Bioinformatics, 36(1), 2020, 272–279
doi: 10.1093/bioinformatics/btz493
Advance Access Publication Date: 21 June 2019
Original Paper

Data and text mining

Deep learning on chaos game representation for proteins

Hannah F. Löchel, Dominic Eger, Theodor Sperlea and Dominik Heider*

Department of Mathematics and Computer Science, Philipps-University of Marburg, Marburg 35032, Germany

*To whom correspondence should be addressed.
Associate Editor: Jonathan Wren

Received on March 13, 2019; revised on May 29, 2019; editorial decision on June 5, 2019; accepted on June 14, 2019

Abstract

Motivation: Classification of protein sequences is one big task in bioinformatics and has many applications. Different machine learning methods exist and are applied on these problems, such as support vector machines (SVM), random forests (RF) and neural networks (NN). All of these methods have in common that protein sequences have to be made machine-readable and comparable in the first step, for which different encodings exist. These encodings are typically based on physical or chemical properties of the sequence. However, due to the outstanding performance of deep neural networks (DNN) on image recognition, we used frequency matrix chaos game representation (FCGR) for encoding of protein sequences into images. In this study, we compare the performance of SVMs, RFs and DNNs, trained on FCGR encoded protein sequences. While the original chaos game representation (CGR) has been used mainly for genome sequence encoding and classification, we modified it to work also for protein sequences, resulting in n-flakes representation, an image with several icosagons.
Results: We could show that all applied machine learning techniques (RF, SVM and DNN) show promising results compared to the state-of-the-art methods on our benchmark datasets, with DNNs outperforming the other methods and that FCGR is a promising new encoding method for protein sequences.
Availability and implementation: https://cran.r-project.org/.
Contact: dominik.heider@uni-marburg.de
Supplementary information: Supplementary data are available at *Bioinformatics* online.

1 Introduction

Protein classification is one big challenge in bioinformatics (Heider *et al.*, 2009), and has therefore many applications, ranging from genomic annotations towards clinical applications, such as drug resistance prediction in human immunodeficiency virus (HIV) for personalized therapies. To this end, different machine learning methods exist and have been applied, e.g. support vector machines (SVM) (Beerenwinkel *et al.*, 2003), random forests (RF) (Heider *et al.*, 2011; Löchel *et al.*, 2018) or neural networks (NN) (Wang and Larder, 2003). Generally, the protein sequences have to be made 'machine-readable' in a first step. Different protein encodings exist, which can be roughly separated into sequence-based or structure-based encodings. These sequence-based encodings include sparse encoding (Hirst and Sternberg, 1992), amino acid composition (Matsuda *et al.*, 2005), reduced amino acid alphabets (Solis and Rackovsky, 2000), physicochemical properties (Heider and Hoffmann, 2011) or Fourier Transformation (Nagarajan *et al.*, 2006). Structure-based encodings include quantitative structure-activity relationship (QSAR) (Cherkasov *et al.*, 2014), Electrostatic Hull (Dybowski *et al.*, 2011) or Delaunay triangulation (Yu *et al.*, 2013). For a comprehensive review on encodings of protein sequences see Spänig and Heider (2019). After encoding, the encoded sequences can be used for training of different machine learning models, such as SVMs, RFs or deep neural networks (DNNs). Due to the fact that DNNs have been shown to outperform other methods in image classification, we will introduce a modified chaos game representation (CGR) for proteins and will show the performance of this encoding on HIV drug resistance prediction datasets in comparison to the state-of-the-art models. Moreover, we made our new frequency matrix chaos game representation (FCGR) for protein-encoding available as an R package kaos.

The chaos game representation (CGR) algorithm is a recurrent iterative function system, which can be used to create fractals from sequences of symbols, i.e. from an alphabet A = {s1,..., sn}. For n = 3 and A = {1, 2, 3}, the CGR algorithm can be used to construct, e.g. the Sirpinski triangle, a fractal structure constructed by smaller triangles (Barnsley, 2012). Jeffrey (1990) was the first who applied the CGR algorithm to DNA sequences, i.e. n = 4 and A = {A, C, G, T}, thus the resulting fractals are constructed from squares instead of triangles. The underlying idea of the CGR algorithm for DNA is summarized in Figure 1. Each symbol is set at one corner (here: 4). Starting from the middle, the next dot is put half the way towards the next symbol in the sequence. The second (and all remaining dots) are put half the way from the last position in the direction to the next symbol (exemplarily shown in Fig. 1B).

Since the development of the CGR and its application in life science, it has been used mainly for the analyses and comparison of whole genome sequences (Joseph and Sasikumar, 2006). It has been shown that CGR is an excellent representation for genomes and that CGR-driven phylogeny leads to reliable predictions (Deschavanne *et al.*, 1999). In particular the comparison between genomes by using CGR is very easy and fast (Hoang *et al.*, 2016). Extensions of CGR include color grids (Deschavanne *et al.*, 1999) and frequency matrix CGR (FCGR) (Almeida *et al.*, 2001). Wang *et al.* (2005) used FCGR to calculate the image distance between genomes in order to generate phylogenetic trees. Rizzo *et al.* (2016) showed that DNNs trained on genomes encoded with FCGR yielded very accurate predictions. They used a convolutional neural network (CNN) to divide bacteria in three different phyla, order, family and genus and showed a very high accuracy for the method. While these studies focused only on FCGR for DNA, there exist also a smaller number of studies dealing with the encoding of protein sequences. Yu *et al.* (2004) employed the CGR algorithm for protein classification by separating the amino acids in four groups based on their properties and used multifractal and correlation analysis to construct a phylogenetic tree of Archaea and Eubacteria. In another approach the amino acids were re-translated into DNA for CGR (Yang *et al.*, 2009). Basu *et al.* (1997) used CGR by grouping the amino acids in twelve groups and used a twelve-sided regular polygon for the representation.

Most of the studies with CGR on proteins have in common that they make use of the original approach to create the CGR, i.e. they go half the way of the distance to the next symbol to produce the CGR images. However, by using this approach, resulting CGR images are very noisy for alphabets with n > 4. In this study, we introduce the use of Sierpinskin-gons, also known as, n-flakes or polyflakes (Tzanov, 2015), which can be constructed by varying the distances and thus result in well-structured fractals. The main advantage of using n-flakes in CGR is that the resulting points do not overlap (see Fig. 2), and thus are precisely separable for n > 4. Moreover, we will make use of DNNs and FCGR for proteins and analyze the impact of the scaling factor as well as the resolution on the classification performance on HIV drug resistance datasets.

2 Materials and methods

2.1 Dataset

HIV-1 is known for its high mutation rate, which offers the virus the opportunity to quickly evolve drug resistance. Thus, prediction of drug resistance is crucial for personalized therapy of the patient. Protein sequences of the HIV-1 protease (PR) and reverse transcriptase (RT) originating from subtype B strains with data for seven PIs (RTV: Ritonavir, IDV: Indinavir, SQV: Saquinavir, NFV: Nelfinavir, APV: Amprenavir, ATV: Atazanavir, LPV: Lopinavir), three NNRTIs (NVP: Nevirapine, EFV: Efavirenz, DLV: Delavirdine) and five NRTIs (3TC: Lamivudine, ABC: Abacavir, AZT: Zidovudine, D4T: Stavudine, DDI: Didanosine) with IC_{50} ratios were collected from the HIV Drug Resistance Database (Rhee *et al.*, 2003). The data was separated into susceptible and resistant by drug-specific cutoffs. Rhee *et al.* (2006) We removed sequences from the datasets for which no resistance information was available and excluded ATV from our classification approach, since too many sequences lacked IC_{50} information. Table 1 gives a summary of the data used in the study for each drug.

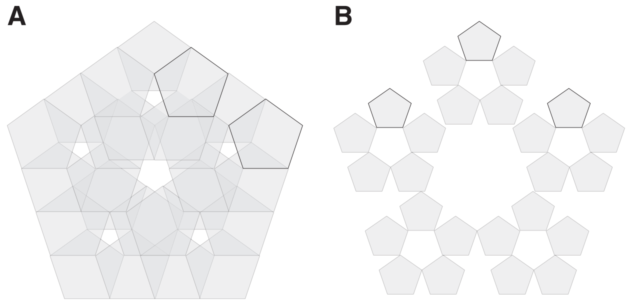

Fig. 2. Construction of n-flakes. (**A**) Overlapping polygons; (**B**) n-flake

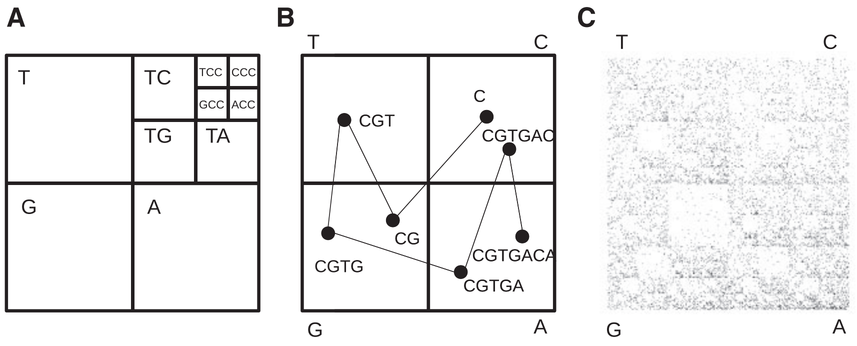

Fig. 1. Chaos Game Representation for DNA. (**A**) Division of the square. (**B**) Way walked to draw points. (**C**) HIV genome (NCBI Reference Sequence: NC_001802.1)

Table 1. Data used in the study

	NNRTI			NRTI					PI					
	DLV	EFV	NVP	3TC	ABC	AZT	D4T	DDI	APV	IDV	LPV	NFV	RTV	SQV
Positive	455	447	415	195	179	322	336	306	424	384	223	303	349	457
Negative	263	274	318	429	440	299	285	317	278	374	278	472	379	304

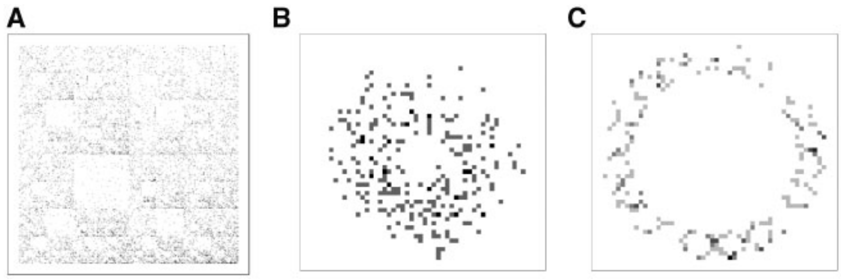

Fig. 3. (**A**) FCGR of genomic DNA sequence of HIV (NCBI Reference Sequence: NC_001802.1) with resolution of 200. (**B**) AAQ18891.1 reverse transcriptase, partial [Human immunodeficiency virus 1] with resolution of 50 and *sf* = 0.5, (**C**) AAQ18891.1 reverse transcriptase, partial [Human immunodeficiency virus 1] resolution = 50, sf_{20}

2.2 Implementation of the chaos game representation algorithm

We implemented an R package kaos (downloadable from CRAN), which can be used to create CGR and FCGR with n-flakes. The kaos package accepts any kind of alphabets and creates the (F)CGR image based on the given sequence and user-specified resolution. The package offers the options to create a CGR image with dots (option 'points') or an FCGR (option 'matrix') with different gray-levels. For the FCGR, the user has to specify a resolution to specify the columns of the matrix. It is also possible to set the scaling factor ('sf') which is needed to construct n-flakes. For protein sequences with twenty proteinogenic amino acids, the CGR representation results in twenty edges and twenty icosagons within a larger icosagon. The contraction ratio between the outer and the inner polygon can be calculated by the following equation (Strichartz, 2000):

$$r = \frac{\sin\left(\frac{\pi}{n}\right)}{\sin\left(\frac{\pi}{n}\right) + \sin\left(\frac{\pi}{n} + \frac{2\pi m}{n}\right)} \quad \text{for} \quad m = \left\lfloor \frac{n}{4} \right\rfloor \tag{1}$$

The ratio for the distance between the actual point and the target edge (i.e. the scale factor sf) can be calculated by the following equation:

$$sf = 1 - r \tag{2}$$

By default, the CGR package automatically creates the alphabet based on the given symbols or words in the sequence (vector of symbols or words) and takes this number as n to calculate the scaling factor by Equation 1. The number is also needed to calculate the coordinates for the edges of a polygon in an unit circle with the following equation:

$$\begin{aligned} x[i] &= r \cdot \sin\left(\frac{2\pi i}{n} + \theta\right) \\ y[i] &= r \cdot \cos\left(\frac{2\pi i}{n} + \theta\right) \end{aligned} \tag{3}$$

i : edge; n : number of edges, θ : angel of orientation

A CGR object contains the gray-level matrix with given resolution as an encoding for further analyses. In case of n = 4, the CGR algorithm fills the whole matrix, otherwise it uses the unit circle.

Figure 3 shows examples created with the CGR package, namely the FCGR representation of the genomic DNA sequence of HIV with a resolution of 200, of the HIV RT sequence with a resolution of 50 and $sf = 0.5$, as well as of the HIV RT sequence with a resolution of 20 and sf_{20}, the scaling factor for protein sequences with n-flakes. As mentioned before, the scaling factor is crucial in order to structure the fractal, which can be clearly seen by comparing the two FCGR representations. The CGR package offers predefined alphabets for numbers between 0 and 9, amino acids, DNA, and for the letters a–z as capital- and lowercase letters, which can be applied to compare sequences with different amounts of letters; e.g. for protein sequences which may consist of less than the twenty proteinogenic amino acids.

2.3 Development of prediction models

In order to evaluate the impact of the resolution and the scaling factor on subsequent classification, we used eight different configurations for the CGR images and trained DNNs, RFs and SVMs, with the settings for protein sequences ('amino'), to force 20-edges (see Fig. 4).

We performed a stratified hold-out validation scheme where 20% of the data was randomly selected for validation and 60% was used for building the models to evaluate the machine learning models. The remaining 20% of the data was used as test data for the DNNs, SVMs and RFs. We then performed a 10-fold cross-validation with the remaining data (i.e. without the validation data). We trained models for SVMs, RFs and DNNs with the different configurations mentioned before. All cells containing only zeros in all data were removed prior training of the SVMs and RFs.

We used the e1071 package for the SVMs with linear kernel and varied the costs C between 0.1 and 1 in steps of 0.1, the randomforest package (Breiman, 2001) for the RFs with default settings, 1000 trees, and varied the nodesize from 1 up to 10 in steps by 1. For the DNNs we used the deepnet package in R. We trained the fully connected DNNs with *tangens hyperbolicus* as activation function a learning rate of 0.8, and a momentum of 0.5. In addition, we varied the numbers of neurons (up to 20) in three hidden layers and number of training epochs (see Fig. 5).

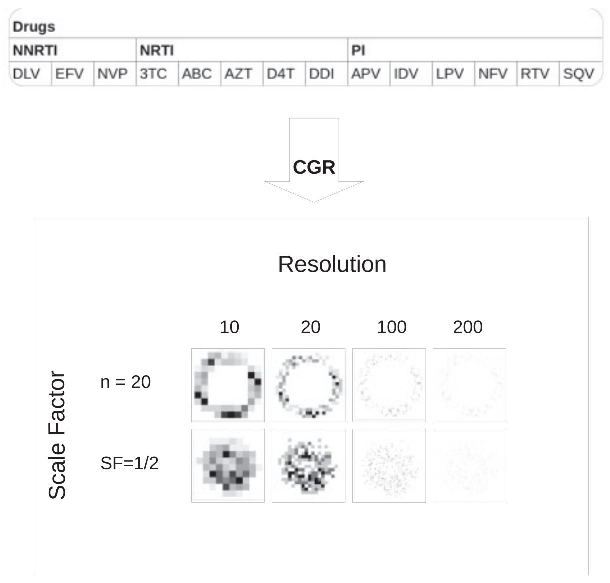

Fig. 4. Different settings for producing the FCGR pictures. We used different combinations of resolution and scale factor to produce the FCGR images. The resolution was set to 10, 20, 100 and 200, while the scaling factor was set to 0.5 and sf_{20}, i.e. the optimal scaling factor for n = 20

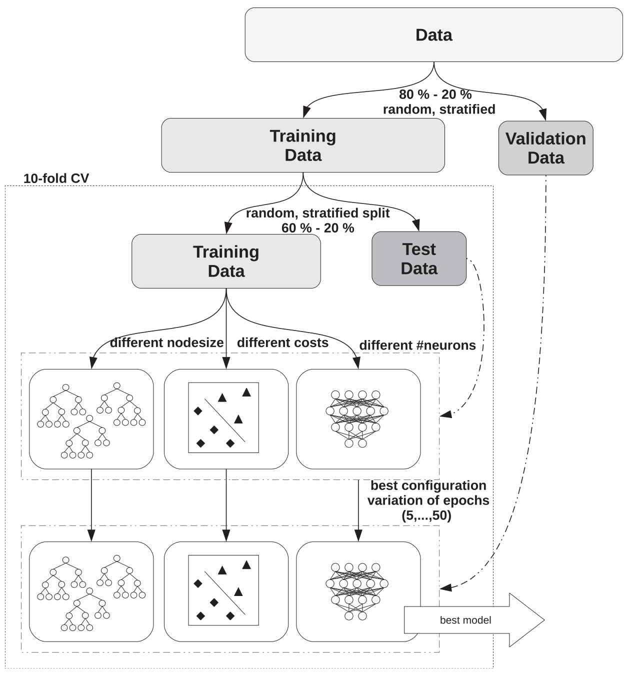

Fig. 5. Development of prediction models

We evaluated and compared the models based on the area under the receiving operating characteristics curve (AUC) with the R package pROC (Robin *et al.*, 2011). The best hidden layer configuration was selected based on the best average AUC. For the DNNs, we calculated the AUC also for the varying training epochs. Moreover, we used the R package ROCR (Sing *et al.*, 2005) to draw precision-recall curves for the best-performing models.

2.4 Evaluation of FCGR as encoding

We calculated the average FCGRs of positive and negative samples, i.e. the average for each cell in the FCGR matrices of positive and negative sequences, respectively, in all datasets. Next, we calculated the differences between the average FCGR of the positive and the average FCGR of the negative samples. Significance of the differences were calculated based on Student's t-tests, resulting p values were corrected for multiple testing by the method of Bonferroni. Moreover, in order to visualize the predictive quality of the different encodings in a model-independent manner, we used $<\phi,\delta>$ diagrams as implemented in the R package phiDelta (Armano and Giuliani, 2018). For this purpose we plotted the $<\phi,\delta>$ diagrams for the encoding used by Heider *et al.* (2011) and the FCGR encoding with the different settings used in the current study. $<\phi,\delta>$ diagrams are two-tiered 2D tools, which have been devised to support the assessment of classifiers or features in terms of accuracy and bias. δ and ϕ are defined as follows:

$$\begin{aligned} \phi &= \rho - \bar{\rho} \\ \delta &= \rho + \bar{\rho} - 1 \\ \rho &= \frac{TP}{TP + FN} \\ \bar{\rho} &= \frac{TN}{TN + FP} \end{aligned} \tag{4}$$

with TP, FN, TN and FP representing the number of true positives, false negatives, true negatives and false positives, respectively (Armano and Giuliani, 2018).

3 Results

We calculated the AUCs for the DNNs with different number of neurons from the cross-validation. For the best performing DNNs, we also evaluated different number of training epochs. Final evaluation of the models was carried out using the validation set. The best DNN configuration (number of neurons and epochs) in comparison to SVMs and RFs for the NRTIs and NNRTI, and the PIs are shown in Figures 6 and 7, respectively.

For the NRTI 3TC, the DNN outperforms the other method in all encoding configuration, i.e. independently from the resolution of the FCGR. However, for the NNRTIs DLV, EFV and NVP, as well as for the NRTIs AZT and D4T, linear SVMs give better prediction results for lower resolutions. In all cases, the accuracy of the DNN models increases with the resolution. Figure 7 shows the results for the PIs. For high resolutions, the DNNs outperform the RFs and the SVMs, or are equally good as the SVMs, comparable to the results of the NNRTIs and NRTIs.

In Table 2 the AUCs of the best RF, SVM and DNN models are summarized. The DNNs and SVMs outperform the RFs. For the PIs, best results are observed with the DNNs, in most cases with sf_{20}, except for IDV and LPV, where the best results are observed at a scaling factor of 0.5. The optimal scaling factor depends on the dataset, e.g. for APV there are higher AUC values with sf_{20}, however, for DDI the best results are obtained with $sf = 0.5$. Some datasets perform quite well at low resolution, especially ABC and RTV, whereas increasing the scaling factor has a barely remarkable influence on the AUC values. While the DNNs have the highest AUC values, the other models still perform quite well, and thus supports the idea of CGR for protein encoding.

Figures 8 and 9 show the precision-recall curves for the best DNNs for the different drugs, supporting the very good prediction results from the DNNs. For the AUCs and mean errors with growing number of epochs for the different training- and testsets see Supplementary Material.

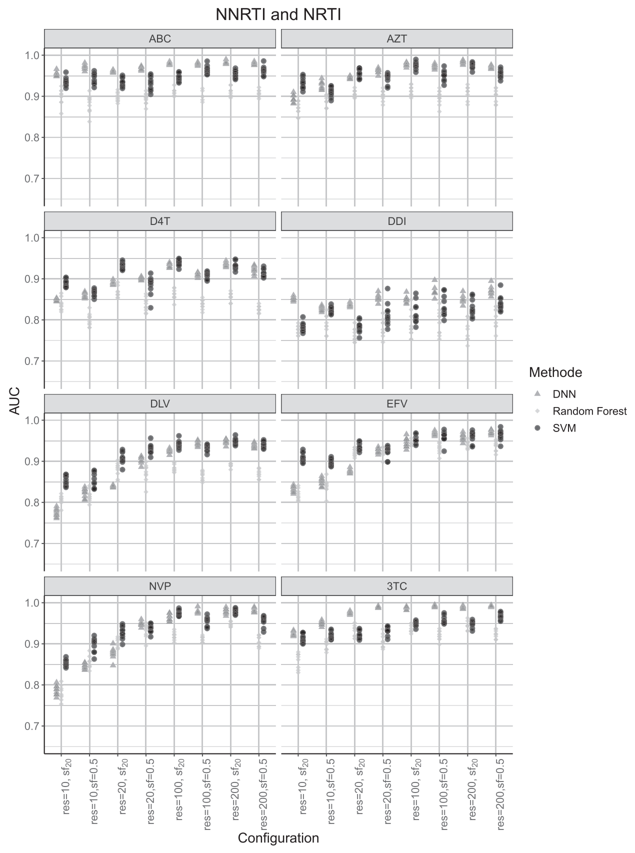

Fig. 6. AUCs for NNRTIs and NRTIs. Results from with different training splits and different configurations evaluated with the validation data. Triangle: DNN; raute: RF; circle: SVM

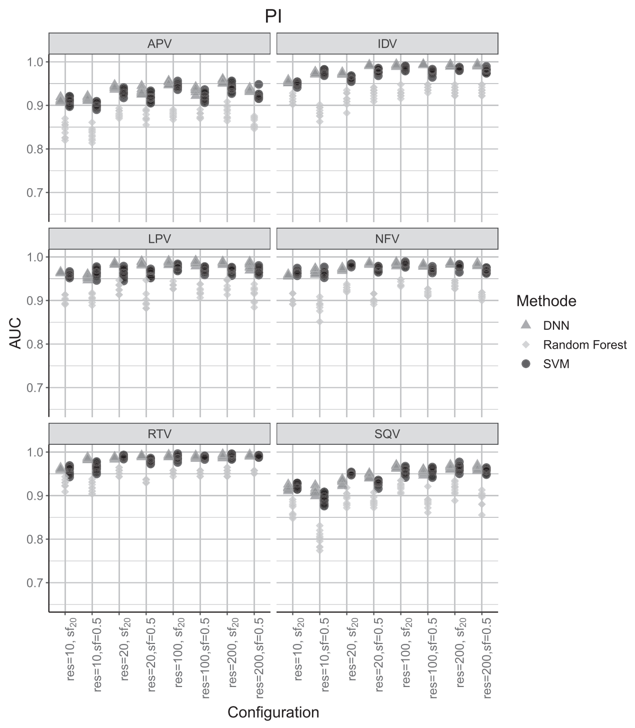

Fig. 7. AUCs for PIs. Results from with different training splits and different configurations valuated with the validation data. Triangle: DNN; raute: RF; circle: SVM

Table 2. Best AUCs for NNRTIs and NRTIs on validation data

	DNN	RF	SVM
NNRTI			
DLV	0.95 ($r = 200$, sf_{20})	0.90 ($r = 200$, sf_{20}	0.96 ($r = 200$, sf_{20})
EFV	0.98 ($r = 200$, $sf = 0.5$)	0.96 ($r = 100$, sf_{20})	0.98 ($r = 200$, $sf = 0.5$)
NVP	0.99 ($r = 100$, $sf = 0.5$)	0.96 ($r = 200$, sf_{20})	0.99 ($r = 200$, sf_{20})
NRTI			
3TC	0.99 ($r = 100$, $sf = 0.5$)	0.96 ($r = 200$, sf_{20})	0.98 ($r = 200$, $sf = 0.5$)
ABC	0.99 ($r = 100$, $sf = 0.5$)	0.93 ($r = 100$, sf_{20})	0.99 ($r = 200$, $sf = 0.5$)
AZT	0.99 ($r = 200$, sf_{20})	0.93 ($r = 100$, sf_{20})	0.99 ($r = 100$, sf_{20})
D4T	0.94 ($r = 200$, sf_{20})	0.89 ($r = 20$, sf_{20})	0.95 ($r = 100$, sf_{20})
DDI	0.90 ($r = 100$, $sf = 0.5$)	0.85 ($r = 100$, sf_{20})	0.88 ($r = 200$, $sf = 0.5$)
PI			
APV	0.96 (200 sf_{20})	0.91 (200 sf_{20})	0.96 (200 sf_{20})
IDV	0.995 (200 $sf = 0.5$)	0.95 (100 sf_{20})	0.99 (100 sf_{20})
LPV	0.99 (100 $sf = 0.5$)	0.95 (200 sf_{20})	0.98 (100 sf_{20})
NFV	0.99 (100 sf_{20})	0.95 (100 sf_{20})	0.99 (100 sf_{20})
RTV	0.995 (200 sf_{20})	0.96 (200 sf_{20})	0.996 (100 sf_{20})
SQV	0.97 (200 sf_{20})	0.93 (100 sf_{20})	0.98 (200 sf_{20})

r, resolution; sf, scaling factor.

3.1 Comparison with other encodings

So far, we only compared the results from the different models, namely DNNs, SVMs and RFs, on the same protein encoding, namely the FCGR. In the following, we will compare our results with the state-of-the-art methods.

Table 3 shows the AUC values of the best models trained on FCGR from our approach in comparison to the models of Heider *et al.* (2011) and Kierczak *et al.* (2009) for NRTIs and NNRTIs. Compared to the approach of Heider *et al.* (2011) and Kierczak *et al.* (2009), we get AUC values between 4% up to 8% and 19% higher, respectively. Even the lower performing SVMs and RFs outperform or at least perform equally well compared the state-of-the-art approaches.

Table 4 shows the calculated accuracy values for the best models, in comparison with Heider *et al.* (2011), Rhee *et al.* (2006) and Hou *et al.* (2009) for the PIs. For all drugs, the FCGR approach outperforms the state-of-the-art models.

The fact that FCGR-based classifiers were consistently outperforming other classification models in this study suggests that FCGR itself is a feature encoding for protein sequences preferable to some others. In order to test this hypothesis with regards to the data analyzed here, we compared the prediction performance of the feature encodings used in this study with the amino acid encoding and interpolation based feature encoding used in Heider and Hoffmann (2011) using $< \phi, \delta >$ diagrams (Armano *et al.*, 2018), which allow for the visual inspection of model-independent feature quality with regards to a given binary classification task (Fig. 10A and B). For all sequence datasets analyzed here, FCGR-based features show superior performance (see supporting information). To explain this behavior of FCGR encodings, we compared FCGR matrices for the positive and negative sequences from the different datasets. These show clear and significant differences in a small number of pixels (see Supporting Information). This is in accordance with the finding that very different machine learning models trained with FCGR-encoded sequences show consistently high performance.

4 Discussion

The performance in terms of AUC of the RFs and SVMs has a higher variance compared to the AUCs of the DNNs, i.e. the split of test and training data might have a larger impact on the training of these

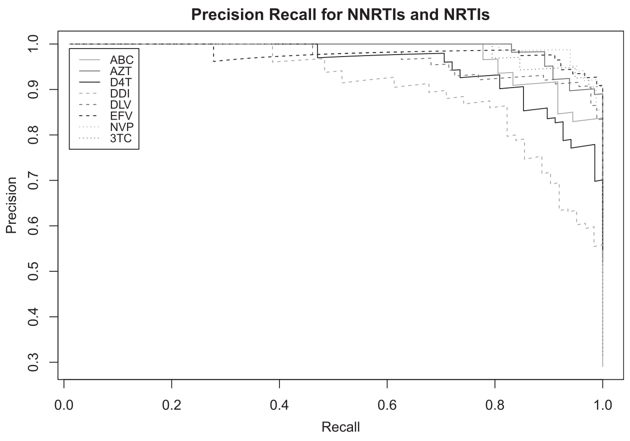

Fig. 8. Precision–recall curves for NNRTI and NRTI

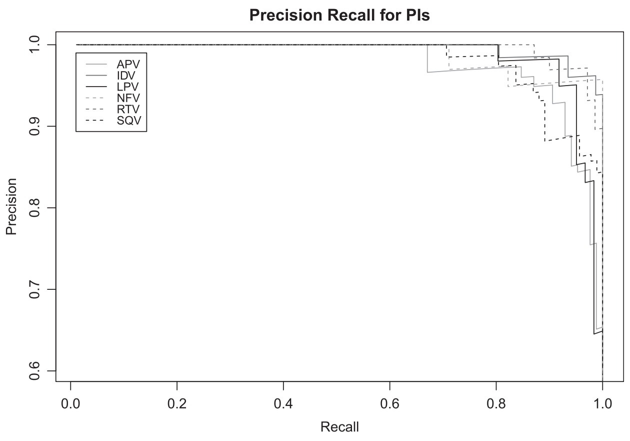

Fig. 9. Precision–recall curves for PIs

Table 3. Comparison of the FCGR approach with state-of-the-art methods for NNRTIs and NRTIs

	This study	Heider *et al.*	Kierczak *et al.*
NNRTI			
DLV	0.95***	0.90	0.76
EFV	0.98***	0.93	–
NVP	0.99***	0.92	0.85
NRTI			
3TC	0.99***	0.94	0.95
ABC	0.99***	0.92	0.83
AZT	0.99***	0.91	0.89
D4T	0.94***	0.90	0.85
DDI	0.90***	0.85	0.82

⁻Kierczak *et al.* analyzed the NRTI and NNRTI datasets except EFV.
$^{***}P \leq 0.001$.

models than on the DNNs. We can also observe that for some drugs low scaling factors work quite well and that the increase barely influences the results, whereas for other drugs the scaling factor leads to better performance until a saturation is reached. This suggests that the scaling factor somehow reveals patterns on some resolution, characteristic for the classification on this dataset. Comparing the course of the different models (Fig. 6), we can see that the SVMs and DNNs perform equally good on a high level.

Table 4. Comparison of the FCGR approach with state-of-the-art methods

	This study	Heider *et al.*	Rhee *et al.*	Hou *et al.*
NNRTI				
DLV	92%***	87%	84%	–
EFV	94%***	88%	87%	–
NVP	96%***	87%	91%	–
NRTI				
3TC	97%***	87%	90%	–
ABC	95%***	88%	84%	–
AZT	94%***	87%	84%	–
D4T	87%***	84%	78%	–
DDI	85%[a]	79%	75%	–
PI				
APV	91%***	88%	84%	89%
IDV	97%***	93%	79%	86%
LPV	98%***	92%	81%	91%
NFV	96%***	91%	82%	87%
RTV	97%***	95%	89%	93%
SQV	90%***	89%	84%	89%

⁻Hou et al. used only the PI datasets.
$^{***}P \leq 0.001$).
[a]Only significant compared to Rhee *et al.*

There might be a saturation for the performance of the DNNs at a given resolution where the application of sf_{20} or using 0.5 has a low impact on the performance. We can observe this for most of the drugs. Except for D4T and DDI where there is a drop in prediction performance. The models trained on FCGR outperform all other evaluated models, independent of the employed machine learning technique. This suggests that FCGR as an encoding for protein sequences might be more appropriate than other encodings. By using the $< \phi, \delta >$ diagrams we could show that the FCGR features show superior predictiveness. In comparison with the method of Heider *et al.* (2011), the FCGR encoding has no information loss on high resolution. Due to the interpolation the sequence-length is changed and this can lead to a loss of information. The advantage of the FCGR encoding is that the amino acid as itself is not transformed in any kind of representation, e.g. physicochemical properties. It can be considered as a kind of black box, where each letter represent different unknown feature lying behind each letter. The order of the letters is more or less kept, depending on the resolution, which explains the increase of performance in a higher resolution. One disadvantage is the increase of memory requirements for one FCGR matrix compared to a string or vector, in case the string or vector is smaller than the selected resolution. However, it can also lead to a compression if the length of the string is larger than the selected resolution (e.g. a whole genome plotted with a resolution of 100×100). The algorithm itself has a linear run time ($O(n)$), due to the fact that it is just depending on the number of symbols in a given sequence. In particular the use of sf_{20}, where most of the space in an FCGR image is never used. Thus, a solution might be to finally erase those elements of the matrix. We used comparatively long protein sequences in this study, thus, one open question is, if the FCGR encoding still works well for shorter sequences, e.g. peptides, since the formation of patterns might be less pronounced for short sequences.

5 Conclusion

FCGR as a feature encoding for proteins reveals a new approach for classification problems, which is particularly well-suited for DNNs. The encoding shows superior behavior compared to other encodings,

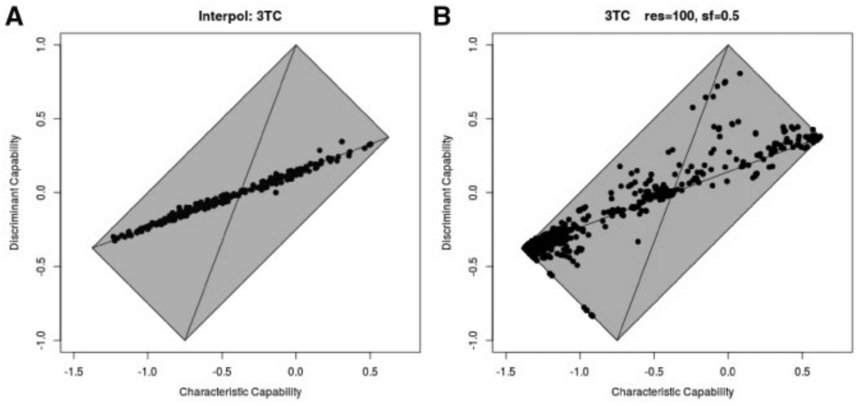

Fig. 10. Comparison of amino acid and FCGR based feature representations. Exemplarily shown for the 3TC sequence dataset. A and B: $<\phi,\delta>$ diagrams displaying the quality of features for the same sequence dataset (3TC) calculated using the R package Interpol (Heider and Hoffmann, 2011) as used in Heider *et al.* (2011) (**A**) and by FCGR with a resolution of 100 and $sf = 0.5$ (**B**). Dots represent features; dots closer to the upper or lower corner of the quadrilateral represent features with a high predictiveness for the classification task

independent from the employed machine learning technique in our study dealing with HIV-1 drug resistance. In fact, it outperforms the state-of-the-art methods and therefore it might be preferable to other protein classification problems. In combination with DNNs, FCGR can give very accurate predictions. The application of the scaling factor, in order to make use of n-flakes for training, can increase the accuracy, especially for RFs. Besides, the resolution of the FCGR plays an important role and can increase the accuracy depending on the classification problem. Since the FCGR method offers the opportunity to encode all kind of sequences, e.g. text and numbers, the use of FCGR in many other kind of applications besides DNA and protein classification problems, might be reasonable. Moreover, FCGR could, in principle, also be applied to microbiomes or viral quasispecies by using an additive approach of FCGR, where the single sequences are all plotted into one FCGR or to flatened network structures (Alcaraz *et al.*, 2011).

Funding

This work was partially funded by the Philipps-University of Marburg and the Paul Ehrlich Institute.

Conflict of Interest: none declared.

References

Alcaraz,N. *et al.* (2011) Keypathwayminer: detecting case-specific biological pathways using expression data. *Internet Math.*, **7**, 299–313.

Almeida,J.S. *et al.* (2001) Analysis of genomic sequences by chaos game representation. *Bioinformatics*, **17**, 429–437.

Armano,G. and Giuliani,A. (2018) A two-tiered 2d visual tool for assessing classifier performance. *Inf. Sci.*, **463**, 323–343.

Armano,G. *et al.* (2018) Phi-delta-diagrams: software implementation of a visual tool for assessing classifier and feature performance. *Mach. Learn. Knowl. Extract.*, **1**, 7.

Barnsley,M.F. (2012) *Fractals Everywhere: New Edition*. Dover Publications, Mineola, New York.

Basu,S. *et al.* (1997) Chaos game representation of proteins. *J. Mol. Graph. Modell.*, **15**, 279–289.

Beerenwinkel,N. *et al.* (2003) Geno2pheno: estimating phenotypic drug resistance from hiv-1 genotypes. *Nucleic Acids Res.*, **31**, 3850–3855.

Breiman,L. (2001) Random forests. *Mach. Learn.*, **45**, 5–32.

Cherkasov,A. *et al.* (2014) QSAR modeling: where have you been? Where are you going to? *J. Med. Chem.*, **57**, 4977–5010.

Deschavanne,P.J. *et al.* (1999) Genomic signature: characterization and classification of species assessed by chaos game representation of sequences. *Mol. Biol. Evol.*, **16**, 1391–1399.

Dybowski,J.N. *et al.* (2011) Improved bevirimat resistance prediction by combination of structural and sequence-based classifiers. *BioData Min.*, **4**, 26.

Heider,D. and Hoffmann,D. (2011) Interpol: an r package for preprocessing of protein sequences. *BioData Min.*, **4**, 16.

Heider,D. *et al.* (2009) A computational approach for the identification of small GTPases based on preprocessed amino acid sequences. *Technol. Cancer Res. Treat.*, **8**, 333–341.

Heider,D. *et al.* (2011) Machine learning on normalized protein sequences. *BMC Res. Notes*, **4**, 94.

Hirst,J.D. and Sternberg,M.J. (1992) Prediction of structural and functional features of protein and nucleic acid sequences by artificial neural networks. *Biochemistry*, **31**, 7211–7218.

Hoang,T. *et al.* (2016) Numerical encoding of DNA sequences by chaos game representation with application in similarity comparison. *Genomics*, **108**, 134–142.

Hou,T. *et al.* (2009) Predicting drug resistance of the HIV-1 protease using molecular interaction energy components. *Proteins Struct. Funct. Bioinform.*, **74**, 837–846.

Jeffrey,H.J. (1990) Chaos game representation of gene structure. *Nucleic Acids Res.*, **18**, 2163–2170.

Joseph,J. and Sasikumar,R. (2006) Chaos game representation for comparison of whole genomes. *BMC Bioinformatics*, **7**, 243.

Kierczak,M. *et al.* (2009) A rough set-based model of hiv-1 reverse transcriptase resistome. *Bioinform. Biol. Insights*, **3**, BBI–S3382.

Löchel,H.F. *et al.* (2018) Scotch: subtype a coreceptor tropism classification in HIV-1. *Bioinformatics*, **34**, 2575–2580.

Matsuda,S. *et al.* (2005) A novel representation of protein sequences for prediction of subcellular location using support vector machines. *Protein Sci.*, **14**, 2804–2813.

Nagarajan,V. *et al.* (2006) A fourier transformation based method to mine peptide space for antimicrobial activity. *BMC Bioinformatics*, **7**, 1–8.

Rhee,S.-Y. *et al.* (2003) Human immunodeficiency virus reverse transcriptase and protease sequence database. *Nucleic Acids Res.*, **31**, 298–303.

Rhee,S.-Y. *et al.* (2006) Genotypic predictors of human immunodeficiency virus type 1 drug resistance. *Proc. Natl. Acad. Sci. USA*, **103**, 17355–17360.

Rizzo,R. *et al.* (2016) Classification experiments of DNA sequences by using a deep neural network and chaos game representation. In: *Proceedings of the 17th International Conference on Computer Systems and Technologies 2016*. ACM, pp. 222–228.

Robin,X. *et al.* (2011) PROC: an open-source package for r and s+ to analyze and compare roc curves. *BMC Bioinformatics*, **12**, 77.

Sing,T. *et al.* (2005) Rocr: visualizing classifier performance in r. *Bioinformatics*, **21**, 7881.

Solis,A.D. and Rackovsky,S. (2000) Optimized representations and maximal information in proteins. *Proteins Struct. Funct. Bioinform.*, **38**, 149–164.

Spänig,S. and Heider,D. (2019) Encodings and models for antimicrobial peptide classification for multi-resistant pathogens. *BioData Min.*, **12**, 7.

Strichartz,R.S. (2000) Evaluating integrals using self-similarity. *Am. Math. Mon.*, **107**, 316–326.

Tzanov,V. (2015) Strictly self-similar fractals composed of star-polygons that are attractors of iterated function systems. arXiv preprint arXiv: 1502.01384.

Wang,D. and Larder,B. (2003) Enhanced prediction of lopinavir resistance from genotype by use of artificial neural networks. *J. Infect. Dis.*, **188**, 653–660.

Wang,Y. *et al.* (2005) The spectrum of genomic signatures: from dinucleotides to chaos game representation. *Gene*, **346**, 173–185.

Yang,J.-Y. *et al.* (2009) Prediction of protein structural classes by recurrence quantification analysis based on chaos game representation. *J. Theor. Biol.*, **257**, 618–626.

Yu,X. *et al.* (2013) Sparse representation for hiv-1 protease drug resistance prediction. In: *Proceedings of the 2013 SIAM International Conference on Data Mining*. SIAM, pp. 342–349.

Yu,Z.-G. *et al.* (2004) Chaos game representation of protein sequences based on the detailed hp model and their multifractal and correlation analyses. *J. Theor. Biol.*, **226**, 341–348.