THE INTERNATIONAL NEURAL NETWORK SOCIETY (INNS)

A New Direction in Membranolytic Anticancer Peptides Classification: Combining Spaced k-mers with Chaos Game Representation.

Taslim Murad, Sarwan Ali, Murray Patterson

INNS 2023

Georgia State University

Table of Content

Motivation

- Membranolytic Anticancer Peptides (ACPs):
  - Small protein sequences.
  - Used as breast cancer therapeutics
    - delay cellular resistance development and eliminate some common chemotherapy challenges, like cytotoxicity, aftereffect, etc.
- Breast cancer is a leading cause of death among women globally.
- Identifying the potent ACPs is a vital contribution towards breast cancer treatment.

Introduction

- ❖ Proposes Chaos Graph Representation (CGR)-based method to convert ACPs into images for their analysis.
- ❖ Explored the use of secant and cosecant functions as an alternatives in CGR.
- ❖ Used Spaced k-mers of a sequence as an alternate way to manipulate its amino acids rather than the original sequence.
- ❖ Enabled the deep learning models application for ACPs analysis.

Existing Works

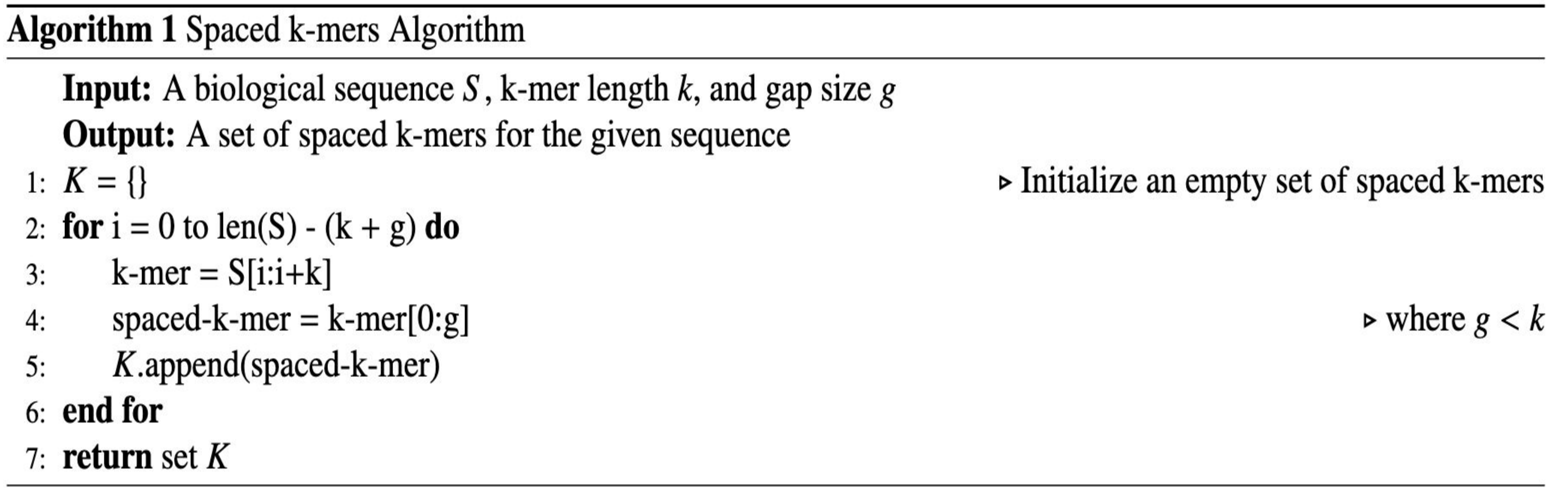
❖ Many works exits to perform protein sequence analysis and some of them are summarized as follow,

Methods	**Drawbacks**
One-hot encoding [2]	Sparsity and curse of dimensionality
WDGRL (neural network based) [3]	Require large training data to get optimal feature representation
K-mer based methods [4]	Sparsity and computationally expensive
FCGR (image based) [1]	Only 1-to-1 mapping of amino acids to pixels.

Proposed System

- Used spaced k-mers to get the images instead of original peptide sequences.
  - spaced k-mers provide more meaningful manipulation of amino acids, so they enable more information about the sequence to be captured in generated image.
  - Enhance predictive performance.
- Spaced k-mers are introduced to overcome the sparsity and high-dimensionality challenges associated with k-mers.
- We use one existing (P-CGR[1]) and 3 different novel strategies (Static, Random, RCGR) to do the image encoding of our peptide sequence data.

Proposed System



Algorithm 1 Spaced k-mers Algorithm

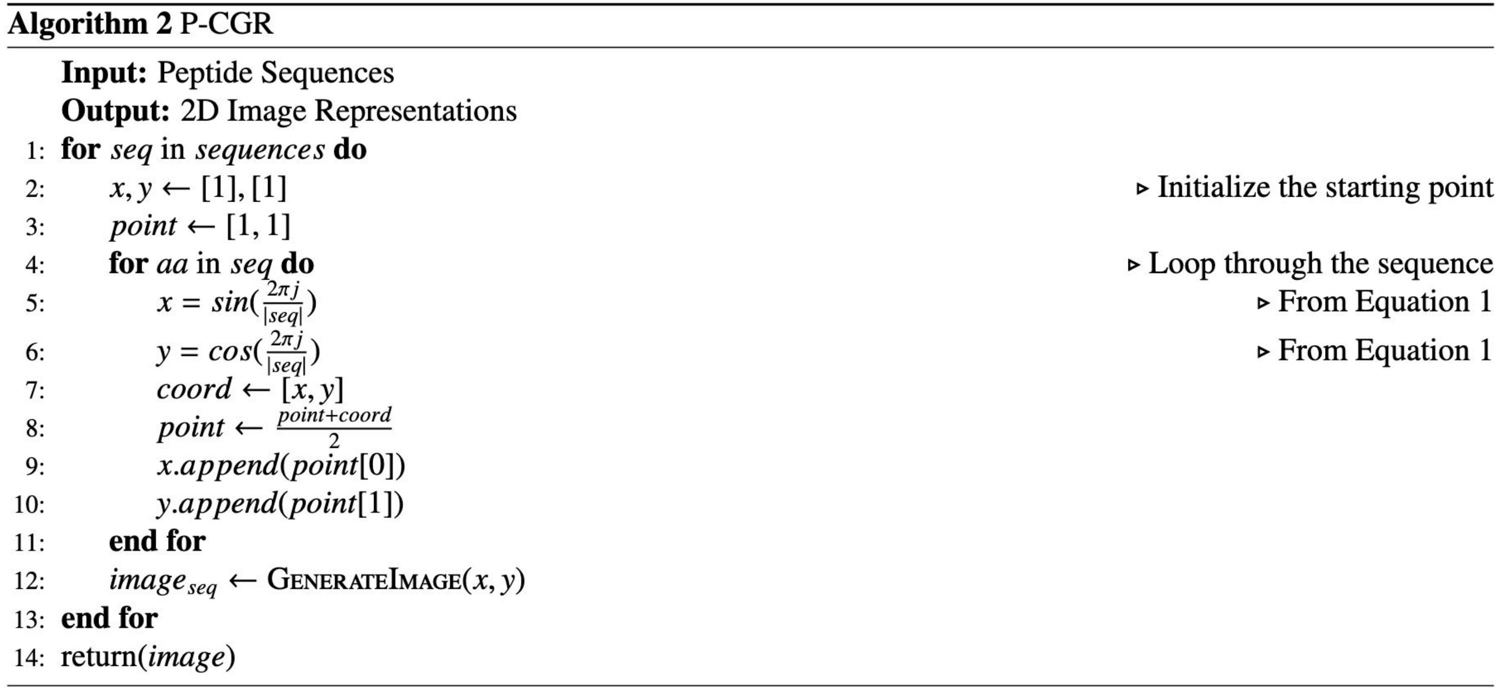
Input: A biological sequence S, k-mer length k, and gap size g

Output: A set of spaced k-mers for the given sequence

```
$K$ = {}                                  ▷ Initialize an empty set of spaced k-mers
for i = 0 to len(S) - (k + g) do
    k-mer = S[i:i+k]
    spaced-k-mer = k-mer[0:g]             ▷ where $g < k$
    $K$.append(spaced-k-mer)
end for
return set $K$
```

Proposed System: Protein-CGR[1]

❖ N-flakes based image generation method for protein sequences.

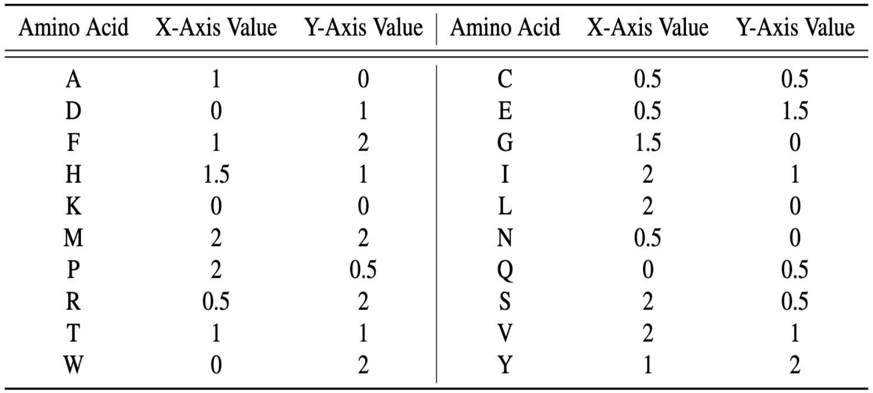


Algorithm 2 P-CGR

```
    Input: Peptide Sequences
    Output: 2D Image Representations
 1: for $seq$ in $sequences$ do
 2:     $x, y \leftarrow [1], [1]$                        ▷ Initialize the starting point
 3:     $point \leftarrow [1, 1]$
 4:     for $aa$ in $seq$ do                              ▷ Loop through the sequence
 5:         $x = sin(\frac{2\pi j}{|seq|})$               ▷ From Equation 1
 6:         $y = cos(\frac{2\pi j}{|seq|})$               ▷ From Equation 1
 7:         $coord \leftarrow [x, y]$
 8:         $point \leftarrow \frac{point+coord}{2}$
 9:         $x.append(point[0])$
10:         $y.append(point[1])$
11:     end for
12:     $image_{seq} \leftarrow \textsc{GenerateImage}(x, y)$
13: end for
14: return($image$)
```

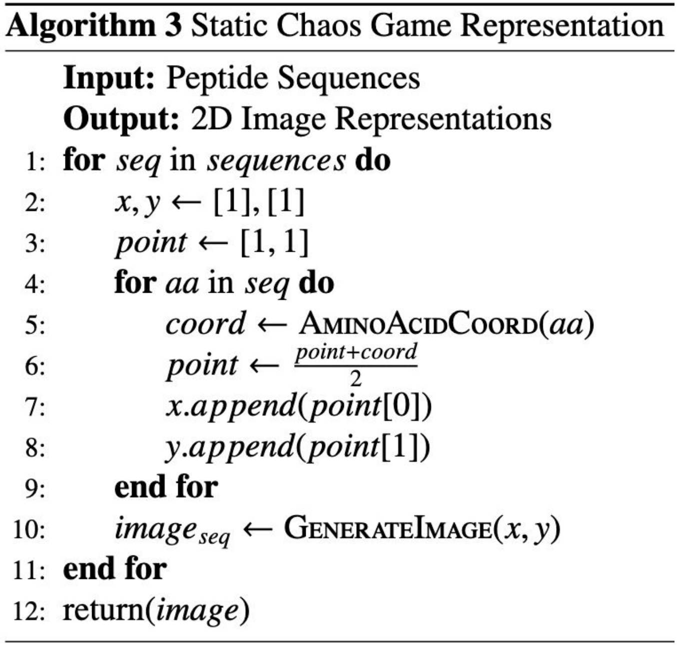
Proposed System: Static-CGR

❖ Employ a set of pre-defined axis values for each amino acid.



Amino Acid	X-Axis Value	Y-Axis Value	Amino Acid	X-Axis Value	Y-Axis Value
A	1	0	C	0.5	0.5
D	0	1	E	0.5	1.5
F	1	2	G	1.5	0
H	1.5	1	I	2	1
K	0	0	L	2	0
M	2	2	N	0.5	0
P	2	0.5	Q	0	0.5
R	0.5	2	S	2	0.5
T	1	1	V	2	1
W	0	2	Y	1	2

Table 1: Static Amino acids positions/coordinates for x and y axis in the 2D image.

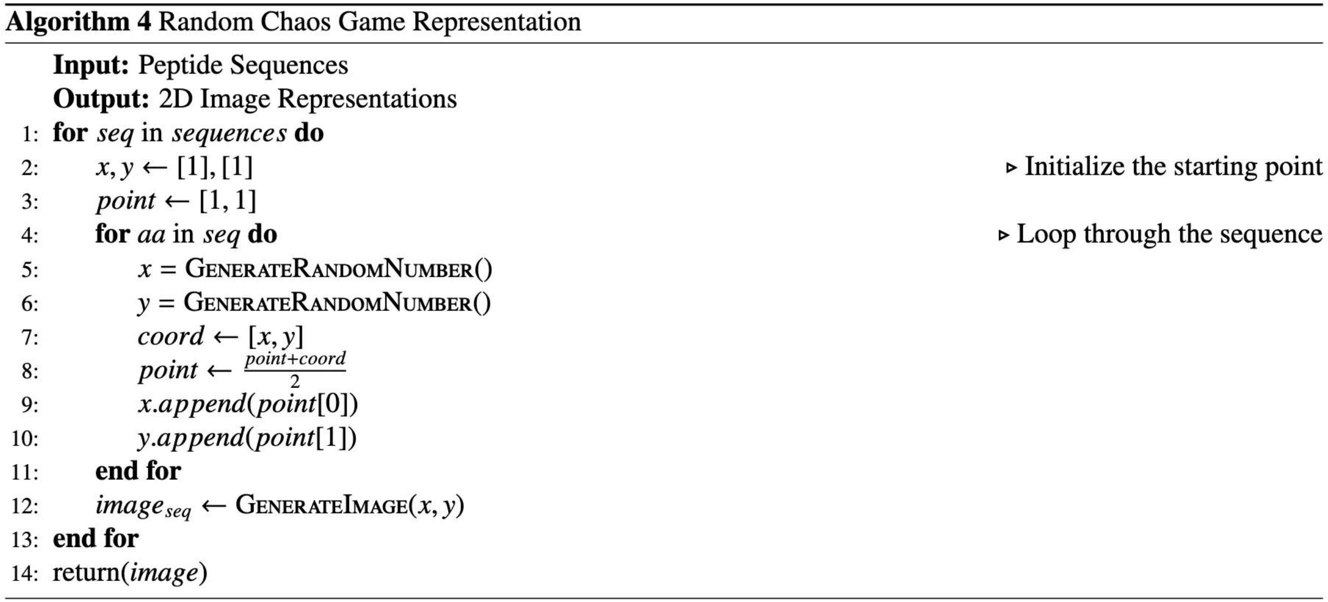


Algorithm 3 Static Chaos Game Representation

```
    Input: Peptide Sequences
    Output: 2D Image Representations
 1: for $seq$ in $sequences$ do
 2:     $x, y \leftarrow [1], [1]$
 3:     $point \leftarrow [1, 1]$
 4:     for $aa$ in $seq$ do
 5:         $coord \leftarrow \textsc{AminoAcidCoord}(aa)$
 6:         $point \leftarrow \frac{point+coord}{2}$
 7:         $x.append(point[0])$
 8:         $y.append(point[1])$
 9:     end for
10:     $image_{seq} \leftarrow \textsc{GenerateImage}(x, y)$
11: end for
12: return($image$)
```

Proposed System: Random-CGR

- Utilizes a random function (instead of using pre-defined static values) to assign axis values to any amino acid.

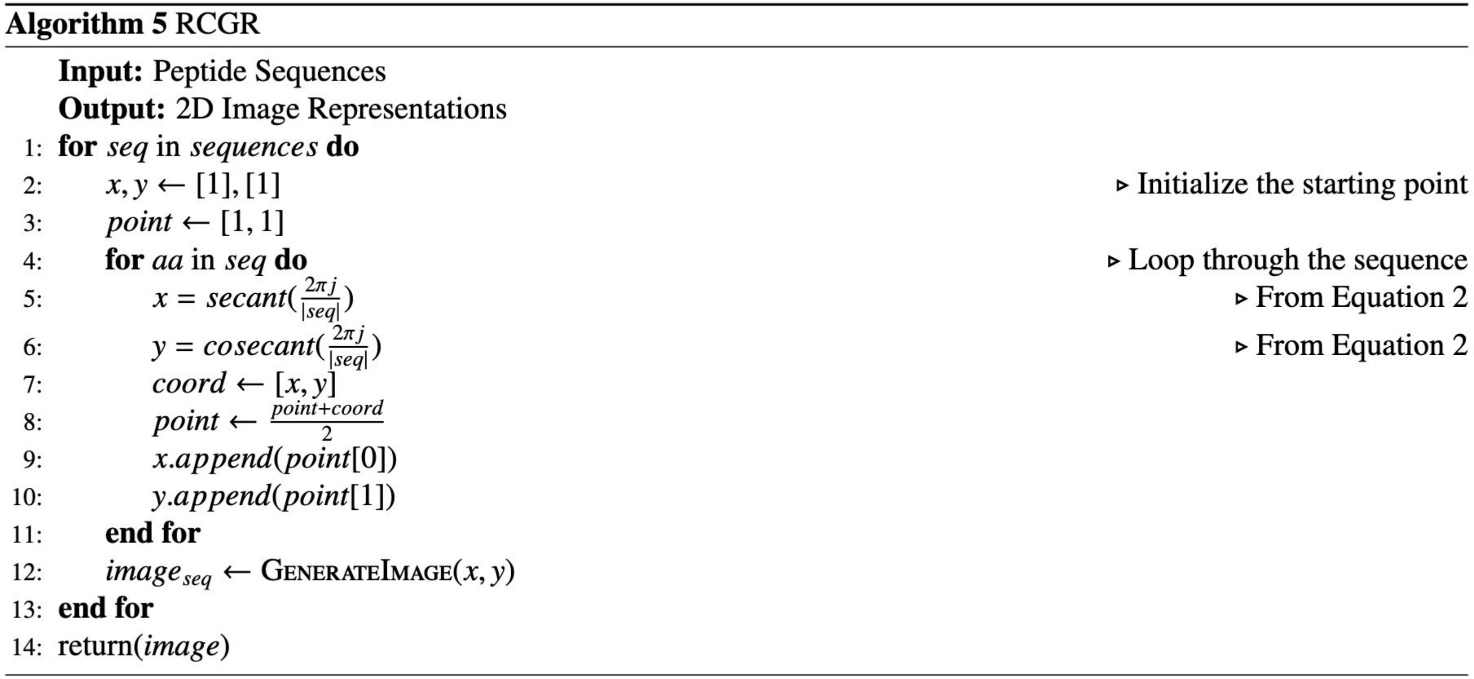


Algorithm 4 Random Chaos Game Representation

```
    Input: Peptide Sequences
    Output: 2D Image Representations
 1: for $seq$ in $sequences$ do
 2:     $x, y \leftarrow [1], [1]$                          ▷ Initialize the starting point
 3:     $point \leftarrow [1, 1]$
 4:     for $aa$ in $seq$ do                                ▷ Loop through the sequence
 5:         $x$ = GenerateRandomNumber()
 6:         $y$ = GenerateRandomNumber()
 7:         $coord \leftarrow [x, y]$
 8:         $point \leftarrow \frac{point+coord}{2}$
 9:         $x.append(point[0])$
10:         $y.append(point[1])$
11:     end for
12:     $image_{seq} \leftarrow$ GenerateImage($x, y$)
13: end for
14: return($image$)
```

Proposed System: Rectangular-CGR

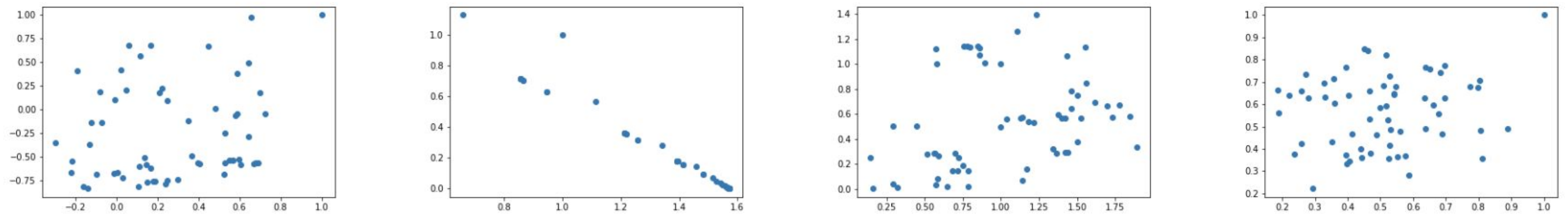
❖ Same as P-CGR but uses secant and cosecant.



Algorithm 5 RCGR

```
    Input: Peptide Sequences
    Output: 2D Image Representations
 1: for $seq$ in $sequences$ do
 2:     $x, y \leftarrow [1], [1]$                                  ▷ Initialize the starting point
 3:     $point \leftarrow [1, 1]$
 4:     for $aa$ in $seq$ do                                         ▷ Loop through the sequence
 5:         $x = secant(\frac{2\pi j}{|seq|})$                        ▷ From Equation 2
 6:         $y = cosecant(\frac{2\pi j}{|seq|})$                      ▷ From Equation 2
 7:         $coord \leftarrow [x, y]$
 8:         $point \leftarrow \frac{point+coord}{2}$
 9:         $x.append(point[0])$
10:         $y.append(point[1])$
11:     end for
12:     $image_{seq} \leftarrow \textsc{GenerateImage}(x, y)$
13: end for
14: return($image$)
```

Proposed System: Example Images



(a) S-P-CGR (b) S-RCGR (c) S-Static (d) S-Random

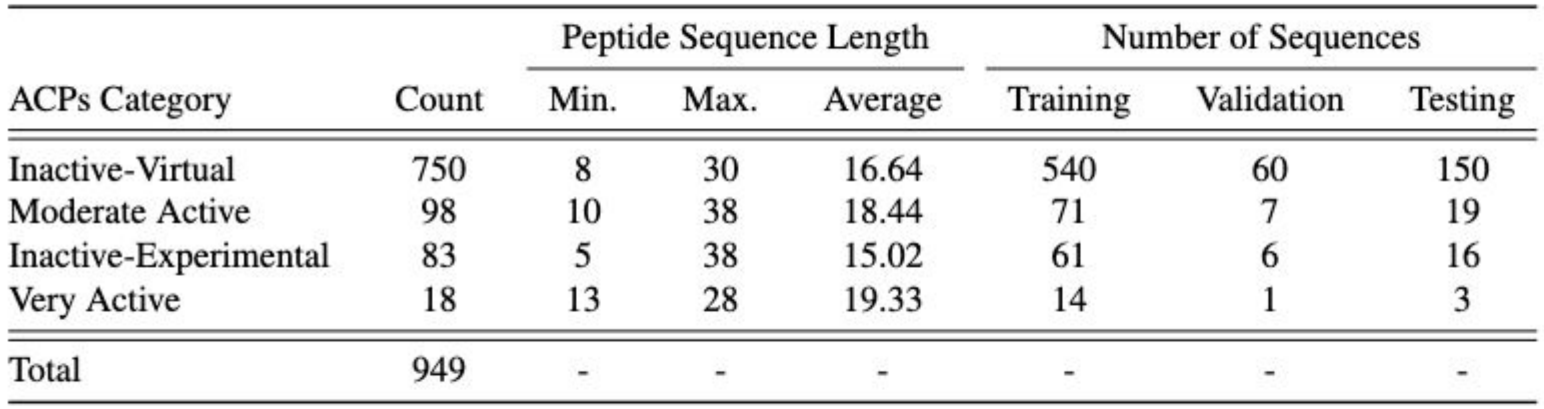
Figure 1: Graphical representations of different methods using a randomly selected peptide sequence belonging to a *moderately active* category generated by different methods using the spaced *k*-mers of the sequence.

Proposed System: Contributions

- ❖ We propose a new image-based approach for ACPs classification.
- ❖ We explore the use of secant and cosecant functions as an alternative to sine and cosine functions in the P-CGR technique to generate a more rectangular mapping.
- ❖ We investigate the usage of a pre-defined set of values and a randomly generated set of values of axis projection to generate 2D images of peptide sequences.
- ❖ We examine the usage of Spaced k-mers as a way to manipulate the amino acids within biological sequences to get the graphical representations.
- ❖ By demonstrating the performance of our proposed approach on an ACP dataset, we show that our system is able to classify the peptide sequences with higher accuracy.
- ❖ Our work provides a new direction for the classification of ACP sequences which could be used for breast cancer treatment.

Experimental Setup: Dataset Statistics

❖ The dataset has peptides (protein sequences) and their anticancer activity (target labels) on breast cancer cell lines.



		Peptide Sequence Length			Number of Sequences		
ACPs Category	Count	Min.	Max.	Average	Training	Validation	Testing
Inactive-Virtual	750	8	30	16.64	540	60	150
Moderate Active	98	10	38	18.44	71	7	19
Inactive-Experimental	83	5	38	15.02	61	6	16
Very Active	18	13	28	19.33	14	1	3
Total	949	-	-	-	-	-	-

Experimental Setup: Baseline Methods

- One-Hot Encoding (OHE)[2]:
  - Create binary feature vectors of the sequences.
- Wasserstein Distance Guided Representation Learning (WDGRL)[3]:
  - Use a neural network to extract numerical embeddings of the sequences.
- P-CGR[1]:
  - Image-based baseline.

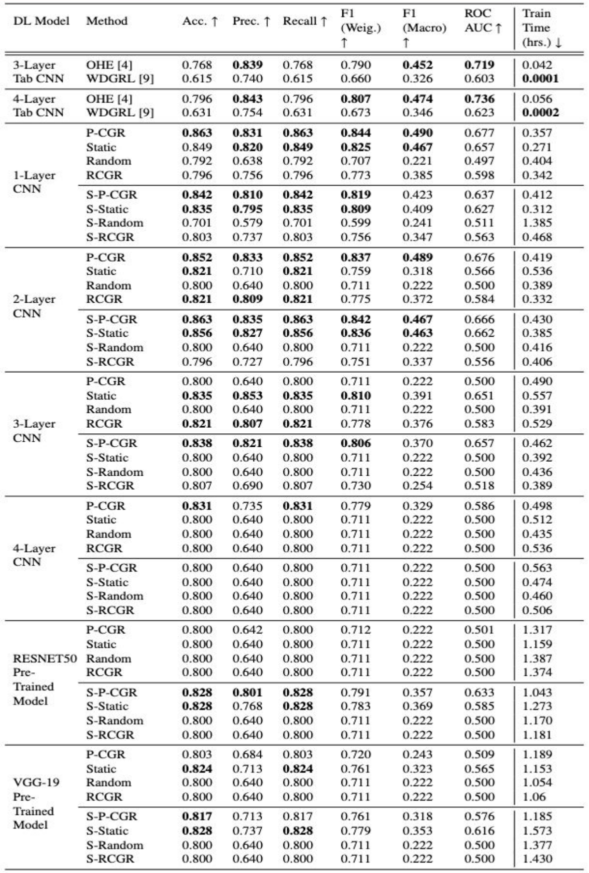
Experimental Setup: Classification Models

- ❖ The evaluation metrics are,
  - ➢ average accuracy, precision, recall, F1 (weighted), F1 (macro), Receiver Operator Characteristic Curve Area Under the Curve (ROC AUC), and training run-time.

Type	Classification Models	
DL	Vision	1-Layer CNN, 2-Layer CNN, 3-Layer CNN, 4-Layer CNN, RESNET50-pretrained, VGG19-pretrained
	Tabular	3-Layer Tab CNN, 4-Layer Tab CNN

- ❖ For training, the DL models follow 80-20% train-test split, with learning rate 0.003 for tabular & 0.001 for vision, batch size 64, epochs 10, optimizer ADAM, and loss function NLL.

Results & Discussion



DL Model	Method	Acc. ↑	Prec. ↑	Recall ↑	F1 (Weig.) ↑	F1 (Macro) ↑	ROC AUC ↑	Train Time (hrs.) ↓
3-Layer Tab CNN	OHE [4]	0.768	**0.839**	0.768	0.790	**0.452**	**0.719**	0.042
	WDGRL [9]	0.615	0.740	0.615	0.660	0.326	0.603	**0.0001**
4-Layer Tab CNN	OHE [4]	0.796	**0.843**	0.796	**0.807**	**0.474**	**0.736**	0.056
	WDGRL [9]	0.631	0.754	0.631	0.673	0.346	0.623	**0.0002**
1-Layer CNN	P-CGR	**0.863**	**0.831**	**0.863**	**0.844**	**0.490**	0.677	0.357
	Static	0.849	**0.820**	**0.849**	**0.825**	**0.467**	0.657	0.271
	Random	0.792	0.638	0.792	0.707	0.221	0.497	0.404
	RCGR	0.796	0.756	0.796	0.773	0.385	0.598	0.342
	S-P-CGR	**0.842**	**0.810**	**0.842**	**0.819**	0.423	0.637	0.412
	S-Static	**0.835**	**0.795**	**0.835**	**0.809**	0.409	0.627	0.312
	S-Random	0.701	0.579	0.701	0.599	0.241	0.511	1.385
	S-RCGR	0.803	0.737	0.803	0.756	0.347	0.563	0.468
2-Layer CNN	P-CGR	**0.852**	**0.833**	**0.852**	**0.837**	**0.489**	0.676	0.419
	Static	**0.821**	0.710	**0.821**	0.759	0.318	0.566	0.536
	Random	0.800	0.640	0.800	0.711	0.222	0.500	0.389
	RCGR	**0.821**	**0.809**	**0.821**	0.775	0.372	0.584	0.332
	S-P-CGR	**0.863**	**0.835**	**0.863**	**0.842**	**0.467**	0.666	0.430
	S-Static	**0.856**	**0.827**	**0.856**	**0.836**	**0.463**	0.662	0.385
	S-Random	0.800	0.640	0.800	0.711	0.222	0.500	0.416
	S-RCGR	0.796	0.727	0.796	0.751	0.337	0.556	0.406
3-Layer CNN	P-CGR	0.800	0.640	0.800	0.711	0.222	0.500	0.490
	Static	**0.835**	**0.853**	**0.835**	**0.810**	0.391	0.651	0.557
	Random	0.800	0.640	0.800	0.711	0.222	0.500	0.391
	RCGR	**0.821**	**0.807**	**0.821**	0.778	0.376	0.583	0.529
	S-P-CGR	**0.838**	**0.821**	**0.838**	**0.806**	0.370	0.657	0.462
	S-Static	0.800	0.640	0.800	0.711	0.222	0.500	0.392
	S-Random	0.800	0.640	0.800	0.711	0.222	0.500	0.436
	S-RCGR	0.807	0.690	0.807	0.730	0.254	0.518	0.389
4-Layer CNN	P-CGR	**0.831**	0.735	**0.831**	0.779	0.329	0.586	0.498
	Static	0.800	0.640	0.800	0.711	0.222	0.500	0.512
	Random	0.800	0.640	0.800	0.711	0.222	0.500	0.435
	RCGR	0.800	0.640	0.800	0.711	0.222	0.500	0.536
	S-P-CGR	0.800	0.640	0.800	0.711	0.222	0.500	0.563
	S-Static	0.800	0.640	0.800	0.711	0.222	0.500	0.474
	S-Random	0.800	0.640	0.800	0.711	0.222	0.500	0.460
	S-RCGR	0.800	0.640	0.800	0.711	0.222	0.500	0.506
RESNET50 Pre-Trained Model	P-CGR	0.800	0.642	0.800	0.712	0.222	0.501	1.317
	Static	0.800	0.640	0.800	0.711	0.222	0.500	1.159
	Random	0.800	0.640	0.800	0.711	0.222	0.500	1.387
	RCGR	0.800	0.640	0.800	0.711	0.222	0.500	1.374
	S-P-CGR	**0.828**	**0.801**	**0.828**	0.791	0.357	0.633	1.043
	S-Static	**0.828**	0.768	**0.828**	0.783	0.369	0.585	1.273
	S-Random	0.800	0.640	0.800	0.711	0.222	0.500	1.170
	S-RCGR	0.800	0.640	0.800	0.711	0.222	0.500	1.181
VGG-19 Pre-Trained Model	P-CGR	0.803	0.684	0.803	0.720	0.243	0.509	1.189
	Static	**0.824**	0.713	**0.824**	0.761	0.323	0.565	1.153
	Random	0.800	0.640	0.800	0.711	0.222	0.500	1.054
	RCGR	0.800	0.640	0.800	0.711	0.222	0.500	1.06
	S-P-CGR	**0.817**	0.713	0.817	0.761	0.318	0.576	1.185
	S-Static	**0.828**	0.737	**0.828**	0.779	0.353	0.616	1.573
	S-Random	0.800	0.640	0.800	0.711	0.222	0.500	1.377
	S-RCGR	0.800	0.640	0.800	0.711	0.222	0.500	1.430

Conclusion

- This study proposed a new direction for ACP sequence classification by converting those sequences to images and applying DL classification models.
- In the future, can further investigate the potential of this method in other applications, such as predicting protein functionality or disease diagnosis, etc.

References

[1] H. F. L ̈ochel, D. Eger, T. Sperlea, D. Heider, Deep learning on chaos game representation for proteins, Bioinformatics 36 (1) (2020) 272–279

[2] K. Kuzmin, et al., Machine learning methods accurately predict host specificity of coronaviruses based on spike sequences alone, Biochemical and Biophysical Research Communications 533 (3) (2020) 553–558

[3] J. Shen, Y. Qu, W. Zhang, Y. Yu, Wasserstein distance guided representation learning for domain adaptation, in: AAAI conference on artificial intelligence, 2018

[4] S. Ali, M. Patterson, Spike2vec: An effi cient and scalable embedding approach for covid-19 spike sequences, in: International Conference on Big Data (Big Data), 2021, pp. 1533–1540.