yHydra: Deep Learning enables an Ultra Fast Open Search by Jointly Embedding MS/MS Spectra and Peptides of Mass Spectrometry-based Proteomics

Tom Altenburg[1], Thilo Muth[2], and Bernhard Y. Renard[1*]

[1] *Hasso Plattner Institute, Digital Engineering Faculty, University of Potsdam*
[2] *Federal Institute for Materials Research and Testing (BAM), Berlin, Germany*
[*] **Corresponding Author: bernhard.renard@hpi.de**

Abstract

Mass spectrometry-based proteomics allows to study all proteins of a sample on a molecular level. The ever increasing complexity and amount of proteomics MS-data requires powerful and yet efficient computational and statistical analysis. In particular, most recent bottom-up MS-based proteomics studies consider either a diverse pool of post-translational modifications, employ large databases – as in metaproteomics or proteogenomics, contain multiple isoforms of proteins, include unspecific cleavage sites or even combinations thereof and thus face a computationally challenging situation regarding protein identification. In order to cope with resulting large search spaces, we present a deep learning approach that jointly embeds MS/MS spectra and peptides into the same vector space such that embeddings can be compared easily and interchangeable by using euclidean distances. In contrast to existing spectrum embedding techniques, ours are learned jointly with their respective peptides and thus remain meaningful. By visualizing the learned manifold of both spectrum and peptide embeddings in correspondence to their physicochemical properties our approach becomes easily interpretable. At the same time, our joint embeddings blur the lines between spectra and protein sequences, providing a powerful framework for peptide identification. In particular, we build an open search, which allows to search multiple ten-thousands of spectra against millions of peptides within seconds. yHydra achieves identification rates that are compatible with MSFragger. Due to the open search, delta masses are assigned to each identification which allows to unrestrictedly characterize post-translational modifications. Meaningful joint embeddings allow for faster open searches and generally make downstream analysis efficient and convenient for example for integration with other omics types.

Availability: (under MIT license) `https://github.com/tzom/yHydra`
Contact: bernhard.renard@hpi.de
Keywords: Deep learning, Proteomics, Open Search, Post-translational Modifications, Joint Embedding

Introduction

Mass spectrometry (MS)-based proteomics provides insights into the content of proteins of living cells through high-throughput experiments. Peptide sequences (short snippets of protein sequences) are acquired in the form of their corresponding tandem mass spectra (MS/MS spectra) [1]. Even though recent types of MS-instruments produce more and higher quality data, the spectra remain a unique and challenging type of raw data. Typically a database search is used to robustly identify peptide sequences of acquired spectra. During a search, acquired spectra are compared to theoretical ones that are *in silico* constructed for each candidate peptide in the database [2]. However, each comparison is computationally expensive as it involves the construction of theoretical spectra, subsequent peak matching and scoring. Even more so, in an open search where considerably wide mass windows result in large search spaces for each spectrum. Hence, principally many comparisons are needed throughout an open search. Here, we present a learned joint embedding that reduces the computational cost per comparison. Furthermore, the embeddings allow us to systematically cover large search spaces and thus being nearly twice as fast as current open search methods.

Here, we use recent deep learning techniques, i.e. Transformers [3] to jointly embed spectra and peptides. As a result, we can easily and efficiently compare both types of data through their corresponding embeddings. In principal, any algorithm that requires real valued vectors of fixed size can be used on learned embeddings. In particular, by using UMAP on our embeddings, we gained insights into physico-chemical properties of spectra and peptides. To perform fast open searches, we use a GPU-accelerated k-nearest-neighbors (k-NN) search, which improves the runtime by nearly twofold compared to existing approaches while achieving similar identification rates. Due to the open search by yHydra, a delta mass profile is constructed such that post-translational protein modifications in the sample can be studied.

In a previous study we showed that end-to-end training of a deep learning models on a vast pool of MS/MS spectra is not only feasible but biochemically reasonable

patterns were learned by our model based on the spectra it had seen during training [4]. Most importantly, it had learned them on its own. This is often not the case in other approaches as typically some type of domain-specific feature engineering was involved which effectively reveals patterns to the model before it has had a chance to come up with them in the first place. In particular, through model interpretation on the peak-level we could show that biochemically relevant pattern between peaks were learned by our previous model AHLF [4].

Here, we extent this concept of being able to properly train a neural network from a large pool of MS/MS spectra by also presenting the respective identified peptides as input to a deep learning model with the goal to bridge the gap between both domains that, although largely distinct representations at first, effectively describe the same entity.

A collection of algorithmic approaches for open searches exist, including index-based methods [5, 6], tag-based methods [7, 8] or spectral library-based methods [9]. For index-based methods, a fragment-index is constructed allowing parallel scoring of PSMs. In contrast, tag-based methods identify short but certain regions of a peptide sequence based on the spectrum and pre-select peptide candidates that share these tags. Lastly, spectral library-based methods index the fragments of spectra in the library (similar to the fragment index-based methods above) and thus speed up the comparison between both types of spectra. Altogether, these methods miss out on providing a universal basis of how to compare spectra and peptides. A learned joint embedding would furthermore provide means to handle both spectra and peptides in a unified and efficient way.

Besides there have been recent approaches of embedding spectra for an error-tolerant search [10]. However, since only the spectra themselves are embedded the respective method is limited to perform spectral searches only. Furthermore, the MS1 precursor mass was incorporated as a main feature and thus the learned embeddings render an open search impossible.

Concurrently to our development, Tariq and Saeed presented a cross-modal approach for learning a similarity function between tandem mass spectra and peptides to leverage a closed search [11].

Here, we present a new approach of jointly embedding spectra and peptides, which allowed us to speed up open searches compared to existing approaches. Therefore, we trained Transformer models on 67 different proteomics repositories containing nearly 20 million identified peptide spectrum matches (PSMs). Besides, we developed a dedicated multiplexed schema for a k-NN search that is able to simultaneously perform searches against multiple indices or constraints on them. Lastly, we implemented a GPU-accelerated peak matching and scoring in order to speed up the downstream analysis and make use of GPU-acceleration in virtually all parts of yHydra.

Results

yHydra jointly embeds peptides and MS/MS spectra

Here, we present a learned joint embedding between peptides and fragmentation spectra. In particular, yHydra embeds both of these two distinct representations into a joint embedding space consisting of real-valued vectors in an euclidean space (Fig. 1A). Having such a metric between fixed-sized vectors allows computationally inexpensive comparisons between embeddings. To achieve such an embedding, we jointly trained dedicated Transformer models (one for each of the two domains) by providing a large collection of previously identified peptide spectrum matches (see Methods section). During training, the task for yHydra was to learn an embedding such that emeddings of a PSM show a smaller euclidean distance than any mismatching paired embedding (by swapping peptides and spectra of PSMs) within each mini-batch. Or in other words, to find embeddings such that the diagonal of the pairwise euclidean distance matrix (Fig. 1A, bottom right) is minimized while the off-diagonal of this matrix is maximized. Note, after both embedders (respectively, Spectrum Transformer and Peptide Transformer) have been trained they can be used separately and independently while both embed into the same joint space. In particular, we implement an open search by embedding MS/MS spectra in order to search them against embeddings of digested peptides from a proteome database (Fig. 1B). Because both embeddings are real-valued vectors of the same fixed size that live in the same joint embedding space we can make use of highly optimized algorithms, such as the GPU-accelerated k-nearest neighbor (k-NN) search [12], as we will demonstrate below.

As mentioned above, comparing spectra with one another typically requires peak-matching. Interestingly, the wavelet encoding (originally developed alongside the Transformer architecture) applied to m/z locations of peaks serves two purposes here. First, it allows us to present a spectrum to the Transformer model and second, it retains the notion of being able to separate peaks in close proximity (Fig. 1C). Actually, the shown metrics render to be even stricter (by providing a higher dynamic range of respective distance values considering the L1- or L2-metric) compared to the plain difference between two m/z values of two peaks in close proximity (Fig. 1C).

Finally, because the learned joint embedding yields real valued vectors of fixed size the entire toolbox of machine learning and statistical tools is open to be used in conjunction with our embeddings. For example, UMAP was used to visualize the manifold of the embeddings from identified PSMs (Fig. 1D). The two dimensional manifold of both peptide and spectrum embeddings of identified PSMs is colored according to charge, precursor mass and modifications. The embeddings are largely ordered from smaller to larger charges. Respectively, they are also sorted by precursor masses. This implies that representations retain information about those properties. In contrast, the PTMs are largely uniformly scattered, which makes the embedding suitable for an open search.

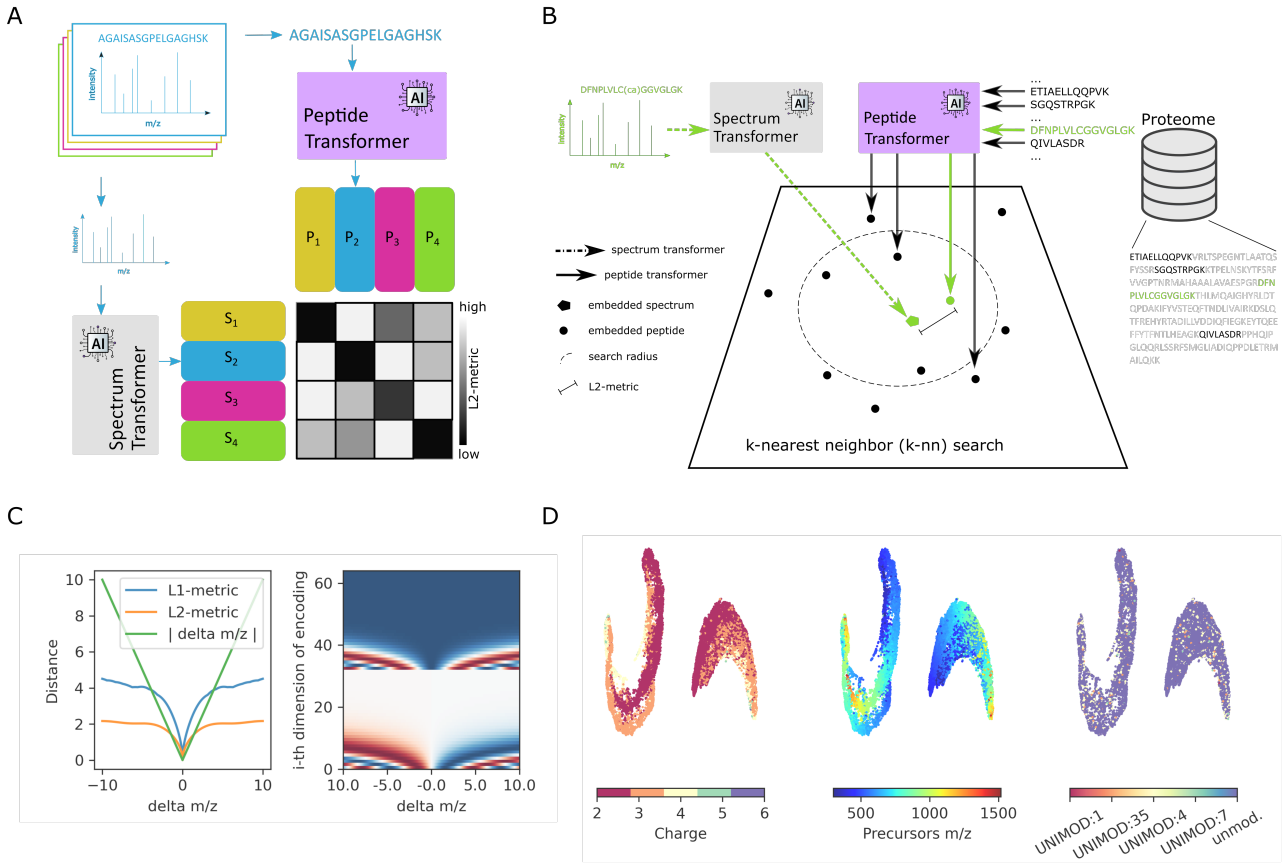

Figure 1: **Our deep learning model yHydra allows to jointly embed spectra and peptides.** **A**: Illustration of the yHydra architecture consisting of two transformers: for spectra (grey) and for peptide sequences (purple), each embed to their respective embeddings (yellow, blue, pink, green) for which pairwise euclidean distances (L2-metric) serve as loss during training. The pairwise distance metrics allows to update model parameters such that the diagonal minimized and the off-diagonal is maximized (see results section). **B**: Illustration of the open search of yHydra, using a trained spectrum transformer and trained peptide transformer. Each transformer embeds their respective input into the joint embedding space (center) in which a k-nearest neighbor search finds peptide candidates within an entire protein database. **C**: Left, comparison of the behavior of distances between the wavelet encoding of the m/z location of a single peak with respect to deviations from this location (-10 m/z to 10 m/z). Right, visualization of the change for each of the i-th dimension of the wavelet encoding of a single peak depending on m/z deviations. **D**: UMAP-visualization of the manifold of joint embeddings for identified peptide spectrum matches. The manifold appears as two clusters, spectra (left) and peptides (right). Each manifold is colored according to either charge, precursor mass or protein modification.

Learned joint embeddings enable an ultra fast open search

For the open search, yHydra starts from generating a set of tryptic peptides for a selected protein database (Fig. 1B). For each peptide, a trained peptide transformer (see Methods) infers a peptide embedding. Similarly, for each MS/MS spectrum in a run, the spectrum embedder infers a spectrum embedding. Subsequently, these spectrum embeddings are queried against the entire set of peptide embeddings using a k-nearest neighbor (k-NN) search. Note that all spectrum embeddings of an entire run (typically multiple ten-thousands) are queried simultaneously against the entire database in a single call in order to achieve lowest possible search times. In order to be able to search spectra against dedicated mass buckets (e.g. to select between a close-, narrow- or open search) while performing a single query per run we developed a method, which we call multiplexed k-NN search (Methods section). As a result, the actual k-NN search only takes a few seconds when using the GPU-acceleration (table 1). Here, we chose a k of 50, meaning for each spectrum we get its 50 nearest peptides. Only, these 50 candidate peptides are then scored, by constructing a theoretical spectrum, matching peaks and subsequently calculated a score in order to estimate a false discovery rate estimation using a target-decoy approach (2D).

For yHydra, the aforementioned combination of a k-NN search with classical scoring yields identification rates that are compatible with identifications by MS-Fragger. In particular, both methods identify peptides resulting in similar peptide length and charge distributions (Fig. 2B,C). The log score of yHydra allows to separate between targets and decoys and additionally reflects a bimodal distribution (in contrast to the hy-

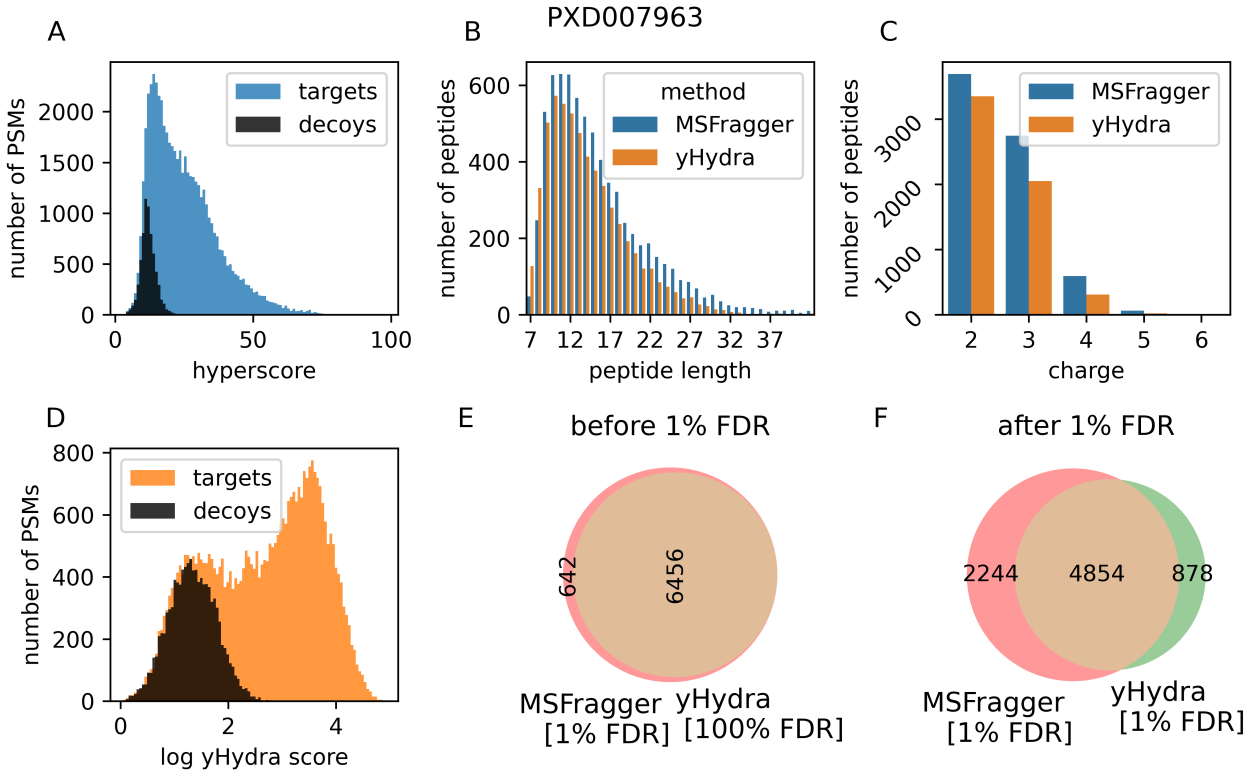

Figure 2: **yHydra enables an ultra fast open search. A**: Target-decoy distribution of hyperscores (as reported by MSFragger). **B**: Number of identified peptides versus peptide lengths according to MSFragger (blue) and yHydra (orange) identifications. **C**: Number of identified peptides versus precursor charge according to MSFragger (blue) and yHydra (orange) identifications. **D**: Target-decoy distribution of log yHydra scores. **E**: Venn diagram consisting of identified peptides (red; MSFragger with 1% FDR) versus top-k peptides found by yHydra (orange, 100% FDR). **F**: Venn diagram representing, a unique set peptides identified by MSFragger (red, 1% FDR), the intersection of identified peptides (orange; 1% FDR for both methods) and a unique set of peptide only identified by yHydra (green, 1% FDR).

perscore distribution by MSFragger).

	yHydra	MSFragger
per run [seconds]	**113.3**	183.0
total runtime [seconds]	**340.0**	549.0

Table 1: Runtimes of yHydra and MSFragger on PXD007963.

Discussion

We proposed a new approach of how to embed peptides and spectra of MS-based proteomics jointly. This allowed us to implement an open search in order to unrestrictedly characterize post-translational modified peptides. Besides, it allows to visualize spectra and peptides alongside each other by using manifold learning methods, e.g. as shown here by using UMAP. Overall, our approach is innovative regarding its underlying methodology while allowing an open peptide identification that is on par with other state-of-the-art open searches in MS-based proteomics.

Currently, we use a k-NN search with a specific k that is fixed for the entire search and thus it is the same for every spectrum. However, a radius-based search might improve search results by dynamically adjusting the k per each individual spectrum. For example, for some spectra it might be important to consider a large set of valid isoforms within the database for downstream scoring (i.e. k should be large). In contrast, for other spectra there might be a specific set of peptides in the database that is smaller than the chosen k (i.e. k should be smaller). Overall, this might improve both the sensitivity and specificity of the set of neighbors.

An open search by yHydra is possible because our embeddings are largely indifferent across PTMs, i.e. a peptide sequence can be identified irregarding possible PTMs (however the reported delta masses can give rise to PTMs). Specifically, this can be seen from the respective UMAP visualizations (Fig. 1D) and in the search results themselves and ultimately from the resulting delta mass profile of identified peptides (Fig. 2A). However, in certain situations it might not always be desirable that embeddings are independent of PTMs. As shown in our previous study [4], PTM detection based on the MS/MS spectrum directly, is a challenging topic on its own. If the PTMs would coincide with particular clusters it may harm the open search but, on the contrary, it would allow a multi-PTM detection. One could argue this is already given by the delta masses. However, as previously shown, a MS/MS spectrum-based

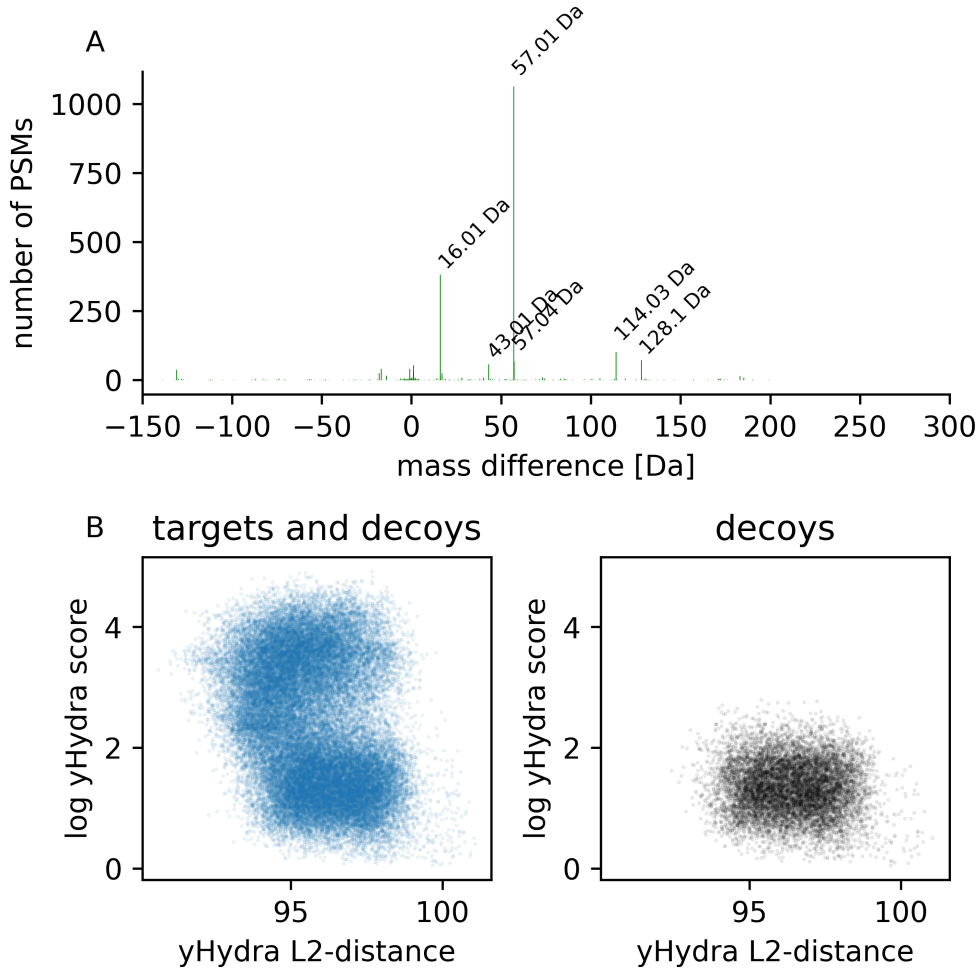

Figure 3: **yHydra allows an unrestrictive characterization of post-translational modifications.** **A**: yHydra matches spectra to peptides in a wide mass range, the resulting delta masses between precursor mass and peptide mass are shown here. The six most common delta masses are labeled. Note, the bulk of identifications with delta masses close to zero (between -1 Da and +1 Da) are not shown here. **B**: Distribution of log yHydra scores and euclidean distances shown for target and decoy PSMs (left) and decoys only (right).

detection of PTMs additional to the delta mass improves the identification rate [4]. Besides, the apparent delta mass (i.e. the difference between precursor mass and peptide mass) only reflects the sum of potentially multiple underlying delta masses.

Currently, we still need to algorithmically compute scores per PSM within the list of candidate peptides, because the L2-distance of yHydra is only accurate up to a certain k that is considerably larger that the one single correct peptide. A comparison between the log yHydra score and L2-distance of yHydra (Fig. 3B) shows that log yHydra score separates targets and decoys much better than the L2-distance alone. However, it also shows that the combination of both features would be better than each individual feature alone. Hence, a rescoring (e.g. via Percolator) of PSMs and therefore an integration of the L2-distance with the log score of yHydra would potentially boost the number of identifications. Otherwise keeping an algorithmic approach of scoring PSMs has the advantage to be flexible in terms of which PTMs, ion-types, etc. should be specifically considered during scoring. In contrast, any incorporation or change of the aforementioned properties within the L2-distance itself would require to retrain the embedders of yHydra.

Methods

Preprocessing and encoding of spectra and peptides

The intensities are normalized such that the intensities $(I_1, ..., I_l, ..., I_N)$ are a unity-length vector (i.e. scaled by $1/\sqrt{\sum I_l^2}$). For each peaks list, up to 500 peaks are kept. Otherwise, if the list contains more than the maximum number of peaks, we keep peaks according to the top-500 highest intensities or append tuples of (0.0,0.0) if the peaks list contained less peaks.

Both spectra and peptides are fed to Transformer architectures, hence we adapt the original positional encoding (also called wavelet encoding) and extend it in

order to suit the context of spectra and peptides.

For the positional encoding of the peaks list of the peaks list (mz_1, ..., mz_l, ..., mz_{500}), each mz_l is multiplied by the radiant rate r_i of the i-th dimension of the encoding vector s: $s_{li} = mz_l \cdot r_i$, where $r_i := 10000.0^{-i/d}$. Furthermore, $S'_{li} = sin(s_{li})$ if i is an even integer and $S'_i = cos(s_i)$ if i is odd. For the final spectrum encoding S, each positional encoded mz_l is scaled by its respective intensity: $S_i = I_l \cdot S'_{li}$. Hence S is a tensor of size (number of peaks, d) = (500,64).

For each peptide sequence, we enumerate each amino acid (starting from zero, up to peptide length n - 1) these integer positions p_n are then multiplied by a radiant rate r_i (same definition as above) of the i-th dimension of the peptide positional encoding vector p: $p_{li} = p_l \cdot r_i$. Up to n=42 amino acids are allowed, if the peptide is shorter it is padded with zeros. Hence, the peptide positional encoding vector has a size of (42,64). Furthermore, the amino acids are replaced by unique indices (starting from 0 up to 25, with zero being a padding character) and function as the index into a lookup table of size (alphabet size, d) = (26, 64), where the parameters of the lookup table are trainable and trained together with the model. For the final peptide sequence encoding P, each positional encoded p_i is multiplied by its amino acids embedding (from the lookup table) : $P_{li} = p_{li} \cdot aa_i$. Hence P is a tensor of size (maximum peptide length, d) = (42,64).

Training of yHydra

Training data

The Transformer models of yHydra were trained on 19,991,263 PSMs (retrieved as USIs) from the following 67 repositories: PXD000702, PXD001072, PXD001344, PXD001351, PXD002054, PXD002147, PXD003094, PXD003261, PXD003364, PXD003556, PXD003718, PXD003779, PXD003916, PXD003976, PXD004398, PXD004825, PXD005009, PXD005117, PXD005196, PXD005306, PXD005341, PXD005654, PXD005744, PXD006033, PXD006084, PXD006316, PXD006375, PXD006389, PXD006645, PXD006823, PXD006836, PXD008592, PXD008602, PXD008622, PXD008647, PXD008667, PXD008895, PXD009387, PXD009665, PXD009698, PXD009713, PXD010000, PXD010641, PXD010827, PXD011042, PXD011583, PXD011712, PXD011714, PXD011984, PXD012827, PXD013274, PXD013304, PXD013684, PXD013711, PXD013712, PXD013724, PXD013890, PXD013897, PXD015153, PXD015296, PXD015698, PXD016833, PXD016846, PXD017308, PXD018714, PXD019095, PXD019134. We specifically selected repositories of non-model organisms (excluding the top-10 most commonly studied organisms in terms of counts of repositories) in order to reduce the bias towards certain proteome-specific sequence patterns. Furthermore, we only included data acquiered on Q Exactive.

Pairwise contrastive loss of yHydra

The loss of yHydra is inspired by the recent approach of CLIP [13] and is based on the idea to directly calculate a contrastive loss based on the pairwise distances within each mini-batch (table 2 and Fig.1A). We found this approach has major advantages over previous types of contrastive-losses or triplet-losses while virtually having none of their shortcomings. Most importantly the pairwise contrastive loss does not not require the creation of artificial negatives as they naturally occur due to the mixed pairs of distances (i.e. off-diagonal elements in Fig.1A). This is not only more elegant than previous contrastive loss formulations but also makes the network learn at anytime, whereas for older types of contrastive losses typically hard-negative mining was essential to get decent training results. Furthermore, we extended this idea by adding label smoothing which should allow the model to also learn from the specific but small distances that mixed negatives naturally have. Label smoothing allows the model to also learn from this regime (i.e. off-diagonal elements in Fig.1A).

IN	:	spectrum embedding $T_{SPEC}(SPEC_{ENC})$ peptide embedding $T_{SEQ}(SEQ_{ENC})$
E_{SPEC}	=	$L2Norm(T_{SPEC}(SPEC_{ENC}))$
E_{SEQ}	=	$L2Norm(T_{SEQ}(SEQ_{ENC}))$
D	=	$L2Dist_{pairwise}(E_{SPEC}, E_{SEQ}.T)$
$logits$	=	$-tf.log(D + eps) \cdot tf.exp(T)$
$targets$	=	$tf.range(N)$
L_{SPEC}	=	tf.sce(targets,logits,smoothing,axis=0)
L_{SEQ}	=	tf.sce(targets,logits,smoothing,axis=1)
$loss$	=	$(L_{SPEC} + L_{SEQ})/2$
OUT	:	$loss$

Table 2: Tensorflow-/Numpy-like pseudocode of the calculation of the pairwise loss between spectra and peptides. The Transformer models T_{SPEC} and T_{SEQ} yield embeddings E_{SPEC} and E_{SEQ} each of size (batch-size=64, embeddings-size=64). The spectrum encoding $SPEC_{ENC}$ and peptide sequence SEQ_{ENC} are described in the main text. Parameters are set to eps=0.001, smoothing=0.1 and T=3.0 and tf.sce is the sparse cross entropy between targets and logits according to the the distance matrix D (illustrated in Fig. 1A).

yHydra algorithm

Multiplexed k-NN search of mass buckets for closed-, narrow- and open searches

Our multiplexed k-NN search allows us to search all spectra embeddings against all peptide embeddings within the same search call while at the same time only spectra against theoretically possible peptides (e.g. determined by certain combinations of precursor mass and respective peptide masses) are searched. Therefore we divide the peptides in the database into buckets according to their theoretical mass. Hence, for a closed search we could have a thousand of small buckets (of +/-1 Da width) and, in contrast, for the open search we have a few but wide buckets (e.g. +/-500 Da width). Each bucket gets a unique vector assigned, which is appended to the peptide embeddings in that bucket (i.e. similar to an 'address' vector). Subsequently, the query embeddings, which are supposed to be searched against a specific bucket also gets the respective address vector appended. Effectively, the L2-norm between the embeddings is dictated by their common 'address'-vector because only those with a common address-vector have meaningful intra-buckets L2-distances but comparatively high inter-bucket L2-

distances. Ultimately, this allows us to achieve multiple mass-compliant search calls while really only performing a single search.

GPU-accelerated peak matching and PSM scoring

The core algorithms of yHydra are GPU-accelerated (i.e. neural networks and k-NN search by faiss [12]). To further speed up the runtime of yHydra we developed a GPU-accelerated peak-matching and PSM scoring (table 3). The idea is to simultaneously score a batch of 64 spectra against their respective k-candidates, i.e. k=50, which means for 3,200 PSMs peaks are matched and scored in parallel.

			tensor size
IN	:	queries $q = mz_{theor}$	[b,k,l]
		keys $k = mz_{acq.}$	[b,n]
		values $v = I_{acq.}$	[b,n]
M	=	$L2Dist_{pairwise}(q, k.T)$	[b,k,l,n]
M	=	tf.where($M <$ tolerance, $1/(M)$, 0.0)	[b,k,l,n]
M	=	tf.reduce_maximum(M,axis=2)	[b,k,n]
M	=	tf.where($M > 0.0, 1.0, 0.0$)	[b,k,n]
S	=	$M \cdot v$	[b,k]
S	=	tf.reduce_maximum(S,axis=-1)	[b]
I	=	tf.argmax(S,axis=-1)	[b]
OUT	:	indices (I) and scores (S) of the best matching peptide for each spectra in the mini-batch of size b	

Table 3: Tensorflow-/Numpy-like pseudocode of peak matching and scoring for PSMs. The inputs are a mini-batch of b spectra, with n peaks considering their mz-values $mz_{acq.}$ and intensities $I_{acq.}$. Furthermore, list of k candidate peptide (result of the k-NN search) is considered as theoretical ions with up to l mz-locations mz_{theor}, see Methods for details on parameters.

Open search parameters for yHydra and MSFragger

For both search open engines (yHydra and MSFragger) the raw files qe2_03132014_1WT-1, qe2_03132014_5WT-2, and qe2_03132014_13WT-3 were searched against the SynPCC7002_Cbase.fasta by considering tryptic peptides with up to 1 miscleavage of length between 7 and 42 amino acids. In both searches a minimum delta mass of -150.0 Da and a maximum delta mass of 500.0 Da is considered.

For yHydra, the matching tolerance was set to 0.01 Da and a mimimum of 4 matching peaks for a PSM was required. For scoring, the globally highest top-100 peaks per spectrum are considered. For each pepetide, b- and y-ions are calculated and up to 200 fragment ions are considered (any excess ions are discarded starting from higher charge states). For MSFragger, we used the standard search parameters of MSFragger version 3.3, the maximum peptide length of 42 and allowing up to 1 miscleavage. For both methods, the PSM-level FDR was set to 1%.

Data availability

Proteomic data were downloaded from public repositories PXD000702, PXD001072, PXD001344, PXD001351, PXD002054, PXD002147, PXD003094, PXD003261, PXD003364, PXD003556, PXD003718, PXD003779, PXD003916, PXD003976, PXD004398, PXD004825, PXD005009, PXD005117, PXD005196, PXD005306, PXD005341, PXD005654, PXD005744, PXD006033, PXD006084, PXD006316, PXD006375, PXD006389, PXD006645, PXD006823, PXD006836, PXD008592, PXD008602, PXD008622, PXD008647, PXD008667, PXD008895, PXD009387, PXD009665, PXD009698, PXD009713, PXD010000, PXD010641, PXD010827, PXD011042, PXD011583, PXD011712, PXD011714, PXD011984, PXD012827, PXD013274, PXD013304, PXD013684, PXD013711, PXD013712, PXD013724, PXD013890, PXD013897, PXD015153, PXD015296, PXD015698, PXD016833, PXD016846, PXD017308, PXD018714, PXD019095, PXD019134 in order to train yHydra. Furthermore, public data from PXD007963 was downloaded in order to evaluate yHydra.

Code availability

An open source implementation with command-line instructions is publicly available (under MIT license) at `https://github.com/tzom/yHydra` .

References

[1] Ruedi Aebersold and Matthias Mann. Mass-spectrometric exploration of proteome structure and function. *Nature*, 537(7620):347–355, sep 2016.

[2] Alexey I Nesvizhskii, Olga Vitek, and Ruedi Aebersold. Analysis and validation of proteomic data generated by tandem mass spectrometry. *Nature Methods*, 4(10):787–797, oct 2007.

[3] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention Is All You Need. jun 2017.

[4] Tom Altenburg, Sven Giese, Shengbo Wang, Thilo Muth, and Bernhard Y. Renard. AHLF: ad hoc learning of peptide fragmentation from mass spectra enables an interpretable detection of phosphorylated and cross-linked peptides. *bioRxiv*, page 2020.05.19.101345, 2021.

[5] Andy T. Kong, Felipe V. Leprevost, Dmitry M. Avtonomov, Dattatreya Mellacheruvu, and Alexey I. Nesvizhskii. MSFragger: Ultrafast and comprehensive peptide identification in mass spectrometry-based proteomics. *Nature Methods*, 14(5):513–520, 2017.

[6] Doruk Beyter, Miin S. Lin, Yanbao Yu, Rembert Pieper, and Vineet Bafna. ProteoStorm: An Ultrafast Metaproteomics Database Search Framework. *Cell Systems*, 7(4):463–467.e6, oct 2018.

[7] Bernhard Y. Renard, Buote Xu, Marc Kirchner, Franziska Zickmann, Dominic Winter, Simone Korten, Norbert W. Brattig, Amit Tzur, Fred A. Hamprecht, and Hanno Steen. Overcoming Species Boundaries in Peptide Identification with Bayesian Information Criterion-driven Error-tolerant Peptide Search (BICEPS). *Molecular & Cellular Proteomics*, 11(7):M111.014167–1–M111.014167–12, jul 2012.

[8] Arun Devabhaktuni, Sarah Lin, Lichao Zhang, Kavya Swaminathan, Carlos G. Gonzalez, Niclas Olsson, Samuel M. Pearlman, Keith Rawson, and Joshua E. Elias. TagGraph reveals vast protein modification landscapes from large tandem mass spectrometry datasets. *Nature Biotechnology*, 37(4):469–479, apr 2019.

[9] Wout Bittremieux, Pieter Meysman, William Stafford Noble, and Kris Laukens. Fast open modification spectral library searching through approximate nearest neighbor indexing. *bioRxiv*, jan 2018.

[10] Damon H May, Jeff Bilmes, and William S Noble. A learned embedding for efficient joint analysis of millions of mass spectra. *bioRxiv*, jan 2018.

[11] Muhammad Usman Tariq and Fahad Saeed. SpeCollate: Deep cross-modal similarity network for mass spectrometry data based peptide deductions. *PLOS ONE*, 16(10):e0259349, oct 2021.

[12] Jeff Johnson, Matthijs Douze, and Herve Jegou. Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data*, pages 1–1, 2019.

[13] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning Transferable Visual Models From Natural Language Supervision. feb 2021.