Computers and Chemical Engineering 131 (2019) 106533

Contents lists available at ScienceDirect

Computers and Chemical Engineering

journal homepage: www.elsevier.com/locate/compchemeng

ProGAN: Protein solubility generative adversarial nets for data augmentation in DNN framework

Xi Han[a], Liheng Zhang[b], Kang Zhou[a,c,*], Xiaonan Wang[a,*]

[a] *Department of Chemical and Biomolecular Engineering, National University of Singapore, Singapore 117585, Singapore*
[b] *Department of Computer Science, University of Central Florida, Orlando, FL 32816, USA*
[c] *Disruptive & Sustainable Technologies for Agricultural Precision, Singapore-MIT Alliance for Research and Technology, Singapore 138602, Singapore*

ARTICLE INFO

Article history:
Received 9 May 2019
Revised 25 July 2019
Accepted 10 August 2019
Available online 11 October 2019

Keywords:
Deep learning
Protein solubility
Deep neural networks
Generative adversarial networks
Model prediction

ABSTRACT

Protein solubility plays a critical role in improving production yield of recombinant proteins in biocatalysis applications. To some extent, protein solubility can represent the function and activity of biocatalysts which are mainly composed of recombinant proteins. In literature, many machine learning models have been investigated to predict protein solubility from protein sequence, whereas parameters of those models were underdetermined with insufficient data of protein solubility. Here we propose a deep neural network (DNN) as a more accurate regression predictive model. Moreover, to tackle the insufficient data problem, a novel data augmentation algorithm, Protein Solubility Generative Adversarial Nets (ProGAN), was proposed for improving the prediction of protein solubility. After adding mimic data produced from ProGAN, the prediction performance measured by R^2 was improved compared with that without data augmentation. A R^2 value of 0.4504 was achieved, which was enhanced about 10% compared with the previous study using the same dataset.

1. Introduction

Developing highly active biocatalysts that can substantially reduce the production cost is a crucial process in biochemical applications (Madhavan et al., 2017). Biocatalysts are mainly enzymes, which are mostly recombinant proteins. *Escherichia coli* is a bacterium commonly used to express recombinant proteins (Chan et al., 2010). However, low activity of those recombinant proteins results in inefficient biocatalytic processes and limits the development of novel enzymes.

Some experimental technologies (e.g. directed evolution, immobilization, using better promoters, optimizing codon usage, or changing culture media, temperature and/or other culture conditions) could enhance the performance of recombinant proteins (Madhavan et al., 2017; Wang et al., 2012; Idicula-Thomas and Balaji, 2005; Magnan et al., 2009). However, such empirical optimizations are time-consuming and labor-intensive and more importantly, they often fail due to ambiguous reasons. A more effective solution is highly desired such as an accurate computational model that can predict protein activity from its amino acid sequence directly. In biological and metabolic engineering, some computational models, such as machine learning models, have emerged and improved the design and productivity of various bioprocesses. Neural networks were designed to optimize a setpoint-tracking control loop for the dissolved oxygen concentration in a biological wastewater treatment process (Sadeghassadi et al., 2018). An artificial neural network was used as a surrogate model to optimize the synthesis of the optimum biodiesel production plant (Fahmi and Cremaschi, 2012). A recurrent neural network was developed to track the dynamics of fed-batch yeast fermentation (Gadkar et al., 2005). Therefore, models obtained by using machine learning may provide an effective approach to improve protein activity.

Since activity and solubility of proteins are correlated to some extent, and a dataset of protein solubility is available for exploration, protein solubility has been used as a proxy for protein activity (Han et al., 2019). The association between amino acid sequence and protein solubility has been investigated by using a large number of machine learning (ML) models and the prediction accuracy is being gradually improved. As a subset of artificial intelligence (AI), ML is a scientific method combining algorithms and statistical models to enable computer systems to perform a specific task effectively by identifying patterns and inference, without having to use explicit instructions (Bishop, 2006). As a branch of ML, supervised learning includes classification modelling which approximates a mapping function from input variables to discrete output variables and regression modelling which

* Corresponding authors.
E-mail addresses: kang.zhou@nus.edu.sg (K. Zhou), chewxia@nus.edu.sg (X. Wang).

https://doi.org/10.1016/j.compchemeng.2019.106533
0098-1354/© 2019 Elsevier Ltd. All rights reserved.

outputs continuous variables (Russell and Norvig, 2016). Most machine learning models can be used both for classification tasks and regression tasks and the two versions of one model are based on the same theory. For example, Support Vector Machine (SVM) for regression assigns two slack variables to each training data point in the optimization function and predicts continuous values, whereas the classification version uses one slack variable and outputs categories. A simple regression method was first developed by Wilkinson and Harrison (1991) to predict protein solubility from amino acid sequence, which achieved an accuracy of 0.88 with 81 protein sequences. Subsequently, the accuracy was improved to 0.94 by a logistic regression model with 212 proteins (Diaz et al., 2010). It was recognized that small dataset was not enough for generating a generic model that can be applied to new proteins, and then different SVM models were developed with more than 1000 samples (Niwa et al., 2009), with 2159 proteins (Agostini et al., 2012), and with 5692 proteins (Xiaohui et al., 2014), respectively. Moreover, other types of machine learning models were explored, such as decision tree (Arrowsmith et al., 2000; Goh et al., 2004; Rumelhart et al., 1985), Naive Bayes (Smialowski et al., 2006), random forest (Fang and Fang, 2013; Hirose et al., 2011), and gradient boosting machine (Rawi et al., 2017). Several software and web servers were also developed for protein solubility prediction, such as ESPRESSO (Hirose and Noguchi, 2013), Pros (Hirose and Noguchi, 2013), PROSOII (Smialowski et al., 2012), SOLpro (Magnan et al., 2009) and PROSO (Smialowski et al., 2006).

Among the large number of databases published in previous studies, most machine learning models were developed based on databases divided into two discrete categories, i.e. soluble or insoluble proteins. However, the prediction of continuous values of solubility is more meaningful to guide experiments in protein engineering. If some mutations of the protein sequence were made to improve the protein solubility, binary values of predicted solubility would fail to differentiate proteins with the same value (0 or 1) and cannot show the change caused by the mutagenesis. For example, with the threshold of 0.5, the mutation of protein sequence that improves solubility from 0.1 to 0.4 would seem meaningless since the binary value of predicted solubility would still be 0. To the best of our knowledge, only one previous work (Han et al., 2019) predicted continuous values of protein solubility ranging from 0 to 1 with the database comprised of continuous values of protein solubility, e.g. eSol database. Other studies either used databases grouped into soluble proteins and insoluble proteins or converted continuous values of protein solubility into binary values by using an arbitrarily determined threshold (Habibi et al., 2014; Chang et al., 2013). However, only 3173 proteins were collected in the eSol database. Compared with other databases with binary values of protein solubility (Habibi et al., 2014), the size of the eSol database might be insufficient, which might result in the overfitting problem during the training of machine learning models. With the development of exploration of protein solubility, the size of datasets was gradually enlarged to build prediction models used widely and improve prediction performance. For example, one recent work utilized a dataset consisting of 58,689 soluble and 70,954 insoluble proteins to train a deep learning framework and attained an accuracy of 0.77 (Khurana et al., 2018). In addition, according to the literature, the predictive power of classifiers largely depended on the size of training samples and would improve with the increasing data size when the training data were not enough (Figueroa et al., 2012). We assumed that enlarging the data size might improve the prediction performance of protein solubility. One approach to validate this hypothesis is using a data augmentation algorithm.

Generative Adversarial Networks (GANs), a state-of-the-art data augmentation algorithm in AI, have mainly been applied in the fields of image recognition and computer vision. GANs can help scientists leverage existing datasets by uncovering patterns in these datasets and reduce labor-intensive hands-on experiments. Many variants of GANs have been developed, which enhanced the performance of GANs to some extent, such as Conditional GAN (CGAN) (Mirza and Osindero, 2014), Wasserstein GAN (WGAN) (Arjovsky et al., 2017), Conditional WGAN (Gulrajani et al., 2017), Localized GAN (LGAN) (Qi et al., 2018). Recently, various data augmentation algorithms have demonstrated their usefulness in generating artificial data in diverse biological areas, such as processing genes, protein sequences, and drugs, using Feedback GAN (FBGAN) (Gupta and Zou, 2018), SeqGAN (Yu et al., 2017), etc. Data augmentation provides a new platform for constructing models in biological research limited by small datasets. A novel feedback-loop architecture, called Feedback GAN (FBGAN), was used to generate artificial DNA sequences for optimizing desired antimicrobial properties based on a recurrent neural network classifier (Gupta and Zou, 2018). In addition, a sequence generating framework, called SeqGAN, was proposed for generating sequences of discrete tokens to reduce the cost of extensive experiments on synthetic data (Yu et al., 2017). The Wasserstein distance was used as the loss function to further improve training stability of SeqGAN. The molecules encoded as text sequence and musical melodies were generated and the information learned from original data and sample diversity can be remained after data augmentation (Guimaraes et al., 2017). Moreover, some architectures of conditional WGAN showed great promise in datasets with labels, which generated mimic data similar to real data, and explored the relationship between features and labels at the same time (Odena et al., 2016; Isola et al., 2017; Zhu et al., 2017). However, conditional WGAN has not been investigated to solve the problem of protein solubility, which aims to predict solubility of a protein from its amino acid sequence. Data augmentation which can alleviate the problem of lacking sufficient data is always highly desired in biological fields and benefits other engineering fields where generating data is troublesome or even impossible.

Deep learning, a branch of machine learning, has advanced rapidly during the last two decades and demonstrated tremendous progress in bioinformatics (Min et al., 2017). Applications of deep learning are attracting more and more attention from both academia and industry. Useful insights can be gained by analyzing biological data through various deep learning architectures (e.g., deep neural networks (DNN) (Hinton et al., 2012), convolutional neural networks (Krizhevsky et al., 2012), recursive neural networks (Socher et al., 2011), recurrent neural networks (Pineda, 1987), and shallow neural networks (Mikolov et al., 2013)). DNN is well known for its advantages of analyzing data in high dimensions, which is one of the key characteristics of complex bioinformatic data. DNN has shown great potential in exploring patterns or correlations in bioinformatics (Aliper et al., 2016; Zhang et al., 2017; Xu et al., 2015). For predicting protein solubility from protein sequence, a convolutional neural network was investigated recently on a dataset with only binary values of protein solubility (Khurana et al., 2018). However, binary values of protein solubility are not suitable for guiding protein engineering as discussed.

In the present study, we first proposed a DNN architecture to predict protein solubility in continuous values from 0 to 1. DNN is more suitable for our relatively small dataset, compared with the convolutional neural network used before (Khurana et al., 2018). To tackle the problem of overfitting, we built a **Pro**tein Solubility **G**enerative **A**dversarial **N**ets (ProGAN) to generate artificial data to improve the performance of DNN. In our study, ProGAN successfully enhanced the regression evaluation metric R^2 of our predictive model DNN by enlarging the dataset with mimic data. We compared our approach with previous work predicting continuous values of solubility from protein sequence (Han et al., 2019), and demonstrated that the proposed method improved R^2 from 0.4115 to 0.4504.

The main contributions of this paper can be summarized as follows:

(1) By proposing the data augmentation algorithm ProGAN, mimic data of protein sequence and protein solubility were generated without experiments *in vivo*. It benefits not only the bioinformatics but also other fields where producing data is time-consuming or challenging. Moreover, by this algorithm, the predictive model was improved with the enlarged dataset.
(2) The regression model for predicting protein solubility from protein sequence achieved a R^2 value of 0.4504, which enhanced about 10% compared with previous regression predictive models with the same dataset. More accurate models are definitely favorable for their further applications in laboratories or industry.
(3) Deep learning framework DNN was utilized to explore the relationship between protein solubility and protein sequence for the first time and performed better compared with previous work using SVM (Han et al., 2019). Deep learning models have advantages over the traditional machine learning models. Our study lays a foundation for further addressing the problem of protein solubility by using deep learning models.

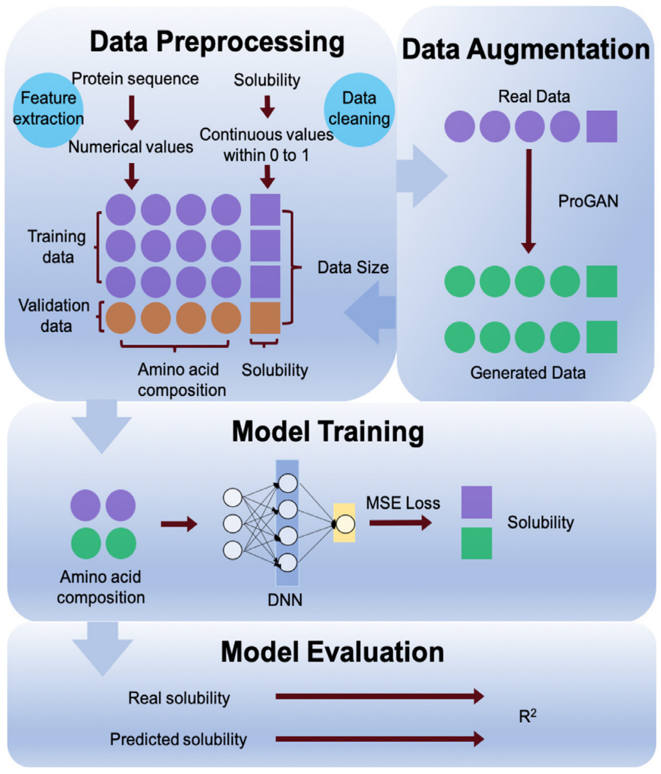

Fig. 1. The workflow of our study.

2. Methods

2.1. Protein database and training workflow

All the protein solubility information used in the study was obtained from eSol database (Niwa et al., 2009). For collecting protein solubility in this dataset, cell-free protein expression technology and plasmids carrying Open Reading Frame (ORF) from the *E. coli* ORF library (ASKA library) were used by Niwa et al. to produce proteins *in vitro* (Kitagawa et al., 2005). Synthesized proteins were then separated into soluble and insoluble fractions by using centrifugation after translation, and the proteins in both fractions were quantified using SDS-PAGE. Finally, solubility was recorded as the percentage of the soluble protein quantity among the total protein quantity. Fig. 1 shows the whole workflow of our work. The data cleaning includes two steps. At first, all the samples from the database were used except the samples containing no sequence information, or poorly determined sequence (containing N instead of A, T, C, or G, or multiple stop-codons). Twenty-five samples were excluded from 3173 samples of the eSol database in our study. Second, protein solubility higher than value 1 was replaced as value 1 since the meaning of protein solubility in our study was a ratio which should be in the range 0–1. And the abnormal values of protein solubility might be due to the error of measurement in experiments. Feature extraction was performed by transforming the characters of protein sequence into numerical values representing the amino acid composition by Amino Acid Composition Descriptor, as it was proved to have the best performance compared with other descriptors according to the previous study (Han et al., 2019). The descriptor can be extracted using protr package (Xiao et al., 2014) within R software, which generates the numerical representation scheme from protein sequences. In the table recording our dataset, each row represents one protein and each column represents the amino acid composition of one type of amino acid, which is the quotient of the number of that type of amino acid and the total number of amino acids of that protein. Therefore, there are 20 columns of features and the 21st column is the target response, protein solubility. After extracting a well-organized dataset including the 20-dim amino acid composition x and protein solubility y where the values of both were within the range 0–1 from the raw data, it was taken as inputs to train DNN model.

The dataset of 3148 proteins was divided into a test dataset (10%) which was selected randomly from the 3148 proteins and a 4-fold dataset (90%) which was divided into four groups with the same size according to the original order in the dataset. Each group was 25% of the original dataset (about 709 proteins) and was selected as validation data once to evaluate the performance of DNN trained by using remaining training data (about 2124 proteins) in the original dataset. The 4-fold dataset enables each part in the dataset to have a chance for training and validation, which is a commonly used method in data mining and makes the selection of data more representative. In Table 3, all the 4 groups were tested 5 times to reduce the variance and the averaged results were used as evaluation metrics. In the process of training DNN, validation data were used to tune the hyperparameters of the model. Then the test data which were held out in the training process were used to test the prediction performance. Subsequently, the training data were taken as inputs of ProGAN to produce doubled dataset including the training data in the original dataset and the generated data in the same size to train DNN again. A new R^2 value was measured on the validation data which were held out in the process of data augmentation and compared with the original R^2. The difference between the two R^2 reflects the contribution of the data augmentation algorithm ProGAN. The frameworks of both DNN and ProGAN are composed by neural networks. DNN was for predicting protein solubility and ProGAN was used for data augmentation to further improve the prediction performance of DNN.

In the eSol dataset, fewer data points of protein solubility in the range of 0.3–0.7 may lead to undesired prediction performance of machine learning models since the characteristics of data with solubility in the range of 0.3–0.7 cannot be fully learned. In the process of balancing the dataset by generating mimic data with solubility in the range of 0.3–0.7, the size of generated data was calculated in the following way. At first, the ratio of proteins with solubility within 0.3–0.7 to proteins with solubility within 0–1 was calculated. If it was distributed evenly in the range 0–1, the ratio of protein within 0.3–0.7 should be 40% rather than 30%. Among the 2361 training data points, 693 proteins were with solubility 0.3–0.7 and the number of proteins should be added to adjust the ratio to 40% was set as n. The proteins in the range of 0.3–0.7 after data augmentation ($693+n$) divided by the total proteins ($2361+n$) should be 40%. The value of n was about 420 after calculation.

2.2. Evaluation metrics

According to the previous study (Han et al., 2019), continuous values of predicted protein solubility from regression models are better for guiding protein engineering since the improvement of protein solubility after mutation can be observed easily and several samples for experimental validation can be selected according to the continuous values of protein solubility. Therefore, a commonly used evaluation metric of the regression model, the coefficient of determination (R^2) was used to quantitatively measure the performance of DNN and ProGAN (Heckmann et al., 2018). R^2 is the ratio of the explained variance (variance of the model's predictions) to the total variance (sample variance of the dependent variable y). R^2 represents how well our regression predictions approximate the real data points. The R^2 of DNN represents the performance of prediction. In addition, the analysis of variance (ANOVA) was implemented to test whether our final regression predictive model was significant. In the ANOVA setting, the observed variance in the response y is partitioned into the variation in mean response and the residual value and it can also be written as SST = SSM + SSE, where SS represents sum of squares and T, M, and E mean total, model, and error, respectively. The mean square value for T, M, and E can be calculated by dividing SST, SSM, SSE by their corresponding degrees of freedom. The test statistic is the ratio MSM/MSE, the mean square model term divided by the mean square error term, which is also called F value. When the MSM is large relative to the MSE, the regression model is significant.

2.3. Machine learning model—DNN

A DNN is a neural network with a certain level of complexity, i.e., a neural network with multiple hidden layers (Bengio et al., 2009; LeCun et al., 2015). The supervised machine learning algorithm DNN was applied to our dataset to predict protein solubility from the amino acid sequence. For continuous values of solubility, the regression algorithm of DNN was used in our study.

The architecture of the DNN we used for the regression of protein solubility is presented in Table 1. It consists of 4 Fully Connected (FC) layers with ReLU activations. The second and third layer contain 512 hidden features. The last layer makes predictions of the continuous value of solubility with a Sigmoid function. Except for the last layer, a Bach Normalization layer follows each FC layer to enable a faster convergence. The loss function is the Mean Square Error (MSE) Loss to regress the groundtruth solubility y. The hyperparameters are tuned through 4-fold cross-validation.

Table 1
Architecture of DNN.

Linear layer	1	2	3	4
Neruons	(20,512)	(512,512)	(512,512)	(512,1)
Activation function	ReLU	ReLU	ReLU	Sigmoid
Loss function	Eq. (7)			

2.4. Data augmentation algorithm—ProGAN

Generative Adversarial Networks are a class of ML algorithms used for data augmentation. GANs consist of two neural networks, Generator and Discriminator, which are trained in opposition to each other. The generator G uses random noise data z from probability distribution $P_z(z)$ as inputs and synthesized data $G(z)$ as outputs. The discriminator D receives generated data from G and real data from training dataset as inputs and produces a probability distribution to distinguish whether the data are artificial or real as outputs. In Eq. (1), the discriminator is trained to maximize the log-likelihood it assigns to the correct data source (termed as $max(V)$), whereas the generator is trained to minimize $max(V)$. This adversarial game between G and D will finish until D cannot distinguish the generated data and real data anymore.

$$\min_G \max_D V(D,G) = E_{x\sim P_{data(x)}}[\log D(x)] + E_{z\sim P_{z(z)}}[\log(1-D(G(z)))] \tag{1}$$

Our goal is to generate data with protein solubility which is very similar to the solubility in our original dataset to augment real data. We propose Protein Solubility Generative Adversarial Nets (ProGAN) which is shown in Fig. 2 to achieve this goal. The input of the generator consists of a noise vector z with 20 dimensions and a latent label $\tilde{y}$. The latent label serves as conditions which were expected to be the protein solubility of the generated data, and follows the uniform distribution P_f from 0 to 1. The generator is trained to produce data $\tilde{x}$ conditioned on the expected protein solubility $\tilde{y}$ to "fool" the discriminator, $G(z,\tilde{y}) \rightarrow \tilde{x} \sim P_g$. In addition to distinguish real and generated data, we introduce an auxiliary regressor (Odena et al., 2016) to predict the protein solubility, D: $x \rightarrow \{D_{adv}, D_{reg}\}$. An adversarial loss is adopted to make the generated data more realistic. To stabilize the training, we use the Wasserstein GAN objective with gradient penalty (Arjovsky et al., 2017; Gulrajani et al., 2017) formulated as

$$\begin{aligned} L_{adv} &= E_{x\sim P_r}[D_{adv}(x)] - E_{\tilde{x}\sim P_g}[D_{adv}(\tilde{x})] \\ &\quad - \lambda E_{\hat{x}\sim P_{\hat{x}}}\left[\left(||\nabla_{\hat{x}} D_{critic}(\hat{x})||_2 - 1\right)^2\right] \end{aligned} \tag{2}$$

where D_{adv} denotes the ProGAN value, $\hat{x}$ is the data sampled uniformly along a straight line between a pair of real data x and generated data $\tilde{x}$, λ is the hyperparameter for gradient penalty and equals 1 in our experiments. It uses the Wasserstein distances instead of the Jensen–Shannon distance as the measurement between the distribution of the mimic data and the real data. The

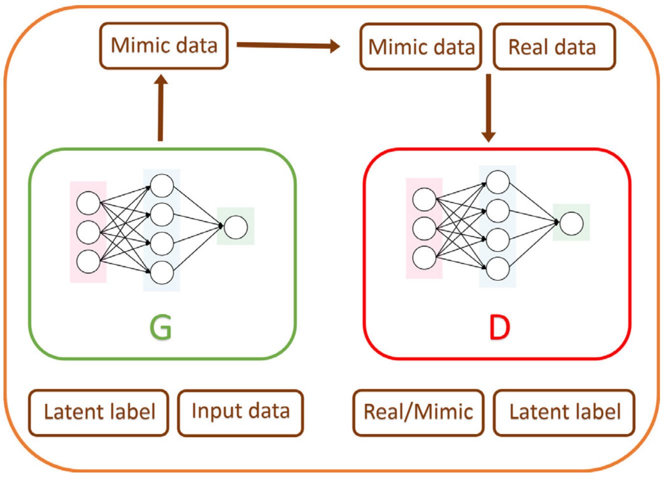

Fig. 2. Diagram of ProGAN.

adversarial loss is maximized while training the discriminator D, but minimized while training the generator G.

To generate data with conditioned protein solubility, we introduce an auxiliary regressor on D. Given a real data x, the discriminator outputs D_{reg} which regresses the corresponding protein solubility y of the real data. It learns the correlation between the protein sequence and the protein solubility. The regression loss for real data is defined as

$$L_{reg}^{r} = E_{x \sim P_r, y \sim P_y} ||D_{reg}(x) - y||_2^2 \tag{3}$$

During the training of generator, the auxiliary regressor provides gradients to constrain the protein solubility of the generated data to be the expected values, which is provided as the condition in the inputs of the generator. Thus, the regression loss for generated data is formulated as

$$L_{reg}^{f} = E_{\tilde{x} \sim P_g, \tilde{y} \sim P_f} ||D_{reg}(\tilde{x}) - \tilde{y}||_2^2 \tag{4}$$

The total objectives to train G and D are

$$L_G = L_{adv} + \alpha L_{reg}^{f} \tag{5}$$

$$L_D = -L_{adv} + \beta L_{reg}^{r} \tag{6}$$

respectively, where α and β balance the importance between the adversarial loss and the regression loss. We use $\alpha = 1$ and $\beta = 1$ in all the experiments. After training the generator and the discriminator iteratively for many epochs, the discriminator cannot distinguish generated data from the generator and real data from the original dataset anymore and the generated data were the final outputs from ProGAN.

After the convergence of ProGAN, the real data are augmented with the generated data to train a DNN regressor R with loss function

$$\min_R E_{x \sim P_r \cup P_g, y \sim P_y \cup P_f} ||R(x) - y||_2^2 \tag{7}$$

The architectures of the generator and discriminator of ProGAN are illustrated in Table 2 respectively. Both generator and discriminator contain four FC layers. Batch Normalization is also applied after each linear layer. Especially, the auxiliary regressor shares the first three layers of features with the discriminator. As shown in Table 2, layer 4(1) and layer 4(2) are two branches in parallel, producing D_{adv} and D_{reg} for the discriminator and the auxiliary regressor respectively.

Table 2
Architecture of ProGAN.

	Linear layer	1	2	3	4	
G	Neruons	(21,512)	(512,512)	(512,512)	(512,20)	
	Activation function	ReLU	ReLU	ReLU	Sigmoid	
	Loss function	Eq. (5)				
	Linear layer	1	2	3	4[a]	
					4(1)	4(2)
D	Neruons	(20,512)	(512,512)	(512,512)	(512,1)	(512,1)
	Activation function	ReLU	ReLU	ReLU	–	Sigmoid
	Loss function	Eq. (6)				

[a] 4(1) and 4(2) are two branches on the top of layer 3 in a parallel fashion. 4(1) together with layer 1–3 performs as the critic to distinguish real data and generated data. 4(2) is introduced as an auxiliary regressor. It learns the correlation between the real sequence data and its corresponding protein solubility.

3. Computational experiments

3.1. Training details

The ProGAN was trained with the Adam optimizer (Kingma and Ba, 2014). The Adam optimization algorithm is an extension to classical stochastic gradient descent and is widely applied on deep learning. It leverages the power of adaptive learning rates methods to find individual learning rates for each parameter and updates network weights iteratively based on training data. The learning rate was 10^{-4} for both discriminator and generator. It was totally trained for 50 epochs. The model converges if the critic values of fake data are close to those of real data, and if the regression losses for both real and fake data are lower than a threshold, which may be different in different trainings. It is approximately 0.06 in our case.

The DNN for protein solubility regressor was trained with stochastic gradient descent (SGD) for 40 epochs. The learning rate was 0.01 and the weight decay was 10^{-4}. A Nesterov momentum (Sutskever et al., 2013) of 0.9 without dampening was also enabled.

3.2. Tuning hyperparameters of DNN and ProGAN

After narrowing down search space firstly, grid search was used to tune the hyperparameters in neural networks including the number of layers, neurons and activation functions. A simpler model with less layers or neurons would not be trained well due to the limited representation power, while a more complex model would lead to the overfitting problem. The architectures we used for DNN and ProGAN were listed in Tables 1 and 2. And the prediction performance achieved by those architectures was recorded in Table 3. It can be seen from Table 3 that ProGAN improved the performance of DNN by doubling the training data in the original dataset through data augmentation. The default size of generated data is the same as the training data. The values of R^2 for fold 1, 2, 3, and 4 indicate the performance for DNN and ProGAN when fold 1, 2, 3, and 4 were taken as the validation data respectively. It was observed that R^2 was improved 1.46%, 2.33%, 1.86%, and 3.47% on average after data augmentation when the fold 1, 2, 3, and 4 were selected as validation data, respectively. The difference of

Table 3
R^2 of DNN with and without ProGAN.

Dataset		Algorithm	R^2
Fold 1		DNN	0.3732 ± 0.0118
		DNN+ProGAN	0.3878 ± 0.0093
Fold 2		DNN	0.3852 ± 0.0064
		DNN+ProGAN	0.4085 ± 0.0055
Fold 3		DNN	0.4035 ± 0.0094
		DNN+ProGAN	0.4221 ± 0.0074
Fold 4		DNN	0.3436 ± 0.0105
		DNN+ProGAN	0.3783 ± 0.0091
Test data		DNN	0.4097 ± 0.0074
		DNN+ProGAN	0.4213 ± 0.0067
	100% data	DNN	0.4258 ± 0.0033
		DNN+ProGAN	0.4383 ± 0.0029
	50% data	DNN+ProGAN	0.4356 ± 0.0060
Balanced dataset	200% data	DNN+ProGAN	0.4356 ± 0.0041
	0.3–0.7[a]	DNN+ProGAN	0.4198 ± 0.0092
	0.3–0.7[b]	DNN+ProGAN	0.4247 ± 0.0055
	Sigmoid	DNN	0.4388 ± 0.0049
	Sigmoid	DNN+ProGAN	**0.4504 ± 0.0018** **MSE(0.0563 ± 0.0007)**

[a] The conditions of protein solubility were 0–1 in the training process, but the generated data in the augmentation process were limited in the range 0.3–0.7.

[b] The conditions of protein solubility were limited in the range 0.3–0.7 in both the data training and augmentation processes.

improvement among different folds in R^2 was mainly due to the uneven data distribution in the original dataset, since the folds were grouped without shuffling. After tuning the hyperparameters by 4-fold cross validation, the test data were used to test the performance of DNN and ProGAN, and R^2 was improved 1.16% after data augmentation.

3.3. Improving model performance by pre-processing the dataset

To further improve the prediction performance of DNN and data augmentation performance of ProGAN, the characteristics of our dataset were investigated. Three major steps, including balancing our dataset by selecting equal data size in different ranges of protein solubility, balancing our dataset by adding data generated from ProGAN in the specific range of protein solubility, and enlarging dataset by adding different data size of generated data were explored one by one in our study.

In the training process of DNN, it was observed that the value of loss function for the training data was low, whereas the value of loss function for the validation data was high. Overfitting occurred and the low value of loss function for training data means the features can be learned by the proposed architecture of DNN. In addition, the performances of different groups were different, which means the original dataset was not distributed evenly and a more balanced dataset was needed to eliminate overfitting and the unbalanced issue. Therefore, we explored the data distribution of protein solubility in our original dataset and plotted it in Fig. 3. It can be found that the values of solubility were not evenly distributed in different ranges between 0–1 and there were fewer points in the range of 0.3–0.7 of solubility. The original dataset was divided into 10 sub-groups for solubility within the range of 0–1, using 0.1 as the step size. Then 75% data of each group were taken out randomly and combined as training data, and the remaining data were validation data.

It can be seen from Table 3, the balanced dataset achieved better results with a R^2 value of 0.4258. The R^2 was further improved by 1.25% after data augmentation with ProGAN. After balancing the dataset in this way, both predictive model and data augmentation algorithm were improved. The reason may be the fact that the models can learn the characteristics of data in various ranges better by training them with the balanced data.

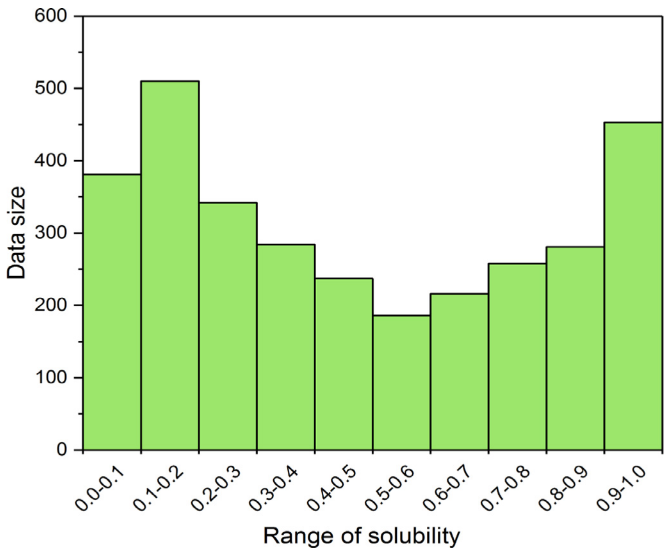

Fig. 3. Data distribution of protein solubility in the original dataset.

With the balanced dataset, we attempted to further improve the model performance by generating more data points within the range of 0.3–0.7 by using ProGAN. Therefore, 420 points with solubility within 0.3–0.7 were generated from ProGAN to balance our training data further. In Table 3, the conditions of protein solubility were limited to the range of 0.3–0.7 where there were fewer samples in the original dataset. However, this method did not improve the prediction performance. Both taking training data in all ranges evenly and generating mimic data in specific regions of original dataset are to balance our dataset, whereas only the former worked. For the second method, insufficient training for samples with solubility within 0.3–0.7 may result in the disability of the ProGAN to generate realistic data, especially only generating mimic data in this region.

For the previous exploration, the generated data from ProGAN had the same size as the training data (100% data). Other data size of generated data was also explored here such as half of training data (50% data) and double data size (200% data) of the training data. It can be observed in Table 3 that there was no obvious improvement for this exploration compared with the R^2 value of 0.4383. The reason may be that the relationship between protein solubility and protein sequence in the original dataset has been learned fully by ProGAN and can be reflected totally in a small amount of generated data. Therefore, there may be a threshold less than 50% and when the size of generated data is less than the threshold, the data augmentation performance would gradually decrease. According to the size of dataset used, the ratio can be reduced to decrease the computing time. Since our dataset only contained 3148 data points in total, the original ratio (100%) was remained in the following work.

3.4. Optimizing activation functions of ProGAN

A sigmoid activation function enables the output of regression for protein solubility from protein sequence within the range of 0–1, but the boundary values 0 and 1 are excluded. Protein solubility in our dataset included points with solubility of 0 or 1. Therefore, the sigmoid activation function was not adopted in the ProGAN. However, a sigmoid function is commonly utilized even though there exists 0 and 1 in the original dataset. Therefore, a sigmoid function was added here and the R^2 was improved to 0.4504 according to Table 3. In addition, the convergence of regression became faster with the sigmoid function. The solubility could infinitely approach 0 or 1, although the exact boundary values were not included. Compared with our previous work (Han et al., 2019), the R^2 was improved from 0.4115 to 0.4504 (a close to 10% improvement) after combining the deep learning predictive model DNN and the data augmentation algorithm ProGAN.

At the same time, the ANOVA test was implemented to evaluate our model in a more comprehensive way. For the final version of our predictive model trained by the original training data and generated data, the test statistic or the F value of our model is 32.2968. It is substantially higher than 1, which shows our regression model is significant and the observed variance can mainly be explained by our model. In addition, the MSE value of 0.0584, which represents the sum of squared distances between the values of solubility in the validation data and predicted values, was calculated with a more accurate degree of freedom shown in Table 4. The MSE of value 0.0563 in Table 4 was an averaged value and the degree of freedom was the number of test data points 784. The small difference between two values of MSE was due to the degree of freedom and can be ignored. Compared with the previous study with the MSE value of 0.0639 (Han et al., 2019), the MSE was decreased by 8.6% in this work, which further showed the improvement of the prediction performance.

Table 4
ANOVA.

Source	Degrees of freedom	Sum of squares	Mean squares	F
Model	19	35.8196	1.8852	32.2968
Error	784-19-1	44.5965	0.0584	
Total	784-1	80.4162		

4. Discussions

4.1. Challenges of implementing data augmentation on data of protein solubility

Generating and collecting a large amount of data by *in vivo* experiments are time-consuming and expensive, which causes a common problem in biotechnology field that the dataset is always too small compared with other fields that use data mining. The machine learning models cannot be trained fully by inadequate volume of data. Therefore, data augmentation algorithms in the biological field are highly desired. However, implementing them in biological data is quite challenging. First, applications of the data augmentation algorithms in biological datasets are more difficult to evaluate compared with applications in image recognition where data visualization is easier. Images plotted by generated data and real data can be observed directly to verify whether they are similar. However, generated and real biological data in tables cannot be differentiated directly by eye. This fact results in a difficult model evaluation process and few applications of data augmentation algorithms on biological data. Second, using data augmentation for protein sequence which contains 20 types of amino acids is more complex than using that for DNA sequence with only four characters (Gupta and Zou, 2018). The data with higher dimensions which use one-hot encoding as inputs may result in much sparser data. Third, it is more complicated than developing an algorithm producing mimic data distribution to implement ProGAN which needs to simulate not only the real data distribution but also the relationship between features and targets (Gupta and Zou, 2018). Compared with the previous exploration about data augmentation algorithm which is undesirable (Han et al., 2019), we improved the architecture by developing a data augmentation algorithm with the latent label for protein solubility in the generator and an auxiliary regressor in the discriminator in this study, and tuning the hyperparameters of our data augmentation algorithm according to the values of loss functions and R^2. Those improvement enables the association between protein solubility and protein sequence of the real data to be maintained in the generated data. In the previous study, the relationship between protein sequence and protein solubility was not considered in the generator and discriminator of GANs, which was not favorable for exploring the relationship between them (Han et al., 2019). In this work, we successfully developed a ProGAN model to improve the deep learning framework DNN for predicting protein solubility from protein sequence by generating artificial data.

4.2. Meaning of improving prediction performance of protein solubility

Compared with the previous work (Han et al., 2019), a more accurate modeling method was achieved by using DNN and ProGAN and R^2 was enhanced about 10%, which may influence the success rate of experimental results in protein engineering. Experimental applications would be based on the prediction results of our model. And in another ongoing project, several proteins with low solubility were mutated to improve their solubility according to optimization algorithms which used the predicted solubility as the dependent variable. More accurate models will definitely improve the applications of predictive models. Therefore, it is a meaningful and promising direction to improve the performance of the predictive model further.

4.3. Applications of deep learning framework for protein solubility prediction

Various machine learning models for predicting protein solubility from protein sequence have been investigated before, whereas the deep learning model DNN has not been used for this research topic. As a part of machine learning, deep learning is increasingly gaining popularity due to its supremacy in many academic fields, especially the computer vision. DNN was utilized to explore the relationship between protein solubility and protein sequence for the first time. The R^2 was improved to 0.4367 compared with the SVM regression model used before (Han et al., 2019). Compared with the traditional machine learning models where features need to be identified by domain experts to recognize patterns and reduce complexity, deep learning models learn high-level features by the hidden architecture, extract important features, and explore the relationship between features. Therefore, our work with predictive model DNN provides a good foundation for predicting protein solubility with deep learning framework.

5. Conclusions and future work

Highly soluble proteins are more effective in biocatalytic processes and can reduce the cost of biocatalysts. However, screening proteins with high solubility by experiments *in vivo* would be costly. First of all, in our study, a data augmentation algorithm, ProGAN enlarged the dataset of protein solubility and improved the performance of the predictive model DNN successfully, which can alleviate the common issue of insufficient data in biotechnology applications for developing machine learning models. Data augmentation opens the door to applications of machine learning models on biological data, as machine learning models always fail to be well trained by small datasets. More importantly, this data augmentation algorithm provides an approach to generate mimic data without hands-on experiments, which brings new insights to the fields where collecting massive data is troublesome. In addition, the prediction performance of protein solubility was improved about 10% compared with other regression models using the same dataset. A more accurate model is beneficial for guiding experimental validations and further optimizations on the protein properties.

In future work, we would like to further improve the performance of DNN and ProGAN. For the first aspect, more features of protein sequence for prediction can be added, such as physiochemical properties, secondary structure information and amino acid sequence. For example, we can input protein sequence as features directly rather than using amino acid composition. The largest length of the protein sequences in our dataset is about 1300 and one-hot encoding can be used to deal with the sequence data in the inputs. In this way, more information can be included in the model training not only amino acid composition. By including the sequence information, optimization of the protein solubility by mutating protein sequences can be more accurate based on the predictive model. In addition, another deep learning model, recurrent neural network which is good at dealing with sequence data, can be explored for using the protein sequence as model inputs directly. However, one difficulty we may come across is that the dimensions of features may be too large compared with the data size, which may cause overfitting in the model training process. There are only 3148 protein sequences in our dataset. In that case, one possible method is to use some dimension reduction algorithms to reduce features. Another one is to use part of the protein sequences, such as some fragments which include obvious structure information of the sequence. This can reduce the length of the protein sequences and make the realization of prediction with one-hot encoding more possible. Finally, when a large amount of features are included, feature selection algorithms can be used to filter features which are more important. On the other hand, this may also be solved by using other methods which can represent the information of the position of each amino acid in a simpler representation scheme rather one-hot encoding.

To further improve the data augmentation algorithms, the hyperparameters of DNN and ProGAN can be tuned. Linear layers in DNN and ProGAN can be replaced by other architectures such as convolutional layers. Convolutional layers can extract the relationships between features and the local spatial coherence in the inputs with relative low computational costs. In addition, DNN can be combined with ProGAN into a whole model to tune hyperparameters and the difference between R^2 without and with data augmentation can be used as the loss function of the whole synthesized model to further enhance the performance of data augmentation.

Acknowledgments

The authors acknowledge the MOE PhD Scholarship, NRF CRP Grant (R-279-000-512-281), MOE Tier 1 Grant (R-279-000-513-133) and NRF CREATE Program “Disruptive & Sustainable Technologies for Agricultural Precision” (R-279-000-531-592) in Singapore.

References

Agostini, F., Vendruscolo, M., Tartaglia, G.G., 2012. Sequence-based prediction of protein solubility. J. Mol. Biol. 421 (2–3), 237–241.
Aliper, A., Plis, S., Artemov, A., Ulloa, A., Mamoshina, P., Zhavoronkov, A., 2016. Deep learning applications for predicting pharmacological properties of drugs and drug repurposing using transcriptomic data. Mol. Pharm. 13 (7), 2524–2530.
Arjovsky, M., Chintala, S., Bottou, L., 2017. Wasserstein gan. arXiv:1701.07875.
Arrowsmith, C., Christendat, D., Yee, A., Dharamsi, A., Kluger, Y., Savchenko, A., Cort, J., Booth, V., Mackereth, C., Saridakis, V., et al., 2000. Structural proteomics of an archaeon. Nat. Struct. Biol. 7, 903–909.
Bengio, Y., et al., 2009. Learning deep architectures for ai. Found. Trends® Mach.Learn. 2 (1), 1–127.
Bishop, C.M., 2006. Pattern Recognition and Machine Learning. Springer.
Chan, W.-C., Liang, P.-H., Shih, Y.-P., Yang, U.-C., Lin, W.-c., Hsu, C.-N., 2010. Learning to predict expression efficacy of vectors in recombinant protein production. BMC Bioinform. 11 (1), S21.
Chang, C.C.H., Song, J., Tey, B.T., Ramanan, R.N., 2013. Bioinformatics approaches for improved recombinant protein production in escherichia coli: protein solubility prediction. Brief. Bioinform. 15 (6), 953–962.
Diaz, A.A., Tomba, E., Lennarson, R., Richard, R., Bagajewicz, M.J., Harrison, R.G., 2010. Prediction of protein solubility in escherichia coli using logistic regression. Biotechnol. Bioeng. 105 (2), 374–383.
Fahmi, I., Cremaschi, S., 2012. Process synthesis of biodiesel production plant using artificial neural networks as the surrogate models. Comput. Chem. Eng. 46, 105–123.
Fang, Y., Fang, J., 2013. Discrimination of soluble and aggregation-prone proteins based on sequence information. Mol. BioSyst. 9 (4), 806–811.
Figueroa, R.L., Zeng-Treitler, Q., Kandula, S., Ngo, L.H., 2012. Predicting sample size required for classification performance. BMC Med. Inform. Decis. Mak. 12 (1), 8.
Gadkar, K.G., Mehra, S., Gomes, J., 2005. On-line adaptation of neural networks for bioprocess control. Comput. Chem. Eng. 29 (5), 1047–1057.
Goh, C.-S., Lan, N., Douglas, S.M., Wu, B., Echols, N., Smith, A., Milburn, D., Montelione, G.T., Zhao, H., Gerstein, M., 2004. Mining the structural genomics pipeline: identification of protein properties that affect high-throughput experimental analysis. J. Mol. Biol. 336 (1), 115–130.
Guimaraes, G. L., Sanchez-Lengeling, B., Outeiral, C., Farias, P. L. C., Aspuru-Guzik, A., 2017. Objective-reinforced generative adversarial networks (organ) for sequence generation models. arXiv:1705.10843.
Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., Courville, A.C., 2017. Improved training of wasserstein gans. In: Advances in Neural Information Processing Systems, pp. 5767–5777.
Gupta, A., Zou, J., 2018. Feedback GAN (FBGAN) for dna: a novel feedback-loop architecture for optimizing protein functions. arXiv:1804.01694.
Habibi, N., Hashim, S.Z.M., Norouzi, A., Samian, M.R., 2014. A review of machine learning methods to predict the solubility of overexpressed recombinant proteins in escherichia coli. BMC Bioinform. 15 (1), 134.
Han, X., Wang, X., Zhou, K., 2019. Develop machine learning based regression predictive models for engineering protein solubility. Bioinformatics.
Heckmann, D., Lloyd, C.J., Mih, N., Ha, Y., Zielinski, D.C., Haiman, Z.B., Desouki, A.A., Lercher, M.J., Palsson, B.O., 2018. Machine learning applied to enzyme turnover numbers reveals protein structural correlates and improves metabolic models. Nat. Commun. 9 (1), 5252.
Hinton, G., Deng, L., Yu, D., Dahl, G.E., Mohamed, A.-r., Jaitly, N., Senior, A., Vanhoucke, V., Nguyen, P., Sainath, T.N., et al., 2012. Deep neural networks for acoustic modeling in speech recognition: the shared views of four research groups. IEEE Signal Process. Mag. 29 (6), 82–97.
Hirose, S., Kawamura, Y., Yokota, K., Kuroita, T., Natsume, T., Komiya, K., Tsutsumi, T., Suwa, Y., Isogai, T., Goshima, N., et al., 2011. Statistical analysis of features associated with protein expression/solubility in an in vivo escherichia coli expression system and a wheat germ cell-free expression system. J. Biochem. 150 (1), 73–81.
Hirose, S., Noguchi, T., 2013. Espresso: a system for estimating protein expression and solubility in protein expression systems. Proteomics 13 (9), 1444–1456.
Idicula-Thomas, S., Balaji, P.V., 2005. Understanding the relationship between the primary structure of proteins and its propensity to be soluble on overexpression in escherichia coli. Protein Sci. 14 (3), 582–592.
Isola, P., Zhu, J.-Y., Zhou, T., Efros, A. A., 2017. Image-to-image translation with conditional adversarial networks. arXiv preprint.
Khurana, S., Rawi, R., Kunji, K., Chuang, G.-Y., Bensmail, H., Mall, R., 2018. Deepsol: a deep learning framework for sequence-based protein solubility prediction. Bioinformatics.
Kingma, D. P., Ba, J., 2014. Adam: a method for stochastic optimization. arXiv:1412.6980.
Kitagawa, M., Ara, T., Arifuzzaman, M., Ioka-Nakamichi, T., Inamoto, E., Toyonaga, H., Mori, H., 2005. Complete set of ORF clones of escherichia coli aska library (a complete s et of e. coli k-12 orf a rchive): unique resources for biological research. DNA Res. 12 (5), 291–299.
Krizhevsky, A., Sutskever, I., Hinton, G.E., 2012. Imagenet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems, pp. 1097–1105.
LeCun, Y., Bengio, Y., Hinton, G., 2015. Deep learning. Nature 521 (7553), 436.
Madhavan, A., Sindhu, R., Binod, P., Sukumaran, R.K., Pandey, A., 2017. Strategies for design of improved biocatalysts for industrial applications. Bioresour. Technol. 245, 1304–1313.
Magnan, C.N., Randall, A., Baldi, P., 2009. Solpro: accurate sequence-based prediction of protein solubility. Bioinformatics 25 (17), 2200–2207.
Mikolov, T., Chen, K., Corrado, G., Dean, J., 2013. Efficient estimation of word representations in vector space. arXiv:1301.3781.
Min, S., Lee, B., Yoon, S., 2017. Deep learning in bioinformatics. Brief. Bioinform. 18 (5), 851–869.
Mirza, M., Osindero, S., 2014. Conditional generative adversarial nets. arXiv:1411.1784.
Niwa, T., Ying, B.-W., Saito, K., Jin, W., Takada, S., Ueda, T., Taguchi, H., 2009. Bimodal protein solubility distribution revealed by an aggregation analysis of the entire ensemble of escherichia coli proteins. Proceedings of the National Academy of Sciences 106 (11), 4201–4206.
Odena, A., Olah, C., Shlens, J., 2016. Conditional image synthesis with auxiliary classifier gans. arXiv:1610.09585.
Pineda, F.J., 1987. Generalization of back-propagation to recurrent neural networks. Phys. Rev. Lett. 59 (19), 2229.
Qi, G.-J., Zhang, L., Hu, H., Edraki, M., Wang, J., Hua, X.-S., 2018. Global versus localized generative adversarial nets. In: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition. IEEE, pp. 1517–1525.
Rawi, R., Mall, R., Kunji, K., Shen, C.-H., Kwong, P.D., Chuang, G.-Y., 2017. Parsnip: sequence-based protein solubility prediction using gradient boosting machine. Bioinformatics 34 (7), 1092–1098.
Rumelhart, D.E., Hinton, G.E., Williams, R.J., 1985. Learning Internal Representations by Error Propagation. Technical Report. California Univ San Diego La Jolla Inst for Cognitive Science.
Russell, S.J., Norvig, P., 2016. Artificial Intelligence: a Modern Approach. Pearson Education Limited, Malaysia.
Sadeghassadi, M., Macnab, C.J., Gopaluni, B., Westwick, D., 2018. Application of neural networks for optimal-setpoint design and MPC control in biological wastewater treatment. Comput. Chem. Eng. 115, 150–160.
Smialowski, P., Doose, G., Torkler, P., Kaufmann, S., Frishman, D., 2012. Proso ii–a new method for protein solubility prediction. FEBS J. 279 (12), 2192–2200.
Smialowski, P., Martin-Galiano, A.J., Mikolajka, A., Girschick, T., Holak, T.A., Frishman, D., 2006. Protein solubility: sequence based prediction and experimental verification. Bioinformatics 23 (19), 2536–2542.
Socher, R., Lin, C.C., Manning, C., Ng, A.Y., 2011. Parsing natural scenes and natural language with recursive neural networks. In: Proceedings of the 28th International Conference on Machine Learning (ICML-11), pp. 129–136.
Sutskever, I., Martens, J., Dahl, G., Hinton, G., 2013. On the importance of initialization and momentum in deep learning. In: International Conference on Machine Learning, pp. 1139–1147.
Wang, M., Si, T., Zhao, H., 2012. Biocatalyst development by directed evolution. Bioresour. Technol. 115, 117–125.
Wilkinson, D.L., Harrison, R.G., 1991. Predicting the solubility of recombinant proteins in escherichia coli. Nat. Biotechnol. 9 (5), 443.
Xiao, N., Xu, Q., Cao, D., 2014. Protr: protein sequence descriptor calculation and similarity computation with r. R package version 0.2-1, URL http://CRAN.R-project.org/package=protr.

Xiaohui, N., Feng, S., Xuehai, H., Jingbo, X., Nana, L., 2014. Predicting the protein solubility by integrating chaos games representation and entropy in information theory. Expert Syst. Appl. 41 (4), 1672–1679.

Xu, Y., Dai, Z., Chen, F., Gao, S., Pei, J., Lai, L., 2015. Deep learning for drug-induced liver injury. J. Chem. Inf. Model. 55 (10), 2085–2093.

Yu, L., Zhang, W., Wang, J., Yu, Y., 2017. Seqgan: Sequence generative adversarial nets with policy gradient.. In: AAAI, pp. 2852–2858.

Zhang, L., Tan, J., Han, D., Zhu, H., 2017. From machine learning to deep learning: progress in machine intelligence for rational drug discovery. Drug Discov. Today 22 (11), 1680–1685.

Zhu, J.-Y., Park, T., Isola, P., Efros, A. A., 2017. Unpaired image-to-image translation using cycle-consistent adversarial networks. arXiv preprint.