Chaos Game Representations & Deep Learning for Proteome-Wide Protein Prediction

Kevin Dick
Department of Systems & Computer Engineering
Carleton University
Ottawa, Canada
kevin.dick@carleton.ca

James R. Green
Department of Systems & Computer Engineering
Carleton University
Ottawa, Canada
jrgreen@sce.carleton.ca

***Abstract*—Chaos Game Representation (CGR) is an emerging means of visualising and representing genomic and proteomic sequences. There exist many open questions related to its effective application to various computational tasks. In this work, we begin to address some of these questions by comparing four variants of the Chaos Game to generate CGR imagery as part of a multi-class classification task to identify the source organism for a given protein. We propose a novel nodal configuration for icosagon and 20-flake CGRs. Using two datasets, we performed fine-tuning using seven deep convolutional neural network (CNN) architectures and report modest performance over random among the 56 test conditions, highlighting certain shortcomings in effectively leveraging CGR in conjunction with deep CNN architectures. Many of the insights from this work will serve to orient subsequent protein-related studies involving CGR-based encoding and be generally applicable to disparate domains seeking to leverage CGR for sequence-type data.**

1. Introduction

The task of classifying proteins into various categories is a major challenge in the field of bioinformatics and effective models contribute to greater insight into the fundamentals of molecular biology. Such tasks may involve predicting whether or not any two proteins are likely to interact [1], predicting protein solubility [2], classification of functional families of proteins [3], or predicting what strain of HIV-1 a given protein may belong to [4]. Common to each of these cited examples is the use of an increasingly popular means of encoding a given protein sequence into a machine-readable format: the Chaos Game Representation (CGR).

Diverse approaches to encoding protein sequences exist including sparse encoding [5], position-specific scoring matrices (PSSM) [6], application of the Fourier Transform [7], or the use of general numerical descriptors [8]. Through the 1980s, the field of physics known as *non-linear dynamics*, *chaotic dynamical systems*, or *chaos* garnered much interest and, in 1990, Dr. H. Joel Jeffery proposed the use of CGR to visually depict gene structure [9]. Specifically, the Chaos Game is an interated function system (IFS) which allows the visual representation of the fractal structure of a sequence of possibly infinite length; that is, a one-dimensional sequence of arbitrary length is encoded into a two dimensional CGR-space with a desirable property that each point in CGR-space encodes both local and global sequence information.

A classical example of the CGR is to consider $n = 3$ distinct and non-co-linear nodes on a plane, assigning to each a pair of outcomes from a six-sided die (*e.g.* node 1: [1,2], node 2: [3,4], node 3: [5,6]). With an initial starting point equidistant from the three nodes, the die is repeatedly cast to generate a sequence of random numbers. For each outcome, a line from the current position is drawn to the node assigned that outcome and the current position is moved to the midpoint of that line. Repeating this process *ad infinitum* results in the famed fractal known as the Sierpiński triangle.

Jeffrey (1990) first proposed the application of the CGR algorithm to genomic sequences comprising an $n = 4$ nodal square with each vertex assigned one character from the alphabet $\sum = \{A, C, G, T\}$; resultant fractals are, therefore, constructed from squares rather than $n = 3$ triangles [9]. As in the example above, an initial point is set equidistant to the four nodes and, for each character in a genomic sequence, a line is drawn from the current point to the corresponding node and the current point is moved to the midpoint of that line, setting a new point in the plane. This is repeated until the sequence is exhausted; see Fig. 1 for a simple example.

The use of $n = 4$ CGRs of genomic sequences has been increasingly studied through the last two decades and has been found to be effective to several tasks including the prediction of phylogenetic relationships [10] and bacterial classification into distinct phyla, order, family, and genus [11]. The latter leveraged deep convolutional neural networks (CNNs) in their work. With the dramatic improvement in performance of CNN through the last decade and the introduction of diverse network architectures, CGR-based imagery promises to be increasingly useful in the classification of sequential information beyond $n = 4$ character alphabets.

Of particular relevance to this work is the CGR-based depiction of protein amino acid sequences. It is only during the last decade that efforts to meaningfully represent pro-

2471-7819/20/$31.00 ©2020 IEEE
DOI 10.1109/BIBE50027.2020.00027

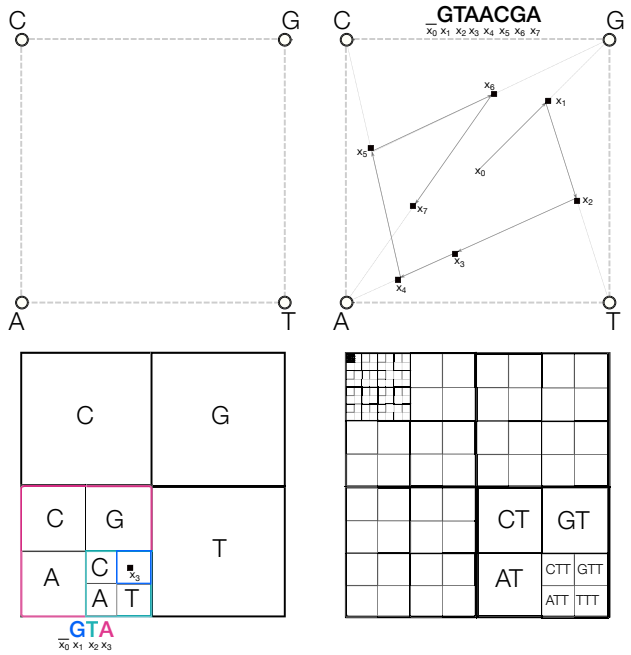

Figure 1. **Pictoral Representation of the the Chaos Game Representation for an Example Sequence.** Top-left: the initial configuration of assigned characters. Top-right: an annotated CGR of the sequence "GTAACGA" showing the construction lines (light-dashed) and travel lines (dark arrows). Bottom-left: multi-scale representation of how the subsequence "GTA" is encoded in a single position within CGR-space. Bottom-right: visualization of how increased pixel resolution of an image captures subsequences of increased length.

teomic sequences by means of CGR have been undertaken. With an $n = 20$ alphabet of amino acids, the meaningful representation of these sequences is still an open question and under continued exploration.

One method is to produce four groups of amino acids sharing similar properties and playing the Chaos Game, akin to the $n = 3$ example above where multiple outcomes of the die are assigned to the same node. This method was used by Yu *et al.* to construct a phylogenetic tree of Archaea and Eubacteria [12].

Another method to project the $n = 20$ alphabet into an $n = 4$ alphabet is to leverage codon mapping. While tri-nuclotide sequences have a many-to-one mapping to an amino acid, it is non-trivial to *reverse translate* an amino acid sequence back to it's original nucleotide sequence. However, Deschavanne & Tuffery proposed a *fixed* reverse translation mapping [3] useful in converting amino acid sequences back into a genomic sequence while conserving performance in predicting protein functional families. Their mapping, also used in this work, is depicted in Table 1.

Finally, we might relax the assumption that an amino acid sequence of $n = 20$ need necessarily be manipulated into $n = 4$ and, rather, redefine the CGR-space such that it accommodates n-gons with node number $n > 4$. For example, much like the work above, Basu *et al.* mapped the amino acids into twelve groups and played a variant of the Chaos

TABLE 1. FIXED REVERSE TRANSLATION MAPPING FROM DECHAVANNE & TUFFERY [3].

1st Base	2nd Base T		2nd Base C		2nd Base A		2nd Base G		3rd Base
T	TTT	F	TCT	S	TAT	Y	TGT	C	**T**
	TTC		TCC		TAC		TGC		**C**
	TTA	L	TCA		TAA	STOP	TGA	STOP	**A**
	TTG		TCG		TAG		TGG	W	**G**
C	CTT	L	CCT	P	CAT	H	CGT	R	**T**
	CTC		CCC		CAC		CGC		**C**
	CTA		CCA		CAA	Q	CGA		**A**
	CTG		CCG		CAG		CGG		**G**
A	ATT	I	ACT	T	AAT	N	AGT	S	**T**
	ATC		ACC		AAC		AGC		**C**
	ATA		ACA		AAA	K	AGA	R	**A**
	ATG	M	ACG		AAG		AGG		**G**
G	GTT	V	GCT	A	GAT	D	GGT	G	**T**
	GTC		GCC		GAC		GGC		**C**
	GTA		GCA		GAA	E	GGA		**A**
	GTG		GCG		GAG		GGG		**G**

The shading indicates the selected codon for each amino acid.

Game within this $n = 12$ regular dodecagon to generate representations [13]. It is important to make a distinction here that playing the Chaos Game on a regular polygon doesn't strictly enforce the self-similar fractal relationship between points that exists in the $n = 4$ representation and to explicitly differentiate these representations, we will refer to them as CGR*. Nonetheless, the n-gon CGR* have been found to be useful in representing amino acid sequences [4], [13]. For the $n = 20$ amino acid alphabet, the 20-sided polygon, known as an *icosagon* is ideally suited.

Promisingly, in 2015, Tzanov generalized the CGR for $n > 4$, upwards to ∞-gons and ∞-flakes [14]. The distinction between an n-gon and an n-flake is that the latter is strictly non-overlapping (*i.e.* variable scaled n-flakes are precisely separable from one another). This generalization enables the creation of strictly self-similar n-flakes, with the most recent protein-based application of these leveraging the $n = 20$-flake by Löchel *et al.*, finding them to be effective in classifying proteins between various strains of HIV-1 [4]. As part of their work, the "kaos" R library was made publicly available to generate n-flakes of arbitrary alphabets to create CGRs as well as *frequency matrix* CGRs (FCGRs) which colour points on a greyscale based on the k-mer frequency within a sequence (as opposed to the binarized representation of a CGR). We denote the class of all possible types of Chaos Game variants discussed in this work as (F)CGR(*).

While there exist promising new directions for the utility of (F)CGR(*)-based representation in protein-related tasks, several open questions and challenges remain:

1) **What is the ideal relative assignment of amino acids to nodes?** There exist on the order of 20! possible combinations of nodal representations, with no clear process for selecting the optimal representation for a particular task. Any permutation of nodes will lead to a unique CGR(*) while encoding the same sequential information.

2) **Does the $n = 20$ (F)CGR(*) of an amino acid sequence encapsulate more information than it's reverse transcribed $n = 4$ variant?** More specifically, can a *fixed* reverse translation mapping suffice?
3) **Does the FCGR(*) improve performance over the CGR(*)?** Normalizing each pixel by subsequence frequency encodes greater information as a trade-off with pixel brightness.
4) **What is the minimum sequence length required before an (F)CGR(*) becomes useful in distinguishing between various classes?** The majority of biological applications have applied $n = 4$ (F)CGR(*) to an organism's entire genome. Are these representations effective on the individual protein level?
5) **Can transfer learning be used to effectively fine-tune a CNN to process (F)CGR(*) representations?** Most models are trained on ImageNet which contains imagery fundamentally different from those encoded by an (F)CGR(*).

In an effort to begin addressing a number of these open questions, here, we propose a new relative node orientation for icosagonal and 20-flake (F)CGR(*)s of protein sequences and evaluate the performance of several fine-tuned deep CNNs in distinguishing proteins between a set of evolutionary diverse organisms. We compare the performance using four different (F)CGR(*)s and our findings promise to orient future research in determining how (F)CGR(*)s might effectively be used in both related and disparate applications.

2. Data & Methods

We first collected proteome-wide sequences for twelve organisms of varying evolutionary distance. Two datasets were then created, the first comprised a stratified sample of 1,000 proteins for 11 organisms ("strat-1K"; Table 2) and the second comprised a stratified sample of 5,000 protein for 6 organisms ("strat-5k"; Table 2). For each protein, we then generated four different (F)CGR(*) representations and split these into distinct training, validation, and independent test sets according to a 60-20-20 split. Each representation was then used to fine-tune seven different pretrained deep CNN models and the performance of each representation using each model was reported.

2.1. Obtaining Proteome-Wide Sequences

To collect proteome-wide sequences for numerous organisms, we first selected twelve model organism with varying evolutionary distance. The twelve organisms and and their phylogenetic relationship are visualized in Fig. 2. We downloaded the twelve reference proteomes from UniProt, selecting for only "Reviewed" proteins resulting in the number of proteins tabulated in Table 2. As this work focuses on the traditional $n = 20$ amino acid alphabet,

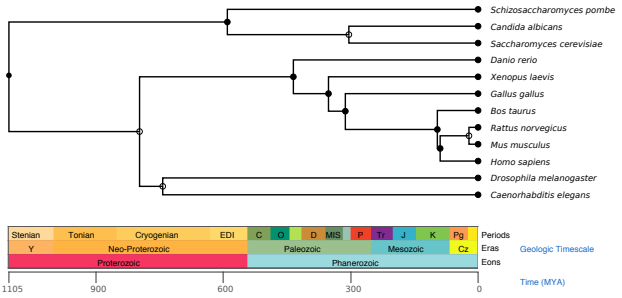

Figure 2. **Phylogenetic Tree of the Estimated Evolutionary Distance between the twelve Organisms considered.** Generated using timetree.org.

we discarded from each proteome a negligible few proteins whose sequences contained non-standard or ambiguous amino acids. To ensure that we had a sufficient minimum number of samples per organism, we restricted ourselves to considering only those organisms for which at least 1,000 proteins were available, eliminating *Xenopus laevis*, the African clawed frog. A second dataset was created for the six organisms with at least 5,000 proteins (the bold organisms in Table 2). To generate a balanced dataset (to account for dramatically differing proteome sizes) we obtained a stratified random sample of 1,000 proteins for each organism and applied a 60%, 20%,20% split: 600-200-200 proteins for training-validation-testing per organism. We repeated this procedure for the second dataset: 3000-1000-1000 proteins for training-validation-testing per organism. This stratified sampling did not explicitly account for isoforms, however, future research might generate more homogenous and evolutionary-focused datasets.

2.2. Generating the Chaos Game Representations

In this work, we consider four different ways of encoding a protein sequence into an (F)CGR(*) image. Of particular significance to this process is the selection of the image resolution. In order to benefit from fractal self-similarity, the image resolution should be selected such that each pixel encodes a subsequence pattern and that its neighbours encode a variant of that subsequence. Formally,

TABLE 2. MODEL ORGANISMS CONSIDERED BASED ON EVOLUTIONARY DISTANCE

Organism	Num. Proteins	Proteome ID
H. sapiens	20,353	UP000005640
M. musculus	17,038	UP000000589
R. norvegicus	8,106	UP000002494
S. cerevisiae	6,049	UP000002311
B. taurus	6,012	UP000009136
S. pombe	5,140	UP000002485
C. elegans	4,128	UP000001940
D. melanogaster	3,581	UP000000803
D. rerio	3,158	UP000000437
G. Gallus	2,295	UP000000539
C. albicans	1,007	UP000000559
X. laevis	43	UP000186698

for the $n = 4$ genomic sequence encoding example, if the CGR image has a resolution of $2^k \times 2^k$ pixels, then each pixel represents a distinct DNA subsequence of length k. To generate the CGR representation, a pixel is coloured white if its corresponding subsequence occurs in the DNA sequence, and it is coloured black otherwise. To generate the FCGR, each pixel's k-mer *frequency of occurrence* in the DNA sequence is determined and then these pixels are Min-Max normalized into a greyscale colour with the maximum occurrence receiving the colour white and no occurrence as black; all other occurrence values are made a relative colour of grey. This is an adaptation of the original CGR where instead of directly coloring a pixel white, we instead increment the frequency of occurrence in a given channel and, upon sequence exhaustion, then Min-Max normalize all frequencies of occurrence into the relative greyscale range $[0, 255]$ over all channels.

We selected $k = 9$ to generate images of size 512 pixels. Each pixel, thus, encodes a 9-mer which, in turn, encodes three amino acids. While this pixel-based self-similarity is lost as we consider $n > 4$, for the sake of consistency, we choose to represent the icosagonal $n = 20$ (F)CGR(*)s as 512×512 pixel images. Moreover, the majority of deep learning models require a minimum of 256×256 pixel images and traditional data manipulation techniques (*e.g.* scaling and cropping) risk destroying some of the information encoded within an (F)CGR(*) image. The four representations considered are:

1) Original 4-node formulation of CGR (DNA sequences)
2) Proposed 20-node nodal orientation of CGR* (amino acid sequences)
3) Proposed 20-node nodal orientation of FCGR* (amino acid sequences)
4) A 20-flake FCGR using the proposed nodal orientation (amino acid sequences)

For the first FCGR, since we are leveraging the $n = 4$ representation which requires a DNA input sequence, we applied the fixed reverse translation mapping proposed by Deschavanne & Tuffery [3], (Table 1). The nucleotides are assigned alphabetically and clockwise starting from the top-left.

For the remaining representations, our proposed relative nodal configuration , illustrated in Fig. 3, was used to arrange the $n = 20$ icosagon to generate both a CGR* and and FCGR*. We first separated amino acids into polar and non-polar groups. Of the polar amino acids, we further separated them into three groups by charge; negative, neutral, and positive. Within these four groups, we then sorted the amino acids by decreasing molecular weight. This created a relative ordering of amino acids about the icosagon and we then rotated this ordering such that Methionine was located at the top node. An illustration of this proposed configuration is provided in Fig. 3.

Finally, the 20-flake representation of each sequence was produced using the Kaos R library developed by Löchel *et al.* as part of their work in [4]. We parameterized the

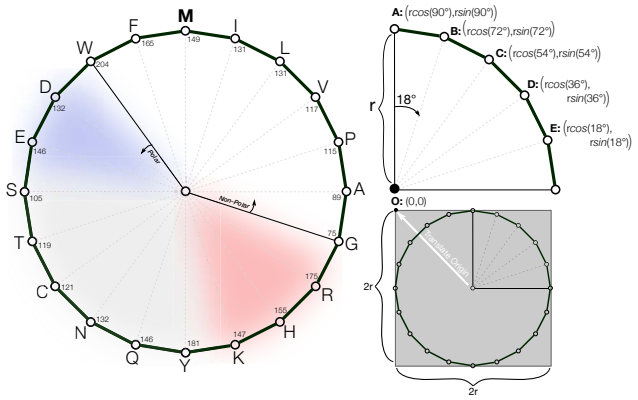

Figure 3. **Proposed Relative Orientation of Amino Acids within an Icosagon.** Polar (coloured) amino acids are separated from non-polar amino acids (white); the polar amino acids are arranged into negative (blue), neutral (grey), and positive (red) groups; each group is then sorted in clockwise order from highest molecular weight to smallest within that group. The two right-most figures depict how an icosagon can bee inscribed within a square parameterized by width of $2r$.

nodal orientations to match our proposed configuration and produced 512×512 pixel FCGRs.

2.3. Comparison of Deep Learning Architectures

To extensively evaluate the utility of (F)CGR(*)s as part of a multi-class fine-tuning task, we leveraged seven pretrained deep CNN architectures. To account for model size, we chose three ResNet variants of variable model complexity: ResNet-18, ResNet-50, and ReseNet-152 from [15]. We also used the 16-layer VGG model (configuration "D") with batch normalization from [16], the SqueezeNet 1.1 model as a more efficient version of SqueezeNet 1.0 from [17], the Densenet-201 model from [18], and finally, the original AlexNet from [19].

For each of the seven models, we freeze the feature extracting layers and fine-tune the final layer using each the four (F)CGR(*) representations over 100 epochs with a batch size of 4 to determine which (if any) of the representation and model combinations resulted in superior performance. We report the 11-class accuracy of each model (strat-1k dataset) and the 6-class accuracy (strat-5k dataset), noting that a random model would produce an accuracy of $9.\overline{09}\%$ for the former and $16.\overline{6}\%$ for the latter.

3. Results & Discussion

In this work we sought to compare four protein sequence-based representations using variants of the Chaos Game. To visualize an example of each of the (F)CGR(*)s, we illustrate the Midasin protein from the *S. cerevisiae* proteome as one of the larger proteins which facilitates the visualization of the sparse pixel-level points of each Chaos Game (Fig. 4). We note that Fig. 4B and Fig. 4C represent the same pixel-level pattern; however, where the CGR* is

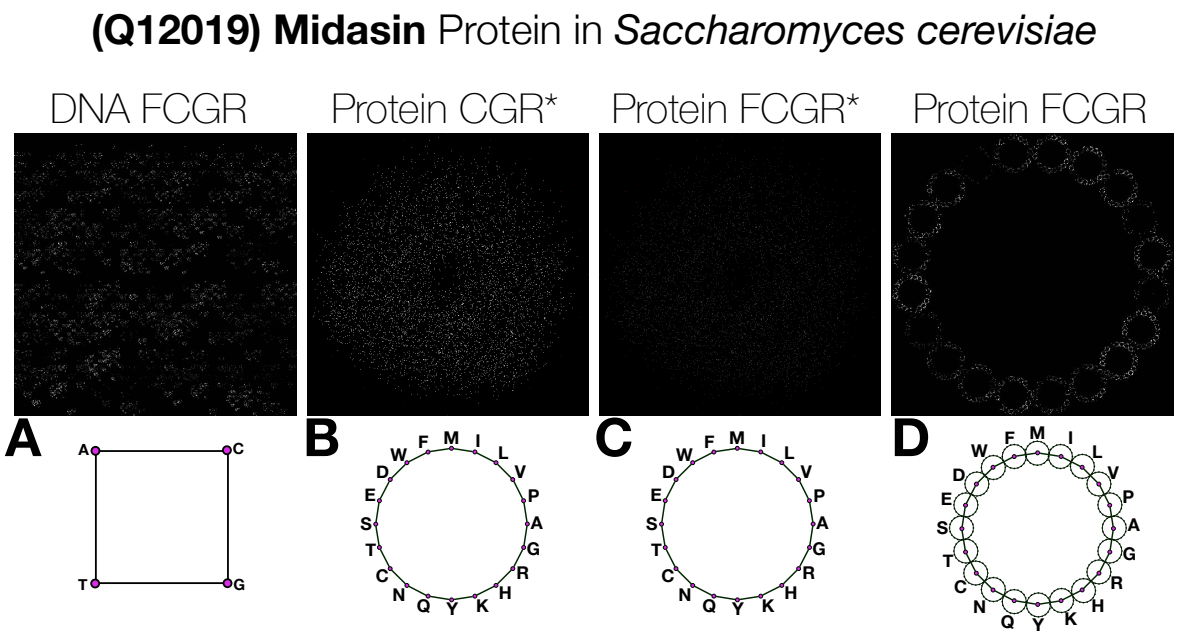

Figure 4. **Example of each of the Four (F)CGR(*) Representations Compared.** The Midasin protein sequence (Q12019) is depicted as a 512×512 pixel image in each case. First as a DNA-based CGR in A, a 20-node amino acid CGR* in B, a 20-node amino acid FCGR* in C, and a 20-flake FCGR in D. Below each image is the reference relative nodal orientations; each of the protein amino acid representations use the nodal orientation proposed in this work.

binary, the pixels in the FCGR* are Min-Max normalized by frequency.

We hypothesized that the 20-flake representation, as a strictly self-similar fractal, would likely outperform the other representations, particularly as it was found by Löchel *et al.* to be an encoding well suited to deep neural networks with evidenced superiority over competing encodings [4]. While the proposed icosagonal (F)CGR*s in Fig. 4B and Fig. 4C will capture positional information, as non-strictly-fractal representations, the information encoded in coloring any one pixel is not scale-free as with the Fig. 4A and Fig. 4D FCGR representations.

We tabulated the performance across the 28 test conditions for each dataset in Tables 3 and 4 noting that, generally, the majority of representation-model pair appear only to achieve a modest accuracy. In fact, both the AlexNet and SqueezeNet models for each dataset do not appear to have learned the task at all and report results equivalent to a random model.

While ResNet-18 has the highest average model performance for both datasets and appears to have benefited from all representations, it is the VGG-16 model with Batch Normalization that has the highest reported accuracy using the 20-flake representation in both cases. Interestingly, both the $n = 4$ DNA FCGR and $n = 20$ amino acid FCGR share the highest average performance among the representations over both datasets. Additionally, contrary to expectation, while the $n = 20$ amino acid FCGR* theoretically contains greater information by capturing the relative frequency of pixels, the binarized $n = 20$ ammino acid CGR* had a larger average performance across the models for both datasets. We posit that this may be due to the increased brightness of the CGR* as compared to the FCGR*. Given that these representations are incredibly sparse and the fine-tuning is being performed upon models pretrained on ImageNet, it may be that generally brighter images would increase performance. This hypothesis may be tested by representing the (F)CGR(*) on coarser-grained resolution imagery to both reduce sparsity and increase overall image brightness.

By representing sequences at the individual protein level, as opposed to representing entire genomes or entire proteomes, the resultant imagery is commensurately darker with considerably less information available from which a model might learn. To evaluate this hypothesis, future work might consider reducing the image resolution to generate more densely represented (F)CGR(*)s.

Among the three ResNet models, it appears that smaller and less complex models lead to higher performance. Given the modest number of training samples available within each class, it is expected that a less complex model may be better suited, as seen with both the ResNet-18 and VGG-16_BN (Table 3 and 4).

With the highest-performing model and representation combination (VGG-16_BN and the 20-flake FCGR), we inspected the training accuracy and loss with respect to each training epoch for both datassets. Interesting, a consistent improvement in the validation performance over the training performance is observed suggestive that the sampled validation set consists of "easier" samples than those in the training set (Fig. 5). This pattern persists over the majority

TABLE 3. MODEL ACCURACY FOLLOWING 100 EPOCHS OF TRAINING USING THE 1K STRATIFIED SAMPLED DATASET (11 CLASSES)

Model	DNA FCGR (4-node)	AA CGR* (20-node)	AA FCGR* (20-node)	AA FCGR (20-flake)	**Model Average**
resnet50	0.1214	0.1268	0.1127	0.1127	**0.1184**
densenet121	0.1373	0.1232	0.1136	0.1077	**0.1205**
resnet152	0.1145	0.1150	0.1168	0.1223	**0.1172**
resnet18	0.1673	0.1505	0.1491	0.1505	**0.1543**
squeezenet	0.0909	0.0955	0.0936	0.0909	**0.0927**
vgg16_bn	0.1273	0.1446	0.0982	0.1750	**0.1363**
alexnet	0.0909	0.0909	0.0909	0.0909	**0.0909**
Repr. Average	**0.1214**	**0.1209**	**0.1107**	**0.1214**	0.1186

Cell colour emphasises relative performance; the darker the better.
Bold denotes the column-wise (representation) and row-wise (model) averages.
Vertical lines denote most performant model.

TABLE 4. MODEL ACCURACY FOLLOWING 100 EPOCHS OF TRAINING USING THE 5K STRATIFIED SAMPLED DATASET (6 CLASSES)

Model	DNA FCGR (4-node)	AA CGR* (20-node)	AA FCGR* (20-node)	AA FCGR (20-flake)	**Model Average**
resnet50	0.2378	0.2605	0.2465	0.2842	**0.2573**
densenet121	0.2640	0.2575	0.2557	0.2730	**0.2626**
resnet152	0.2562	0.2028	0.2447	0.2947	**0.2496**
resnet18	0.3120	0.2567	0.2648	0.3092	**0.2857**
squeezenet	0.1667	0.1667	0.1667	0.1668	**0.1667**
vgg16_bn	0.2257	0.2480	0.1667	0.3183	**0.2397**
alexnet	0.1667	0.1667	0.1667	0.1667	**0.1667**
Repr. Average	**0.2327**	**0.2227**	**0.2160**	**0.2590**	0.2326

Cell colour emphasises relative performance; the darker the better.
Bold denotes the column-wise (representation) and row-wise (model) averages.
Vertical lines denote most performant model.

of models and each of the representations. This effect could be mitigated by instead leveraging a k-fold cross-validation.

Finally, to better understand why the test accuracy is only modest, we look at the confusion matricies themselves (Fig. 6). It is immediately apparent that the majority of the samples of the strat-1k dataset are being predicted to belong to *D. melanogaster*. A number of expected errors are also apparent for each dataset. Many of the *M. musculus* errors are attributed to evolutionarily proximal organisms such as *R. norvegicus*, *H. sapiens*, and *B. taurus*. Similarly, many of the *S. cerevisiae* errors are attributed to *S. pombe* and *C. albicans*. The converse is also true for *C. albicans*. The model performed the best for *D. melanogaster*, *C. albicans*, *M. musculus*, and *S. cerevisiae* and failed to effectively classify the proteins for the remaining seven organisms. Combining *S. cerevisiae* with *C. albicans*, these three groupings are all evolutionarily distant from one another suggestive that the dataset may not be sufficiently large to represent more distinct evolutionary relationships, making this task much more difficult. This hypothesis is evidenced in the strat-5k results where the majority of the errors between *S. cerevisiae* are attributed to *S. pombe* and vice versa, and seen also among the four mammals. Many prior studies applying CGR have achieved considerably improved performance by both leveraged denser CGRs and considered more evolutionary proximal organisms [4], [10].

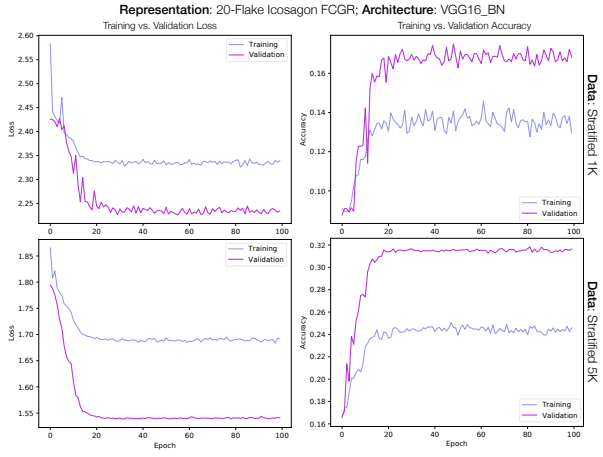

Figure 5. **Training and Validation Performance of the Most Performant Models over 100 Epochs.** The VGG16_BN pretrained models were fine-tuned using the the 20-flake icogagon representation and rapidly converged within the first 20 epochs.

3.1. Limitations, Insights, & Future Directions

This work only achieved modest model performance and is unlikely to be directly useful to the task of differentiating proteins between various organisms; however, both the limitations and insights gleaned from this work shed light on the use of CGR as part of related protein-based research as well as their general applicability to other sequence-type information. Below we list the promising future directions and notable limitations of this work:

1. **Limitation: This work relies upon a representation that is, by definition, minimalist**. The task of distinguishing to which organism a given protein belongs is abstracted into CGR-space with no information about the protein sequence itself, physical or chemical properties of the amino acids, or context about the protein. While the resultant accuracies from these datasets may not appear impressive, when compared to the accuracy expected from a random classifier, it becomes apparent that many of the

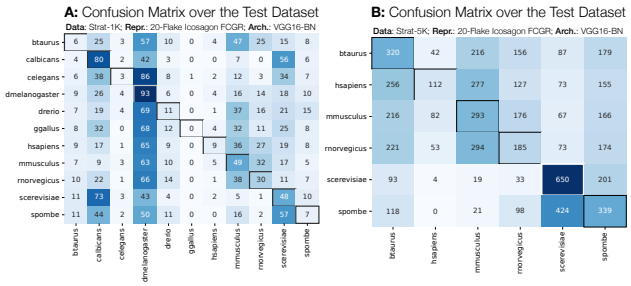

Figure 6. **Confusion Matricies of the Most Performant Models & Representations.**

models were, in fact, able to learn salient features from these representations.

2. **Limitation: This work leveraged pretrained deep learning models originally developed to interpret full-colour images.** In the vast majority of ImageNet samples, the semantic meaning of pixels are often densely clustered in a region of the image (*e.g.* a cat is represented as a connected patch of pixels and not a smattering of disjoint pixels). The (F)CGR(*) of a protein sequence is fundamentally different than those images used in the prototypical fine-tuning of classifiers for multi-class classification tasks. Consequently, only modest performance should be expected from transfer learning. Future work will investigate a fully trained-from-scrath model and additionally explore means of augmenting the (F)CGR(*) datasets (to generate much larger training datasets) such that end-to-end training of these deep model architectures might capture the salient features of the sparse and disjoint representations. A translation-invariant deep CNN is also expected to better capture the position-specific information encoded within CGR-space.

3. **Future Work: An open question in leveraging the (F)CGR(*) representations for amino acid sequences is the determination of the appropriate assignment of amino acids to nodes.** At present, the consensus view is that, given the tremendously large space of possible combinations of orientations (on the order of 20!; vastly more configurations than there are atoms in the universe), selection of an arbitrary configuration (such as alphabetical ordering) is reasonable. In this work we propose a new orientation informed by the physical and chemical characteristics of each amino acid to approximately cluster them about the icosagon. Furthermore, we propose a paradigm shift in the thinking that the enormous nodal-configuration-space is a weakness to CGR-based methods and, rather, view it as a not-yet exploited means of data augmentation. That is, in permuting the nodal orientations, we can generate unique and complimentary images representative of the same sequence. Intuitively, this might be considered analogous to the traditional data augmentation techniques used in computer vision (rotation, cropping, mirroring, scaling, *etc.*). This form of augmentation is amenable to CGR-space while fully preserving sequence information; the traditional data augmentation techniques would destroy some amount of information.

4. Conclusion

The use of Chaos Game Representation has been demonstrated for a growing number of bioiniformatic applications and several open questions must be explored to effectively apply CGR to each task. In this work, we sought to address some of these questions by comparing four variants of the Chaos Game to generate (F)CGR(*) imagery used as part of an 11-class and 6-class classification task to assign a given protein to an organism. Only modest performance was observed over the 56 test conditions, highlighting certain shortcomings in effectively leveraging CGR. Many of the insights from this work promise to orient subsequent studies.

References

[1] J. Jia, X. Li, W. Qiu, X. Xiao, and K.-C. Chou, "ippi-pseaac (cgr): Identify protein-protein interactions by incorporating chaos game representation into pseaac," *Journal of theoretical biology*, vol. 460, pp. 195–203, 2019.

[2] N. Xiaohui, S. Feng, H. Xuehai, X. Jingbo, and L. Nana, "Predicting the protein solubility by integrating chaos games representation and entropy in information theory," *Expert systems with applications*, vol. 41, no. 4, pp. 1672–1679, 2014.

[3] P. Deschavanne and P. Tuffery, "Exploring an alignment free approach for protein classification and structural class prediction," *Biochimie*, vol. 90, no. 4, pp. 615–625, 2008.

[4] H. F. Löchel, D. Eger, T. Sperlea, and D. Heider, "Deep learning on chaos game representation for proteins," *Bioinformatics*, vol. 36, no. 1, pp. 272–279, 2020.

[5] J. D. Hirst and M. J. Sternberg, "Prediction of structural and functional features of protein and nucleic acid sequences by artificial neural networks," *Biochemistry*, vol. 31, no. 32, pp. 7211–7218, 1992.

[6] S. Ahmad and A. Sarai, "Pssm-based prediction of dna binding sites in proteins," *BMC bioinformatics*, vol. 6, no. 1, p. 33, 2005.

[7] V. Nagarajan, N. Kaushik, B. Murali, C. Zhang, S. Lakhera, M. O. Elasri, and Y. Deng, "A fourier transformation based method to mine peptide space for antimicrobial activity," in *BMC bioinformatics*, vol. 7, no. S2. Springer, 2006, p. S2.

[8] Y. B. Ruiz-Blanco, W. Paz, J. Green, and Y. Marrero-Ponce, "Protdcal: A program to compute general-purpose-numerical descriptors for sequences and 3d-structures of proteins," *BMC bioinformatics*, vol. 16, no. 1, p. 162, 2015.

[9] H. J. Jeffrey, "Chaos game representation of gene structure," *Nucleic acids research*, vol. 18, no. 8, pp. 2163–2170, 1990.

[10] P. J. Deschavanne, A. Giron, J. Vilain, G. Fagot, and B. Fertil, "Genomic signature: characterization and classification of species assessed by chaos game representation of sequences." *Molecular biology and evolution*, vol. 16, no. 10, pp. 1391–1399, 1999.

[11] R. Rizzo, A. Fiannaca, M. La Rosa, and A. Urso, "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*, 2016, pp. 222–228.

[12] Z.-G. Yu, V. Anh, and K.-S. Lau, "Chaos game representation of protein sequences based on the detailed hp model and their multifractal and correlation analyses," *Journal of theoretical biology*, vol. 226, no. 3, pp. 341–348, 2004.

[13] S. Basu, A. Pan, C. Dutta, and J. Das, "Chaos game representation of proteins," *Journal of Molecular Graphics and Modelling*, vol. 15, no. 5, pp. 279–289, 1997.

[14] V. Tzanov, "Strictly self-similar fractals composed of star-polygons that are attractors of iterated function systems," *arXiv preprint arXiv:1502.01384*, 2015.

[15] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[16] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[17] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, "Squeezenet: Alexnet-level accuracy with 50x fewer parameters and¡ 0.5 mb model size," *arXiv preprint arXiv:1602.07360*, 2016.

[18] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4700–4708.

[19] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.