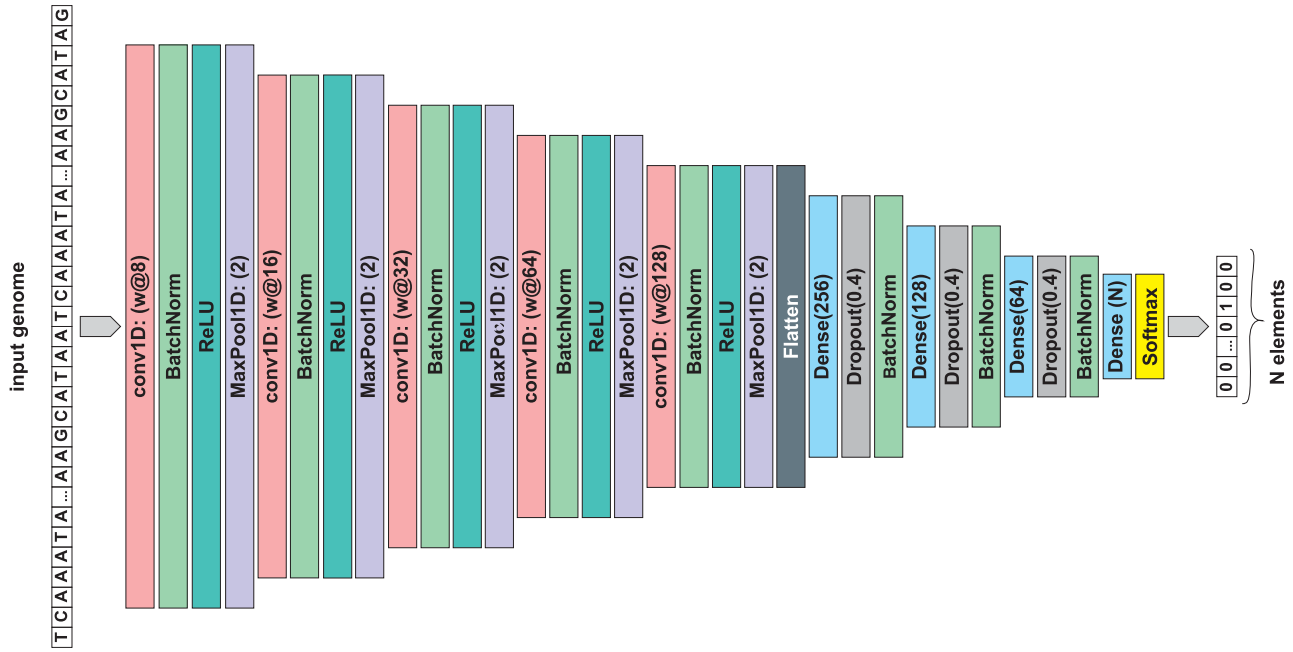


FIGURE 1. The architecture of the CNN behind the Viral Genome Deep Classifier, where: Conv1D (*w*@*n*) - 1D convolutional layer with the filter size of *w*, and a number of filters equal to *n*; BatchNorm - batch normalization layer; ReLU - Rectified Linear Unit activation; MaxPool1D (2) - 1D max pooling layer with a pool size of 2.

propagates them to the dense layers. The output of the last max-pooling layer is flattened (i.e., transformed) into a 1D feature vector and used as an input to the classifier part of the model.

In the classifier part of the model, there are four dense layers with the decreasing number of neurons. These range from 256 neurons in the first dense layer, through 128 and 64 neurons in two following dense layers, to N neurons in the last dense layer, where N is the number of virus subtypes to be recognized. Except for the last one, the dense layers are followed by the dropout layer and the batch normalization layer. The last layer is the softmax activation which outputs the probability of a genome sequence to belong to each class. Finally, the genome is classified as the subtype for which the probability is the highest.

The architecture of VGDC, common for all shown experiments, was deployed via trial and error.

Different numbers of layers (network depths) were tested balancing between the universality, network performance and training time. Particularly, it was observed that the optimal depth of the architecture depends on the genome length. For the shortest genomes (i.e., Influenza A and Hepatitis B) shallower architectures were sufficient.

More concretely, two pairs of convolutional/pooling layers used for feature extraction allowed to obtain high classification rates. Additional convolutional layers did not influence the resulting accuracy but increased the training time. However, a shallow network was insufficient for classification of longer genomes (i.e., Dengue, HIV-1, and Hepatitis C). In such a case two pairs of convolutional/pooling layers allowed classifying about 50% of the samples. Increasing the network depth up to 5 pairs of convolutional/pooling layers significantly increased the network performance. Deeper architectures did not exhibit improvement in the resulting accuracy.

D. TRAINING PARAMETERS

To assess the performance of the proposed deep learning approach, for each of the considered datasets of viral genomes, five-fold cross-validation was performed (with 80% of samples used for training and the remaining 20% used for validation). In each experiment, the network was trained for 1000 epochs unless the early stopping condition was fulfilled. Particularly, the training was stopped after 10 successive epochs with no training improvement. Usually, the training converged at after about 200 epochs.

Adam optimizer [35] with the learning rate of 0.002 was used to minimize the categorical cross-entropy loss function. To measure the performance of the model, the mean squared error was used. The batch size was equal to 50 except for the Influenza A virus. In the latter case, the batch size of 1000 was used due to a large number of training samples. The above hyperparameters of the model were set in a trial-and-error manner balancing between the training time and the training performance.

The values of CNN parameters, particularly the size of the filters w, were selected with respect to the genome length. In the case of the shortest genomes, namely Hepatitis B and Influenza A, w was set to 7. This was the maximum filter size that could be applied for the CNN architecture behind the VGDC. In the case of longer genomes, namely Dengue, Hepatitis C and HIV-1, the filter size w was increased to 9 to

better capture genome spatial features. The size n of the CNN input was equal to the length of the longest genome within each considered virus family. Finally, the size N of the vector output by the CNN was equal to the number of virus subtypes to be predicted (see column *No. classes* in Tab. 2).

E. EXPERIMENTAL SETUP

The proposed CNN was implemented in Python 3.6, with the Keras library running on top of Theano. The source code of our method is available in the GitHub repository https://github.com/afabijanska/VGDC. The experiments were performed on a desktop computer (i7-960 3.2 GHz CPU, 24 GB RAM) with GeForce GTX TITAN X GPU equipped with 12 GB of DDR5 RAM.

The training time varied depending on the virus type, particularly the genome length and the number of training samples (see Tab. 3). However, since the training is performed once, the relatively high training times are not problematic. The classification took less than one second per genome.

TABLE 3. Average time per epoch of training.

virus type	max. gen. length	no. samples	time [s]
Dengue	11195	4064	6
Hepatitis B (1)	3257	4911	2
Hepatitis B (2)	3257	5459	5
Hepatitis C	24751	1496	10
HIV-1 (1)	24307	5734	11
HIV-1 (2)	24307	5755	11
Influenza A (1)	2867	251026	151
Influenza A (2)	2867	254136	153

III. RESULTS AND DISCUSSION

The performance of the proposed approach was assessed using popular classification performance metrics including sensitivity (SEN), specificity (SPEC), precision (PREC), accuracy (ACC) and F1-score (F1) given by Equations 2– 5 respectively.

$$SEN = \frac{TP}{TP + FN} \tag{1}$$

$$SPEC = \frac{TN}{TN + FP} \tag{2}$$

$$PREC = \frac{TP}{TP + FP} \tag{3}$$

$$ACC = \frac{TP + TN}{TP + TN + FP + FN} \tag{4}$$

$$F1 = \frac{2\,TP}{2\,TP + FP + FN} \tag{5}$$

where TP, TN, FP, and FN stand for true positives, true negatives, false positives, and false negatives, respectively.

To alleviate an unbalanced number of instances of virus subtypes the above metrics were obtained for each class and then weighted by the number of instances to obtain the overall classification score. A similar procedure was also used by the authors of a competitive approach to virus subtyping [1].

To compare the proposed algorithm against known solutions, we selected four competitors. Two of them, CASTOR and COMET, are well established state-of-the-art approaches described in Section I. Alignment and phylogenetic-based methods, like USEARCH [3] and REGA [4], were excluded from the comparison since they have been proven to perform noticeably worse than COMET and CASTOR (see [8] and [1] for comparison). Two other methods considered in the assessment, namely C-measure and the compression-based approach, are well-known in the area of textual similarity matching.

C-measure [36] makes use of k-mers. For a query sequence $Q_{1\ldots m}$ it checks its $m-k+1$ k-mers (i.e., $Q_{1\ldots k}$, $Q_{2\ldots k+1}$,..., $Q_{m-k+1\ldots m}$) for occurrence in each of the training sequences. The class of the training sequence which maximizes the number of query k-mers occurring in it is the output label (with ties resolved arbitrarily). In our solution, the said training sequences are concatenations of all training genomes from a given class. For example, in the Dengue dataset, there are only four such training sequences. Using whole classes rather than individual genomes makes the classification much faster.

The last algorithm uses a compression-based approach to text classification. Given c training sequences T_i, $1 \leq i \leq c$, and a query Q, the class label j, $1 \leq i \leq c$, is chosen, such that the compressed size of the concatenation $T_j \circ Q$ minus the compressed size of T_j is minimized. In other words, as the training sequence T_j forms the most helpful context for the compression of Q, it can also be seen as the sequence most similar to Q. This idea has been successfully used for authorship attribution and text classification [37], [38], but possibly its oldest incarnation was applied to DNA sequence classification [39], namely for distinguishing bacterial promoters from non-promoters and for recognizing non-bacterial splice-junction sites in protein-coding regions of DNA. In our experiments, the used compression method was PPMTrain (http://compression.ru/ds/ppmtrain.rar), a variant of PPMd by Dmitry Shkarin, in which the PPM compressor is first trained on a given file and then used to compress another file. We ran it with -o8 -m32 switches, as in the example given in the documentation.

Both CASTOR and COMET were tested through the web interfaces available at http://castor.bioinfo.uqam.ca/ and https://comet.lih.lu/, respectively. The C-measure classifier and the compression-based approach were implemented for the evaluation purposes.

Wherever possible, the considered methods were evaluated via five-fold cross-validation. It means the whole dataset was randomly partitioned into five equal parts (''folds'') and each part, in turn, became the testing data while the remaining four parts constituted the training data.

The exception is COMET, which is both limited to HIV and Hepatitis C viruses, and cannot be trained with any new data since this functionality is unincluded in the COMET web interface. Despite efforts, we were unable to obtain a standalone Java jar file to train the method with any data (as mentioned in the COMET source paper). Therefore, for the

experiments regarding COMET, we used the already trained (available on-line) classifier to predict subtypes of our testing folds.

In the case of CASTOR, the classifier performing best was selected for comparison. However, due to limitations imposed by the web interface (particularly the limited size of the training data allowed for upload), it was impossible to repeat the experiment for the Influenza A virus for the whole training folds. Therefore, the validation was performed with the training folds reduced randomly by a factor of three. For comparison purposes, the same experiment was carried out for the proposed VGDC approach. For both experiments, the evaluation was performed with respect to complete (i.e., not reduced) testing folds.

The Influenza A dataset was excluded from testing in the case of the compression-based approach (PPMT) due to a heavy disk load, particularly a large number of temporary files that need to be created and then removed. More precisely, the number of files produced in a PPMT session is equal to the number of test sequences multiplied by the number of classes, a product whose value may easily go into millions, cf. Table 2.

The results of the above experiments are summarized in Table 4. For each of the considered virus type, the weighted average sensitivity (SEN), i.e., true positive rate, specificity (SPEC), i.e. true negative rate, precision (PREC), accuracy (ACC) and F1-score (F1) are presented. The results obtained for Influenza A using the training data reduced by a factor of three are denoted by an asterisk. For each experiment, the results exhibiting the highest metric values are given in bold. Additionally, the confusion matrices of the proposed VGDC approach (summed up for all five folds) are presented in Appendix.

The results presented in Table 4 clearly show that the proposed method performs reasonably well for all five considered virus families. The performance, however, varies slightly depending on the virus type and the number of corresponding subtypes.

The best results were obtained for Dengue, where four pure virus subtypes were considered. In this case, VGDC correctly classified all the genomes. All considered classification performance metrics reaching 1 were also obtained for an experiment with eight pure subtypes of Hepatitis B where the number of misclassified genomes on average did not exceed two per fold. A similar number of classification errors appeared in the case of Hepatitis C dataset what also resulted in specificity nearing 1 and remaining metrics equal to 0.996.

The performance of VGDC slightly decreased with the increasing number of subtypes. For the Hepatitis B (2) dataset including genomes of both pure subtypes and retro-transcribing viruses on average 62 genomes out of 1364 were misclassified what resulted in the F1-score at the level of 0.952 and specificity nearing 0.990.

Moderately higher scores (i.e., F1-score equal to 0.955 and specificity at the level of 0.994) were obtained for the extended HIV-1 (2) dataset which included both pure subtypes and recombinants resulting in total in 37 subtypes. In this case on average, about 59 genomes out of 1433 were misclassified. On the HIV-1 (1) dataset containing 56 most frequent HIV-1 subtypes, VGDC performed visibly better, which manifested by the significantly higher value of F1-score (i.e., 0.978). For this dataset, on average, 28 misclassifications per testing 1308 genomes were registered in each fold. The worst classification results were obtained for the Influenza A datasets with the specificity and accuracy

TABLE 4. Classification performance metrics yielded by various virus subtyping methods. Results obtained on the training set reduced by a factor of three are marked by *.

Virus	SEN				
	VGDC	CASTOR	COMET	C-Measure	PPMT
Dengue	**1.000**	**1.000**	N/A	**1.000**	**1.000**
Hep. B(1)	0.999	**1.000**	N/A	**1.000**	**1.000**
Hep. B(2)	**0.954**	0.949	N/A	**0.954**	0.953
Hep. C	**0.996**	**0.996**	0.958	**0.996**	0.953
HIV-1(1)	**0.979**	0.942	0.904	0.951	0.973
HIV-2(2)	**0.960**	0.912	0.864	0.919	0.942
Infl. A(1)	**0.846**	N/A	N/A	0.829	N/A
Infl. A(1)*	**0.847**	0.811	N/A	0.829	N/A
Infl. A(2)	**0.841**	N/A	N/A	0.824	N/A
Infl. A(2)*	**0.849**	0.803	N/A	0.821	N/A
Virus	SPEC				
	VGDC	CASTOR	COMET	C-Measure	PPMT
Dengue	**1.000**	**1.000**	N/A	**1.000**	**1.000**
Hep. B(1)	**1.000**	**1.000**	N/A	**1.000**	**1.000**
Hep. B(2)	**0.988**	0.987	N/A	0.987	**0.988**
Hep. C	0.999	**1.000**	0.984	**1.000**	0.988
HIV-1(1)	**0.993**	0.985	0.964	0.973	0.988
HIV-2(2)	**0.994**	0.988	0.970	0.970	0.988
Infl. A(1)	**0.983**	N/A	N/A	0.961	N/A
Infl. A(1)*	**0.981**	**0.981**	N/A	0.961	N/A
Infl. A(2)	**0.982**	N/A	N/A	0.963	N/A
Infl. A(2)*	**0.985**	0.981	N/A	0.967	N/A
Virus	PREC				
	VGDC	CASTOR	COMET	C-Measure	PPMT
Dengue	**1.000**	**1.000**	N/A	**1.000**	**1.000**
Hep. B(1)	0.999	**1.000**	N/A	**1.000**	**1.000**
Hep. B(2)	**0.953**	0.945	N/A	0.952	0.950
Hep. C	**0.996**	**0.996**	0.962	**0.996**	0.950
HIV-1(1)	**0.978**	0.940	0.862	0.954	0.974
HIV-2(2)	**0.956**	0.907	0.783	0.915	0.937
Infl. A(1)	**0.844**	N/A	N/A	0.839	N/A
Infl. A(1)*	**0.845**	0.817	N/A	0.839	N/A
Infl. A(2)	**0.835**	N/A	N/A	0.833	N/A
Infl. A(2)*	**0.848**	0.802	N/A	0.824	N/A
Virus	ACC				
	VGDC	CASTOR	COMET	C-Measure	PPMT
Dengue	**1.000**	**1.000**	N/A	**1.000**	**1.000**
Hep. B(1)	0.999	**1.000**	N/A	**1.000**	**1.000**
Hep. B(2)	**0.987**	0.986	N/A	**0.987**	**0.987**
Hep. C	0.999	0.999	0.984	**1.000**	0.987
HIV-1(1)	**0.995**	0.984	0.975	0.984	0.992
HIV-2(2)	**0.995**	0.985	0.976	0.980	0.990
Infl. A(1)	**0.977**	N/A	N/A	0.951	N/A
Infl. A(1)*	**0.977**	**0.977**	N/A	0.951	N/A
Infl. A(2)	**0.978**	N/A	N/A	0.953	N/A
Infl. A(2)*	**0.979**	0.977	N/A	0.959	N/A
Virus	F1				
	VGDC	CASTOR	COMET	C-Measure	PPMT
Dengue	**1.000**	**1.000**	N/A	**1.000**	**1.000**
Hep. B(1)	0.999	**1.000**	N/A	**1.000**	**1.000**
Hep. B(2)	**0.952**	0.945	N/A	0.949	0.949
Hep. C	**0.996**	**0.996**	0.957	**0.996**	0.949
HIV-1(1)	**0.978**	0.940	0.870	0.943	0.971
HIV-2(2)	**0.955**	0.909	0.816	0.904	0.935
Infl. A(1)	**0.842**	N/A	N/A	0.827	N/A
Infl. A(1)*	**0.843**	0.811	N/A	0.827	N/A
Infl. A(2)	**0.835**	N/A	N/A	0.821	N/A
Infl. A(2)*	**0.847**	0.802	N/A	0.817	N/A

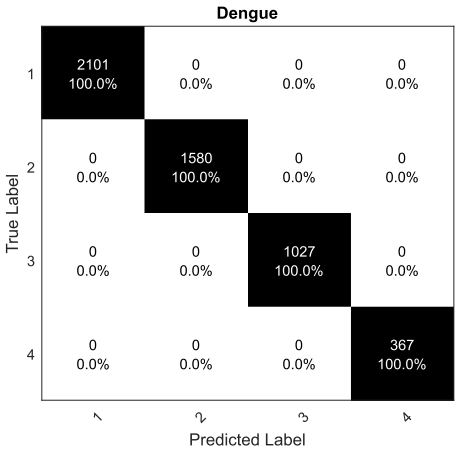


FIGURE 2. Confusion matrix obtained for the dengue dataset.

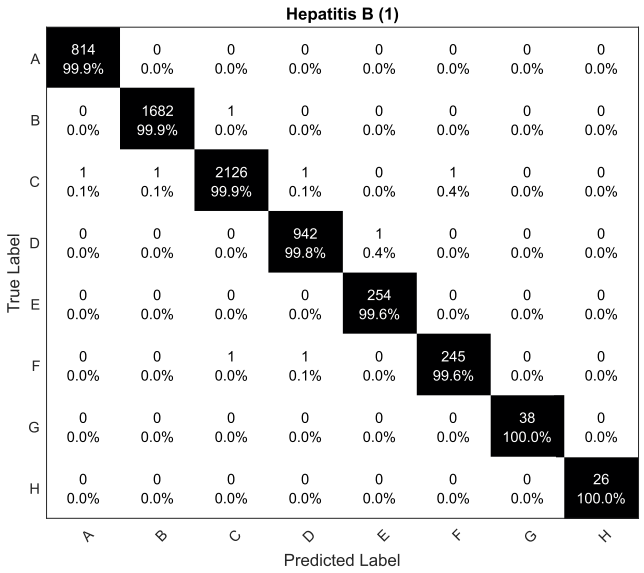


FIGURE 3. Confusion matrix obtained for the hepatitis B (1) dataset.

around 0.98 and other metrics nearing but not exceeding 0.85. This dataset, however, was the most challenging since it contained the largest number of virus subtypes equal to 56 and 113 for the basic (i.e., Influenza A (1)) and the extended (i.e., Influenza A (2)) dataset respectively.

When compared to the other approaches, in most cases VGDC is the winner. This manifests by the resulting values of all the considered classification performance metrics which for the proposed approach are the highest for most of the cases. However, the differences in the VGDC performance vary depending on the dataset and thus are related to the considered family of viruses.

On the Dengue data, all the considered methods performed equally well resulting in 100% correct classification. On the Hepatitis C dataset, VGDC performed equally well as CASTOR and C-Measure, visibly outperforming COMET and PPMT by means of all considered metrics and achieving F1 score at the level of 0.996.

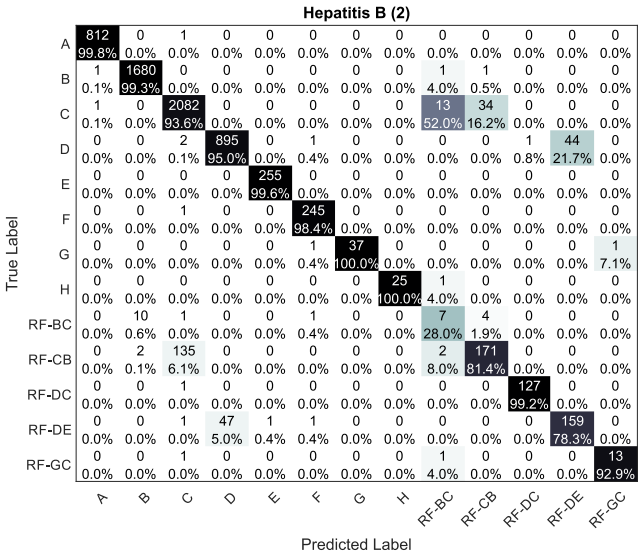


FIGURE 4. Confusion matrix obtained for the hepatitis B (2) dataset.

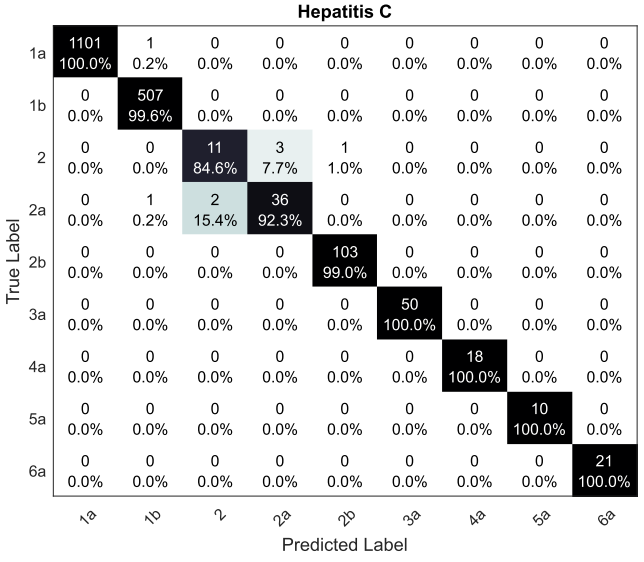


FIGURE 5. Confusion matrix obtained for the hepatitis C dataset.

Specificity equal 1 and other metrics nearing or equal 1 were equally obtained by all considered methods on the Hepatitis B (1) dataset containing the genomes of eight pure subtypes. However, for the extended Hepatitis B (2) dataset including both pure subtypes and recombinants, the proposed approach outperformed the other competitors. Although sensitivity, specificity and accuracy of VGDC are occasionally equal to their counterparts exhibited by some competitors, the F1-score determined for the results of the VGDC was the highest. Thus, the advantage of VGDC manifests when the recombinants need to be classified. This is especially visible when the HIV-1 virus is considered. For the twelve most frequent subtypes all considered classification performance metrics of VGDC were higher than the corresponding scores obtained by the competitors including CASTOR and

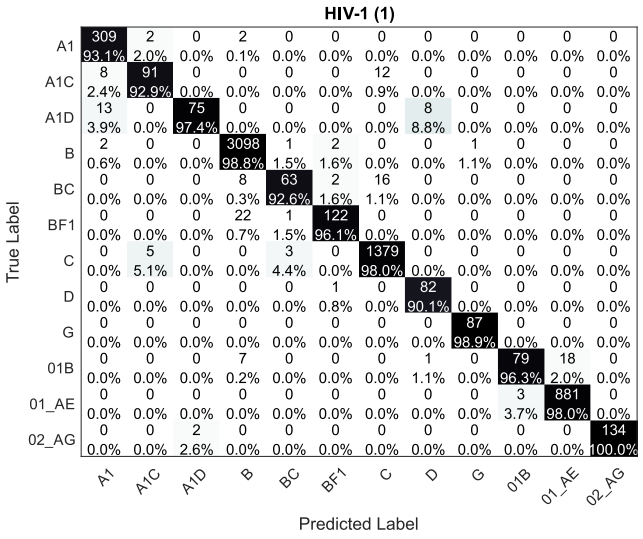
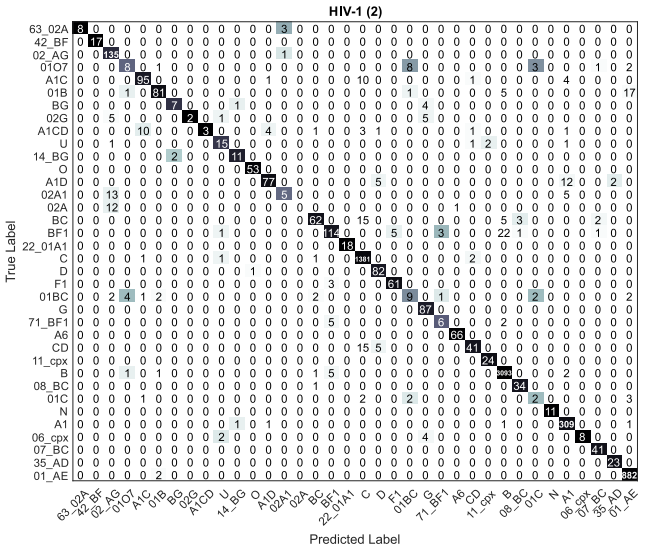
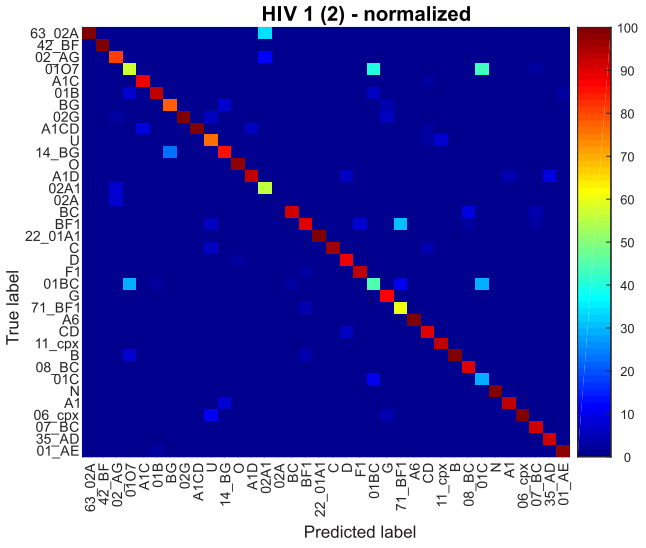


FIGURE 6. Confusion matrix obtained for the HIV-1 (1) dataset.



(a)



(b)

FIGURE 7. Confusion matrix obtained for the HIV-1 (2) dataset; a) original; b) normalized.

COMET which are the leading methods for subtyping of the HIV-1 virus. The proposed approach correctly classified 3.9% more samples than CASTOR and 8.3% more samples than COMET. The classification rate was also by 2.9% higher than for C-measure and 0.6% than for PPMT. The superiority of VGDC is even more visible when more HIV-1 subtypes are considered. In such a case the difference in the classification rates ranges from 11.1% in the case of COMET to 1.9% for PPMT.

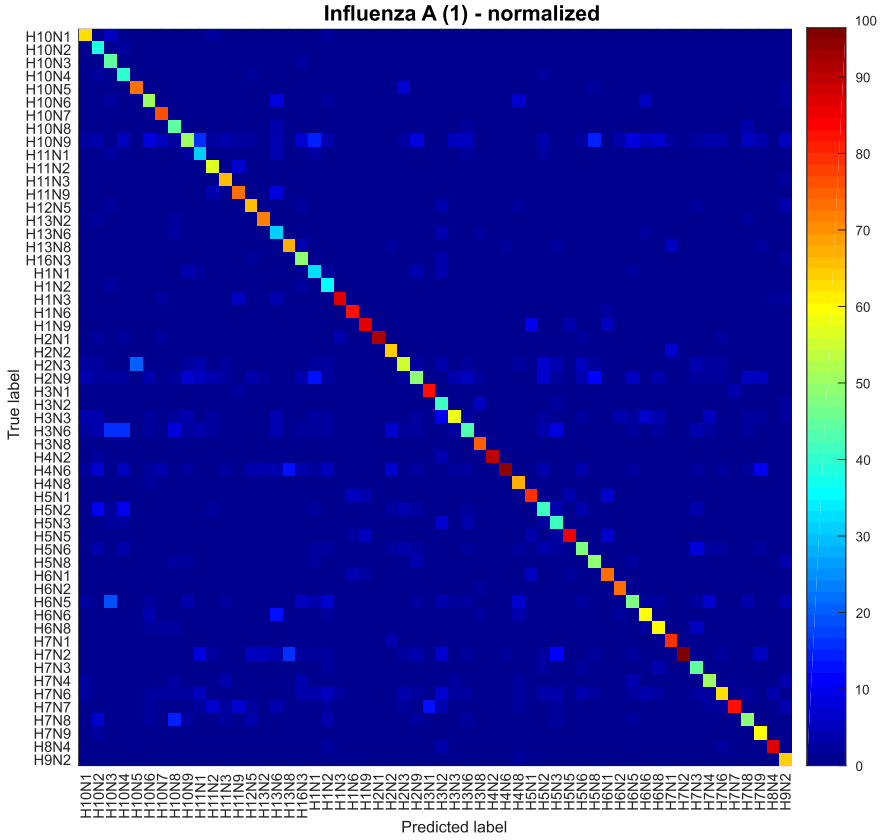
The results of the virus subtyping obtained on the Influenza A dataset also prove the superiority of the proposed approach. For all the considered variants of the dataset, all metrics scored by VGDC are the highest. Particularly, in the case of the basic Influenza A dataset (c.f. Influenza A(1)), F1 measure scored by VGDC is about 2% higher than the corresponding score for C-Measure and about 4% higher than in the case of CASTOR. A similar trend can also be observed for the extended Influenza A dataset (c.f. Influenza A(2)).

Finally, in the case of Influenza A (1) dataset, the proposed approach performed equally well when trained with both reduced and full training set. However, the results obtained on the Influenza A (2) dataset are a bit surprising. In this case, VGDC seems to better generalize to the less amount of training data. Particularly, when trained with the use of all available training data on the extended Influenza dataset, the VGDC performed slightly worse (F1 at the level of 0.847) than in the case when one-third of the training data was used (F1 at the level of 0.835). In the latter case, even when the number of subtypes increased (and thus hindered classification) VGDC performed better than for main Influenza subtypes. This is an interesting issue of training data preparation that will be investigated in our future work.

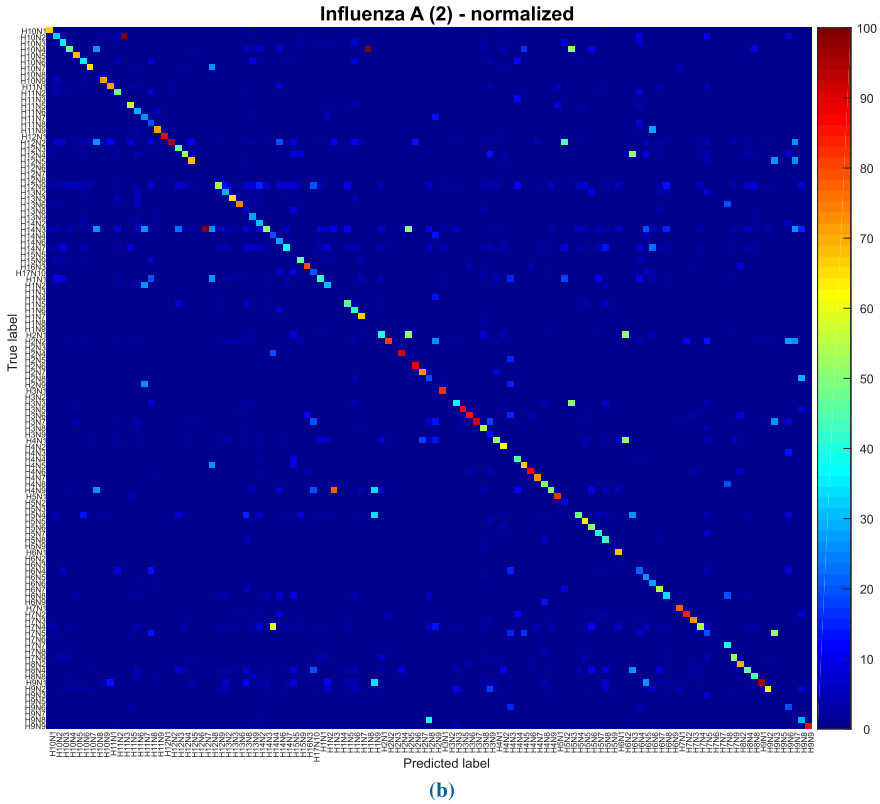
IV. CONCLUSION

The Viral Genome Deep Classifier proposed in this paper is the first approach to virus subtyping that uses the convolutional neural network. The method is universal and can be successfully applied for the subtyping of many virus families. It also outperforms the selected state-of-the-art approaches in the virus classification task. This can be especially seen in the case of the HIV viruses and Influenza A virus where the VGDC approach proved to be a few percent more accurate than the considered competitors. The latter includes CASTOR and COMET which are well established and commonly used tools for HIV virus subtyping.

An additional advantage of the VGDC approach is the feasibility of usage with a large amount of training data. As shown in our experiments with Influenza A, some of the state-of-the-art algorithms experience serious efficiency problems when trained with an enormous amount of data. Particularly, both CASTOR and PPMT required several days



(a)



(b)

FIGURE 8. Normalized confusion matrices obtained for influenza dataset. a) Influenza A (1). b) Influenza A (2).

of training when a training set of more than 250 thousand genomes was utilized. Additionally, PPMT used the hard disk drive heavily. The VGDC is free from these drawbacks since it takes advantage of GPU processing.

Although VGDC provides convincing results, the method still needs further investigation. Particularly, the results on Influenza A dataset have shown that the accuracy of the classification could be increased by tuning the network and training parameters. One may also notice that the simple C-measure classifier is faster than VGDC, often exceeding 100 classified samples per second. Exploring the influence of the training setup on the resulting virus classification accuracy will thus be a subject of our future work.

APPENDIX
CONFUSION MATRICES

This appendix presents confusion matrices for an automatic classification of genomic sequences using the proposed VGDC approach. Particularly, the consecutive figures refer to the results obtained for Dengue dataset (Fig. 2), two variants of Hepatitis B dataset (Fig. 3 and Fig. 4), Hepatitis C dataset (Fig. 5), two variants of HIV-1 dataset (Fig. 6 and Fig. 7) and two variants of Influenza A dataset (Fig. 8). Due to a large number of classes for Influenza A, only heat maps representing normalized confusion matrices are presented. In the remaining cases, the confusion matrices are presented in both variants, i.e., with a number of samples (upper row) and after normalization by the cardinality of a particular virus subtype. In all the cases, the resulting confusion matrices were obtained by summing up matrices obtained for all five folds. The *True Label* denotes real virus subtype while the *Predicted Label* refers to virus subtype assigned by the proposed VGDC approach.

ACKNOWLEDGMENT

The authors would like to thank Laurence Guillorit from The Luxemburg Institute of Health for her help in running the COMET experiments.

REFERENCES

[1] M. A. Remita, A. Halioui, A. A. M. Diouara, B. Daigle, G. Kiani, and A. B. Diallo, "A machine learning approach for viral genome classification," *BMC Bioinf.*, vol. 18, no. 1, p. 208, Apr. 2017.

[2] S. L. K. Pond, D. Posada, E. Stawiski, C. Chappey, A. F. Y. Poon, G. Hughes, E. Fearnhill, M. B. Gravenor, A. J. L. Brown, and S. D. W. Frost, "An evolutionary model-based algorithm for accurate phylogenetic breakpoint mapping and subtype prediction in HIV-1," *PLoS Comput. Biol.*, vol. 5, no. 11, Nov. 2009, Art. no. e1000581. [Online]. Available: http://dx.plos.org/10.1371/journal.pcbi.1000581

[3] R. C. Edgar, "Search and clustering orders of magnitude faster than BLAST," *Bioinformatics*, vol. 26, no. 19, pp. 2460–2461, Aug. 2010.

[4] L. C. J. Alcantara, S. Cassol, P. Libin, K. Deforche, O. G. Pybus, M. Van Ranst, B. Galvao-Castro, A.-M. Vandamme, and T. De Oliveira, "A standardized framework for accurate, high-throughput genotyping of recombinant and non-recombinant viral sequences," *Nucleic Acids Res.*, vol. 37, no. 2, pp. W634–W642, May 2009.

[5] A.-C. Pineda-Peña, N. R. Faria, S. Imbrechts, P. Libin, A. B. Abecasis, K. Deforche, A. Gómez-López, R. J. Camacho, T. de Oliveira, and A.-M. Vandamme, "Automated subtyping of HIV-1 genetic sequences for clinical and surveillance purposes: Performance evaluation of the new REGA version 3 and seven other tools," *Infection, Genet. Evol.*, vol. 19, pp. 337–348, Oct. 2013.

[6] S. Solis-Reyes, M. Avino, A. Poon, and L. Kari, "An open-source k-mer based machine learning tool for fast and accurate subtyping of HIV-1 genomes," *PLoS ONE*, vol. 13, no. 11, Nov. 2018, Art. no. e0206409. doi: 10.1371/journal.pone.0206409.

[7] P. K. Attaluri, Z. Chen, and G. Lu, "Applying neural networks to classify influenza virus antigenic types and hosts," in *Proc. IEEE Symp. Comput. Intell. Bioinf. Comput. Biol.*, May 2010, pp. 1–6.

[8] D. Struck, G. Lawyer, A.-M. Ternes, J.-C. Schmit, and D. P. Bercoff, "COMET: Adaptive context-based modeling for ultrafast HIV-1 subtype identification," *Nucleic Acids Res.*, vol. 42, no. 18, p. e144, Oct. 2014.

[9] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, pp. 436–444, May 2015.

[10] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.

[11] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds. Red Hook, NY, USA: Curran Associates, 2012, pp. 1097–1105. [Online]. Available: http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf

[12] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *CoRR*, vol. abs/1409.1556, pp. 1–14, Apr. 2014. [Online]. Available: http://arxiv.org/abs/1409.1556

[13] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *CoRR*, vol. abs/1512.03385, pp. 1–12, Dec. 2015. [Online]. Available: http://arxiv.org/abs/1512.03385

[14] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proc. IEEE Conf. CVPR*, Jun. 2015, pp. 1–9.

[15] R. W. Swiniarski and D. Waagen, "A neural network approach to genome sequence alignment," *Int. J. Appl. Math. Comput. Sci.*, vol. 4, no. 3, pp. 371–395, 1994.

[16] J. Zhou and O. G. Troyanskaya, "Predicting effects of noncoding variants with deep learning–based sequence model," *Nature Methods*, vol. 12, no. 10, p. 931, 2015.

[17] J. Lanchantin, R. Singh, B. Wang, and Y. Qi, "Deep gdashboard: Visualizing and understanding genomic sequences using deep neural networks," *CoRR*, vol. abs/1608.03644, pp. 1–11, Aug. 2016. [Online]. Available: http://arxiv.org/abs/1608.03644

[18] T. Yue and H. Wang, "Deep learning for genomics: A concise overview," Feb. 2018, *arXiv:1802.00810*. [Online]. Available: https://arxiv.org/abs/1802.00810

[19] C. Cao, F. Liu, H. Tan, D. Song, W. Shu, W. Li, Y. Zhou, X. Bo, and Z. Xie, "Deep learning and its applications in biomedicine," *Genomics, Proteomics Bioinf.*, vol. 16, no. 1, pp. 17–32, 2018.

[20] T. Ching *et al.*, "Opportunities and obstacles for deep learning in biology and medicine," *J. Roy. Soc. Interface*, vol. 15, no. 141, 2018, Art. no. 20170387.

[21] C. Angermueller, T. Pärnamaa, L. Parts, and O. Stegle, "Deep learning for computational biology," *Mol. Syst. Biol.*, vol. 12, no. 7, p. 878, 2016.

[22] Y. Chen, Y. Li, R. Narayan, A. Subramanian, and X. Xie, "Gene expression inference with deep learning," *Bioinformatics*, vol. 32, no. 12, pp. 1832–1839, 2016.

[23] S. Zhang, J. Zhou, H. Hu, H. Gong, L. Chen, C. Cheng, and J. Zeng, "A deep learning framework for modeling structural features of RNA-binding protein targets," *Nucleic Acids Res.*, vol. 44, no. 4, p. e32, 2016. [Online]. Available: http://dx.doi.org/10.1093/nar/gkv1025

[24] D. Quang and X. Xie, "DanQ: A hybrid convolutional and recurrent deep neural network for quantifying the function of DNA sequences," *Nucleic Acids Res.*, vol. 44, no. 11, p. e107, 2016. doi: 10.1093/nar/gkw226.

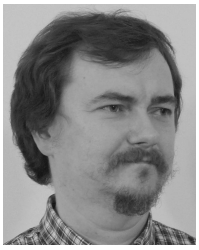
[25] D. R. Kelley, J. Snoek, and J. L. Rinn, "Basset: Learning the regulatory code of the accessible genome with deep convolutional neural networks," *Genome Res.*, vol. 26, no. 7, pp. 990–999, 2016.

[26] J. Eickholt and J. Cheng, "DNdisorder: Predicting protein disorder using boosting and deep networks," *BMC Bioinf.*, vol. 14, no. 1, p. 88, Mar. 2013. doi: 10.1186/1471-2105-14-88.

[27] M. Spencer, J. Eickholt, and J. Cheng, "A deep learning network approach to ab initio protein secondary structure prediction," *IEEE/ACM Trans. Comput. Biol. Bioinf.*, vol. 12, no. 1, pp. 103–112, Jan. 2015.

[28] S. Wang, S. Weng, J. Ma, and Q. Tang, "DeepCNF-D: Predicting protein order/disorder regions by weighted deep convolutional neural fields," *Int. J. Mol. Sci.*, vol. 16, no. 8, pp. 17315–17330, 2015. [Online]. Available: http://www.mdpi.com/1422-0067/16/8/17315
[29] R. Rizzo, A. Fiannaca, M. La Rosa, and A. Urso, "A deep learning approach to DNA sequence classification," in *Computational Intelligence Methods for Bioinformatics and Biostatistics*, C. Angelini, P. M. Rancoita, and S. Rovetta, Eds. Cham, Switzerland: Springer, 2015, pp. 129–140.
[30] S. Khawaldeh, U. Pervaiz, M. Elsharnoby, A. E. Alchalabi, and N. Al-Zubi, "Taxonomic classification for living organisms using convolutional neural networks," *Genes*, vol. 8, no. 11, p. 326, 2017.
[31] B. Yin, M. Balvert, D. Zambrano, A. Schoenhuth, and S. Bohte, "An image representation based convolutional network for DNA classification," in *Proc. Int. Conf. Learn. Represent.*, 2018, pp. 1–17. [Online]. Available: https://openreview.net/forum?id=HJvvRoe0W
[32] X. Gao, J. Zhang, Z. Wei, and H. Hakonarson, "DeepPolyA: A convolutional neural network approach for polyadenylation site prediction," *IEEE Access*, vol. 6, pp. 24340–24349, 2018.
[33] M. Li, C. Ling, and J. Gao, "An efficient CNN-based classification on G-protein coupled receptors using TF-IDF and N-gram," in *Proc. IEEE Symp. Comput. Commun. (ISCC)*, Jul. 2017, pp. 924–931.
[34] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *CoRR*, vol. abs/1502.03167, pp. 1–11, Feb. 2015. [Online]. Available: http://arxiv.org/abs/1502.03167
[35] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *CoRR*, vol. abs/1412.6980, pp. 1–15, Dec. 2014. [Online]. Available: http://arxiv.org/abs/1412.6980
[36] D. S. Hunnisett and W. J. Teahan, "Context-based methods for text categorisation," in *Proc. 27th Annu. Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, 2004, pp. 578–579.
[37] D. Benedetto, E. Caglioti, and V. Loreto, "Language trees and zipping," *Phys. Rev. Lett.*, vol. 88, no. 4, 2002, Art. no. 048702.
[38] D. V. Khmelev and W. J. Teahan, "A repetition based measure for verification of text collections and for text categorization," in *Proc. 26th Annu. Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, 2003, pp. 104–110.
[39] D. Loewenstern, H. Hirsh, P. Yianilos, and M. Noordewier, "DNA sequence classification using compression-based induction," Center Discrete Math. Theor. Comput. Sci., Rutgers Univ., Piscataway, NJ, USA, Tech. Rep. LCSR-TR-240, 1995.

ANNA FABIJAŃSKA was born in 1982. She received the M.E., Ph.D., and Habilitation degrees in computer science from the Faculty of Electrical, Electronic, Computer, and Control Engineering, Lodz University of Technology, Poland, in 2006, 2007, and 2013, respectively, where she is currently an Associate Professor in computer science with the Institute of Applied Computer Science. She has authored/coauthored over 100 scientific papers. Her scientific interests focus on digital image processing and analysis, machine vision, and artificial intelligence, especially deep learning. In particular, they concern the development of the dedicated image processing pipelines for computer-aided diagnosis systems and applications of computer vision in various fields of science and industry. Since 2016, she has been a member of The Polish Young Academy of the Polish Academy of Sciences and the Committee on Informatics of the Polish Academy of Sciences. She was a beneficiary of the Ministry of Science and Higher Education Fellowship for outstanding young scientists, from 2013 to 2015, a beneficiary of the Foundation for Polish Science (FNP) START Fellowship, in 2011, and the Leader of scientific grants, including the project within the framework of the Iuventus Plus Programme, from 2013 to 2015.



SZYMON GRABOWSKI was born in 1973. He received the Ph.D. and Habilitation degrees in computer science, in 2003 and 2011, respectively. He is currently an Associate Professor in computer science with the Institute of Applied Computer Science, Lodz University of Technology, Poland. His former research, including Ph.D. dissertation, involved nearest neighbor classification methods in pattern recognition, also with applications in image processing. He has published about 130 papers in journals and conferences. His current interests include string matching, text indexing, and data compression, with applications in bioinformatics. Some of his particular research topics include various approximate string matching problems, compressed text indexes, and XML compression.