Prosit Transformer: A transformer for prediction of MS2 spectrum intensities

Markus Ekvall[1] Patrick Truong[1] Wassim Gabriel[2] Mathias Wilhelm[2]
Lukas Käll[1],*

[1]Science for Life Laboratory, School of Engineering Sciences in Chemistry, Biotechnology and Health, Royal Institute of Technology – KTH, Box 1031, SE-17121 Solna, Sweden
[2]Computational Mass Spectrometry, Technical University of Munich (TUM), D-85354 Freising, Germany
* Corresponding author, lukas.kall@scilifelab.se

Abstract

Machine learning has been an integral part of interpreting data from mass spectrometry-based proteomics for a long time. Relatively recently, a machine-learning structure appeared successful in other areas of bioinformatics, Transformers. Furthermore, the implementation of Transformers within bioinformatics has become relatively convenient due to transfer learning, i.e., adapting a network trained for other tasks to new functionality. Transfer learning makes these relatively large networks more accessible since it generally requires less data, and the training time improves substantially. We implemented a Transformer based on the pre-trained model TAPE to predict MS2 intensities. TAPE is a general model trained to predict missing residues from protein sequences. Despite being trained for a different task, we could modify its behavior by adding a prediction head at the end of the TAPE model and fine-tune it using the spectrum intensity from the training set to the well-known predictor Prosit. We demonstrate that the predictor, which we call Prosit Transformer, outperforms the recurrent neural network-based predictor Prosit, increasing the median angular similarity on its hold-out set from 0.908 to 0.929. We believe that transformers will significantly increase prediction accuracy for other types of predictions within mass spectrometry-based proteomics.

Keywords: Machine Learning; Proteomics; MS2 spectra; Transformers

Introduction

Just as in many other areas involving analysis of large and complex datasets, different types of machine learning are tremendously helpful for the modern analysis of mass spectrometry-based proteomics data.[1,2] For example, we nowadays can use machine learning to predict tryptic digestion,[3] chromatographic retention time,[4–6] collisional cross-section,[7] the accuracy of peptide-spectrum matches,[8] and the accuracy of transitions in DIA data[9] are tasks that utilize machine learning.

One task that has gained traction in the last couple of years is predicting MS2 spectra from peptide sequences.[10,11] Such predictors can predict relative intensities of a given peptide sequence's b- and y-ions. Together with the m/z values of the ions, which one can derive from first principle, one can subsequently form a full MS2 spectrum. MS2 spectrum prediction has in a short time established itself as a means to rescore peptide spectrum matches,[12] increasing the sensitivity in large search spaces,[13] and target-decoy strategies for DIA interpretation.[14]

Many types of frameworks are available for training a predictor, such as Support Vector Machines and Recurrent Neural Networks (RNNs) used within mass spectrometry-based proteomics. However, in the last couple of years, a structure first in natural language processing[15] known as Transformers[16] has successfully been employed within bioinformatics, e.g., structure prediction,[17,18] gene expression prediction,[19] and even within MS-based proteomics, e.g. peptide detection problem,[20] DIA library generation for the phosphoproteome[21] and *de novo* interpretation of MS2 spectra.[22]

Transformers are, like RNNs, designed to handle sequential input data and do so through attention mechanisms, i.e., mechanisms that enhance the essential parts of the input sequence for its output.

However, unlike RNNs, the Transformers do not use recurrence, thus enabling a significant speed-up by parallelizing their training. The encoder-decoder structure is the basis of the Transformers, where both the encoder and decoder adopt the multi-headed attention mechanism.[16]

Notably, the Tasks Assessing Protein Embeddings (TAPE) model[17] is exciting; a Transformer-based autoencoder of protein sequences is formed by withholding one amino acid at a time in a large set of protein sequences and subsequently predicting which is the missing amino acid. One can subsequently employ the model for higher-level tasks by plugging them into some extra layers of neurons in a process known as transfer learning.[17,18]

Here, we argue that Transformers can greatly aid mass spectrometry-based proteomics. We demonstrate that TAPE's BERT sub-model, can predict MS2 spectrum intensities from peptide sequences. We are using the training and test sets of the popular Prosit[11] predictor and demonstrate that the transformer-based predictor, which we named Prosit Transformer, drastically outperforms the old implementation of Prosit.

Methods

Data

We downloaded the Prosit training data from `https://figshare.com/projects/Prosit/35582`. This set is composed of spectra from PXD004732, PXD010595, and PXD021013.[11,13] The Prosit data had to be converted from HDF5 to LMDB to be compatible with the Tape framework. The LMDB data files used during training and validation are accessible at `https://figshare.com/articles/dataset/LMDB_data_Tape_Input_Files/16688905`.

Architecture

The TAPE model consists of twelve 768 hidden units attention layers, with the attention dropout (DropHead) rate[23] and regular dropout rate set to 0.1. We downloaded weights for the pre-trained model that has been trained on the raw protein sequences in the protein families database (Pfam) to predict the amino acid at each protein position given the previous amino acids and the following amino acids.[17] The Prosit-specific transformer has the same parameter but consists of 9 attention layers. The meta-data layer is a multilayer perceptron (MLP) with two layers of size 512 units followed by a dropout rate of 0.1 each. The final prediction layer has the same structure, except for no dropout after the final layer. The activation function is ReLU, except for the prediction layer where the first layer uses a ReLU6,[24] i.e., a max(0,min(6,x)) function as an activation function, and the final layer uses a linear layer.

Metrics

We measure angular distance, $d_{AB} = \frac{2}{\pi}\cos^{-1}\left(\frac{A\cdot B}{(||A||\cdot||B||)}\right)$, and angular similarity, $s_{AB} = 1d_{AB}$ as measures of accuracy of the predicted intensities. Here, A is the vector of predicted intensities, and B is the vector of observed intensities for the ion series included in the prediction. However, we introduced a few extra steps during training to avoid undefined behavior. Firstly, to avoid undefined values using angular similarity during training, we had to clip the inputs to $\cos^{-1}$ with $-(1-\epsilon)$ and $(1-\epsilon)$ to avoid undefined values. This implementation was necessary since some predictions were too similar to their target after training, resulting in an undefined loss. However, there was no clipping during the evaluation, so it will not affect the final result. Lastly, we also had to introduce a small ϵ in the denominator in the cosine similarity, i.e., $\max(||A||\cdot||B||, \epsilon)$, to assure no undefined behavior during training. The sum of all d_{AB} for all peptides in the test set was used as a loss function for the training of the networks.

We calculated the $FDR = FP/(FP + TP)$ and $FNR = FN/(FN + TP)$ for each predicted spectrum to measure the number of erroneous peak predictions. Here FP is the number of peaks predicted in excess to be present in a spectrum that was absent in the observed spectrum, FN is the number of peaks deficently predicted to be absent in a spectrum that was present in the observed spectrum, and TP is the number of peaks accurately predicted to be present in a spectrum that was present in the observed spectrum.

Post-processing of Predicted intensities

We use the same post-processing on the predicted spectrum used in Prosit [11] for the final result. To clarify, we set ions with a predicted negative intensity to zero, i.e., a negative intensity indicates an absent peak. Furthermore, we set all ion's intensity that's not obtainable for any given peptide due to too low a charge state or too low peptide length to -1. However, we exclude such peaks for similarity measurements.

Hardware

The model was trained on the Berzelius SuperPOD, a GPU cluster consisting of 60 NVIDIA DGX A100 systems, linked on a 200 Gbit/second NVIDIA Mellanox InfiniBand HDR network.

Results

We set out to test whether Transformers are a technology fit for spectrum intensity predictions, i.e., to predict the intensities of the most commonly observed ion-series ($b^+, b^{2+}, b^{3+}, y^+, y^{2+}, and\ y^{3+}$) of product-ion spectra from peptide fragmentation. The length of the peptides ranged between 7 to 30 amino acids long. We used the train/test data and the preprocessing coming with the Prosit predictor as a testbed. Prosit's scripts calculating the intensity vectors, adopting meta-data, and calculating predictions' angular similarity has been found robust after years of usage. We also found it straightforward to set up a benchmark, as we could reuse the Prosit test sets just out of the box. We will refer to the traditional Prosit predictor as Prosit RNN from hereon to avoid confusion.

Model

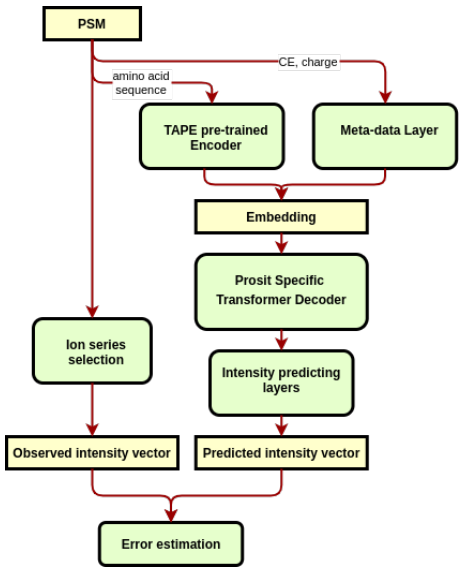

Figure 1: **The architecture of the Prosit Transformer.** The model depends on a pre-trained encoder from the TAPE project and uses the TAPE design for a Prosit Specific decoder. However, our model implements many of the design features of Prosit RNN, i.e., layers handling meta-data and final intensity prediction.

We set out to use the setup previously used for training and testing the Prosit model but with a transformer. We used the pre-trained TAPE model[17] and retrofitted it with a Prosit-specific decoder and some additional application-specific code (See Figure 1). The TAPE model will encode the peptide into a 512-dimensional embedding. Furthermore, just as for the original RNN-based Prosit model, we used layers for handling meta-data consisting of the charge state of the spectrum and its collision energy (CE). The charge states range from one to six, represented as 6-dimensional

one-hot-encoding. Hence, the meta-data layer has seven input nodes to account for the charge state and CE. The meta-layer transforms the metadata into a 512-dimensional vector that is subsequently is combined with the encoded peptide by element-wise multiplication. Then a Prosit-specific Transformer will decode this combined embedding. Lastly, a two-layered Multilayer perception (MLP) follows the decoding layer, serving as a prediction layer to predict the spectrum intensity. The MLP used activation by a hinge loss function constrained between 0 and 6 (a *RELU6* function) to activate the two final layers to avoid a so-called gradient explosion. For the training, the objective function was to minimize the sum of the angular distances between the observed and predicted spectrum intensity vectors.

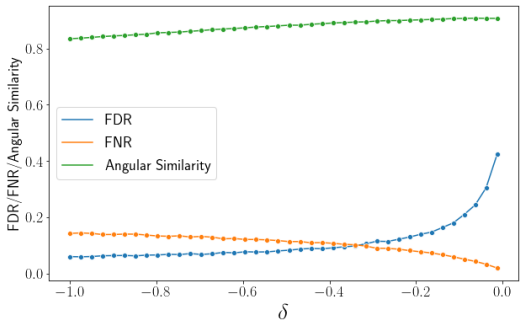

Figure 2: **The effect of adjusting the hyperparameter δ on predicting the absence/presence of individual MS2 peaks.** To obtain better prediction accuracy of present and absent MS2 peaks, we adjusted the intensities of absent peaks from zero to δ. We measured the false discovery rate (*FDR*) and the false-negative rate (*FNR*) of each spectrum and then plotted the average angular similarity, the *FDR*, and the *FNR* for different choices of δ. We selected $\delta = 0.34$ for the final training. The predicted spectra were not post-processed for the measurements in this figure (See Methods).

Training of model

During the training, we used a batch size of 1024, the learning rate of 0.0001, gradient accumulation step of one, and a linear learning rate schedular with 10000 warmup steps. The training proceeded until no further improvement over ten epochs.

To better predict present and absent peaks, we introduced a hyperparameter, δ, setting an artificial offset of the intensities of absent peaks to $\delta_p = \delta/|\text{number of considered peaks}|$. This hyperparameter adds an extra penalty if the model predicts intensities for absent peaks. By varying the size of δ, we can control the model's propensity to predict peaks as absent, and by such means, tune the model's false positive and false negative predictions. We measured the false discovery rate and the false-negative rate of each spectrum and then plotted the average angular similarity, the *FDR*, and the *FNR* for different choices of δ. We selected $\delta = 0.34$ for the final training. (See Figure 2)

Comparison of performance to regular Prosit

To test the performance of our final Prosit Transformer, we investigated its performance on the same held-out test set as used when initially training Prosit RNN. We calculated the so-called angular similarity between the predicted and observed intensities for both predictors. Overall, we see that the predictions from Prosit Transformer have a higher angular similarity than Prosit RNN and are hence more accurate (Figure 3A). The Prosit Transformer increased the median angular similarity from Prosit RNN's 0.908 to 0.929. We also see that Prosit Transformer obtained a higher angular similarity than Prosit RNN in 75.7% of the spectra, while the opposite was true in 24.3% of the spectra. The same pattern was also true when dividing the PSMs based on their peptide's lengths (Figure 3B). We also wanted to compare the predictors' ability to predict present and absent (zero intensity) fragment peaks. Our choice of hyperparameter delta for Prosit Transformer resulted in a lower fraction of observed absent peaks among the predicted non-zero intensity peaks (Figure 3C)

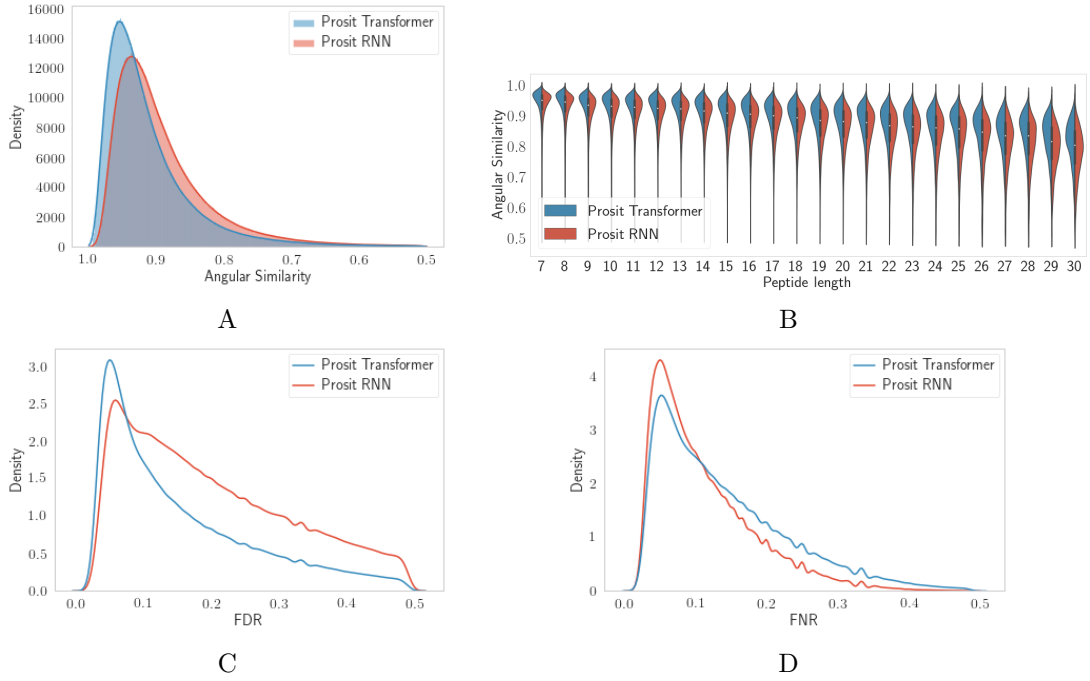

Figure 3: **Comparison of the accuracy of Prosit Transformer and Prosit RNN.** (A) We made separate histograms and smoothed them with a kernel density estimator to observe the distribution of angular similarity for the spectra predicted with Prosit Transformer and Prosit RNN. (B) The same angular similarity was also stratified by the length of peptides. (C) We also measured the false discovery rate, i.e., the fraction of observed absent peaks among the predicted non-zero intensity peaks for each spectrum, and (D) the false-negative rate, i.e., the fraction of predicted absent peaks among the observed non-zero intensity peaks.

while observing a higher fraction of predicted absent peaks among the observed non-zero intensity peaks (Figure 3D) for Prosit Transformer compared to Prosit RNN.

Comparison of a Transformer to an extended RNN for prediction of spectra

We set out to eliminate other explanations for Prosit Transformer's elevated performance than the Transformers themselves. A notable difference between Prosit RNN and Prosit Transformer is their difference in size. Prosit RNN contains 3 million parameters, while Prosit Transformer contains 164 million parameters, which gives the Transformer an unfair advantage. Hence, we stacked long short-term memory layers to create RNN models of similar size to the ones of the Transformers. This extended RNN gave a median angular similarity of 0.892 compared to Prosit Transformer's 0.929. Further, Prosit Transformer also outperformed the extended RNNs encoder in combination with Prosit Transformer's decoder (median angular similarity of 0.927), as well as Prosit Transformer's encoder in combination with Prosit RNN's decoder (median angular similarity of 0.915). See Table 1 for an overview of the permutations of encoder decode architectures and their sizes.

When training the RNN-models, the learning rate had to be decreased from 0.0001 to 0.00008 to get the model to learn. Everything else was the same as for the Transformer-Transformer model. We also had to switch the gated recurrent unit (GRU) of the Prosit RNN to an LSTM to use the TAPE framework, leading to minor differences between the Extended Prosit RNN and Prosit RNN.

Surprisingly, the extended RNN-RNN model got worse results than regular Prosit. The decrease could be due to that increase from 3M to 178M parameters leading to overfitting, requiring more data to justify such a massive model for the type of architecture. However, a performance increase was observed in all cases when adding a Transformer to the architecture. The most significant increase in performance appeared when implementing the Transformer as a decoder, i.e., after the peptide has been encoded and combined with the meta-data, and not in the peptide's encoding, although this improves the results as well.

At first the conclusion that the Transformer-Transformer model performed best might seem to contradict the results of others. Particularly, DeepPhospho,[21] reports a better performance for their LSTM-Transformer model that their Transformer-Transformer model. However, it is worth noting that the circumstance were different, their LSTM decoder was larger than their Transformer decoder (34M vs 6M parameters).[21] One would expect that the Transformer's performance would increase with a larger model, while the LSTM wouldn't benefit as much (perhaps even getting worse) with a larger model.

Table 1: **Extended RNN models' size and performance.** We trained and tested different permutations of expanded RNNs and Transformers of comparable size and compared their prediction accuracy.

Architecture for Encoder-Decoder	Encoder size	Encoder layers	Encoder units	Decoder size	Decoder layers	Decoder units	Total size	Median Angular similarity
Transformer-Transformer	85M	12	768	64M	9	768	164M	0.929
RNN-RNN	77M	5	1028	93M	5	2056	178M	0.892
Transformer-RNN	64M	9	768	94M	10	768	172M	0.9156
RNN-Transformer	53M	6	768	113M	6	768	173M	0.927

Time comparision of sprectrum prediciton

The models in table 1 using a Transformer as an encoder required ~ 3 GPU days, while the models using an RNN encoder needed ~ 6 GPU days.

Both regular Prosit and Prosit Transformer were timed for predicting 1000, 10000, and 100000 spectra to see how much the increased model complexity for Prosit Transformer affects prediction time; see table 2 below. Prosit Transformer requires roughly 40 times more time, so there is a clear trade-off for prediction accuracy and time requirements.

Table 2: **The larger Transformer model needs more time to predict spectra.** We measured the required time to predict spectra from peptides for both Prosit RNN and Prosit Transformer.

Number of predicted spectra	1k	10k	100k
Prosit RNN	0.05 s	0.5 s	4.7 s
Prosit Transformer	2 s	18 s	180 s

Prosit Transformer's ability to model collision energy

We also wanted to test that the improved ability of Prosit Transformer to predict MS2 intensities did not affect the predictor's ability to model collision energy's (CE's) influence on predicted spectra. We hence isolated batches of spectra with $CE = \{0.2, 0.25, 0.3, 0.35, 0.4\}$ and measured the median Angular Similarity when predicting the spectra for a range of different collision energies (Figure 4). The highest angular similarity was found between the observed and predicted spectra when setting CE to the set's actual specified value.

Discussion

Here we have used a Transformer trained to predict protein sequence and transferred its functionality into predicting intensities of the b- and y- ions of MS2 spectra. The resulting predictor's performance outperformed a predictor built by a classical recurrent neural network. This type of structure can likely improve other types of peptide property prediction.

One interesting finding was that the most significant improvement was when using Transformers as a decoder when comparing different combinations of RNNs and Transformers as decoders and encoders. A possible interpretation of this result is that Transformer architecture better utilizes the metadata, i.e., the collision energy and charge state information. A future direction of the project

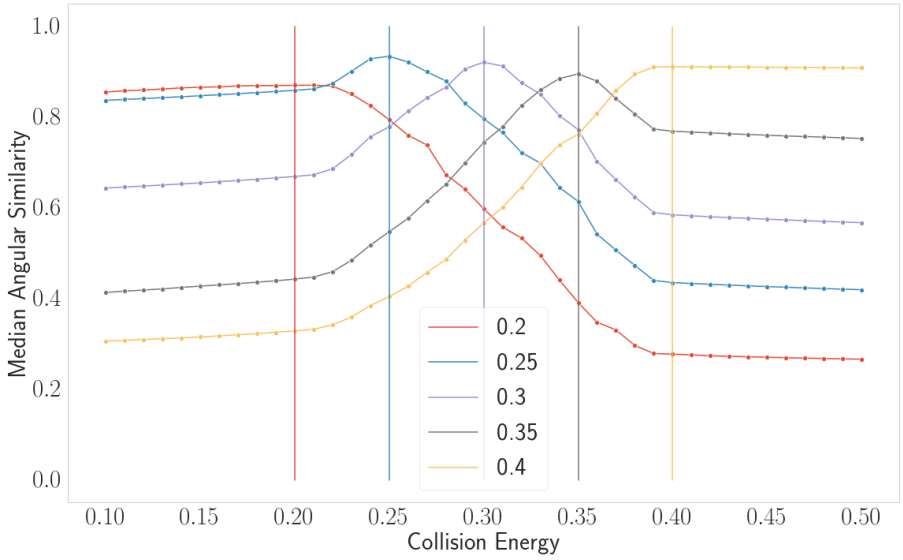

Figure 4: **The mean spectral angle as a function of the collision energy (CE) for spectra acquired with different CEs.**

could be to investigate the source of the improved accuracy by examining the effects of removing this information from the different decoders.

Here we made use of the framework provided by the original Prosit project. It was essential to access the scripts and data sets provided and hardened by the previous team of algorithm designers. In general, it is of utmost importance to keep this type of resource easy to access. If we want to attract the attention of the machine learning community, which often wants a precise problem formulation and does not like to get into the details of how to generate datasets from scratch, we need to help them.

Acknowledgements

Vital parts of this manuscript were conceived during the Dagstuhl Seminar 21271 on Computational Proteomics, July 2021.

The training of the Prosit Transformer model was enabled by the supercomputing resource Berzelius provided by National Supercomputer Centre at Linköping University and the Knut and Alice Wallenberg foundation.

Funding

This work has been supported by a grant from the Swedish Foundation for Strategic Research (BD15-0043).

References

[1] Jesse G Meyer. Deep learning neural network tools for proteomics. *Cell Reports Methods*, page 100003, 2021.

[2] Matthias Mann, Chanchal Kumar, Wen-Feng Zeng, and Maximilian T Strauss. Artificial intelligence for proteomics and biomarker discovery. *Cell Systems*, 12(8):759–770, August 2021.

[3] Jinghan Yang, Zhiqiang Gao, Xiuhan Ren, Jie Sheng, Ping Xu, Cheng Chang, and Yan Fu. DeepDigest: Prediction of protein proteolytic digestion with deep learning. *Analytical Chemistry*, 93(15):6094–6103, April 2021.

[4] Luminita Moruz, Daniela Tomazela, and Lukas Käll. Training, selection, and robust calibration of retention time models for targeted proteomics. *Journal of Proteome Research*, 9(10):5209–5216, October 2010.

[5] 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, September 2018.

[6] Robbin Bouwmeester, Ralf Gabriels, Niels Hulstaert, Lennart Martens, and Sven Degroeve. Deeplc can predict retention times for peptides that carry as-yet unseen modifications. *BioRxiv*, pages 2020–03, 2021.

[7] Florian Meier, Niklas D Köhler, Andreas-David Brunner, Jean-Marc H Wanka, Eugenia Voytik, Maximilian T Strauss, Fabian J Theis, and Matthias Mann. Deep learning the collisional cross sections of the peptide universe from a million experimental values. *Nature Communications*, 12(1):1185, February 2021.

[8] Lukas Käll, Jesse D Canterbury, Jason Weston, William Stafford Noble, and Michael J MacCoss. Semi-supervised learning for peptide identification from shotgun proteomics datasets. *Nature Methods*, 4(11):923–925, November 2007.

[9] Vadim Demichev, Christoph B Messner, Spyros I Vernardis, Kathryn S Lilley, and Markus Ralser. DIA-NN: neural networks and interference correction enable deep proteome coverage in high throughput. *Nature Methods*, 17(1):41–44, January 2020.

[10] Sven Degroeve, Davy Maddelein, and Lennart Martens. MS2PIP prediction server: compute and visualize MS2 peak intensity predictions for CID and HCD fragmentation. *Nucleic Acids Research*, 43(W1):W326–30, July 2015.

[11] Siegfried Gessulat, Tobias Schmidt, Daniel Paul Zolg, Patroklos Samaras, Karsten Schnatbaum, Johannes Zerweck, Tobias Knaute, Julia Rechenberger, Bernard Delanghe, Andreas Huhmer, Ulf Reimer, Hans-Christian Ehrlich, Stephan Aiche, Bernhard Kuster, and Mathias Wilhelm. Prosit: proteome-wide prediction of peptide tandem mass spectra by deep learning. *Nature Methods*, 16(6):509–518, June 2019.

[12] Ana S C Silva, Robbin Bouwmeester, Lennart Martens, and Sven Degroeve. Accurate peptide fragmentation predictions allow data driven approaches to replace and improve upon proteomics search engine scoring functions. *Bioinformatics*, 35(24):5243–5248, December 2019.

[13] Mathias Wilhelm, Daniel P Zolg, Michael Graber, Siegfried Gessulat, Tobias Schmidt, Karsten Schnatbaum, Celina Schwencke-Westphal, Philipp Seifert, Niklas de Andrade Krätzig, Johannes Zerweck, et al. Deep learning boosts sensitivity of mass spectrometry-based immunopeptidomics. *Nature Communications*, 12(1):1–12, 2021.

[14] Brian C Searle, Kristian E Swearingen, Christopher A Barnes, Tobias Schmidt, Siegfried Gessulat, Bernhard Küster, and Mathias Wilhelm. Generating high quality libraries for DIA MS with empirically corrected peptide predictions. *Nature Communications*, 11(1):1–10, 2020.

[15] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[16] 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.

[17] Roshan Rao, Nicholas Bhattacharya, Neil Thomas, Yan Duan, Xi Chen, John Canny, Pieter Abbeel, and Yun S Song. Evaluating protein transfer learning with TAPE. *Advances in Neural Information Processing Systems*, 32:9689–9701, December 2019.

[18] Tristan Bepler and Bonnie Berger. Learning the protein language: Evolution, structure, and function. *Cell Systems*, 12(6):654–669.e3, June 2021.

[19] Ziga Avsec, Vikram Agarwal, Daniel Visentin, Joseph R Ledsam, Agnieszka Grabska-Barwinska, Kyle R Taylor, Yannis Assael, John Jumper, Pushmeet Kohli, and David R Kelley. Effective gene expression prediction from sequence by integrating long-range interactions. *Nature Methods*, 18:1196–1203, 2021.

[20] Hao Cheng, Bing Rao, Lei Liu, Lizhen Cui, Guobao Xiao, Ran Su, and Leyi Wei. PepFormer: End-to-End transformer-based siamese network to predict and enhance peptide detectability based on sequence only. *Analytical Chemistry*, 93(16):6481–6490, 2021.

[21] Ronghui Lou, Weizhen Liu, Rongjie Li, Shanshan Li, Xuming He, and Wenqing Shui. Deepphospho accelerates dia phosphoproteome profiling through in silico library generation. *Nature communications*, 12(1):1–15, 2021.

[22] Melih Yilmaz, William E Fondrie, Wout Bittremieux, Sewoong Oh, and William Stafford Noble. De novo mass spectrometry peptide sequencing with a transformer model. *bioRxiv*, 2022.

[23] Wangchunshu Zhou, Tao Ge, Ke Xu, Furu Wei, and Ming Zhou. Scheduled Drophead: A regularization method for transformer models. *arXiv preprint arXiv:2004.13342*, 2020.

[24] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

For TOC Only: