Build Virtual Protein Library for Protein Identification by Neural Network

Weizhen Liu
School of Information Science and Technology
ShanghaiTech University
liuwzh@shanghaitech.edu.cn

1. Introduction

Taking advantage of the vast number of peptides retention time and tandem mass spectra data, we report a new deep learning tool termed DeepPhospho. DeepPhospho could learn and predict the chromatographic retention time and the fragment ion intensity of any peptide with extremely high quality. We use the LSTM + Transformer as its architecture; the LSTM could learn a good amino acid representation for the downstream Transformer module. By exploiting self-attention, the Transformer module could capture the difference of amino acids much more precisely. More importantly, the self-attention mechanism could potentially force model to attend the pair of b ion and y ion as these two ions are broke down at the same time in mass spectrometer.

In the supplementary, we show the superiority of LSTM + Transformer compared to solely Transformer or LSTM. Based on the observation that the Transformer module needs the good initial embedding of amino acid, we wonder that if whether or not a convolutional neural network (CNN) could replace the LSTM module to learn a good representation of amino acid, and the results show that the LSTM module is better than the CNN module configured before the Transformer. In further, we want to know a purely deep CNN whether if better than the LSTM + Transformer; however, results show a deep CNN could not generalize well compared with LSTM + Transformer architecture. To the best of our knowledge, DeepPhospho is the first work to utilize the Transformer on the peptide spectrum prediction though it has been prevalently used in the natural language processing domain. There have been several tools that successfully utilized deep learning in RT and Ion intensity prediction, such as DeepRT[4] and pDeep2[8]. DeepRT use the capsule network (CapsNet) [5] model to predict RT. However, the DeepRT did not explain very well for the choice of CapsNet, and CapsNet's structure cannot fully utilize the sequence's characteristics. pDeep2 just use the bi-directional LSTM model to predict the ion intensity, and it does not explicitly take the relations of amino acids into the consideration of model design. We demonstrate the merits of DeepPhospho on a number of challenging examples and provide the scientific community with ready-to-use tools.

2. Data

2.1. RT Dataset

The sequence is represented by the symbol of amino acids such as L, K, M, etc., typically 7-50 in length. Specially, we use 1 to represent the oxidation of methionine (M), and we use 2,3,4 to represent the phosphorylation of serine(S), threonine(T), tyrosine(Y), respectively. And those representation is also used in the Ion Intensity task. Retention time (RT) is a measure of the time taken for a solute to pass through a chromatography column. It is calculated as the time from injection to detection.

The RT datasets are comprised of pairs of $\{X, y\}$. $X := \{< x_1, x_2, x_3, \dots, x_n >\}$, x_i is amino acid, and y is the retention time. For building the virtual library, we split the dataset into train : validation = 9 : 1, selecting the best model on validation set; for model comparison, the dataset is split into train : validation : test = 8 : 1 : 1, reporting results on the test set.

As the retention time is distributed in the real-world unit, such as minutes or seconds, we scale each dataset by its max and min of retention time to 0 - 1 by the following formula. To cover all RT dataset distributions, we set the max(RT) as 200, and min(RT) as -100.

In further, we support the peptide with N-terminal acetyl modification. We use the * symbol to indicate modification, @ to indicate no modification. We pad all sequences to the length of the longest sequence in the dataset to form a matrix feeding into the neural network.

2.2. Ion Intensity Dataset

Like RT dataset, the ion intensity datasets are comprised of pairs of $\{X, y\}$. $X := \{<x_1, x_2, x_3, \ldots, x_i, \ldots, x_n, +q>\}$, x_i is amino acid, $+q$ is the charge carried by the peptide sequence before it is fragmented in the mass spectrometer. And y is the spectrum of the peptide. Each y is composed of pairs of key and value. The key is the ion's name, such as y2+1, b6+2, and the value is their corresponding raw intensity. We divide each intensity by the maximum of the intensities within a peptide sequence to normalize each intensity into 0-1. As kinds of ions in the dataset is severely imbalanced, we only select the 8 types of ions same as pdeep2[8], that is b(y)i+1-noloss, b(y)i+2-noloss, b(y)i+1-1,H3PO4 and b(y)i+2-1,H3PO4, i indicating the site of b(y)ion, to feed the neural network and predict those 8 types of ion intensity. Those ion intensities are formed as the matrix of shape 8 * length of the sequence (illustrated in the supplementary). There are two types of fragment ion intensity values that have no contribution in the loss calculation and are removed in the prediction. One is that related to padding, like y20 for a 7-mer; the other is related to the phosphorylation site. For example, the phosphorylation site is located in the b5, then the b1,b2, b3, and b4 ions cannot lose phosphate so that the ion intensity with phosphate loss must be 0. This ignorance of impossible phosphorylation site potentially help model to learn the implicitly rule of phosphorylation, benefiting the prediction accuracy of intensity of ion with phosphate loss. Otherwise, the only way for model to learn this rule is from the ion intensity data which would be much more inefficient than the injecting the prior knowledge directly to the model learning. The data split, N-terminal acetyl modification indicator and padding operation are the same as the RT dataset.

3. Models

3.1. Model interpretation

It could be formulated as a regression problem from sequence to value: given sequence X, a learned function F could map the X to the y, either retention time or ion intensity. The mathematical expression is as follows:

$$\hat{y} = F(X; \Theta)$$

Θ is the parameters of the model.we find the optimal parameters $\Theta^\star$ by minimizing the loss function L.

$$\Theta^\star = \arg\min_{\Theta} L(\hat{y}, y)$$

We use the LSTM + Transformer model (illustrated in the Figure) to solve the retention time (RT) and ion intensity prediction task. The long short-term memory (LSTM)[2] module comprises of two stacks of bi-directional LSTM. For each stack of LSTM, it has two layers and the dimension of input embedding and hidden state are 256 and 512, respectively. After one stack of LSTM, a combination of LeakyReLU-dropout-linear layer is configured. LSTM is a kind of recurrent neural network(RNN) trying to use gate function to capture the long dependency in the input sequence. The bi-directional LSTM, expecting to learn a good token embedding for the network's downstream layers, is scheduled as the first module of our model.

Transformer[6] is the second module composed of n Transformer encoder. Each Transformer encoder has 8 attention head, then a feedforward layer configured after attention head. The position encoding by sine and cosine functions is added to the output of LSTM module then feed into the Transformer module. We take the same way of position encoding as [6] which is as follows:

$$PE_{(pos,2i)} = \sin\left(pos/10000^{\frac{2i}{d_{model}}}\right)$$
$$PE_{(pos,2i+1)} = \cos\left(pos/10000^{\frac{2i}{d_{model}}}\right)$$

where pos is the position, i is the dimension and d_{model} have the same dimensions with input embeddings. Transformer is an entirely different model by exploiting the self-attention compared to RNN. Self-attention is an attention mechanism relating different positions of a single sequence to compute a representation of the sequence. Self-attention has been used successfully in a variety of tasks, including reading comprehension, textual entailment, and learning task-independent sentence representations. The Transformer is the first transduction model relying entirely on self-attention to compute its input and output representations without using sequence-aligned RNNs or convolution. It has achieved state-of-the-art performance in multiple natural language tasks, such as language translation, language entailment classification, language modeling, etc. We use the pre-layer form of Transformer, which is proposed in [7] and converges much faster.

3.2. RT prediction

For this task, we implement 5 models with two units of two layers bi-directional LSTM and 4, 5, 6, 7, 8 Transformer encoder layers, correspondingly. We train and select those 5 models independently. After obtaining the best models, those 5 predictions of retention time are averaged as the final prediction for each peptide.

The amino acid tokens are embedded into 256 dimensions to the neural network. Especially since the output of the Transformer is the same length as the input sequence, we need to take it down to one scaler for retention time. By adding the time distributed linear layer to assign varied weights for different amino acids dynamically, we obtain RT prediction. Herein we use the square root of mean squared error (RMSE) as loss function. The $\Delta t_{95\%}$ metric is used as the main metric, which represents the minimal time window containing the deviations between observed and predicted RTs for 95% of the peptides.

$$\Delta t_{95\%} = 2 * |y - \hat{y}|_{95\%}$$

The subscript 95% means the 95% rank of the deviations. We select the model by $\Delta t_{95\%}$ metric. Pearson Correlation Coefficient (PCC) is also referred.

3.3. Ion intensity prediction

We implement one model for ion intensity prediction, which has two units of two layers bi-directional LSTM and 8 layers of Transformer encoder. Like the RT prediction task, we first embed the amino acids token, the charge to 192, 64 dimensions separately then concatenate those two vectors, forming the 256 dimension vectors to the neural network. The last layer is a linear layer, which projects the feature from high dimension to 8 dimensions with length unchanged as our prediction of ion intensity.

We use mean squared error (MSE) as our loss function. We compute each peptide's PCC and select the median of those PCCs as the final evaluation metric. Primarily, we follow Prosit[1] using normalized spectral angle(SA) as another metric and the median of those SAs is reported as the final evaluation result. SA's formula is as follows:

$$SA = 1 - 2 * \frac{cos^{-1}(\hat{V_a} \cdot \hat{V_b})}{\pi}$$

$\hat{V}$ is a vector whose L2 norm equals 1. We select the model by the median PCC metric. For both RT and Ion intensity tasks, we use Adam optimizer[3], and the learning rate is 1e-4, the learning rate decay at the milestone epochs during the training. We implement our models by the Python and Pytorch, and train the model on multiple GPUs. The related code could be found in https://github.com/weizhenFrank/DeepPhospho.git.

References

[1] Siegfried Gessulat, Tobias Schmidt, Daniel Paul Zolg, Patroklos Samaras, Karsten Schnatbaum, Johannes Zerweck, Tobias Knaute, Julia Rechenberger, Bernard Delanghe, Andreas Huhmer, et al. Prosit: proteome-wide prediction of peptide tandem mass spectra by deep learning. Nature methods, 16(6):509–518, 2019. 3

[2] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. Neural computation, 9(8):1735–1780, 1997. 2

[3] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017. 3

[4] Chunwei Ma, Yan Ren, Jiarui Yang, Zhe Ren, Huanming Yang, and Siqi Liu. Improved peptide retention time prediction in liquid chromatography through deep learning. Analytical chemistry, 90(18):10881–10888, 2018. 1

[5] Sara Sabour, Nicholas Frosst, and Geoffrey E Hinton. Dynamic routing between capsules. In Advances in neural information processing systems, pages 3856–3866, 2017. 1

[6] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In Advances in neural information processing systems, pages 5998–6008, 2017. 2

[7] Ruibin Xiong, Yunchang Yang, Di He, Kai Zheng, Shuxin Zheng, Chen Xing, Huishuai Zhang, Yanyan Lan, Liwei Wang, and Tie-Yan Liu. On layer normalization in the transformer architecture, 2020. 2

[8] Wen-Feng Zeng, Xie-Xuan Zhou, Wen-Jing Zhou, Hao Chi, Jianfeng Zhan, and Si-Min He. Ms/ms spectrum prediction for modified peptides using pdeep2 trained by transfer learning. Analytical chemistry, 91(15):9724–9731, 2019. 1, 2