Deep Protein Feature Learning for Intrinsically Disordered Regions in Proteins Using Image Inpainting

Samiha Mahin[1], Marc Singleton[2], Michael Eisen[1,2,3]

1-Department of Integrative Biology, UC Berkeley, CA, USA, 2-Biophysics Graduate Group, UC Berkeley, CA, USA,
3-Howard Hughes Medical Institute, UC Berkeley CA, USA

ABSTRACT

Intrinsically disordered regions (IDRs) make up an estimated 33% of eukaryotic proteins, but despite its prevalence, many features and properties about IDRs are unknown. IDRs violate the traditional structure-function paradigm that a protein's amino acid sequence determines a well defined structure which in turn performs a specific function. IDRs are portions of the protein's structure that do not fold into a stable well-defined structure but instead fluctuate through a range of conformations. Many recent studies have shown that machine learning models can uncover unknown features and organizations of biological systems. Generative adversarial networks (GANs) are a class of machine learning framework widely used for image inpainting, a task where masked portions of an image are regenerated using the surrounding context. Thus, in this study, we aimed to uncover unrecognized features and organizations of IDRs by training a series of GANs to solve the image inpainting problem as applied to IDR sequences. We treated the surrounding amino acid sequence as the surrounding context of the image and the disordered region as the masked portion of the image. By regenerating the disordered region, the GAN would learn a rich internal representation that would uncover features and organizations of IDRs. However, our trained models only generated repeated sequences of the most common amino acids found in the disordered region when given the surrounding context. The model's inability to produce accurate amino acid sequences of disordered regions when given surrounding context suggests that the surrounding contexts of disordered regions do not influence their composition, and therefore disordered regions are independent of its local sequence context. Code available on GitHub repository (github.com/Discovery-IDRs/predIDR/tree/inpainting).

Introduction

In contrast to structured domains, intrinsically disordered regions (IDRs) do not fold into stable well-defined three-dimensional structures but instead fluctuate through a range of conformations (Fig. 1). IDRs are "characterized by their biased amino acid composition and low sequence complexity, as well as by their low proportions of bulky hydrophobic amino acids and high proportions of charged and hydrophilic amino acids" (Wright *et al.*, 2015). Beyond these broad biophysical characteristics, however, it is unknown if there are other organizing principles or features that characterize IDRs.

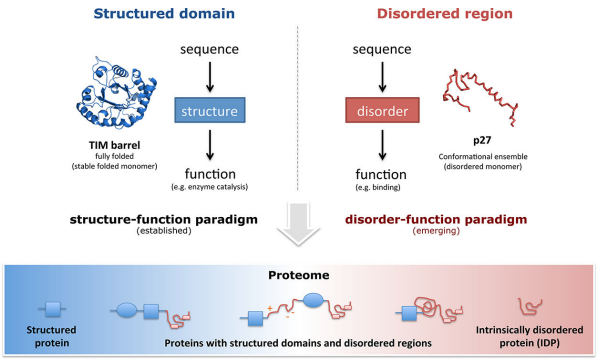

Figure 1 | Structured domains versus disordered regions in proteins. Intrinsically disordered regions violate the classical structure-function paradigm, which states a protein's sequence determines a well-defined structure for a specific function. Reproduced from Van Der Lee *et al.*, 2014.

A. Machine learning applications to intrinsically disordered regions

Previous studies have shown that machine learning algorithms can successfully identify disorder in protein sequences (e.g., IUPred2A, DisPROT, Espritz) with a high degree of accuracy (Mészáros, 2018; Piovesan et al., 2016; Walsh *et al.*, 2011). The latent representations of these models can reveal meaningful relationships and organizations of these sequences. For example, a study used natural language processing principles to predict the next residue in the sequence based on the model's hidden state. By revising its construction of the hidden state to maximize accuracy (Fig. 2), the model successfully condensed the fundamental features of a protein into a numerical representation that encodes its structural, evolutionary, and functional information

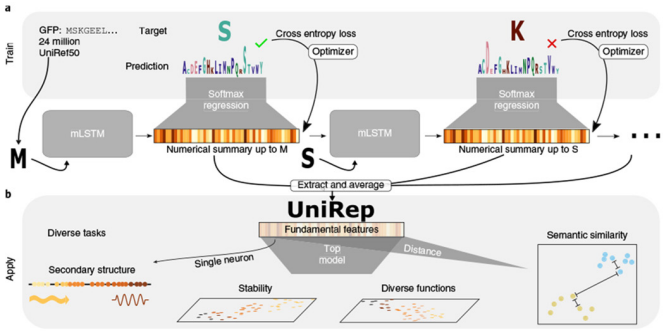

Figure 2 | Model architecture for deep protein representation. Workflow for next amino-acid prediction model from "Unified rational protein engineering with sequence based deep representation learning." Natural language processing techniques were used to develop a rich latent representation of proteins to uncover protein organizations and features. Reproduced from Alley *et al.*, 2019.

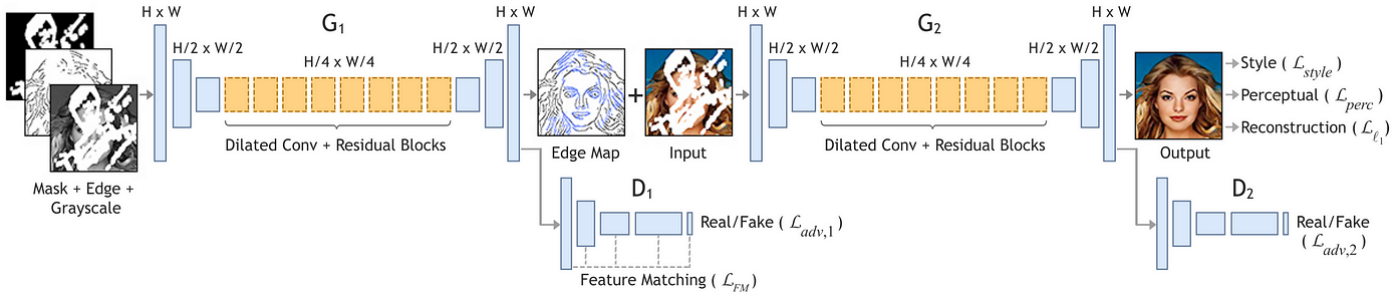

Figure 3 | Model architecture for image inpainting task for EdgeConnect. EdgeConnect is a model trained to predict color and image from a masked image. Reproduced from Nazeri *et al.*, 2019.

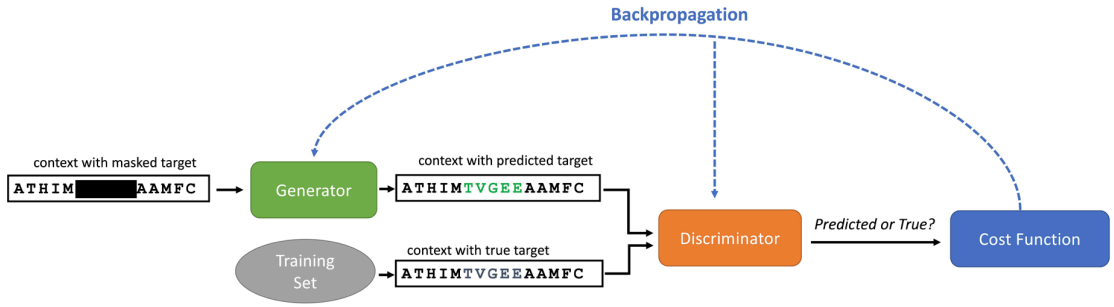

Figure 4 | Generative adversarial network implementation with an example masked amino acid input. The model and functions are in colored boxes, and the inputs and outputs are in black outlined boxes.

(Alley *et al.*, 2019).

Alley *et al.*'s success illustrates that such algorithms can encode biologically relevant representations of protein sequences. This in turn suggests a similar approach based on both next amino acid and disorder state prediction could yield insights into the structure and composition of IDRs. However, as ordered residues compose most annotations, the networks primarily predicted the most common annotation, ordered, rather than learning any relationships about the structure of the sequences themselves. To avoid these issues, we therefore sought to develop a self-supervised approach for learning rich representations of disordered regions that does not rely on the balance of disordered and ordered annotations.

B. Image inpainting problem application

Image inpainting is the task of generating plausible images given contextual pixels in a masked image (Fig. 3). Though many statistical and image processing techniques can regenerate the masked data given the surrounding image context, recent studies have made significant progress by applying deep-learning techniques (Pathak *et al.*, 2016; Yu *et al.*, 2018; Nazeri *et al.*, 2019). In these cases, the models are composed of several layers of learned convolutional filters which allow them to build a rich latent representation of the missing "target" region using only the contextual data. Therefore, training models to learn representations of disordered regions by image inpainting addresses the imbalance of order to disorder annotations because the models specifically learn to reproduce the disordered regions using their surrounding context.

Furthermore, recent studies have shown generative adversarial networks (GANs) can significantly improve the accuracy of the reconstructed images (Jam *et al.*, 2021; Wang *et al.*, 2020; Elharrouss *et al.*, 2020). GANs are made up of a generative and discriminator model. The generative model attempts to generate plausible data to "fool" the discriminator model, and the discriminator model must differentiate the generated data from the true data in the training set. Based on the result of the discriminator model, the generative model revises its internal state to produce more plausible data, and the discriminative model in turn revises its internal state to make better predictions.

C. Problem formulation

Therefore, in this work we sought to learn representations of disordered regions by applying a GAN-based image inpainting model to predict sequences of disordered regions. The surrounding amino acid sequence of the disordered region plays the role of the surrounding image or "context", and the disordered region plays the role of the masked region or "target." The generator generates an amino acid sequence of the disordered region when given the surrounding context. The discriminator then distinguishes the true target from training set between the generated target and true target. The correct or incorrect result of the discriminator is used as a training signal, which allows the generator to create more realistic sequences and therefore build richer latent representations of IDRs (Fig. 4). We find that despite training several models of varying complexity, they did not accurately predict the amino acid sequences of disordered regions, which suggests the properties of disordered regions are not encoded by their surrounding context.

Methods

A. MobiDB database extraction

Annotations of ordered and disordered regions and their amino acid sequences were extracted from MobiDB (Piovesan *et al.*, 2022). In these databases and our data processing pipeline, ordered and disordered residues were labeled as "0" and "1," respectively. To ensure the quality of the data set, the labels were obtained from only "curated" and "derived" annotations. Curated annotations are manually curated

experimental annotations from partner databases: UniProtKB (Bairoch & Apweiler, 1997), DisProt (Piovesan *et al.*, 2016), and IDEAL (Fukuchi *et al.*, 2011). Derived annotations are automatically inferred from primary data in the Protein Data Bank (Berman *et al.*, 2000) and are categorized into two types depending on the experimental technique: missing (from X-ray structures) and mobile (from NMR structures). MobiDB also defines levels of consensus within these categories, which indicate the amount of agreement required between sources for a residue to receive an annotation of disorder. The curated annotations used merge consensus, and both types of derived annotations used strict consensus (th_90). To create a single source of truth for model training, the annotations were merged into a single sequence of binary labels for each protein (Fig. 5). The curated annotations were used if they existed for the protein; otherwise the missing and mobile residues were merged using a bitwise OR operation.

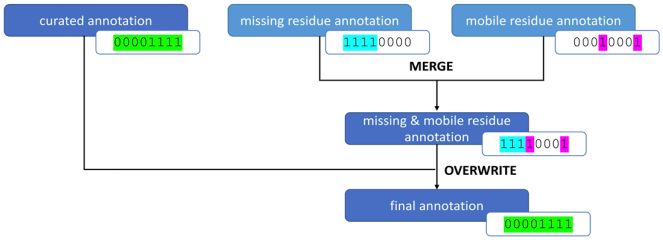

Figure 5 | Creation of the final annotations for a protein sequence. Annotations derived from various sources were merged in a process which prioritized higher quality evidence.

B. Data filtering

The resulting annotations and proteins were filtered by several criteria. First, annotations of disordered regions composed of fewer than 10 consecutive residues were removed as most disordered regions of these lengths are "flexible loops annotated from X-ray experiments that do not represent disorder-related functional sites" (Hatos *et al.*, 2020). These annotations made up roughly 16% of the annotated residues in the dataset. Proteins with nonstandard or ambiguous amino acid symbols (i.e., O, U, B, Z, X, J) were also removed. Because the protein titin has over 20,000 residues, and all other proteins contain fewer than 10,000, its annotations were excluded. These annotated residues made up around 7% of the dataset.

C. Data manipulation

Each training example consists of a masked region (the disordered region or target) and its surrounding region (surrounding amino acid sequence or context). As discussed in the "Data exploration" section of the Results, most disordered regions in the dataset had a length of 0 to 90 residues. To ensure the training examples have sufficient context for the generator to create a plausible target, their contexts were required to constitute at least 50% of their total lengths. This created an upper bound on the length of residues of the training example: 180 residues total with a maximum masked region length of 90 residues. Regions fulfilling these criteria were extracted from protein sequences and their annotations using the multidimensional image processing module "ndimage" from SciPy (Virtanen *et al.*, 2020) and then organized into a pandas dataframe (McKinney, 2010) with a total of 111,528 records. The data was split into train, test, and validation sets with an 80-10-10 ratio.

D. Model architecture

Two GAN architectures were tested. The first was a complex architecture which was modeled after a deep convolutional GAN used to predict masked images of two-dimensional protein distance maps (Li *et al.*, 2017). The second was a simple architecture which reduced the number of channels and layers from the complex architecture in an attempt to improve training stability. The networks were implemented with TensorFlow (Abadi *et al.*, 2016).

1. Complex model

The complex GAN was made up of a generator and discriminator (Fig. 6). The generator was composed of five convolutional layers with 8, 16, 32, 64, 128 filters in each layer, respectively, and five deconvolutional layers with 128, 64, 32, 16, 8 filters in each layer, respectively. Each layer of the generator had a kernel size of 3 by 3 with a stride of 1. The discriminator was composed of four convolutional layers with 25, 13, 7, 4 filters in each layer, respectively, and one fully connected layer. Each convolutional layer of the discriminator had a kernel size of 4 by 4 with a stride of 1.

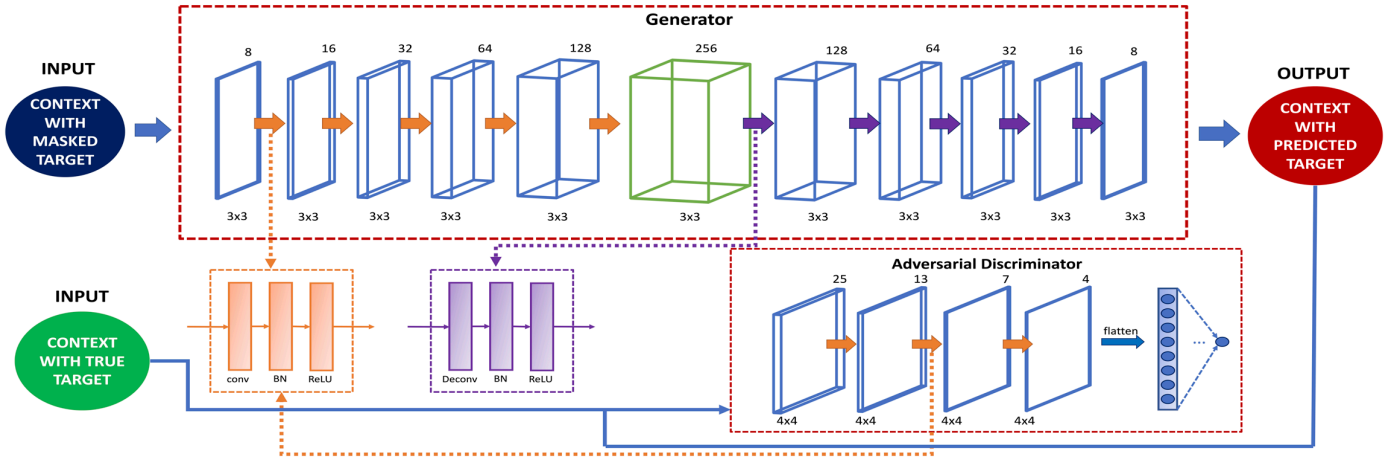

Figure 6 | Model architecture for complex GAN. Dashed boxes represent models and/or components of the models. Each box represents a layer with the depth representing the number of channels per layer. Number of channels in each layer are above the box, and kernel size in each layer is under the box. Orange arrows represent convolutional layers and purple arrows represent deconvolutional layers. Adapted from Li *et al.*

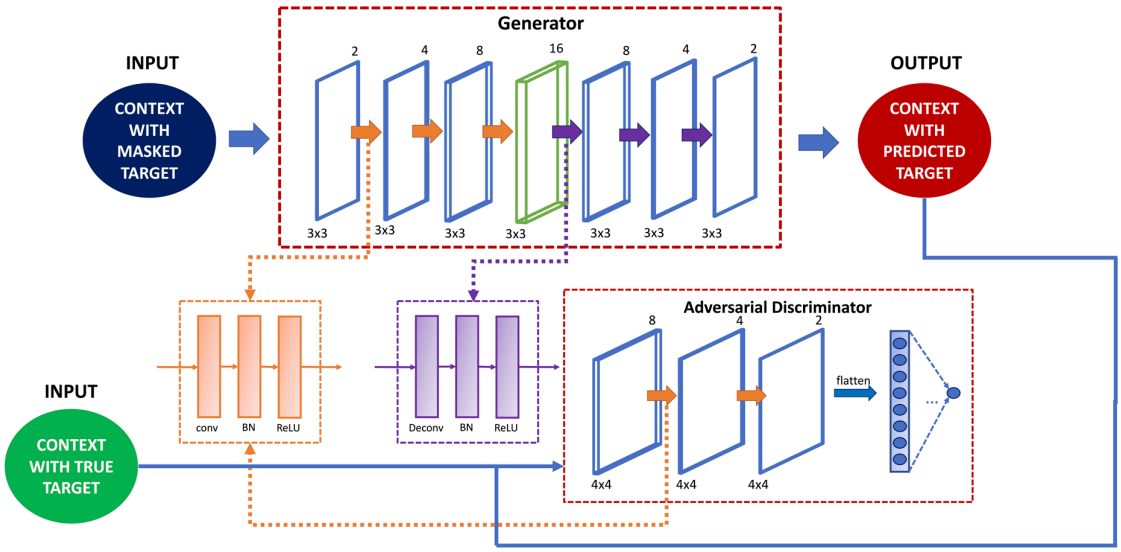

Figure 7 | Model architecture for simple GAN. See Fig.6 for details of model components.

2. Simple model

The simple GAN was made up of a generator and discriminator (Fig. 7). The generator was composed of three convolutional layers with 2, 4, 8 filters in each layer, respectively, and three deconvolutional layers with 8, 4, 2 filters in each layer, respectively. Each layer of the generator had a kernel size of 2 by 2 with a stride of 1. The discriminator was composed of three convolutional layers with 8, 4, 2 filters in each layer, respectively, and one fully connected layer. Each convolutional layer of the discriminator had a kernel size of 4 by 4 with a stride of 2.

E. Model training

The input to the generator is a one-hot-encoded training example, which is the context (surrounding amino acid sequence) with the target (disordered region) masked out. The input is fed into a neural network made up of convolutional and deconvolutional layers. The convolutional layers capture the context information, and the deconvolutional layers use the context information to generate the masked region. Each convolutional layer contains a convolutional (conv) operation, batch normalization (BN) operation, and rectified linear unit (ReLU) activation function. Each deconvolutional layer contains a deconvolutional (deconv) operation, batch normalization (BN) operation, and rectified linear unit (ReLU) activation function (Fig. 6-7). The output of the generator is a one-hot-encoded generated target.

The output of the generator and the one-hot-encoded true target are fed into the convolutional layers and a fully connected layer of the discriminator. Each convolution layer in the discriminator has the same composition as the generator. The dense layer contains a softmax activation function. The output of the discriminator is the probability that the input is a true example.

The loss functions for the generator and discriminator are calculated with equations in Section F of Methods. We performed gradient descent using the Adam optimizer (Kingma & Ba, 2014) with a learning rate of 0.0001. We trained with a batch size of 30 and 300 epochs.

F. Loss functions

The model is composed of two networks, the generator and the discriminator, which are trained simultaneously on separate loss functions. The generator minimizes the difference between the generated and true targets as well as the difference between the discriminator's evaluation of the generated target and the expected evaluation of the generated target. The discriminator minimizes the difference between the discriminator's evaluation of the true and generated targets and an expected evaluation of true and generated targets, respectively. We achieve this by defining a loss function for each model: generator loss and discriminator loss.

1. Generator loss

The generator loss made up of the reconstructive and discriminative loss of the generated target. The reconstructive loss uses the categorical cross entropy loss function to measure the dissimilarity of the generated target (g) and true target (t).

$$Loss_{recon}(g,t) = -\sum_{x} t\,log(g)$$

The discriminative loss of the generated target uses the binary cross entropy loss function to measure the dissimilarity between the discriminator's probability distribution of the generated target (y) and the expected probability distribution of a true target ($\hat{z}$), defined as tensors of ones with the length of the batch size.

$$Loss_{disc}(y,\hat{z}) = -\hat{z} * log(y) - (1-\hat{z}) * log(1-y)$$

The total generator loss is defined as:

$$Loss_{gen} = Loss_{recon}(g,t) + Loss_{disc}(y,\hat{z})$$

2. Discriminator loss

The discriminator loss is the sum of two different discriminative loss terms. The first discriminative loss uses the binary cross entropy

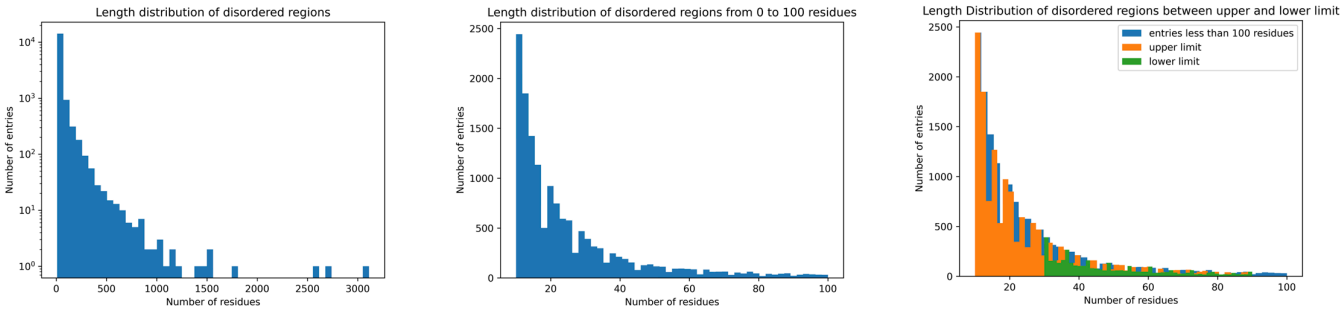

Figure 8| Length distribution of disordered regions in dataset. (A) Disordered regions. (B) Disordered regions from 0 to 100 residues. (C) Disordered regions between upper and lower limit.

loss function to measure the dissimilarity between the discriminator's probability distribution of the true target (z) and the expected probability distribution of the true target ($\hat{z}$), defined a tensors of ones with the length of the batch size.

$$Loss_{disc}(z, \hat{z}) = -\hat{z} * log(z) - (1 - \hat{z}) * log(1 - z)$$

The second discriminative term uses the binary cross entropy loss function to measure the dissimilarity between the discriminator's probability distribution of the generated target (y) and the expected probability of the generated target ($\hat{y}$), defined as a tensors of zeroes with the length of the batch size.

$$Loss_{disc}(y, \hat{y}) = -\hat{y} * log(y) - (1 - \hat{y}) * log(1 - y)$$

The total discriminator loss is defined as:

$$Loss_{disc} = Loss_{disc}(z, \hat{z}) + Loss_{disc}(y, \hat{y})$$

G. Random masking and alternate training

We applied random masking to determine if unmasking a fraction of the target increased the accuracy of the model. In these experiments, 10% of the target was randomly masked, meaning 90% of the disordered region was unmasked and available as an input to the model. To prevent modal collapse in GANs and stabilize the gradients during training, alternate training (Chavdarova & Fleuret, 2017) was implemented into both model architectures using a 10:1 ratio of gradient updates for the generator and discriminator, respectively.

H. Model metrics

We evaluated model performance as the accuracy of the generated targets. Accuracy was calculated with the following formula:

$$accuracy = \frac{number\ of\ correctly\ predicted\ target\ residues}{length\ of\ target}$$

We evaluated model training as the accuracy and loss at the end of each epoch. To evaluate the latent representation of disordered regions, the composition of generated targets was compared with true targets. Visualizations were created with Matplotlib (Hunter, 2007).

Results

A. Data exploration

Each training example contains a masked region (disordered amino acid sequence) and unmasked region (surrounding amino acid sequence). To ensure the generator has sufficient information to create a plausible target, we required that the unmasked contexts constitute at 50% of total lengths of each example. We explored the dataset described in the "Data Manipulation" section of the Methods to find a length range of masked regions that would yield a sufficient number of examples fulfilling this restriction.

The masked region (called "entries" in figures) had lengths that varied between 10 to 3200 residues (Fig. 8A). However, most masked regions had 10 to 500 residues, and their numbers sharply decreased after a length of only 90 residues (Fig. 8B). The maximum length or "upper limit" of the masked regions was therefore set to 90 residues. Accordingly, the training example length was 180 residues to ensure each example was 50% or more unmasked.For the training example to be at least 15% masked, the minimum length or "lower limit" of the training example was set to 30 residues (Fig. 8C).

B. Model performance metrics

Model performance and learning was measured with loss curves and accuracy for both training and validation sets. The loss curve is used to measure model error over time, and model accuracy measures the accuracy of the generated symbols or how close the generated target is to the true target. The model accuracy should be opposite to the loss curve where the model increases insymbol accuracy and decreases its error over subsequent epochs. The loss curve for the training dataset indicates how well the model fits to the training dataset, and the loss curve for the validation dataset indicates how well the model fits to new unseen data or the validation dataset.

To investigate if the model had learned meaningful representations of disordered sequences or only repeating the most overrepresented disordered amino acid to hit the highest accuracy, the amino acid composition of both generated and true disordered regions were compared. The true and generated amino acid for each position in the disordered regions were compared in a heatmap to determine if generated and true amino acids were correlated or the model generated the same amino acid regardless of the true amino acid.

C. Model exploration

Six models were trained, and the model performance was evaluated with metrics described in "Model performance metrics" in

Results. The corresponding name of the model in the code repository is in parathesis next to the title of the experiment. Networks with few layers and a small number of filters had higher training stability as shown by the decreased oscillations in their loss curves (Fig. 10A) than networks with more layers and a large number of filters which had significant oscillations in their loss curves (Fig. 9A). Alternate training led to little to no significant oscillations in loss curves (Fig. 11A, 13A, 14A). Random masking of the target led to less overrepresentation of a single amino acid in generated targets (Fig. 13C, 14C). The accuracy of models did not increase significantly through experiments. This indicates that although simpler architectures and implementation of alternate training and random masking stabilized training for the network it did not lead better predictions of generated targets.

1. Complex model (model_0-00)

Constant oscillation in the loss curve (Fig. 9A) throughout epochs for the training and validation losses show unstable network training and learning. There is little to no difference in training and validation sets showing that training did not improve the model's loss and accuracy (Fig. 9A, Fig. 9B). Looking at the generated targets for the model, indicated that the model predicted the one of the most common amino acids in the true target, serine (S), for the generated target with no correlation for any amino acid (Fig. 9C, Fig 9D).

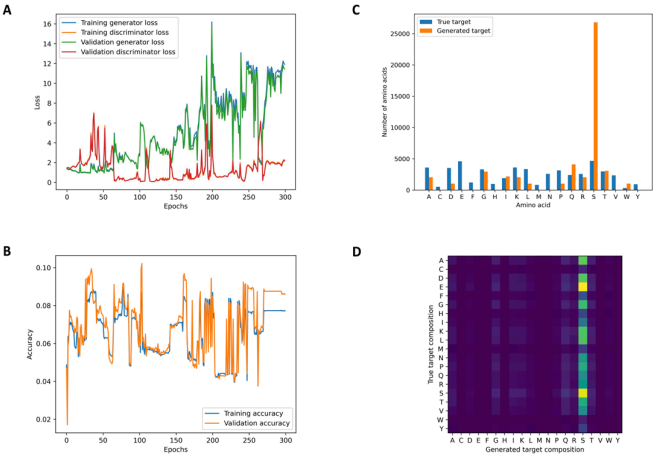

Figure 9 | Result metrics of complex model. (A) Generator and discriminator loss for the training and validation dataset. (B) Symbol accuracy of generator. (C) Amino acid composition of generated and true target. (D) Correlation of true and generated amino acid.

2. Simple model (model_0-01)

Given the constant oscillation in the loss curve from the previous model (Fig. 9A), we hypothesized that the large number of network layers and filters per layer were factors in the unstable training. Based on the size of the dataset, a network with many layers and filters per layer may be inappropriate for training. Therefore, we simplified the architecture of the model in an attempt to stabilize training and increase accuracy.

By decreasing the number of layers and filters per layer, the model was able to achieve more stable training than the complex model demonstrated by decreased oscillations in the loss curve (Fig. 9A). The increase in stability of training, however, did not significantly impact the accuracy of the model (Fig. 10B). Looking at the amino acid composition for the generated targets (Fig. 10C), indicated that the model continued to predict one of the most common amino acids in the true target, isoleucine (I), for the generated target with no correlation for any amino acid (Fig. 10D). This result is indicative of modal collapse, where the generator continues to predict the same amino acid regardless of previous incorrect predictions.

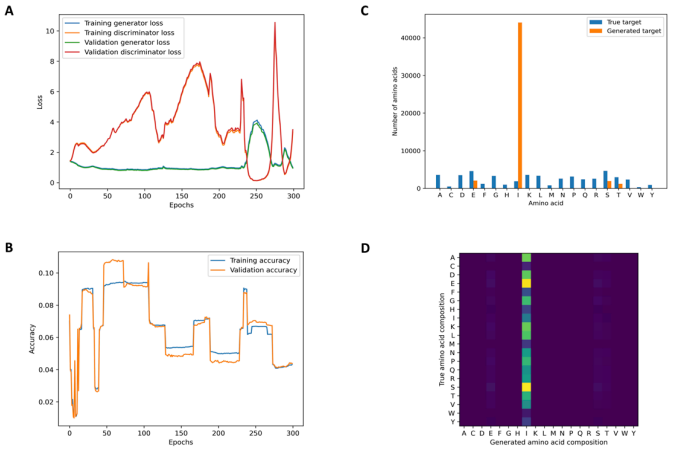

Figure 10 | Result metrics of simple model. (A) Generator and discriminator loss for the training and validation dataset. (B) Symbol accuracy of generator. (C) Amino acid composition of generated and true target. (D) Correlation of true and generated amino acid.

3. Simple model with alternate training (model_0-02)

To diagnose the modal collapse, we implemented alternate training on the simple model. The generator is trained for 10 epochs and then updated using the current discriminator's parameters. This allows for the generator to focus more on generating plausible and diverse targets closer to the true targets rather than only trying to fool the discriminator. The training is then alternated where the discriminator is trained for 10 epochs and then updated using the current generator's parameters. This allows for the discriminator to better distinguish between the generated target and the true target and recognize the pattern in the generated targets.

Alternate training improved the stability of the model, as demonstrated by the absence of oscillations in the loss curves (Fig. 11A). However, this approach resulted in deviation from the typical behavior of loss curves in GANs, where the generator and discriminator losses diverge initially and then converge in parallel. Although the loss curves show the divergence, there is a lack of parallel convergence. Improving training stability did not lead to more accurate generated targets (Fig. 11B). The generated targets continued to show overrepresentations of a single amino acid, glycine (G), (Fig. 11C) with no correlation for any true amino acid (Fig. 11D). The generated target shows inclusion of other amino acids but not as significantly as glycine.

4. Simple model with only generator(model_1-00)

To determine if a simpler training task would be more successful in building a more accurate composition of disordered regions, we trained a simple model with only a generator. The model only had the task of generating targets given the surrounding context. A simpler task for the model can lead to convergence because the model is only sensitive to the optimal hyperparameters for the generator whereas having a discriminator and generator makes the model sensitive to the choice of hyperparameters for both.

The simpler training task resulted in more stable training, as seen in loss curves (Fig. 12A); however, did not significantly impact the accuracy (Fig. 12B). The accuracy overall increases through training

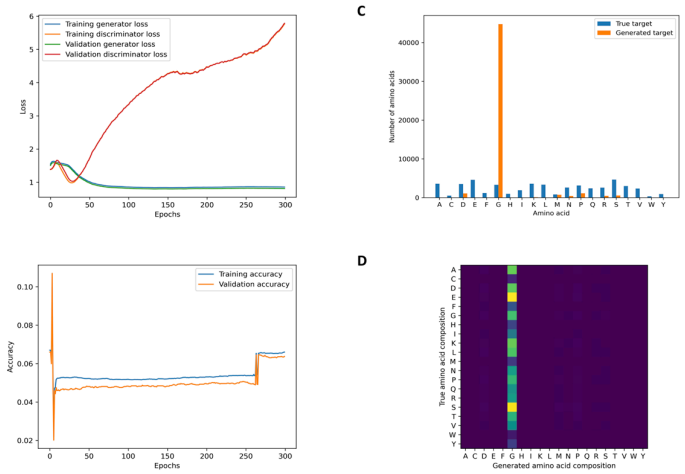

Figure 11 | Result metrics of simple model with alternate training. (A) Generator and discriminator loss for the training and validation dataset. (B) Symbol accuracy of generator. (C) Amino acid composition of generated and true target. (D) Correlation of true and generated amino acid.

even though the resulting accuracy is low. The generated targets are comprised of only two common true target amino acids, glutamate (E) and serine (S), with an overrepresentation of glutamate (Fig. 12C). The presence of only two different amino acids in the generated target is unlike other model's generated targets (Fig. 9A, 10A, 11A) where the targets are comprised of a few different amino acids. The generated targets do not show correlation for any amino acid of the true target (Fig 12D).

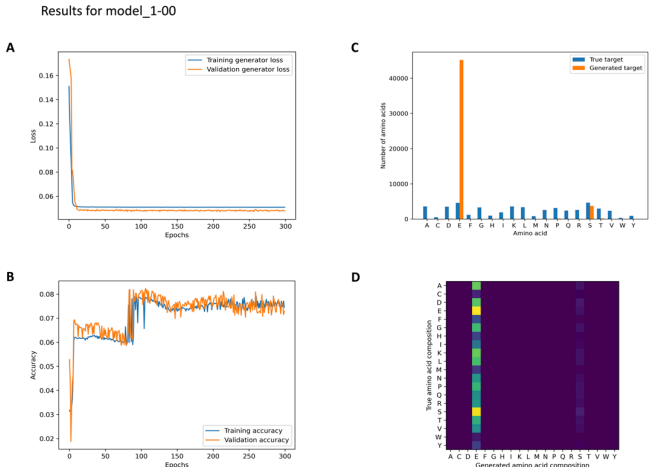

Figure 12| Result metrics of simple model with only generator. (A) Generator and discriminator loss for the training and validation dataset. (B) Symbol accuracy of generator. (C) Amino acid composition of generated and true target. (D) Correlation of true and generated amino acid.

5. Simple model with random masking and alternate training(model_0-02rm)

To address the low accuracy of the model, we hypothesized that the complete masking of the disordered region may have contributed to the model's poor latent space representation of disordered regions. Therefore, we partially masked the target instead of fully masking as done in previous models (i.e., model_0-00, model_0-01, model_1-00).

The implementation of random masking is explained in the "Random masking and alternate training" section of the Methods. By increasing the surrounding context to the disordered region, the generator would gain more information about the disordered region itself as opposed to only the surrounding context.

The loss curves (Fig. 13A) show stable training but lack parallel convergence of the generator and discriminator. There is a notable difference between the validation and training accuracy, showing that training improved the model's performance on the validation dataset (Fig. 13B). Although the improved training, the resulting accuracy is not significantly different than the previous models. The amino acid composition of the generated target shows increased representation of different amino acids (Fig. 13C) compared to overrepresentation of a single amino acid in previous model's generated targets (Fig. 10C, 11C, 12C). Two amino acids, serine (S) and valine (V), are significantly represented in the model's generated targets with no correlation for any other amino acid for the true target (Fig. 13D).

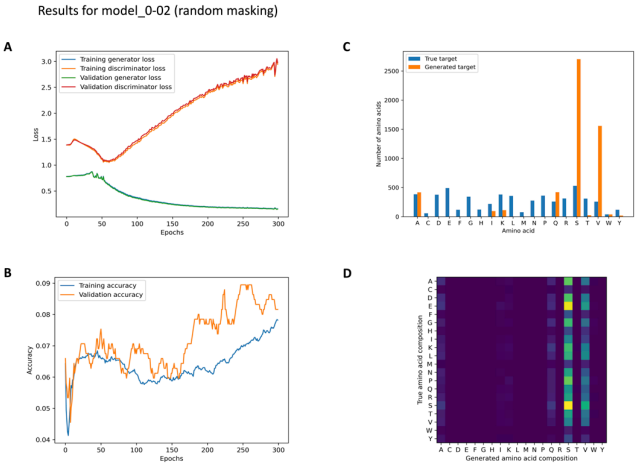

Figure 13 | Result metrics of simple model with random masking and alternate training. (A) Generator and discriminator loss for the training and validation dataset. (B) Symbol accuracy of generator. (C) Amino acid composition of generated and true target. (D) Correlation of true and generated amino acid.

6. Simple model with only generator and random masking (model_1-00rm)

To determine if the discriminator adds unnecessary complexity to model_0-02rm, we hypothesized that removing the discriminator would lead to a higher accuracy. As seen in model_1-00, the model has a simpler training task by removing the discriminator because the model's only focus is on generating targets close to the true targets.

Although the loss curves demonstrate stable training of the generator (Fig. 14A), the accuracy of the model (Fig. 14B) has not significantly increased than the accuracy of previous models (Fig. 9B, 10B, 11B, 12B, 13B). The trend of the accuracy over epochs mirrors the trend of accuracy of a model similarly with only a generator (i.e. model_1-00), demonstrating that a discriminator may help to improve accuracy over epochs even though the resulting accuracy will be similar. The generated targets show representation of several common amino acids found in the true target with a high representation of glutamate (E) and serine (S) (Fig. 14C). The amino acids of the generated target show no strong correlation to any amino acids of the target (Fig. 15C).

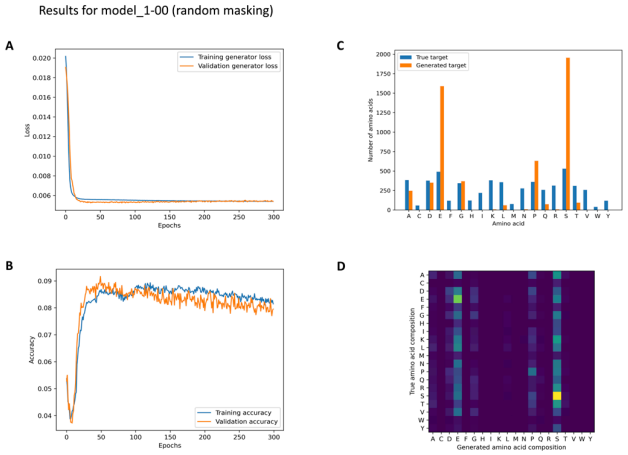

Figure 14 | Result metrics of simple model with only generator and random masking. (A) Generator and discriminator loss for the training and validation dataset. (B) Symbol accuracy of generator. (C) Amino acid composition of generated and true target. (D) Correlation of true and generated amino acid.

Discussion

Motivated by the similarity between images and sequences, we adapted the architecture of a deep convolution generative adversarial network trained to predict protein loops on 2D distance maps/images to instead predict the amino acid composition of disordered regions in proteins.

However, when this model demonstrated unstable training, we implemented a model with simpler architecture. The generated disordered regions (targets) of this model showed large repeats of common amino acids found in the true targets, a sign of modal collapse. Modal collapse refers to when generators only produce a limited number of outputs, in our study, the generator only produced a single or few common amino acids of the true target in repeated segments in the generated target. To diagnose the modal collapse and low accuracy of the generated targets, we implemented alternate training and random masking of the target.

Inspired by the success of GANs in improving the accuracy of neural networks trained to reconstruct missing regions of images, we adopted the GAN architecture to increase the accuracy of generated targets and build a richer representation of disordered regions. However, despite exploring many architectures and training methods, the accuracy remained low. A possible explanation for these results is the available data is not sufficient to train a model this complexity. Successfully trained GANs for image inpainting have hundreds of thousands of images in their training data. In contrast, the dataset used in this model only had about two thousand context-target pairs. Alternatively, the amino acid sequence surrounding the disordered region may not impact the amino acid sequence of the disordered region, resulting in random amino acid sequences.

The small size of our dataset is a consequence of using only high-quality "gold standard" data from experimental sources. Because experimental determination of ordered and disordered regions remain technically challenging and low throughput, relatively few proteins have comprehensive disorder annotations. Though disorder annotations inferred from computational predictions or homology are plentiful, we chose to exclude these sources to prevent circular dependencies or propagate biases inherent in the datasets used to create these algorithms. Datasets for convolutional neural networks and machine learning architectures are large and the size of these datasets are one of the reasons models like ChatGTP are so successful in their respective tasks. Thus, using GANs to learn representations of disordered region may be more successful when there is more high-quality experimental data.

One method to improve the prediction accuracy is to adapt this model to exclude the surrounding context region and instead generate disordered regions from a random seed as in more traditional applications of the architecture. This method may be more successful because as our results suggest that disordered regions are not encoded by their context. Another prospective future of better understanding IDRs is using homology information from disordered regions. This modelwould learn the latent statistical representation of disordered regions by discerning the "correct" homologous disordered region in a set of unrelated disordered regions or homology sequences. This approach implemented in a recent study using a set of homologous yeast IDRs (Lu *et al.*, 2022) but could be extended to other systems such as Drosophila using recently released genomic resources (Singleton & Eisen, 2023). Thus, many promising techniques for revealing the latent "structure" of these enigmatic protein sequences remain unexplored.

REFERENCES

1. Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., … Zheng, X. (2016). TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems. In arXiv [cs.DC]. arXiv. http://arxiv.org/abs/1603.04467
2. Alley, E. C., Khimulya, G., Biswas, S., AlQuraishi, M., & Church, G. M. (2019). Unified rational protein engineering with sequence-based deep representation learning. Nature Methods, 16(12), 1315– 1322. https://doi.org/10.1038/s41592-019-0598-1
3. Babu, M. M., Kriwacki, R. W., & Pappu, R. V. (2012). Versatility from Protein Disorder. Science, 337(6101), 1460–1461. https://doi.org/10.1126/science.1228775
4. Bairoch, A., & Apweiler, R. (1997). The SWISS-PROT protein sequence database: its relevance to human molecular medical research. Journal of Molecular Medicine, 75(5), 312–316. https://www.ncbi.nlm.nih.gov/pubmed/9181472
5. Berman, H. M., Westbrook, J., Feng, Z., Gilliland, G., Bhat, T. N., Weissig, H., Shindyalov, I. N., & Bourne, P. E. (2000). The Protein Data Bank. Nucleic Acids Research, 28(1), 235–242. https://doi.org/10.1093/nar/28.1.235
6. Chavdarova, T., & Fleuret, F. (2017). SGAN: An Alternative Training of Generative Adversarial Networks. In arXiv [stat.ML]. arXiv. http://arxiv.org/abs/1712.02330
7. Dosztányi, Z., Csizmok, V., Tompa, P., & Simon, I. (2005). IUPred: web server for the prediction of intrinsically unstructured regions of proteins based on estimated energy content. Bioinformatics, 21(16), 3433–3434. https://doi.org/10.1093/bioinformatics/bti541
8. Elharrouss, O., Almaadeed, N., Al-Maadeed, S., & Akbari, Y. (2020). Image Inpainting: A Review. Neural Processing Letters, 51(2), 2007–2028. https://doi.org/10.1007/s11063-019-10163-0
9. Frasca, M., Bertoni, A., & Valentini, G. (2015). UNIPred: Unbalance-Aware Network Integration and Prediction of Protein Functions. Journal of Computational Biology: A Journal of Computational Molecular Cell Biology, 22(12), 1057–1074. https://doi.org/10.1089/cmb.2014.0110
10. Fukuchi, S., Sakamoto, S., Nobe, Y., Murakami, S. D., Amemiya,

T., Hosoda, K., Koike, R., Hiroaki, H., & Ota, M. (2012). IDEAL: Intrinsically Disordered proteins with Extensive Annotations and Literature. Nucleic Acids Research, 40(Database issue), D507–D511. https://doi.org/10.1093/nar/gkr884
11. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., & Bengio, Y. (2020). Generative adversarial networks. Communications of the ACM, 63(11), 139–144. https://doi.org/10.1145/3422622
12. Hatos, A., Hajdu-Soltész, B., Monzon, A. M., Palopoli, N., Álvarez, L., Aykac-Fas, B., Bassot, C., Benítez, G. I., Bevilacqua, M., Chasapi, A., Chemes, L., Davey, N. E., Davidović, R., Dunker, A. K., Elofsson, A., Gobeill, J., Foutel, N. S. G., Sudha, G., Guharoy, M., … Piovesan, D. (2020). DisProt: intrinsic protein disorder annotation in 2020. Nucleic Acids Research, 48(D1), D269– D276. https://doi.org/10.1093/nar/gkz975
13. Hunter. (2007). Matplotlib: A 2D Graphics Environment. Computing in Science & Engineering, 9, 90–95. https://doi.org/10.1109/MCSE.2007.55
14. Jam, J., Kendrick, C., Walker, K., Drouard, V., Hsu, J. G.-S., & Yap, M. H. (2021). A comprehensive review of past and present image inpainting methods. Computer Vision and Image Understanding: CVIU, 203, 103147. https://doi.org/10.1016/j.cviu.2020.103147
15. Kingma, D. P., & Ba, J. (2014). Adam: A Method for Stochastic Optimization. In arXiv [cs.LG]. arXiv. http://arxiv.org/abs/1412.6980
16. Li, Z., Nguyen, S. P., Xu, D., & Shang, Y. (2017). Protein Loop Modeling Using Deep Generative Adversarial Network. 2017 IEEE 29th International Conference on Tools with Artificial Intelligence (ICTAI), 1085–1091. https://doi.org/10.1109/ICTAI.2017.00166
17. Lu, A. X., Lu, A. X., Pritišanac, I., Zarin, T., Forman-Kay, J. D., & Moses, A. M. (2022). Discovering molecular features of intrinsically disordered regions by using evolution for contrastive learning. PLoS Computational Biology, 18(6), e1010238. https://doi.org/10.1371/journal.pcbi.1010238
18. McKinney, W. (2010). Data Structures for Statistical Computing in Python. Proceedings of the 9th Python in Science Conference. Python in Science Conference, Austin, Texas. https://doi.org/10.25080/majora-92bf1922-00a
19. Mészáros, B., Erdos, G., & Dosztányi, Z. (2018). IUPred2A: context-dependent prediction of protein disorder as a function of redox state and protein binding. Nucleic Acids Research, 46(W1), W329–W337. https://doi.org/10.1093/nar/gky384
20. Nazeri, K., Ng, E., Joseph, T., Qureshi, F. Z., & Ebrahimi, M. (2019). EdgeConnect: Generative Image Inpainting with Adversarial Edge Learning. In arXiv [cs.CV]. arXiv. http://arxiv.org/abs/1901.00212
21. Pathak, D., Krahenbuhl, P., Donahue, J., Darrell, T., & Efros, A. A. (2016). Context Encoders: Feature Learning by Inpainting. In arXiv [cs.CV]. arXiv. http://arxiv.org/abs/1604.07379
22. Piovesan, D., Del Conte, A., Clementel, D., Monzon, A. M., Bevilacqua, M., Aspromonte, M. C., Iserte, J. A., Orti, F. E., Marino-Buslje, C., & Tosatto, S. C. E. (2023). MobiDB: 10 years of intrinsically disordered proteins. Nucleic Acids Research, 51(D1), D438–D444. https://doi.org/10.1093/nar/gkac1065
23. Piovesan, D., Tabaro, F., Mičetić, I., Necci, M., Quaglia, F., Oldfield, C. J., Aspromonte, M. C., Davey, N. E., Davidović, R., Dosztányi, Z., Elofsson, A., Gasparini, A., Hatos, A., Kajava, A. V., Kalmar, L., Leonardi, E., Lazar, T., Macedo-Ribeiro, S., Macossay-Castillo, M., … Tosatto, S. C. E. (2017). DisProt 7.0: a major update of the database of disordered proteins. Nucleic Acids Research, 45(D1), D219–D227. https://doi.org/10.1093/nar/gkw1056
24. Singleton, M., & Eisen, M. (2023). Leveraging genomic redundancy to improve inference and alignment of orthologous proteins. In bioRxiv (p. 2023.01.24.525427). https://doi.org/10.1101/2023.01.24.525427
25. Van Der Lee, R., Buljan, M., Lang, B., Weatheritt, R. J., Daughdrill, G. W., Dunker, A. K., Fuxreiter, M., Gough, J., Gsponer, J., Jones, D. T., Kim, P. M., Kriwacki, R. W., Oldfield, C. J., Pappu, R. V., Tompa, P., Uversky, V. N., Wright, P. E., & Babu, M. M. (2014). Classification of intrinsically disordered regions and proteins. Chemical Reviews, 114(13), 6589–6631. https://doi.org/10.1021/cr400525m
26. Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S. J., Brett, M., Wilson, J., Millman, K. J., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., … SciPy 1.0 Contributors. (2020). SciPy 1.0: fundamental algorithms for scientific computing in Python. Nature Methods, 17(3), 261–272. https://doi.org/10.1038/s41592-019-0686-2
27. Walsh, I., Martin, A. J. M., Di Domenico, T., & Tosatto, S. C. E. (2012). ESpritz: accurate and fast prediction of protein disorder. Bioinformatics, 28(4), 503–509. https://doi.org/10.1093/bioinformatics/btr682
28. Wang, L., Chen, W., Yang, W., Bi, F., & Yu, F. R. (2020). A State-of-the-Art Review on Image Synthesis With Generative Adversarial Networks. IEEE Access, 8, 63514–63537. https://doi.org/10.1109/ACCESS.2020.2982224
29. Wright, P. E., & Dyson, H. J. (2015). Intrinsically disordered proteins in cellular signalling and regulation. Nature Reviews. Molecular Cell Biology, 16(1), 18–29. https://doi.org/10.1038/nrm3920
30. Yu, J., Lin, Z., Yang, J., Shen, X., Lu, X., & Huang, T. S. (2018). Generative Image Inpainting with Contextual Attention. In arXiv [cs.CV]. arXiv. http://arxiv.org/abs/1801.07892