DeepPhospho: A Transformer-based Deep Network for Peptide Tandem Mass Spectra Prediction

Weizhen Liu[1]*, Rongjie Li[1]*, Ronghui Lou[2,3,5], Wenqing Shui[2,3]†, Xuming He[1,4]†
[1]School of Information Science and Technology, ShanghaiTech University
[2]School of Life Science and Technology, ShanghaiTech University
[3]iHuman Institute, ShanghaiTech University
[4]Shanghai Engineering Research Center of Intelligent Vision and Imaging
[5]University of Chinese Academy of Sciences
{liuwzh, lirj2, lourh, shuiwq, hexm}@shanghaitech.edu.cn

1. Prepare Data

1.1. Data composition and preprocess

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. In further, we support the peptide with N-terminal acetyl modification. We use the * symbol to indicate modification, @ to indicate no modification.

For RT datasets, they are comprised of $\{X, y\}$ pair. $X := \{< x_0, x_1, x_2, x_3, \ldots, x_i, \ldots, x_n >\}$. x_0 is the symbol of * or @. x_i $(i >= 1)$ is amino acid. n is the length of peptide. y is the retention time. As the retention time is distributed in the real-world unit, such as minutes or seconds, we scale each dataset by its maximal and minimal of retention time to 0 - 1 by the following formula.

$$RT_{normalized} = \frac{RT - min(RT)}{max(RT) - min(RT)}$$

Ion intensity datasets are also comprised of $\{X, y\}$ pair. $X := \{< x_0, x_1, x_2, x_3, \ldots, x_i, \ldots, x_n, +q >\}$. $+q$ is the charge carried by the peptide sequence before it is fragmented in the mass spectrometer. 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, 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 train and predict. The shape of ion intensity input is illustrated in the supplementary.

1.2. RT experiments

We sequentially train the model on the four pre-training datasets, HumanPhosDB [7], Jeff [9], VeroE6 [2] and R2P2 [8], and obtain the best model by the validation sets. Then we fine-tune the model on the three downstream datasets, U2OS-DIA [12], RPE1-DIA [1] and RPE1-DDA [1], respectively. The downstream datasets are split into training : validation : test = 8 : 1 : 1, and we select model on the validation set, reporting the metric on the test set. Correspondingly, we download the DeepRT model provided by the [10], and fine-tune the model on those downstream datasets. The detailed result is shown in Table 2. We could see that ours is the better in all three datasets, which means our method's performance is very stable though these three datasets are quite dissimilar. We improve -4.981%, -7.416% and -6.979% in $\Delta t_{95\%}$, respectively for the three downstream dataset compared to DeepRT.

1.3. Ion Intensity experiments

Similarly to the RT task, we obtain the best pre-trained model from three pre-training datasets, Jeff [9], VeroE6 [2] and R2P2 [8], and fine-tune the model on the downstream three datasets, U2OS-DIA [12], RPE1-DIA [1] and RPE1-DDA [1] with a same split ratio of training, validation, and test. Also, we download the pDeep2 model from [14] and fine-tune the pDeep2 on those three downstream datasets. The detailed result is shown in Table 3. From the table, we could conclude that ours is the better in all three datasets. Although the ion intensity task is relatively more complicated than the RT task, we have significantly boosted ours' performance. We improve 3.060%, 3.987%, and 1.446% in

*Both authors contributed equally to the work.
†Both are conrresponding authors.

data description	no. of peptides	no. of spectra
HumanPhosDB [7]	204,558	-
Jeff [9]	67,552	89,437
VeroE6 [2]	43,405	54,004
R2P2 [8]	35,808	43,312
U2OS-DIA [12]	48,327	58,843
RPE1-DIA [1]	33,576	39,977
RPE1-DDA [1]	129,109	165,719

Table 1. Retention time datasets

method	U2OS-DIA	RPE1-DIA	RPE1-DDA
DeepRT	11.563/0.996	14.752/0.995	16.421/0.983
DeepPhospho	**10.987/0.997**	**13.658/0.996**	**15.275/0.986**

Table 2. RT Dataset results. The left number in cell is $\Delta t_{95\%}$ where the lower is the better, and the right is PCC where the higher is the better.

method	U2OS-DIA	RPE1-DIA	RPE1-DDA
pDeep2	0.887/0.778	0.867/0.767	0.954/0.855
DeepPhospho	**0.915/0.804**	**0.903/0.791**	**0.968/0.881**

Table 3. Ion Intensity Dataset results. The left number in cell is median PCC and the right is median SA where the higher is the better for both metrics.

median PCC, and 3.234%, 3.034%, and 2.951% in median SA, respectively, on the three downstream datasets compared to the pDeep2.

1.4. Ablation study

To illustrate our model design's efficacy, we compare our model architecture with removed LSTM module and removed transformer module. In further, we compare our model with the replacement of LSTM module with convolutional neural network (CNN) module.

The CNN module is built like the ResNet34 [4] except the kernel size of the first convolution layer changed to be 9, and the kernel size in residual block changed to be 7, and it is composed of 3 residual blocks. The comparison experiments are trained on ion intensity data of Jeff [9] dataset. We split the Jeff dataset into training : validation = 9 : 1, reporting the best validation result for each model. The result is seen in Table 4. We could see that once removed any module or replace with CNN module, the performance would decrease.

model	Median PCC	Median SA
w/o LSTM	0.949	0.834
w/o Transformer	0.951	0.835
CNN-Transformer	0.949	0.839
DeepPhospho	**0.955**	**0.844**

Table 4. Ablation study on Jeff dataset

2. Model Architecture

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 to solve the retention time (RT) and ion intensity prediction task. The long short-term memory (LSTM) [5] 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 [11] is the second module composed of n Transformer encoder. Each Transformer encoder has 8 attention head, then a feedforward layer configured after attention head.

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 [13] and converges much faster.

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 [11] 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.

2.1. RT prediction model

For this task, we use model ensemble method and 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.

2.2. Ion intensity prediction model

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.

2.3. Metric

For the RT task, the $\Delta t_{95\%}$ metric is used as the primary metric, representing 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. Pearson Correlation Coefficient (PCC) is also referred to, but we select the model by $\Delta t_{95\%}$ metric as the PCC metric could not reflect the difference between different methods as could be seen in the following experimental results.

For the Ion Intensity task, We compute each peptide's PCC and select the median of those PCCs as the final evaluation metric. Primarily, we follow Prosit [3] using normalized spectral angle(SA) as another metric, and the median of those SAs is reported. 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.

3. Pre-training

To train a better model, we sequentially pre-train the models in four datasets called HumanPhosDB [7], Jeff [9], VeroE6 [2] and R2P2 [8]. We split those pre-training datasets into training and validation set, selecting the best model on the validation set. The model is initialized by the selected model before training on the next pre-training dataset until those four datasets are all trained on. There are three downstream datasets called U2OS-DIA [12], RPE1-DIA [1] and RPE1-DDA [1]. For the three downstream datasets, we manually set the $min(RT)$ and $max(RT)$ equals -100 and 200, respectively. -100 and 200 cold cover all the RTs in the three datasets, and the following researcher could directly use our well-trained model and the fixed $min(RT)$ and $max(RT)$ to predict the unknown RTs of their interested peptides. Herein we use the square root of mean squared error (RMSE) as loss function.

Similarly to the RT task, we also use those ion intensity in three of four pre-training dataset Jeff [9], VeroE6 [2] and R2P2 [8], and fine-tune the pre-trained model on the downstream dataset U2OS-DIA [12], RPE1-DIA [1] and RPE1-DDA [1]. The summary of the datasets used in this work is shown in Table 1. We use mean squared error (MSE) as our loss function. For both RT and Ion intensity tasks, we use Adam optimizer [6], 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.

4. Transfer learning

To train a better model, we sequentially pre-train the models in four datasets called HumanPhosDB [7], Jeff [9], VeroE6 [2] and R2P2 [8]. We split those pre-training datasets into training and validation set, selecting the best model on the validation set. The model is initialized by the selected model before training on the next pre-training dataset until those four datasets are all trained on. There are three downstream datasets called U2OS-DIA [12], RPE1-DIA [1] and RPE1-DDA [1]. For the three downstream datasets, we manually set the $min(RT)$ and $max(RT)$ equals -100 and 200, respectively. -100 and 200 cold cover all the RTs in the three datasets, and the following researcher could directly use our well-trained model and the fixed $min(RT)$ and $max(RT)$ to predict the unknown RTs

of their interested peptides. Herein we use the square root of mean squared error (RMSE) as loss function.

Similarly to the RT task, we also use those ion intensity in three of four pre-training dataset Jeff [9], VeroE6 [2] and R2P2 [8], and fine-tune the pre-trained model on the downstream dataset U2OS-DIA [12], RPE1-DIA [1] and RPE1-DDA [1]. The summary of the datasets used in this work is shown in Table 1. We use mean squared error (MSE) as our loss function. For both RT and Ion intensity tasks, we use Adam optimizer [6], 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.

References

[1] Dorte B Bekker-Jensen, Oliver M Bernhardt, Alexander Hogrebe, Ana Martinez-Val, Lynn Verbeke, Tejas Gandhi, Christian D Kelstrup, Lukas Reiter, and Jesper V Olsen. Rapid and site-specific deep phosphoproteome profiling by data-independent acquisition without the need for spectral libraries. *Nature communications*, 11(1):1–12, 2020. 1, 2, 3, 4

[2] Mehdi Bouhaddou, Danish Memon, Bjoern Meyer, Kris M White, Veronica V Rezelj, Miguel Correa Marrero, Benjamin J Polacco, James E Melnyk, Svenja Ulferts, Robyn M Kaake, et al. The global phosphorylation landscape of sars-cov-2 infection. *Cell*, 182(3):685–712, 2020. 1, 2, 3, 4

[3] 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

[4] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015. 2

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

[6] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017. 3, 4

[7] Robert T Lawrence, Brian C Searle, Ariadna Llovet, and Judit Villén. Plug-and-play analysis of the human phosphoproteome by targeted high-resolution mass spectrometry. *Nature methods*, 13(5):431–434, 2016. 1, 2, 3

[8] Mario Leutert, Ricard A Rodríguez-Mias, Noelle K Fukuda, and Judit Villén. R2-p2 rapid-robotic phosphoproteomics enables multidimensional cell signaling studies. *Molecular systems biology*, 15(12):e9021, 2019. 1, 2, 3, 4

[9] Jeffrey J Liu, Kirti Sharma, Luca Zangrandi, Chongguang Chen, Sean J Humphrey, Yi-Ting Chiu, Mariana Spetea, Lee-Yuan Liu-Chen, Christoph Schwarzer, and Matthias Mann. In vivo brain gpcr signaling elucidated by phosphoproteomics. *Science*, 360(6395), 2018. 1, 2, 3, 4

[10] 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

[11] 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, 3

[12] Shisheng Wang, Wenxue Li, Liqiang Hu, Jingqiu Cheng, Hao Yang, and Yansheng Liu. Naguider: performing and prioritizing missing value imputations for consistent bottom-up proteomic analyses. *Nucleic acids research*, 48(14):e83–e83, 2020. 1, 2, 3, 4

[13] 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

[14] 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