Novel Interpretable Amino Acid Property-Based Peptide Embeddings for Improved Activity Prediction

Evgeniy V. Nam[1], Yevgeniya Din[1], [*]Nikita S. Serov[1]

[1]International Institute "Solutional Chemistry of Advanced Materials and Technologies", ITMO University

9 Lomonosov Street, Saint-Petersburg.

[*]serov@scamt-itmo.ru

Abstract

Peptides gain popularity in various fields of biotechnology and medicine due to their unique properties and functionality. Compared to small molecules, peptides offer higher selectivity and biocompatibility, making them attractive for the development of new therapeutic agents. However, peptide engineering and design remain challenging tasks requiring both experimental and computational approaches. In recent years, artificial intelligence (AI) has shown significant advances in chemistry, especially with the breakthrough of AlphaFold in protein structure prediction. This has stimulated growing interest in AI-driven peptide engineering. Unfortunately, *de novo* peptide design remains a challenging task. Many deep learning models first perform engineering and feature selection before solving the actual problem, and their performance is limited by the formation of suitable mathematical representations. At present, there are no interpretable and compact representations that consider both chemical and sequential context. In this paper, we propose a novel approach to interpretable peptide representation that overcomes the current limitations. The developed methodology allows us to create models with transparent decision making, improving their applicability by experimental scientists. Our development opens new possibilities for automated peptide design, accelerating the process of creating new therapeutic agents and expanding the boundaries of AI applications in chemistry and biotechnology.

Code — https://github.com/GenerativeMolMachines/SeQuant

1. Introduction

The significant improvement of machine learning (ML) and artificial intelligence (AI) seems to have made it possible to dramatically accelerate the process of developing new drugs, reducing the cost of time and resources. They have already demonstrated their effectiveness in several areas, such as organic chemistry, where ML methods can predict the products of organic chemical reactions (Jorner et al. 2021) and their activity (Lee, Yoo, and Kang 2020). Also, in the field of forecasting protein properties, a great success has been the creation of AlphaFold, an AI-based tool capable of predicting the three-dimensional structure of a protein based on its amino acid sequence (Abramson et al. 2024). AI and ML approaches are already used in the study of new protein drugs, ranging from classical ML based on various descriptors to deep learning (DL) models (Notin et al. 2024 & Yang, Wu, and Arnold 2019). Nevertheless, it is important to note that the application of AI in this field is still in its early stages and *de novo* design remains a challenging task with only a hundred proteins known to be developed from scratch, which is remarkably small in comparison with the total number possible (Woolfson 2021). The complexity of creating new protein drugs highlights the necessity for advanced AI techniques in this area.

Protein drugs find application in areas such as medicine and biotechnology, acting as drugs (Wang et al. 2022), diagnostic agents (Uribe et al. 2021) and catalysts for various reactions (Lovelock et al. 2022). Currently, more than 80 protein drugs are registered including 33 non-insulin peptide drugs and more than 170 are in active clinical development (Wang et al. 2022). These medicines are used in the treatment of many diseases, including oncology (Fisher et al. 2019), cardiovascular diseases (Peterson and Barry 2018), and there are also applications as antimicrobials (Torres et al. 2019). However, there is a need to develop novel active agents both to deal with new healthcare area challenges and to improve the treatment of long-known ones, such as cancer. The *de novo* design of protein drugs can take over 10 years and require significant costs ranging from $161 million to $4.54 billion (2019 US$) (Schlander et al. 2021). Such expenses are mainly caused by the most common method for studying new protein therapeutics - experimental screening, while modeling, although used, is resource-intensive, time-consuming and does not always provide accurate results that correlate well with experimental data.

Several crucial challenges have been highlighted in terms of processing protein data: firstly, to the best of our knowledge, there are no interpretable descriptors, and exist-

ing approaches take into account either contextual information or physicochemical properties. Secondly, most of the developed models use 20 basic proteinogenic amino acids, and there is no functionality to consider any modifications, which are capable of significantly influencing the resulting properties of the proteins. These include the difficulty of versatile model development, which creates many tools suitable for use only in a narrow area with various limitations. Thirdly, existing AI and ML tools are often demanding on the amount of data, which is not always available, and for some specific areas, accumulation of it in sufficient quantities is not even possible.

This study presents SeQuant, a tool for creating peptide descriptors based on the physicochemical properties of amino acids. The neural net of SeQuant implies architecture capable of considering the context in the amino acid sequence, and the resulting descriptors also carry information about the properties of amino acids and the peptide chain, making it possible to interpret them. At the same time, the size of the feature space is significantly smaller than in other approaches. Thus, the proposed solution is expected to be superior to other available options**.** The tool is based on the convolutional autoencoder (CAE) architecture which allows transforming matrices of properties of individual monomers into more compact representations, where one-dimensional convolutions allow preserving the original physical meaning of individual properties of monomers, while considering the monomer context. The method involves reducing the dimensionality along the amino acid chain to obtain a vector, where each element represents a specific property of the amino acids. The model offers interpretable peptide embeddings, which can be used for activity prediction tasks and during benchmarking they appear to be comparable with existing approaches, but with remaining physicochemical properties, which are important for the real application of the models in practice.

2. Related works

In classical ML, various models are used to solve classification and regression problems (Woolfson 2021), such as random forests (RF) (Breiman 2001) and support vector machines (SVM) (Cortes and Vapnik 1995). A common element is the implementation of descriptors which are numerical characterization of peptides (Xu et al. 2020). Often these are physico-chemical and composition properties, amino acid indices, or topological descriptors. In each specific case, a search is carried out for the most informative ones. Modern research uses combinations of various descriptors, and also introduces new ones that are specific to a particular field of study. A common problem with this approach is that it does not consider the context of the systems under study. One can learn more about the various descriptors, as well as the tools for obtaining them, in the review of Emonts and Buyel (2023).

Researchers also use the DL approach for peptides' properties prediction. In this case, the initial data are most often the amino acid sequences themselves. They are encoded and fed into neural networks. Due to their high computing power and the ability to find patterns in data, such models demonstrate prominent results in a wide range of problems, including the prediction of protein functions (Bileschi et al. 2022 & Luo et al. 2021), their structure (Kandathil, Greener, and Jones 2019 & Ismi, Pulungan, and Afiahayati 2022), and stability (Blaabjerg et al. 2023). A significant advantage of DL models lies in their capacity to consider both local and global contexts, thereby enabling the accommodation of the complexities inherent in protein structures. However, such models use sequence-only information without considering both amino acids as well as entire peptide properties and are extremely demanding on computing power, which are serious drawbacks.

3. Methods

Our approach was to use pre-calculated physicochemical descriptors of 20 main proteinogenic amino acids and 2 most common modifications as the initial data. For our research, we classified them as molecular and DFT-derived descriptors. Each of the two groups is described in the corresponding subsection. To describe the context and generate descriptors applicable in ML models, we implemented CAE architecture. Briefly, a matrix of descriptors is calculated based on a dictionary of known monomers, then it is used to form arrays of descriptors for each amino acid sequence, where the rows correspond to properties and the columns to amino acids. The resulting arrays are used as an input to CAE, and the resulting peptide descriptors are extracted from the CAE bottleneck layer; these are vectors, each of which characterizes a specific peptide. It is important to note that the decrease in dimensionality occurs along the axis of the amino acid chain length via 1-dimensional pooling to preserve the initial physical meaning behind amino acid descriptors. The vector data, called latent representations, can then be used as descriptors for ML or DL models. The CAE learning was carried out on a server, the overall configuration of which consists of 6 GPU A6000, 256 cores, AMD EPYC 7763 64-Core Processor, 512 GB RAM, using 2 GPU and 50 GB of RAM. The final model was trained in 75 epochs for 4 days.

Data collection and preprocessing

To train the model, unlabeled peptide sequence data was extracted using a special self-made program. We parsed the NCBI database, and at this stage, filtering for duplicates and unknown monomers was performed according to the formed

dictionary (20 natural proteinogenic amino acids + 2 modifications with a common single-letter designation). During the procedure, we tried to collect the same number of peptide sequences with a length from 5 to 96 for each to avoid bias in the trained model due to more common lengths. A total of 6,749,334 peptide sequences were obtained, the length distribution can be seen in **Figure 1**. This data was divided into training and testing sets in a ratio of 0.7 to 0.3. The resulting sets were initially used to train the CAE.

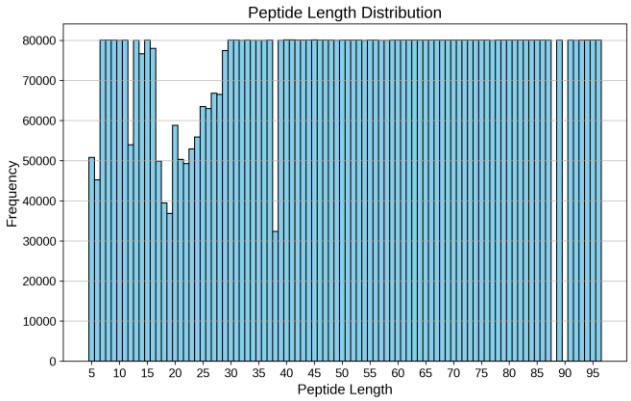

Figure 1. Peptide length distribution in collected data.

Pre-calculated molecular descriptors

This group uses 43 molecular descriptors from the rdMolDescriptors module of the RDKit Python library. After descriptor extraction, normalization within the range from -1 to 1 is performed within each parameter using the MinMaxScaler module of the scikit-learn library. Then, based on the obtained matrix and the matrix of descriptors of the second group (see **Section DFT-derived descriptors**), arrays are formed for each amino acid sequence - each of them is a unique representation of the peptides, while padding up to a length of 96 amino acids was used: this length is sufficient to cover most peptides, the dimensionality itself is determined by the sequence of one-dimensional convolutions in the architecture of the developed neural net.

DFT-derived descriptors

This group of descriptors includes 3, namely descriptors based on the interaction energy of amino acids with divalent metal cations - calcium, magnesium and barium. They were obtained from the results of quantum-chemical calculations based on the density functional theory. The calculation itself is given in the work of Hu, Lenz-Himmer and Baldauf (2022) and the obtained data contained in the NOMAD database. In the work calculations for 23,243 amino acid conformers and their complexes with the calcium, magnesium and barium cations including alternative possible side chain protonation states were performed. The authors report that the data cover 21,909 stationary points on the corresponding potential energy surfaces over a wide range of relative energies up to 4 eV (390 kJ/mol).

The database API, as well as a written parser were used to extract the free energy values of these conformations. All data were divided into 4 categories: free energy of amino acid complexes with magnesium, barium, calcium cations and free energy of simple amino acid conformations. Also, the free energy values of magnesium, calcium and barium cations were obtained from the NOMAD database. Then, for each amino acid, averaging was performed over the conformations to obtain specific values for each. To obtain the interaction energy descriptors themselves, the following operation was performed: the free energy of the amino acid and the free energy of the cation were subtracted from the free energy of the amino acid-cation complex. The result of the processing is a matrix of shape (22x3), where 22 is the number of amino acids, 3 is the number of descriptors obtained. Then, the descriptors, as in the first group, were normalized within each parameter using the MinMaxScaler module of the scikit-learn library. Next, as mentioned in the previous section, arrays are formed for each amino acid sequence.

CAE architecture

This study uses a CAE from the Keras framework to obtain peptide descriptors applicable in ML and DL. Autoencoders consist of 3 parts: an encoder, a decoder, and a latent space. Their operating principle is to compress input data and then restore the compressed representations to the original format. Training is based on the loss function - the difference between the restored and initial data. Autoencoders are mainly used to reduce the dimensionality of input data and feature selection. Compressed representations are extracted from a part of the latent space.

The developed architecture consists of 6 blocks of layers in each encoder and decoder parts. Each encoder block consists of the following layers: Conv2D, BatchNormalization, LeakyReLU, AveragePooling2D, Dropout. The convolution layer was used to extract local dependencies in sequences. This was followed by a normalization layer to stabilize the distribution of activations. This is necessary to speed up training and improve the model's performance. Next came the activation layer using the LeakyReLU function. The fourth layer is the average pooling layer, which reduces the data dimensionality by averaging over (n, n) matrix elements, where n is the pooling parameter. This was necessary to highlight the most significant information and reduce the dimensionality. The last layer of the block is the dropout layer. It was used to regularize the model and prevent overfitting. During training, N (where N depends on the parameter specified in the layer) neurons were randomly reset to zero, which helps the model become more stable and generalized.

The decoder uses fewer layers. One block consists of the following parts: Conv2DTranspose, BatchNormalization, LeakyReLU. In the last block of the decoder, the activation function is changed to the hyperbolic tangent function. Conv2DTranspose Layer is a transposed (inverse) version of the Conv2D layer and was used to increase the dimensionality of the data and restore the original patterns that were compressed in the coding part. Next, a normalization layer was applied, as in the coding part, it accelerates convergence and helps stabilize the training of the model. The last layer is the activation layer. It is important to note that in the last block, the activation function is the hyperbolic tangent function. Its use was due to the format of the input data: as described earlier, we normalized all arrays within the range from -1 to 1, and this function transforms the input data into this range. Thus, the use of this function was necessary for normal data recovery to the original format.

CAE training and optimization

When developing the architecture, we searched for the optimal one. For this purpose, from the entire data array, the acquisition of which is described in the **Data collection and preprocessing Section**, we made a sample of 10,000 sequences for the training and test samples with stratification by the length of the sequences for faster architecture screening. On this sample, we searched for both the optimal architecture and selected the necessary hyperparameters. At the first stage, optimization was carried out by the number of average pooling layers; overall, 6 architectures were tested. After identifying the best result, the parameters of the convolution layer, namely the number of filters, were varied. At the same time, with the move into the depth of the neural network, the number of filters was reduced to 1 for reasons of reducing the dimensionality. At this stage, 7 architectures were tested.

Then, we began training the model on the full data. Due to its quantity, batching was applied, with preprocessing of each batch performed immediately before feeding it for training to reduce the requirements for RAM. The first results of training on full data turned out to be worse than expected (the error exceeded that of training on a sample by almost 3 times). When solving the problem, deep architecture using attention layers were tested, and additional convolution layers were also checked. In each case, overtraining or no reduction in error was observed, learning was interrupted by the early stopping function (at this stage, we have tested 10 different architecture options). Since deepening the architecture did not improve the situation, we returned to the hyperparameters of the model, the initially selected optimal architecture was used with a change in convolution parameters: here, the number of filters increases with the depth of the neural network to compensate for data loss when reducing dimensionality. It was trained with a batch size of 32 and a learning rate of 10^{-3}. However, results on benchmarking tasks seemed to be insufficient. The problem was identified in CAE embeddings, where a high sparsity was observed; therefore, we returned to architecture optimization. In this iteration, the original data was sampled into 3 datasets with stratification based on amino acid frequency. The medium one (that consists of 370.413 peptides) was used for the neural net depth selection, where a small set (123.471 peptides) was used for hyperparameter tuning. This was done with Keras Tuner using the hyperband optimization strategy. After optimization, we obtained several models which were compared on a benchmark task. The best one was fitted with entire balanced data. The learning curve graph shows that the model was trained with a mean squared error of reconstructing normalized data on the test set equal to 0.0803.

4. Experiments

Benchmark studies

To evaluate the performance of the proposed approach we selected several benchmark datasets for classification task: prediction of the antimicrobial (Cao et al. 2023), anti-inflammatory (Raza et al. 2023), antidiabetic (Chen, Huang, and He 2022) and antioxidative (Qin et al. 2023) peptides. Their short description can be seen in **Table 1**.

Dataset	Number of peptides	Average length	Balanced
Antimicrobial	8268	18.53	+
Anti-inflammatory	3790	16.36	-
Antidiabetic	472	19.60	+
Antioxidative	2120	5.92	+

Table 1. Brief description of benchmark datasets.

For comparison, we selected 3 common peptide encoding strategies, namely one-hot, BLOSUM62 and threemers encodings. For each dataset we obtained all encodings and using the same model, which is XGBClassifier, with default hyper parameters, binary classification was performed.

Results and Discussion

This subsection demonstrates how SeQuant can be used to accelerate *de novo* peptide design through *in silico* screening

of the candidate compounds. SeQuant as a tool is implemented as a Python programming language class with several methods initialized when the class is declared. First, the extraction of the descriptors described in the previous section occurs, a total of 46. Then, the original peptide sequences of a dataset are filtered by the maximum length of 96 amino acids, as well as the presence of unknown monomers (not included in the original dictionary). If a mismatch is found, a corresponding error is raised, reporting the presence of too long peptides or unknown amino acids. Then, latent representations are obtained using the model described in the **CAE architecture Section**. They are the necessary peptide embeddings, which can then be used in ML algorithms. The code itself, as well as the documentation for the tool and examples of use are available in the GitHub repository. Also, there is an API for simplified access. For comparison we used several datasets described in the previous subsection. For each of them we provided binary classification with the same ML model. **Table 2** shows the obtained results.

SeQuant embeddings appear to be promising for most metrics they seem to be comparable with known encoding strategies. At the same time, One-Hot encoding tends to be surprisingly effective on these benchmarks in the binary classification task. This might be related to the comparatively small size of the datasets, insufficient to reveal the potential of more complex encoding options. It is especially noticeable on the smallest of all datasets - Antidiabetic, a drop in key metrics (F1 score and MCC) can be seen when using SeQuant embeddings. Also, lower indicators are typical for the Anti-inflammatory dataset for all encoding strategies. We attribute this to its imbalance - the fraction of records with a positive label is approximately one third, which was not enough for good model training; the FN results were greater than the TP. Additionally, all datasets have mean peptide length lower than 20, which seems to be a problem for SeQuant, since we use CAE architecture with paddings for maximum length of 96 amino acids. It makes our tool sensitive for peptides' length. In further work we plan to deal with this problem by investigating other architectures. Nevertheless, unlike other options, SeQuant embeddings have an important advantage - the possibility of chemical

Task	Encoding type	Accuracy	Precision	Recall	F1 score	ROC AUC	MCC
Antimicrobial	one-hot	**0.798**	**0.811**	**0.778**	**0.794**	**0.798**	**0.597**
	threemers	0.753	0.770	0.723	0.746	0.753	0.508
	blosum62	0.684	0.696	0.652	0.673	0.684	0.368
	sequant	0.711	0.726	0.677	0.701	0.711	0.423
Antidiabetic	one-hot	0.632	0.644	0.604	0.624	0.632	0.264
	threemers	0.589	0.592	0.604	0.598	0.589	0.179
	blosum62	**0.663**	**0.660**	**0.688**	**0.673**	**0.663**	**0.326**
	sequant	0.589	0.582	0.667	0.621	0.589	0.180
Anti- inflammatory	one-hot	0.664	0.579	0.441	0.501	0.621	0.260
	threemers	0.679	**0.595**	**0.507**	**0.547**	**0.647**	**0.304**

	blosum62	0.644	0.543	0.431	0.481	0.603	0.218
	sequant	**0.668**	0.585	0.473	0.523	0.631	0.276
Antioxidative	one-hot	**0.906**	**0.913**	**0.896**	**0.905**	**0.906**	**0.811**
	threemers	0.778	0.778	0.778	0.778	0.778	0.557
	blosum62	0.604	0.622	0.528	0.571	0.604	0.210
	sequant	0.750	0.750	0.750	0.750	0.750	0.500

Table 2. SeQuant benchmarking results on selected datasets.

interpretation of the results, since the elements of the embeddings retain the original meaning of the descriptors used to obtain them. **Table 3** provides information on the top 10 most important features of the SeQuant.

It can be noted that for the Antidiabetic dataset, even the top 10 features by importance do not allow dividing it by label, which is reflected in the metrics. For the remaining sets, at least 2 features have a p-value less than 0.05,

Antimicrobial			Antidiabetic			Anti-inflammatory			Antioxidative		
Feature	IS	p-value	Feature	IS	p-value	Feature	IS	p-value	Feature	IS	p-value
eNSR	0.045	**$2.1x10^{-48}$**	echi2v	0.049	0.290	eNSC	0.101	**$2.8x10^{-29}$**	eNSR	0.047	**$1.4x10^{-14}$**
eBIE	0.037	**0.029**	eNAB	0.042	0.892	eLBD	0.055	**0.0006**	eCIE	0.035	0.927
eMIE	0.035	**$1.2x10^{-5}$**	eNAH	0.041	0.780	eBIE	0.029	**$9.1x10^{-23}$**	eNAH	0.034	0.444
ePhi	0.034	**$1.8x10^{-21}$**	eEM	0.034	0.407	eNSH	0.027	0.135	eLBD	0.033	0.085
eNSC	0.031	**$1.6x10^{-10}$**	eNRB	0.033	0.698	eNH	0.025	**0.0002**	echi4n	0.031	0.213
eNA	0.029	**$2.0x10^{-16}$**	echi0v	0.031	0.987	eLBA	0.025	**0.012**	eHKA	0.031	0.977
eNAR	0.028	**$8.4x10^{-50}$**	eMIE	0.030	0.777	eMIE	0.024	**$2.1x10^{-53}$**	eNBA	0.031	**0.002**
eHKA	0.027	0.112	eCCP	0.029	0.790	eEM	0.023	0.636	echi3n	0.031	0.440
eNBA	0.026	0.121	eNSH	0.029	0.492	echi0n	0.023	**$4.0x10^{-11}$**	eNAlR	0.030	0.672
eCCP	0.025	0.405	eCMR	0.028	0.701	eNAH	0.023	**$7.5x10^{-9}$**	echi0v	0.029	0.804

Table 3. Feature Importance (FI) of the SeQuant embeddings for various tasks. Top 10 models features, their importance scores (IS) and significance of differences (p-value) based on the t-test results. The p-value indicates the probability that the observed differences occurred by chance, with smaller values indicating a more significant relationship, the significance level here is 0.05. IS reflects the contribution of each feature to the model predictions. Below is a transcript of the feature names from the table: eNSR - embedded NumSaturatedRings, eBIE - embedded Ba interaction energy, eMIE - embedded Mg interaction energy, ePhi - embedded Phi, eNSC - embedded NumAtomStereoCenters, eNA - embedded NumAtoms, eNAR - em-

bedded NumAromaticRings, eHKA - embedded hallKierAlpha, eNBA - embedded NumBridgeheadAtoms, eCCP - embedded CrippenClogP, echi2v - embedded chi2v, eNAB - embedded NumAmideBonds, eNAH - embedded NumAromaticHeterocycles, eEM - embedded exactmw, eNRB - embedded NumRotatableBonds, echi0v - embedded chi0v, eNSH - embedded NumSaturatedHeterocycles, eCMR - embedded CrippenMR, eLBD - embedded lipinskiHBD, eNH - embedded NumHeteroatoms, eLBA - embedded lipinskiHBA, echi0n - embedded chi0n, eCIE - embedded Ca interaction energy, echi4n - embedded chi4n, echi3n - embedded chi3n, eNAlR - embedded NumAliphaticRings, echi0v - embedded chi0v

indicating a low probability that the distributions of these feature values belong to the same group. This, in turn, suggests the potential for label-based object separation with their usage. In addition, DFT descriptors appear to show statistically significant associations with peptide activity, making them useful for its prediction. For Antimicrobial dataset this is probably related to the antimicrobial peptides' mechanism of action: a common case is their effect on cell membranes (Zasloff 2002). In turn, Mg^{2+} and Ca^{2+} form bridges between negatively charged groups of phospholipids and lipopolysaccharides, stabilizing the cell wall and making it more resistant to external influences (Vaara 1992). Thus, the interaction energy of metal cations with peptides may be significant for predicting their antibacterial activity. Ba^{2+}, although less abundant in biological systems, has properties similar to magnesium and calcium. It can replace these ions in certain biological structures. The interaction of peptides with Ba^{2+} might reflect their ability to bind to divalent cations, which is important for their activity against bacterial membranes. Also, the influence of DFT descriptors on predicting peptide activities may be due to the effect of metal ions on peptide conformation, which is important for their functions.

5. Conclusions

In this study, we developed SeQuant, a CAE-based tool for predicting peptide properties. It offers embeddings for ML and DL models that consider the physicochemical properties of compounds. On benchmark datasets, they appeared to be comparable to known variants of amino acid sequence encoding. However, the metrics obtained with SeQuant did not show superiority in the tasks taken. We attribute this to several fundamental limitations of the used architecture: convolutional autoencoders are highly dependent on the dimensionality of the input data, in this case, the length of peptides. We tried to mitigate this problem by balancing the training data by length and amino acid frequency, but it seems that the problem was not completely solved. The second drawback that affects the result, we believe, is the operating principle of the applied autoencoder - during training it was fitted with discrete values, and the resulting embeddings have a low density. This might affect the performance of the model using the latent representations of SeQuant. In the continuation of this work, we plan to consider the second drawback by using a variational autoencoder - this type of architecture, unlike the convolutional one, gives dense representations.

Acknowledgments

The authors thank Priority 2030 Federal Academic Leadership Program for infrastructure support.

References

Abramson, J.; Adler, J.; Dunger, J. et al. 2024. Accurate Structure Prediction of Biomolecular Interactions with AlphaFold 3. *Nature* 630: 493–500. doi.org/10.1038/s41586-024-07487-w.

Asgari, E., and Mofrad, M. R. K. 2015. Continuous Distributed Representation of Biological Sequences for Deep Proteomics and Genomics. arXiv:1503.05140.

Bileschi, M.L.; Belanger, D.; Bryant, D.H. et al. 2022. Using Deep Learning to Annotate the Protein Universe. *Nat Biotechnol* 40: 932–937. doi.org/10.1038/s41587-021-01179-w.

Blaabjerg, L.M.; Kassem, M.M.; Good, L.L. et. al. 2023. Rapid Protein Stability Prediction Using Deep Learning Representations. *eLife* 12, e82593. doi.org/10.7554/eLife.82593.

Breiman, L. 2001. Random Forests. *Mach Learn* 45: 5–32. doi.org/10.1023/A:1010933404324.

Cai, Y.D.; Liu, X.J.; Xu, X.B. et al. 2002. Support Vector Machines for Prediction of Protein Subcellular Location by Incorporating Quasi-Sequence-Order Effect. *J of Cellular Biochemistry* 84(2): 343-348. doi.org/10.1002/jcb.10030.

Cao, Q.; Ge, C.; Wang, X. et al. 2023. Designing Antimicrobial Peptides Using Deep Learning and Molecular Dynamic Simulations. *Briefings in Bioinformatics* 24(2). bbad058. doi.org/10.1093/bib/bbad058.

Chen, X.; Huang, J.; and He, B. 2022. AntiDMPpred: a Web Service for Identifying Anti-Diabetic Peptides. *PeerJ* 10, e13581. doi.org/10.7717/peerj.13581.

Chou, K.C. 2001. Prediction of Protein Cellular Attributes Using Pseudo-Amino Acid Composition. *Proteins* 43(3): 246-255. doi.org/10.1002/prot.1035.

Cortes, C., and Vapnik, V. 1995. Support-Vector Networks. *Mach Learn* 20: 273–297. doi.org/10.1007/BF00994018.

Davda, J.; Declerck, P.; Hu-Lieskovan, S. et al. 2019. Immunogenicity of Immunomodulatory, Antibody-Based, Oncology Therapeutics. *J Immunother Cancer* 7(105). doi.org/10.1186/s40425-019-0586-0.

Dubchak, I.; Muchnik, I.; Holbrook, S.R. et. al. 1995. Prediction of Protein Folding Class Using Global Description of Amino Acid Sequence. *Proc Natl Acad Sci USA* 92(19): 8700-8704. doi.org/10.1073/pnas.92.19.8700.

Elnaggar, A.; Heinzinger, M.; Dallago, C. et al. 2022. ProtTrans: Toward Understanding the Language of Life Through Self-Supervised Learning. In *Institute of Electrical and Electronics Engineers*

Transactions on Pattern Analysis and Machine Intelligence 44(10): 7112-7127. doi.org/10.1109/TPAMI.2021.3095381.

Emonts, J., and Buyel, J.F. 2023. An Overview of Descriptors to Capture Protein Properties - Tools and Perspectives in the Context of QSAR Modeling. *Comput Struct Biotechnol J* 21: 3234-3247. doi.org/10.1016/j.csbj.2023.05.022.

Fisher, E.; Pavlenko, K.; Vlasov, A. et al. 2019. Peptide-Based Therapeutics for Oncology. *Pharm Med* 33: 9–20. doi.org/10.1007/s40290-018-0261-7.

Gainza, P.; Sverrisson, F.; Monti, F. et al. 2020. Deciphering Interaction Fingerprints from Protein Molecular Surfaces Using Geometric Deep Learning. *Nat Methods* 17: 184–192. doi.org/10.1038/s41592-019-0666-6.

Geary, R. C. 1954. The Contiguity Ratio and Statistical Mapping. *Incorp Statistician* 5: 115–145. doi.org/10.2307/2986645.

Hu, X.; Lenz-Himmer, M.O.; and Baldauf, C. 2022. Better Force Fields Start with Better Data: A Data Set of Cation Dipeptide Interactions. *Sci Data* 9(327). doi.org/10.1038/s41597-022-01297-3

Ismi, D.P.; Pulungan, R.; and Afiahayati. 2022. Deep Learning for Protein Secondary Structure Prediction: Pre and PostAlphaFold. *Comput Struct Biotechnol J* 20: 6271-6286. doi.org/10.1016/j.csbj.2022.11.012.

Jorner, K.; Tomberg, A.; Bauer, C. et al. 2021. Organic Reactivity from Mechanism to Machine Learning. *Nat Rev Chem* 5: 240–255. doi.org/10.1038/s41570-021-00260-x.

Kandathil, S.M.; Greener, J.G.; and Jones, D.T. 2019. Recent Developments in Deep Learning Applied to Protein Structure Prediction. *Proteins* 87: 1179–1189. doi.org/10.1002/prot.25824.

Kawashima, S., and Kanehisa, M. 2000. AAindex: Amino Acid Index Database. *Nucleic Acids Research* 28(1): 374. doi.org/10.1093/nar/28.1.374.

Lee, B.; Yoo, J.; and Kang, K. 2020. Predicting the Chemical Reactivity of Organic Materials Using a Machine-Learning Approach. *Chem Sci* 11: 7813-7822. doi.org/10.1039/D0SC01328E.

Liang, Y.; Liu, S. Y.; and Zhang, S. L. 2015. Prediction of Protein Structural Class Based on Different Autocorrelation Descriptors of Position-Specific Scoring Matrix. *MATCH: Communications in Mathematical and in Computer Chemistry* 73(3): 765-784.

Lin, Z.; Akin, H.; Rao, R. et al. 2023. Evolutionary-Scale Prediction of Atomic-Level Protein Structure with a Language Model. *Science* 379(6637): 1123–1130. doi.org/10.1126/science.ade2574.

Lovelock, S.L.; Crawshaw, R.; Basler, S. et al. 2022. The Road to Fully Programmable Protein Catalysis. *Nature* 606: 49–58. doi.org/10.1038/s41586-022-04456-z.

Luo, Y.; Jiang, G.; Yu, T. et al. 2021. ECNet is an Evolutionary Context-Integrated Deep Learning Framework for Protein Engineering. *Nat Commun* 12(5743). doi.org/10.1038/s41467-021-25976-8.

Moran, P. A. 1950. Notes on Continuous Stochastic Phenomena. *Biometrika* 37: 17–23. doi.org/10.2307/2332142.

Moreau, G., and Broto, P. 1980. Autocorrelation of Molecular Structures, Application to SAR Studies. *Nour J Chim* 4: 757–764.

Notin, P.; Rollins, N.; Gal, Y. et al. 2024. Machine Learning for Functional Protein Design. *Nat Biotechnol* 42: 216–228. doi.org/10.1038/s41587-024-02127-0.

Ong, S.A.; Lin, H.H.; Chen, Y.Z. et al. 2007. Efficacy of Different Protein Descriptors in Predicting Protein Functional Families. *BMC Bioinformatics* 8(300). doi.org/10.1186/1471-2105-8-300.

Peterson, S. C., and Barry, A. R. 2018. Effect of Glucagon-Like Peptide-1 Receptor Agonists on All-Cause Mortality and Cardiovascular Outcomes: a Meta-Analysis. *Curr Diabetes Rev* 14: 273–279. doi.org/10.2174/1573399813666170414101450.

Qin, D.; Jiao, L.; Wang, R., et al. 2023. Prediction of Antioxidant Peptides Using a Quantitative Structure−Activity Relationship Predictor (AnOxPP) Based on Bidirectional Long Short-Term Memory Neural Network and Interpretable Amino Acid Descriptors. *Computers in Biology and Medicine* 154(106591). doi.org/10.1016/j.compbiomed.2023.106591.

Raza, A.; Uddin, J.; Almuhaimeed, A. et al. 2023. AIPs-SnTCN: Predicting Anti-Inflammatory Peptides Using FastText and Transformer Encoder-Based Hybrid Word Embedding with Self-Normalized Temporal Convolutional Networks. *Journal of chemical information and modeling* 63(21): 6537-6554. doi.org/10.1021/acs.jcim.3c01563.

Schlander, M.; Hernandez-Villafuerte, K.; Cheng, C.Y.. et al. 2021. How Much Does It Cost to Research and Develop a New Drug? A Systematic Review and Assessment. *PharmacoEconomics* 39: 1243–1269. doi.org/10.1007/s40273-021-01065-y.

Shen, J.W.; Zhang, J.; Luo, X.M. et al. 2007. Predicting Protein–Protein Interactions Based Only on Sequences Information. *P Natl Acad Sci USA* 104(11): 4337–41. doi.org/10.1073/pnas.0607879104.

Torres, M. D. T.; Sothiselvam, S.; Lu, T. K. et al. 2019. Peptide Design Principles for Antimicrobial Applications. *J Mol Biol* 431: 3547–3567. doi.org/10.1016/j.jmb.2018.12.015.

Uribe, K. B.; Guisasola, E.; Aires, A. et al. 2021. Engineered Repeat Protein Hybrids: The New Horizon for Biologic Medicines and Diagnostic Tools. *Acc Chem Res* 54 (22): 4166–4177. doi.org/10.1021/acs.accounts.1c00440.

Vaara, M. 1992. Agents That Increase the Permeability of the Outer Membrane. *Microbiological reviews* 56(3): 395-411. doi.org/10.1128/mr.56.3.395-411.1992.

Wang, Y.C.; Wang, Y.; Yang, Z.X. et al. 2011. Support Vector Machine Prediction of Enzyme Function with Conjoint Triad Feature and Hierarchical Context. *BMC Syst Biol* 5. doi.org/10.1186/1752-0509-5-S1-S6.

Wang, L.; Wang, N.; Zhang, W. et al. 2022. Therapeutic Peptides: Current Applications and Future Directions. *Sig Transduct Target Ther* 7(48). doi.org/10.1038/s41392-022-00904-4.

Wang, D.; Zeng, S.; Xu, C. et al. 2017. MusiteDeep: a Deep-Learning Framework for General and Kinase-Specific Phosphorylation Site Prediction, *Bioinformatics* 33(24): 3909–3916. doi.org/10.1093/bioinformatics/btx496.

Woolfson, D. N. 2021. A Brief History of *de Novo* Protein Design: Minimal, Rational, and Computational. *J Mol Biol* 433(20): 167160. doi.org/10.1016/j.jmb.2021.167160.

Xu, Y.; Verma, D.; Sheridan, R.P. et. al. 2020. Deep Dive into Machine Learning Models for Protein Engineering. *Journal of Chemical Information and Modeling* 60(6): 2773-2790. doi.org/10.1021/acs.jcim.0c00073.

Yang, K.K.; Wu, Z.; and Arnold, F.H. 2019. Machine-Learning-Guided Directed Evolution for Protein Engineering. *Nat Methods* 16: 687–694. doi.org/10.1038/s41592-019-0496-6.

Zasloff, M. 2002. Antimicrobial Peptides of Multicellular Organisms. *Nature* 415(6870): 389-395. doi.org/10.1038/415389a.