A learned embedding for efficient joint analysis of millions of mass spectra

Wout Bittremieux[1], **Damon H. May**[2], **Jeffrey Bilmes**[2,3], **William Stafford Noble**[2,3,*]

[1]Skaggs School of Pharmacy and Pharmaceutical Science, University of California San Diego, La Jolla, CA 92093, USA; [2]Department of Genome Sciences, University of Washington, Seattle, WA 98195, USA; [3]Paul G. Allen School of Computer Science and Engineering, University of Washington, Seattle, WA 98195, USA

Corresponding author: william-noble@uw.edu

Abstract

Despite the rapidly increasing amount of data in public mass spectrometry repositories, peptide mass spectra are usually analyzed by each laboratory in isolation, treating each experiment as if it has no relationship to any others. This approach fails to exploit the wealth of existing, previously analyzed mass spectrometry data. Alternatively, although large spectral data can be jointly analyzed using spectrum clustering methods, this unsupervised approach does not utilize information from peptide identifications. Here, we propose to train a deep neural network in a supervised fashion based on previous assignments of peptides to spectra. The network, called "GLEAMS," learns to embed spectra into a low-dimensional space in which spectra generated by the same peptide are close to one another. We empirically demonstrate the utility of this learned embedding by propagating annotations from labeled to unlabeled spectra. We further use GLEAMS as the basis for a large-scale spectral clustering, detecting groups of unidentified, proximal spectra representing the same peptide, and we show how to use these clusters to explore the dark proteome of repeatedly observed yet consistently unidentified mass spectra. We provide a software implementation of our approach, along with a tool to quickly embed additional spectra using a pre-trained model, to facilitate large-scale analyses.

Keywords: mass spectrometry, proteomics, machine learning, deep learning

1 Introduction

Since the publication of the SEQUEST search algorithm in 1994,[1] the dominant approach to assigning peptide sequences to tandem mass spectrometry (MS/MS) data has been to derive a list of candidates for each spectrum

from a database, generate a theoretical spectrum for each candidate, and then score each putative peptide-spectrum match (PSM) based on the similarity between the observed and theoretical spectrum fragments. Major advances have been made in search algorithm development and various downstream analyses,[2] but an individual lab searching their spectra against a sequence database remains the dominant paradigm for MS/MS spectrum identification.

Over the last decade, public proteomics repositories such as the PRoteomics IDEntifications (PRIDE) database[3] and MassIVE have grown to include billions of MS/MS spectra from tens of thousands of assays. As these repositories have become more comprehensive, efforts have been undertaken to make the spectra useful to researchers analyzing new datasets. Some approaches, such as PeptideAtlas,[4] the Global Proteome Machine Database,[5] and MassIVE,[6] involve re-searching the spectra using a common workflow. Typically, the output of the analysis is a spectral library, in which sets of spectra corresponding to the same peptide sequence are condensed into a single spectrum either by averaging or selecting a single, representative spectrum per peptide. The spectral library can then be used to analyze new datasets using a search algorithm. A drawback to any method that relies on standard database searching, however, is that prior to false discovery rate (FDR) estimation each spectrum is treated as an independent observation. By failing to jointly consider all of the spectra together during the assignment of peptides to spectra, these pipelines miss out on the opportunity to exploit valuable structure in the data.

An alternative to simple re-searching of the data is to employ clustering algorithms, several of which have been developed specifically for clustering mass spectra, including MS-Cluster,[7] spectra-cluster,[8,9] MaRaCluster,[10] msCRUSH,[11] and falcon.[12] Although spectra-cluster and MS-Cluster have been used to process hundreds of millions of spectra from PRIDE[8,9] and MassIVE,[6] respectively, in practice, however, it remains challenging to apply these methods on the repository scale. Furthermore, clustering approaches are limited in their ability to incorporate new data sets. Although new spectra that correspond to previously detected peptides can easily be added to the corresponding clusters, and some greedy clustering methods can use newly acquired spectra to create new clusters by merging existing ones,[6] running an entire clustering algorithm from scratch is an extremely expensive operation that becomes progressively more expensive as the repository grows.

More fundamentally, clustering is problematic because it is an *unsupervised* approach. The input to a clustering algorithm is an unlabeled set of spectra. In practice, the labels (i.e. the associated peptide sequences) are used only in a *post hoc* fashion, to choose how many clusters to produce or to split up large clusters associated with multiple peptides. In recent years, a revolution has occurred in machine learning, with deep neural networks proving to have applicability across a wide array of problems.[13] Accordingly, within the field of proteomics, deep neural networks have recently been applied to several problems, including *de novo* peptide sequencing[14,15] and simulating MS/MS spectra.[16,17] However, to our knowledge no one has yet applied deep neural networks to

the problem of making public repository data contribute to the analysis of new mass spectrometry experiments. We hypothesize that we can obtain more accurate and useful information about a large collection of spectra by using a supervised deep learning method that directly exploits peptide–spectrum assignments during joint analysis. Specifically, we posit that peptide labels can be used during training of a large-scale learned model of MS/MS spectra to achieve a robust, efficient, and accurate model.

Accordingly, we propose GLEAMS (GLEAMS is a Learned Embedding for Annotating Mass Spectra), which is a deep neural network that has been trained to embed MS/MS spectra into a 32-dimensional space in such a way that spectra generated by the same peptide, with the same post-translational modifications (PTMs) and charge, are close together. Our work builds upon methods that have been used successfully to embed various types of data, including text documents[18] and images.[19] The learned spectral embedding offers the advantage that new spectra can efficiently be mapped to the embedded space without requiring re-training. Our approach is fundamentally different from previous (unsupervised) spectrum clustering applications, in the sense that it uses peptide assignments generated from standard database search methods as labels in a supervised learning setting.

We validate the embedding by demonstrating its ability to place spectra assigned the same label by database search close together, and we describe a method for using this embedding to assign peptide labels to new spectra. We also demonstrate how to use GLEAMS in conjunction with the DBSCAN algorithm[20] to induce a repository-scale clustering of spectra in the MassIVE database. We use these clusters to help characterize the so-called "dark matter" of proteomics. We provide an open-source software implementation of our method, as well as a pre-trained model that can be used to efficiently embed spectra for joint analysis (https://github.com/bittremieux/GLEAMS).

2 Results

2.1 A deep neural network learns to embed spectra into a common latent space

The learned model consists of a "Siamese network,"[21] in which two copies of an embedding network operate side by side (Figure 1A). During training, the network is provided with pairs of spectra S_1 and S_2 and an associated label Y, where $Y = 1$ indicates that the spectra were generated by the same peptide, and $Y = 0$ indicates that they were generated from different peptides. Each embedder transforms a spectrum S_i into its embedded representation $G_W(S_i)$, where W represents the learned network weights. The key to the learning process is

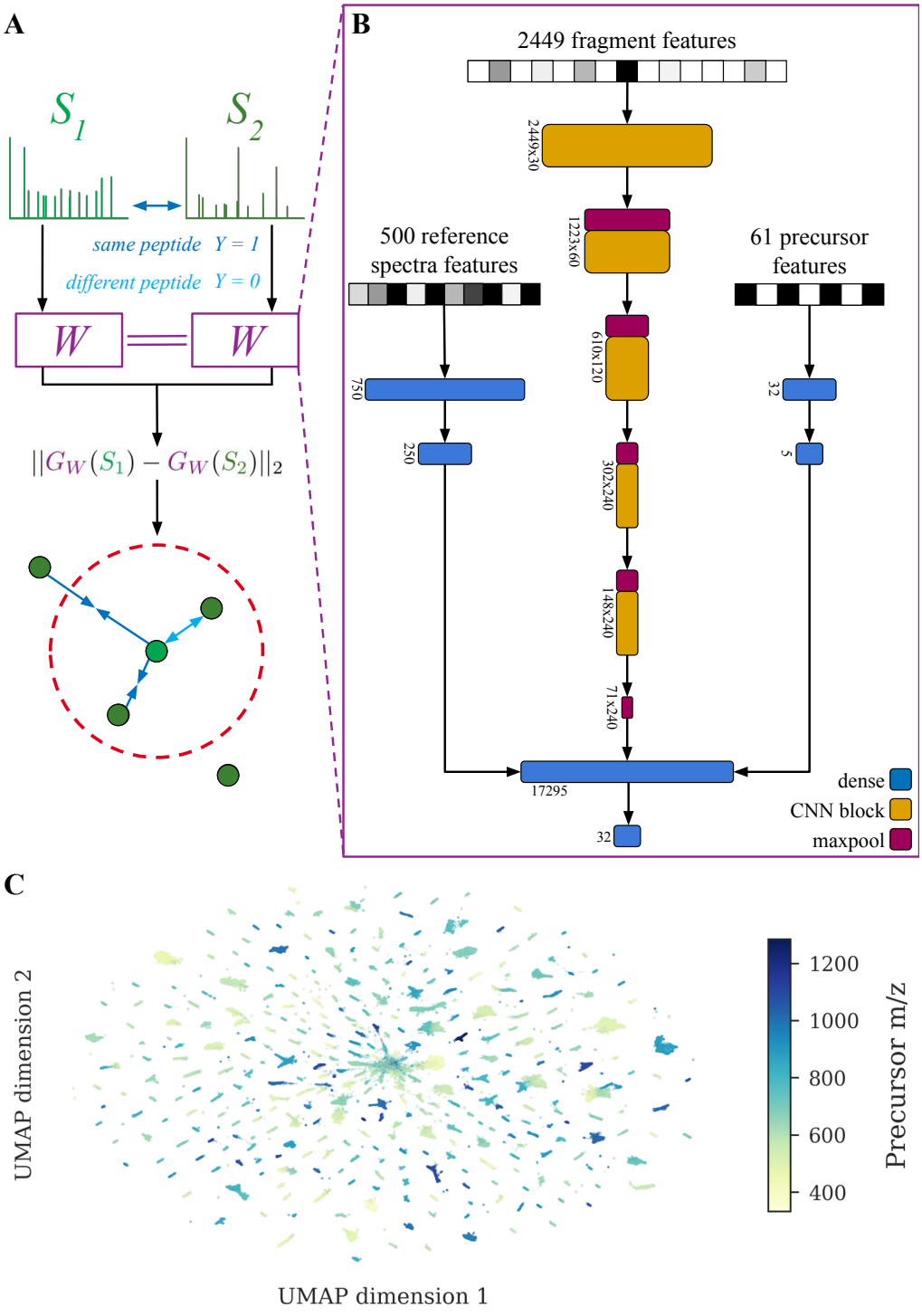

Figure 1: **(A)** Two spectra, S_1 and S_2, are encoded to vectors and passed as input to two instances of the embedder network with tied weights. The Euclidean distance between the two resulting embeddings, $G_W(S_1)$ and $G_W(S_2)$, is passed to a contrastive loss function that penalizes dissimilar embeddings that correspond to the same peptide and similar embeddings that correspond to different peptides, up to a margin of 1. **(B)** The embedder network separately receives each of three feature types as input. Precursor features are processed through a fully-connected network with two layers of sizes 32 and 5. Binned fragment intensities are processed through five blocks of one-dimensional convolutional layers and max pooling layers. Reference spectra features are processed through a fully-connected network with two layers of sizes 750 and 250. The output of the three subnetworks is concatenated and passed to a final fully-connected layer of size 32. **(C)** UMAP projection of 685 337 embeddings from frequently occurring peptides in 10 million randomly selected identified spectra from the test dataset. As indicated by the coloration, precursor m/z has a large influence on the location of spectra in the embedded space.

the contrastive loss function adapted from Hadsell et al. [21] defined as

$$L(W, Y, S_1, S_2) = Y(\min(||G_W(S_1) - G_W(S_2)||_2), 1)^2$$
$$+(1 - Y)(\max(0, 1 - ||G_W(S_1) - G_W(S_2)||_2))^2.$$

Intuitively, this loss function pulls the two spectra together if they are associated with the same peptide ($Y = 1$) and pushes them apart if they are associated with different peptides ($Y = 0$). Backpropagation from this loss function is used to update the weights in the network.

The heart of the model itself is the embedder network (Figure 1B) which takes as input a spectrum and embeds it into a 32-dimensional space. The model contains two copies of the embedder, with weights tied so that updates to one embedder are always reflected in the other. For input to the embedder, each spectrum is encoded using three sets of features representing, respectively, attributes of the precursor ion, binned fragments, and similarities to an invariant set of reference spectra. Each of the different feature types is processed through a separate deep neural subnetwork, after which the outputs of the three networks are concatenated and passed to a final, fully-connected layer to produce vector embeddings with dimension 32.

GLEAMS was trained using a set of 30 million high-quality PSMs derived from the MassIVE knowledge base (MassIVE-KB).[6] Importantly, peptide sequence information is only required during initial supervised training of the Siamese network. Subsequent processing using an individual embedder instance is agnostic to the peptide label and can be performed on identified and unidentified spectra in a similar fashion. After training, the embedder model was used to process 669 million spectra from 227 public human proteomics datasets included in MassIVE-KB. As an initial evaluation of the learned embeddings, these spectra were further projected down to two dimensions using UMAP[22] for visual inspection (Figure 1C). The visualizations suggest that precursor mass (Figure 1C) and precursor charge (Supplementary Figure 1) strongly influence the structure of the embedded space, and that similar spectra are indeed located close to each other. Additionally, several of the individual embedding dimensions show a correlation with the precursor mass, peptide sequence length, or whether the peptides have an arginine or lysine terminus (Supplementary Table 1). This indicates that the GLEAMS embeddings capture latent characteristics of the spectra. Interestingly, although some of these properties were provided as input to the neural network, such as precursor mass, other properties were derived from the data without explicitly encoding them.

2.2 Clustering of spectrum embeddings to explore spectral similarity

If our training worked well, then spectra generated by the same peptide should lie close together, according to a Euclidean metric, in the embedded space. Accordingly, we investigated, for 10 million randomly chosen

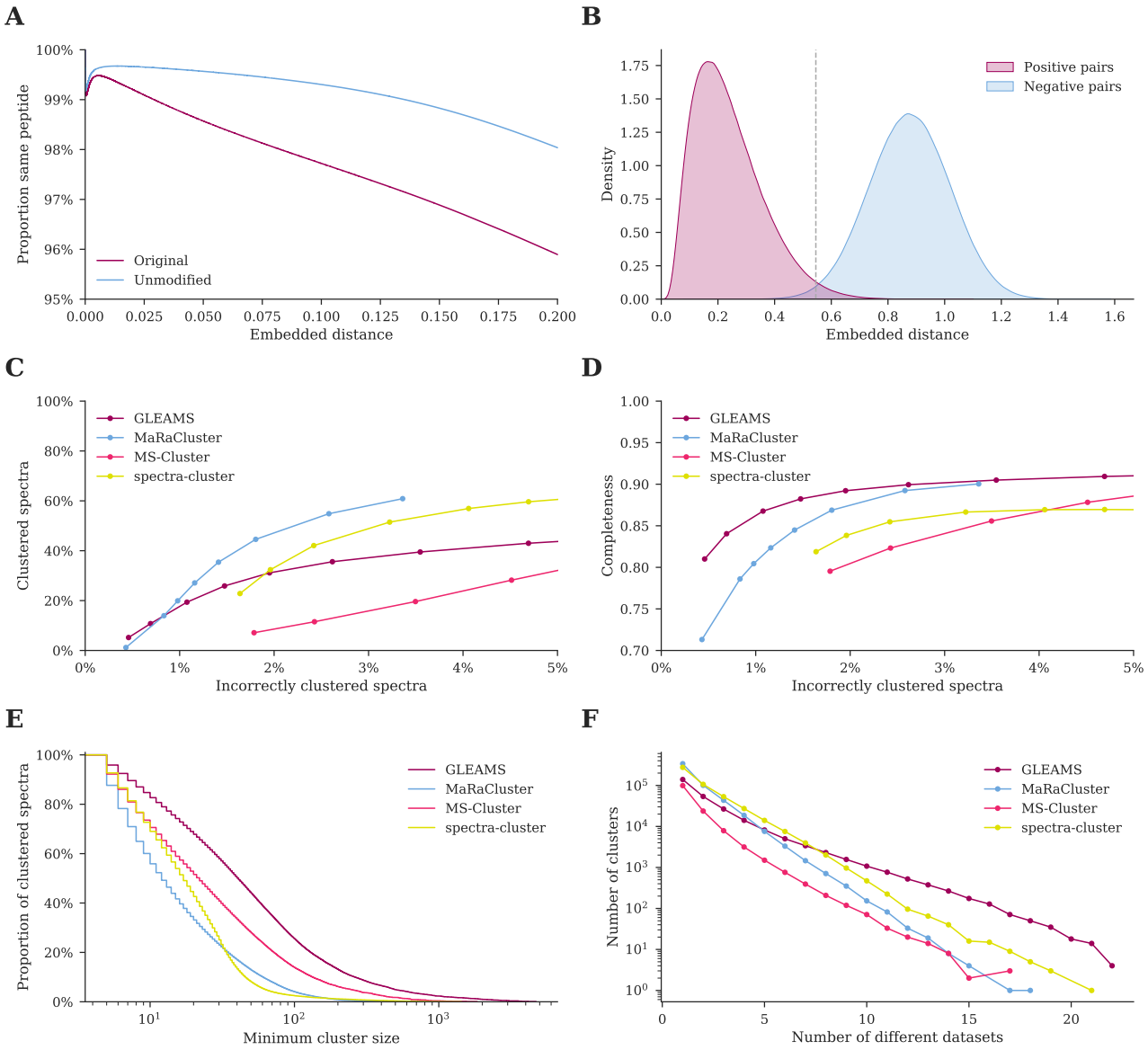

Figure 2: **(A)** Proportion of neighbors that have the same peptide label as a function of the distance threshold for 186 865 330 pairwise distances between 10 million randomly selected embeddings from the test dataset (on average 18.7 neighbors per embedding). Embeddings at small distances represent the same peptide ("Original"), while the majority of close neighbors with different peptide labels correspond to peptides with ambiguously localized modifications ("Unmodified"). **(B)** The false negative rate between positive and negative embedding pairs, for 10 million randomly selected pairs from the test dataset, at distance threshold 0.5455 (grey line), corresponding to 1% FDR, is only 1%, indicating excellent separation between positive and negative pairs. **(C+D)** Average clustering performance over three random folds of the test dataset containing 28 million MS/MS spectra each. **(C)** The number of clustered spectra versus the number of incorrectly clustered spectra per clustering algorithm. GLEAMS and MaRaCluster succeed in clustering the highest number of spectra at low rates of incorrectly clustered spectra, whereas MS-Cluster and spectra-cluster are unable to produce highly pure clusters. **(D)** Cluster completeness versus the number of incorrectly clustered spectra per clustering algorithm. GLEAMS produces the most complete clustering result at different levels of incorrectly clustered spectra. This indicates that GLEAMS achieves a greater data reduction than alternative clustering tools, without sacrificing the cluster quality. **(E+F)** Clustering result characteristics at approximately 1% incorrectly clustered spectra over three random folds of the test dataset (Supplementary Table 2). **(E)** Complementary empirical cumulative distribution of the cluster sizes. GLEAMS produces larger clusters than alternative clustering tools, grouping similar spectra into a single cluster rather than splitting them over multiple related clusters. **(F)** The number of datasets that spectra in the test dataset originate from per cluster (24 datasets total). GLEAMS successfully groups related spectra from heterogeneous datasets into single clusters.

embedded spectra, the relationship between neighbor distance and the proportion of labeled neighbors that have the same peptide label. The results show that neighbors at small distances overwhelmingly represent the same peptide (Figure 2A). Furthermore, the few different-peptide labels at very small distances almost entirely represent virtually indistinguishable spectra that have identical peptide labels but differ in ambiguous modification localizations. We also investigated the false negative rate, for 10 million randomly chosen embedding pairs, to understand the extent to which embeddings that correspond to the same peptide are distant in the embedded space (Figure 2B). This analysis shows an excellent separation between same-labeled embeddings and embeddings corresponding to different peptides, with a very small false negative rate of only 1% at a distance threshold corresponding to 1% FDR. Furthermore, the embeddings are robust to different types of mass spectrometry data. Phosphorylation modifications were not included in the MassIVE-KB dataset, and GLEAMS thus did not see any phosphorylated spectra during its training. Nonetheless, GLEAMS was able to embed spectra from a phosphoproteomics study with high accuracy (Supplementary Figure 2).[23]

To further investigate the utility of the GLEAMS embedding, we performed DBSCAN clustering[20] in the embedded space to find dense clusters of spectra, and we compared the performance to that of the commonly used spectrum clustering tools MaRaCluster,[10] MS-Cluster,[7] and spectra-cluster.[8,9] The comparison indicates that clustering in the GLEAMS embedded space is of a similar or higher quality than clusterings produced by state-of-the-art tools, especially at low rates of incorrectly clustered spectra (Figure 2C–D). Notably, even with the most stringent hyperparameters, MS-Cluster and spectra-cluster struggled to produce pure clustering results (i.e. low numbers of incorrectly clustered spectra), whereas specifying a low Euclidean distance threshold for GLEAMS clustering succeeds in producing highly pure clusters (Figure 2C). Additionally, GLEAMS generates the most "complete" clustering result (Figure 2D). Completeness measures the extent to which multiple spectra corresponding to the same peptide are concentrated in few clusters. By minimizing the extent to which spectra generated from the same peptide are assigned to different clusters, GLEAMS achieves improved data reduction from spectrum clustering compared to alternative clustering tools. This property is especially relevant when performing spectrum clustering at the repository scale, to maximally reduce the data volume for efficient downstream processing of the clustered data.

To further understand the clustering performance of GLEAMS, we investigated clusters produced by the different tools at a low rate (∼1%) of incorrectly clustered spectra (Supplementary Table 2). These results indicate that GLEAMS and MaRaCluster succeed in clustering the highest number of spectra while minimizing the number of incorrectly clustered spectra. Furthermore, although GLEAMS and MaRaCluster cluster a similar total number of spectra, MaRaCluster groups these spectra into twice as many distinct clusters as GLEAMS. Overall, compared to alternative clustering tools, GLEAMS clearly produces the largest clusters (Figure 2E). In contrast, MaRaCluster and spectra-cluster predominantly generate clusters containing fewer than 100 spectra. We hypothesize that GLEAMS' supervised training allows the model to focus on relevant spectrum features

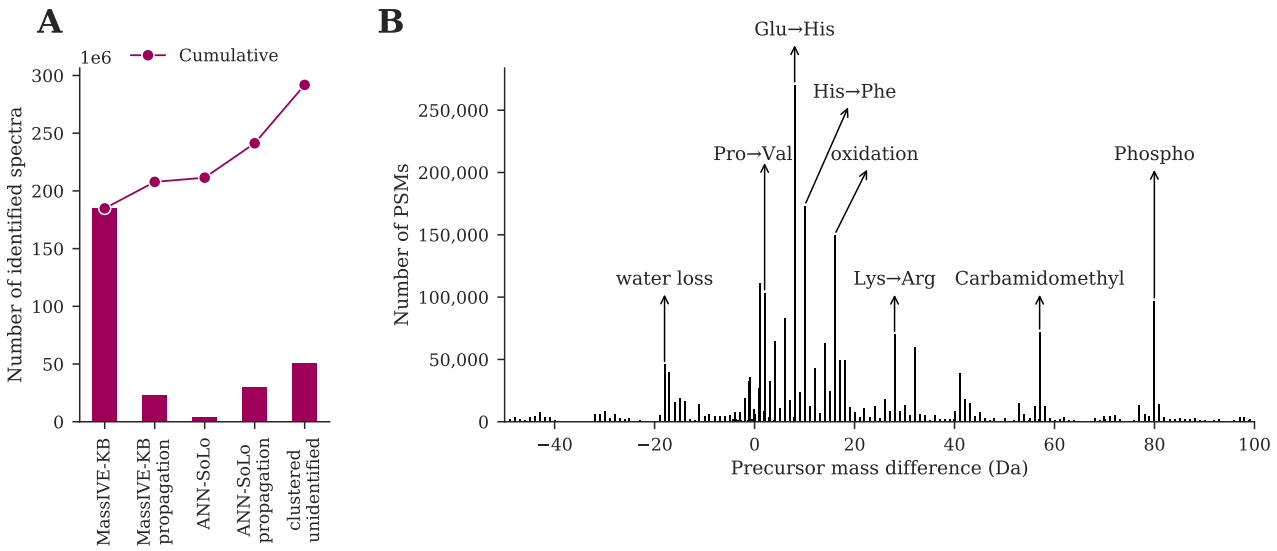

Figure 3: Exploration of the dark proteome using GLEAMS to process previously unidentified spectra. **(A)** GLEAMS succeeded in identifying 31% additional PSMs compared to the original MassIVE-KB results by performing targeted open modification searching of cluster medoid spectra and propagating peptide labels within clusters. **(B)** Precursor delta masses observed from open modification searching. Some of the most frequent delta masses are annotated with their likely modifications, sourced from Unimod.[24]

while ignoring confounding features (for example, peaks corresponding to a ubiquitous contaminant within a single study, boosting intra-study spectrum similarity). This hypothesis is supported by the observation that GLEAMS produces clusters with data drawn from more diverse studies, compared to clusters produced by the other tools (Figure 2F).

2.3 Elucidating the "dark proteome" of unidentified spectra

A key outstanding question in protein mass spectrometry analysis concerns the source of spectral "dark matter," i.e. spectra that are observed repeatedly across many experiments but consistently remain unidentified. Frank et al. [25] have previously used MS-Cluster to identify 4 million unknown spectra included in "spectral archives," and Griss et al. [9] have used spectra-cluster to obtain identifications for 9 million previously unannotated spectra in the PRIDE repository.

The original MassIVE-KB results[6] include identifications for 185 million PSMs out of 669 million MS/MS spectra (1% FDR), leaving a vast amount of spectral data unexplored. To characterize the unidentified spectra, we performed GLEAMS clustering (1% incorrectly clustered spectra) to group 193 million spectra in 17 million clusters. Among the clustered spectra, 86 million spectra were assigned a peptide label by MassIVE-KB, while 107 million spectra remained unidentified. The enrichment in identified spectra (45% of clustered spectra versus 28% of all spectra) indicates that, even though clustering is agnostic to peptide labels, it helps to extract repeatedly occurring, high-quality spectra that can be successfully identified.

We used a combination of strategies to process the unidentified spectra and explore the dark proteome (Figure 3A). First, peptide identifications were propagated within clusters. For clusters that consist of a mix of identified and unidentified spectra, the unidentified spectra were assigned the same peptide label as their identified cluster neighbors. In this fashion, 23 million PSMs could be identified.

Second, open modification searching (OMS) was used to investigate the remaining unidentified spectra. By using a large precursor mass window, spectra corresponding to peptides carrying any type of modification can be identified without explicitly having to specify the modifications of interest. However, this is a very computationally intensive task, due to the large increase in search space by opening up the precursor mass window. Therefore, only medoids of fully unidentified clusters were used, instead of having to perform OMS using all spectra. For 84 million clustered and unidentified spectra, 11 million medoid spectra were extracted and searched using the ANN-SoLo open modification spectral library search engine.[26,27] This search resulted in 3.7 million newly identified PSMs (1% FDR) corresponding to modified peptides. Finally, peptide labels of the identified medoid spectra were propagated to other cluster members, resulting in 30 million additional PSMs.

In total, these combined strategies succeeded in assigning peptides to 56 million previously unidentified PSMs, increasing the number of identified spectra by 31% (Figure 3A). Additionally, the OMS results provide information on the presence of PTMs in the human proteome (Figure 3B). Several of the observed precursor mass differences correspond to single amino acid substitutions, and modifications that were not considered in the original search settings were recovered. Besides abundant modifications that can be artificially introduced during sample processing, such as carbamidomethylation and oxidation, biologically relevant modifications from enrichment studies, such as phosphorylation, were frequently observed.

Importantly, given its large computational requirements, performing an open search at a repository scale is a highly challenging endeavor. However, using GLEAMS embedding and clustering, we succeeded in efficiently boosting the identification rate by up to a third. Notably, while propagating peptide labels from 86 million original MassIVE-KB PSMs resulted in an increase of 23 million PSMs, similarly propagating peptide labels obtained using OMS from only 3.7 million high-quality spectra filtered using GLEAMS achieved an increase of 30 million PSMs. Additionally, there are 51 million clustered spectra that remained unidentified. Because these spectra are repeatedly observed and expected to be high-quality, they likely correspond to true signals. Consequently, this is an extremely important collection of spectra to investigate using newly developed computational methods to further explore the dark proteome. Overall, these results demonstrate how GLEAMS efficiently enables in-depth analysis at a repository scale.

3 Discussion

We have demonstrated the utility of the 32-dimensional embedding learned by GLEAMS. By mapping spectra from diverse experiments into a common latent space, we can efficiently add an additional 31% to the identifications derived from database search. A key factor in GLEAMS' strong performance is its ability to efficiently operate on hundreds of millions to billions of spectra, corresponding to the size of an entire proteomics repository. Once the embedder is trained, new spectra representing previously unobserved peptides can be embedded and used for analysis without performing any expensive operations as long as they have sufficiently similar characteristics to the distribution of training spectra. Because the computational power required to find the nearest neighbors of a given spectrum in the embedded space increases only sub-linearly with the size of the repository, this task can smoothly scale to billions of vectors. Additionally, specialized hardware, such as graphics processing units, can be used to speed up nearest neighbor searching.[28] This makes it possible in principle to assign new spectra to spectrum clusters nearly instantaneously upon submission to a repository, giving researchers the immediate benefit of the combined analysis efforts of the entire proteomics community.

One caveat to the GLEAMS approach is that training the embedder relies upon the availability of peptide labels. A public repository typically contains datasets of varying quality and with varying types of analyses applied to them. The latter may even include invalid labels or labels not subjected to FDR control. Accordingly, we have exploited labels derived from systematic, repository-wide processing of the MassIVE database[6] to attempt to alleviate variability due to differences in analysis. This type of processing is expensive, but is hidden from the typical end user of GLEAMS, who will interact primarily with a pre-trained embedding network.

We hypothesize that the learned embedding can have potential utility beyond simply transferring identifications among nearby spectra. For example, it may be that semantic relationships among spectra generated by related molecular species can be derived from the latent space, in analogy to semantic relationships encoded by neural word embeddings.[18] If such relationships could be mapped, then it might be possible to, for instance, predict where in the embedded space a spectrum generated by a peptide with a particular PTM would be found, based on the known location of the unmodified species. It may also be possible to develop a joint embedding of peptide sequences and spectra, allowing arbitrary peptide sequences to be embedded. The embedded space could then be used like a search engine, assigning peptide identifications to spectra based on closeness to an embedded peptide sequence.

The embedding also opens up possibilities for transfer learning. For example, it may be possible to train a separate neural network to predict a spectrum's quality, or potential for being identified, from its location in embedded space, or to classify spectra as "chimeric" (generated by more than one peptide) or not. Furthermore, the GLEAMS embedding may have potential for applications at the level of the mass spectrometry runs or

entire experiments, using each experiment's embedded spectra to predict its tissue of origin or, in the case of metaproteomics experiments, taxonomic makeup.

A clear direction for future work is the development of statistical confidence estimation procedures suitable for this type of learned embedding. On the one hand, propagating peptide annotations between proximal pairs of spectra may risk introducing false positive assignments. On the other hand, when multiple identified spectra lie close to an unidentified spectrum, our confidence in such a propagation should intuitively increase relative to propagation with respect to a single spectrum–spectrum pair. Target-decoy methods for confidence estimation are widely used and provably correct under reasonable assumptions about the database search procedure,[29] but these methods do not generalize in a straightforward fashion to a method based on propagation in the GLEAMS embedded space.

Furthermore, the training procedure might be augmented by the inclusion of additional features as input to the embedder network. For example, retention time is a valuable feature to validate spectrum identifications. Similarly, it is sometimes used as a filter to minimize the number of incorrect spectrum groupings during clustering. Currently, we did not include this information because the training data originated from a wide diversity of experimental set-ups, including varied chromatography conditions. However, an exploration of retention time-related features to use during training of the GLEAMS neural network could prove a fruitful avenue of research to further boost its performance.

4 Methods

4.1 Encoding mass spectra for network input

Each spectrum is encoded as a vector of 3010 features of three types: precursor attributes, binned fragment intensities, and dot product similarities with a set of reference spectra.

Precursor mass, m/z, and charge are encoded as a combined 61 features. Precursor mass and m/z are each extremely important values for which precision is critical, and so they are poorly suited for encoding as single input features for a neural network. Accordingly, we experimented with several binary encodings of precursor mass and m/z, each of which gave superior performance on validation data than a real-value encoding, and settled on the encoding that gave moderately better performance than the others: a 27-bit "Gray code" binary encoding, in which successive values differ by only a single bit, preserving locality and eliminating thresholds at which many bits are flipped at once. Precursor values may span the range 400 Da to 6000 Da, so the Gray

code encoding has a resolution of 4×10^{-5} Da. Fragment values may span the range 50.5 m/z to 2500 m/z, so the Gray code encoding has a resolution of 2×10^{-5} m/z. Spectrum charge is one-hot encoded: seven features represent charge states 1–7, all of which are set to 0 except the charge corresponding to the spectrum (spectra with charge 8 or higher are encoded as charge 7).

Fragment peaks are encoded as 2449 features. Fragment intensities are square-root transformed and then normalized by dividing by the sum of the square-root intensities. Fragments outside the range 50.5 m/z to 2500 m/z are discarded, and the remaining fragments are binned into 2449 bins at 1.000 507 9 m/z, corresponding to the distance between the centers of two adjacent clusters of physically possible peptide masses,[30] with bins offset by half a mass cluster separation width so that bin boundaries fall between peaks. This bin size was chosen in order to accommodate data acquired using various instruments and protocols, and in deference to practical constraints on the number of input features for the deep learning approach. Consequently, it is unrelated to the optimal fragment mass tolerance for database search for a given run.

Similarities of each spectrum to an invariant set of reference spectra are encoded as 500 features. Each such feature is the normalized dot product between the given spectrum and one of an invariant set of 500 reference spectra chosen randomly from the training dataset. This can be considered as an "empirical kernel map",[31] allowing GLEAMS to represent the similarity between two spectra A and B by "paths" of similarities through each one of the reference spectra R via the transitive property; i.e., A is similar to B if A is similar to R and R is similar to B. In contrast to the fragment binning strategy described previously, similarities to the reference spectra are computed at native resolution. The 500 reference MS/MS spectra were selected from the training data by using submodular selection, as implemented in the apricot Python package (version 0.4.1).[32] First, 1000 peak files were randomly selected from the training data, containing 22 million MS/MS spectra that were downsampled to 200 000 MS/MS spectra. These spectra were used to compute a pairwise similarity matrix (normalized dot product with fragment m/z tolerance 0.05 m/z) that was used to perform submodular selection using the facility location function to select 500 representative reference spectra (Supplementary Figure 3).

4.2 Repository-scale MS/MS data

A large-scale, heterogeneous dataset derived from the MassIVE knowledge base (MassIVE-KB; version 2018-06-15)[6] was used to develop GLEAMS. As per Wang et al. [6], the MassIVE-KB dataset consists of 31 TB of human data from 227 public proteomics datasets. In total 28 155 peak files in the mzML[33] or mzXML format were downloaded from MassIVE, containing over 669 million MS/MS spectra.

All spectra were processed using a uniform identification pipeline during initial compilation of the MassIVE-KB

dataset.[6] MSGF+[34] was used to search the spectra against the UniProt human reference proteome database (version May 23, 2016).[35] Cysteine carbamidomethylation was set as a fixed modification, and variable modifications were methionine oxidation, N-terminal acetylation, N-terminal carbamylation, pyroglutamate formation from glutamine, and deamidation of asparagine and glutamine. MSGF+ was configured to allow one ^{13}C precursor mass isotope, at most one non-tryptic terminus, and 10 ppm precursor mass tolerance. The searches were individually filtered at 1% PSM-level FDR. A dynamic search space adjustment was performed during processing of the synthetic peptide spectra from the ProteomeTools project[36] and affinity purification mass spectrometry runs from the BioPlex project[37] to account for differences in sample complexity and spectral characteristics.[6] Next, the MassIVE-KB spectral library was generated using the top 100 PSMs for each unique precursor (i.e. combination of peptide sequence and charge), corresponding to 30 million high-quality PSMs (uniformly 0% PSM-level FDR from the original searches).[6]

The MSGF+ identification results for the full MassIVE-KB dataset were obtained from MassIVE in the mzTab format[38] and combined in a single metadata file containing 185 million PSMs. Additionally, information for the 30 million filtered PSMs to create the MassIVE-KB spectral library was independently retrieved.

4.3 Neural network architecture

The embedder network (Figure 1B) takes each of the three types of inputs separately. The precursor features are processed through a two-layer fully-connected network with layer dimensions 32 and 5. The 2449-dimensional binned fragment intensities are processed through five blocks of one-dimensional convolutional layers and max pooling layers inspired by the VGG architecture.[39] The first two blocks consist of two consecutive convolutional layers, followed by a max pooling layer. The third, fourth, and fifth blocks each consist of three consecutive convolutional layers, followed by a max pooling layer. The number of output filters of each of the convolutional layers is 30 for the first block, 60 for the second block, 120 for the third block, and 240 for the fourth and fifth blocks. All blocks use convolutional layers with convolution window length 3 and convolution stride length 1. All max pooling layers consist of pool size 1 and stride length 2. In this fashion the first dimension is halved after every block to ultimately convert the 2449x1 dimensional input tensor to a 71x240 dimensional output tensor. The 500-dimensional reference spectra features are processed through a two-layer fully-connected network with layer dimensions 750 and 250. The output of the three networks is concatenated and passed to a final, L2-regularized, fully-connected layer with dimension 32.

All network layers use the scaled exponential linear units (SELU) activation function.[40] The fully-connected layers are initialized using LeCun normal initialization,[41] and the convolutional layers are initialized using the Glorot uniform initialization.[42]

To train the embedder, we construct a "Siamese network" containing two instances of the embedder with tied weights W forming function G_W (Figure 1A). Pairs of spectra S_1 and S_2 are transformed to embeddings $G_W(S_1)$ and $G_W(S_2)$ in each instance of the Siamese network, respectively. The output of the Siamese network is the Euclidean distance between the two embeddings: $||G_W(S_1) - G_W(S_2)||_2$. The Siamese network is trained to optimize the following contrastive loss function:[21]

$$L(W, Y, S_1, S_2) = Y(\min(||G_W(S_1) - G_W(S_2)||_2), 1)^2$$
$$+(1 - Y)(\max(0, 1 - ||G_W(S_1) - G_W(S_2)||_2))^2 \ ,$$

where Y is the label associated with the pair of spectra S_1 and S_2.

4.4 Training the embedder

The GLEAMS model was trained using the 30 million high-quality PSMs used for compilation of the MassIVE-KB spectral library. PSMs were randomly split by their MassIVE dataset identifier so that the training, validation, and test sets consisted of approximately 80%, 10%, and 10% of all PSMs respectively (training set: 24 986 744 PSMs / 554 290 510 MS/MS spectra from 184 datasets; validation set: 2 762 210 PSMs / 30 386 035 MS/MS spectra from 11 datasets; test set: 2 758 019 PSMs / 84 699 214 MS/MS spectra from 24 datasets).

The Siamese neural network was trained using positive and negative spectra pairs. Positive pairs consist of two spectra with identical precursors, and negative spectra consist of two spectra that correspond to different peptides within a 10 ppm precursor mass tolerance with at most 25% overlap between their theoretical b and y fragments. In total 317 million, 205 million, 43 million, and 5 million positive training pairs were generated for precursor charges 2 to 5, respectively; and 8.347 billion, 3.263 billion, 182 million, and 5 million negative training pairs were generated for precursor charges 2 to 5, respectively.

The Siamese neural network was trained for 50 iterations using the rectified Adam optimizer[43] with learning rate 0.0002. Each iteration consisted of 40 000 steps with batch size 256. The pair generators per precursor charge and label (positive/negative) were separately shuffled and rotated to ensure that each batch consisted of an equal number of positive and negative pairs and balanced precursor charge states. After each iteration the performance of the network was assessed using a fixed validation set consisting of 512 000 spectrum pairs.

Training and evaluation were performed on a Intel Xeon Gold 6148 processor (2.4 GHz, 40 cores) with 768 GB memory and four NVIDIA GeForce RTX 2080 Ti graphics cards.

4.5 Phosphoproteomics embedding

An independent phosphoproteomics dataset by Hijazi et al. [23], generated to study kinase network topology, was used to evaluate the robustness of the GLEAMS embeddings for unseen post-translational modifications. All raw and mzIdentML[44] files were downloaded from PRIDE (project PXD015943) using ppx (version 1.1.1)[45] and converted to mzML files[33] using ThermoRawFileParser (version 1.3.4).[46] As per Hijazi et al. [23], the original identifications were obtained by searching with Mascot (version 2.5)[47] against the SwissProt database (SwissProt_Sep2014_2015_12.fasta), with search settings of up to two tryptic missed cleavages; precursor mass tolerance 10 ppm; fragment mass tolerance 0.025 Da; cysteine carbamidomethylation as a fixed modification; and N-terminal pyroglutamate formation from glutamine, methionine oxidation, and phosphorylation of serine, threonine, and tyrosine as variable modifications. The identification results included 3.7 million PSMs at 1% FDR (of which 98.5% are phosphorylated) for 18.6 million MS/MS spectra. All spectra were embedded with the previously trained GLEAMS model, and 1.185 billion positive pairs consisting of PSMs with identical peptide sequences and 293 million negative pairs consisting of PSMs with different sequences within a 10 ppm precursor mass tolerance were generated.

4.6 Density-based embedding clustering

Approximate nearest neighbor indexing and density-based clustering[12] were used to efficiently cluster the spectrum embeddings at a repository scale. First, the MS/MS spectra were converted to embeddings using the trained GLEAMS model. Next, the embeddings were split per precursor charge and partitioned into $1\,m/z$ buckets based on their corresponding precursor mass. The Faiss library was used for efficient similarity searching.[28] Faiss was used to construct an inverted index per m/z bucket by assigning the embeddings to centroids determined using k-means clustering. Conceptually, this corresponds to a Voronoi diagram, with each embedding assigned to its nearest representative centroid in the inverted index. The number of centroids to form the inverted index was dynamically set based on the number of embeddings in a bucket N. For buckets that consisted of up to one million embeddings $2^{\lfloor \log_2 \frac{N}{39} \rfloor}$ centroids were used; for buckets that consisted of up to ten million embeddings 2^{16} centroids were used; and for larger buckets 2^{18} centroids were used.

Next, the inverted index was used for efficient similarity searching. Instead of having to perform all pairwise embedding comparisons in each bucket to find each embedding's nearest neighbors, after mapping the embeddings to their Voronoi representatives they only needed to be compared to a limited number of embeddings in the inverted index. A maximum of 1024 lists in the inverted index were explored per query during searching, and for each embedding the Euclidean distances to its 50 nearest embeddings within a 10 ppm precursor mass window were stored in a sparse pairwise distance matrix.

This pairwise distance matrix was used to cluster the data using the DBSCAN algorithm.[20] Briefly, if a given number of embeddings are close to each other and form a dense data subspace, with closeness defined relative to a user-specified Euclidean distance threshold, then they are grouped in clusters. However, some clusters produced by DBSCAN violated the 10 ppm precursor mass tolerance because embeddings within a cluster can be connected through other embeddings with intermediate precursor mass. To avoid such false positives, the clusters reported by DBSCAN were postprocessed by hierarchical clustering with maximum linkage of the cluster members' precursor masses. In this fashion, some clusters are split into smaller, coherent clusters so that none of the embeddings in a single cluster have a pairwise precursor mass difference that exceeds the precursor mass tolerance.

An important advantage of this clustering approach is that the number of clusters is not required to be known in advance. Instead, DBSCAN is able to find clusters in dense regions, whereas embeddings in low-density regions, without a sufficient number of close neighbors, are marked as noise. Additionally, the approach is scalable: using the sparse pairwise distance matrix it was possible to efficiently process hundreds of millions of data points simultaneously.

4.7 Cluster evaluation

Four clustering algorithms—GLEAMS clustering, MaRaCluster,[10] MS-Cluster,[7] and spectra-cluster[8,9]—were run using a variety of parameter settings for each. For GLEAMS clustering Euclidean distance thresholds of 0.005, 0.01, 0.02, 0.03, 0.04, 0.05, 0.06, 0.07, 0.08, 0.09, and 0.10 were used. MaRaCluster (version 1.01)[10] was run with a precursor mass tolerance of 10 ppm, and with identical P-value and clustering thresholds −3.0, −5.0, −10.0, −15.0, −20.0, −25.0, −30.0, or −50.0. Other options were kept at their default values. MS-Cluster (version 2.00)[7] was run using its "LTQ_TRYP" model for three rounds of clustering with mixture probability 0.000 01, 0.0001, 0.001, 0.005, 0.01, 0.05, or 0.1. The fragment mass tolerance and precursor mass tolerance were 0.05 Da and 10 ppm, respectively, and precursor charges were read from the input files. Other options were kept at their default values. spectra-cluster (version 1.1.2)[8,9] was run in its "fast mode" for three rounds of clustering with the final clustering threshold 0.999 99, 0.9999, 0.999, 0.99, 0.95, 0.9, or 0.8. The fragment mass tolerance and precursor mass tolerance were 0.05 Da and 10 ppm, respectively. Other options were kept at their default values.

The clustering tools were evaluated using 85 million MS/MS spectra originating from 24 datasets in the test set. The spectra were split in three randomly generated folds, containing approximately 28 million MS/MS spectra each, and exported to MGF files for processing using the different clustering tools. To evaluate cluster quality, the mean performance over the three folds was used. Valid clusters were required to consist of minimum five

spectra, and smaller clusters (including singleton clusters) were considered as noise.

The following evaluation measures were used to assess cluster quality:

Clustered spectra. The number of spectra in non-noise clusters divided by the total number of spectra.

Incorrectly clustered spectra. The number of incorrectly clustered spectra in non-noise clusters divided by the total number of identified spectra in non-noise clusters. Spectra are considered incorrectly clustered if their peptide labels deviate from the most frequent peptide label in their clusters, with unidentified spectra not considered.

Completeness. Completeness measures the fragmentation of spectra corresponding to the same peptide across multiple clusters and is based on the notion of *entropy* in information theory. A clustering result that perfectly satisfies the completeness criterium (value "1") assigns all PSMs with an identical peptide label to a single cluster. Completeness is computed as one minus the conditional entropy of the cluster distribution given the peptide assignments divided by the maximum reduction in entropy the peptide assignments could provide.[48]

4.8 Clustering peptide annotation

GLEAMS was used to embed all 669 million spectra in the MassIVE-KB dataset and cluster the embeddings using DBSCAN. The Euclidean distance threshold was 0.013, clustering 193 million spectra (29%) with 0.97% incorrectly clustered spectra and 0.821 completeness.

To assign peptide labels to previously unidentified spectra, first peptide annotations were propagated within pure clusters. For 17 million clusters that contained a mixture of unidentified spectra and PSMs with identical peptide labels, the unidentified spectra were assigned the same label, resulting in 23 million new PSMs.

Second, open modification searching was used to process the unidentified spectra. Medoid spectra were extracted from clusters consisting of only unidentified spectra by selecting the spectra with minimum embedded distances to all other cluster members. This resulted in 11 million medoid spectra representing 84 million clustered spectra. The medoid spectra were split into two groups based on cluster size —size two and size greater than two— and exported to two MGF files.

Next, the ANN-SoLo[26,27] (version 0.3.3) spectral library search engine was used for open modification searching.

Search settings included preprocessing the spectra by removing peaks outside the 101 m/z to 1500 m/z range and peaks within a 1.5 m/z window around the precursor m/z, precursor mass tolerance 10 ppm for the standard searching step of ANN-SoLo's built-in cascade search and 500 Da for the open searching step, and fragment mass tolerance 0.05 m/z. Other settings were kept at their default values. As reference spectral library the MassIVE-KB spectral library was used. Duplicates were removed using SpectraST[49] (version 5.0 as part of the Trans-Proteomic Pipeline version 5.1.0[50]) by retaining only the best replicate spectrum for each individual peptide ion, and decoy spectra were added in a 1:1 ratio using the shuffle-and-reposition method.[51] PSMs were filtered at 1% FDR by ANN-SoLo's built-in subgroup FDR procedure.[52]

ANN-SoLo managed to identify 3.7 million PSMs (32% of previously unidentified cluster medoid spectra). Finally, peptide labels from the ANN-SoLo PSMs were propagated to other cluster members, resulting in 30 million additional PSMs.

4.9 Code availability

GLEAMS was implemented in Python 3.8. Pyteomics (version 4.3.2)[53] was used to read MS/MS spectra in the mzML,[33] mzXML, and MGF formats. spectrum_utils (version 0.3.4)[54] was used for spectrum preprocessing. Submodular selection was performed using apricot (version 0.4.1).[32] The neural network code was implemented using the Tensorflow/Keras framework (version 2.2.0).[55] Faiss (version 1.6.3)[28] was used for efficient similarity searching. Scikit-Learn (version 0.23.1)[56] was used for DBSCAN clustering, and fastcluster (version 1.1.28)[57] was used for hierarchical clustering. Additional scientific computing was done using NumPy (version 1.19.0),[58] SciPy (version 1.5.0),[59] Numba (version 0.50.1),[60] and Pandas (version 1.0.5).[61] Data analysis and visualization were performed using Jupyter Notebooks,[62] matplotlib (version 3.3.0),[63] Seaborn (version 0.11.0),[64] and UMAP (version 0.4.6).[22]

All code is available as open source under the permissive BSD license at `https://github.com/bittremieux/GLEAMS`. Code used to analyze the data and to generate the figures presented here is available on GitHub (`https://github.com/bittremieux/GLEAMS_notebooks`).

References

(1) Eng, J. K., McCormack, A. L., Yates, J. R. I. An Approach to Correlate Tandem Mass Spectral Data of Peptides with Amino Acid Sequences in a Protein Database. *Journal of the American Society for Mass Spectrometry* **1994**, *5*, 976–989, DOI: 10.1016/1044-0305(94)80016-2.

(2) Tabb, D. L. The SEQUEST Family Tree. *Journal of the American Society for Mass Spectrometry* **2015**, *26*, 1814–1819, DOI: 10.1007/s13361-015-1201-3.

(3) Perez-Riverol, Y., Csordas, A., Bai, J., Bernal-Llinares, M., et al. The PRIDE Database and Related Tools and Resources in 2019: Improving Support for Quantification Data. *Nucleic Acids Research* **2019**, *47*, D442–D450, DOI: 10.1093/nar/gky1106.

(4) Deutsch, E. W., Lam, H., Aebersold, R. PeptideAtlas: A Resource for Target Selection for Emerging Targeted Proteomics Workflows. *EMBO reports* **2008**, *9*, 429–434, DOI: 10.1038/embor.2008.56.

(5) Fenyö, D., Eriksson, J., Beavis, R. In *Computational Biology*, Fenyö, D., Ed.; Methods in Molecular Biology, Vol. 673; Humana Press: Totowa, NJ, 2010, pp 189–202.

(6) Wang, M., Wang, J., Carver, J., Pullman, B. S., et al. Assembling the Community-Scale Discoverable Human Proteome. *Cell Systems* **2018**, *7*, 412–421.e5, DOI: 10.1016/j.cels.2018.08.004.

(7) Frank, A. M., Bandeira, N., Shen, Z., Tanner, S., et al. Clustering Millions of Tandem Mass Spectra. *Journal of Proteome Research* **2008**, *7*, 113–122, DOI: 10.1021/pr070361e.

(8) Griss, J., Foster, J. M., Hermjakob, H., Vizcaíno, J. A. PRIDE Cluster: Building a Consensus of Proteomics Data. *Nature Methods* **2013**, *10*, 95–96, DOI: 10.1038/nmeth.2343.

(9) Griss, J., Perez-Riverol, Y., Lewis, S., Tabb, D. L., et al. Recognizing Millions of Consistently Unidentified Spectra across Hundreds of Shotgun Proteomics Datasets. *Nature Methods* **2016**, *13*, 651–656, DOI: 10.1038/nmeth.3902.

(10) The, M., Käll, L. MaRaCluster: A Fragment Rarity Metric for Clustering Fragment Spectra in Shotgun Proteomics. *Journal of Proteome Research* **2016**, *15*, 713–720, DOI: 10.1021/acs.jproteome.5b00749.

(11) Wang, L., Li, S., Tang, H. msCRUSH: Fast Tandem Mass Spectral Clustering Using Locality Sensitive Hashing. *Journal of Proteome Research* **2019**, DOI: 10.1021/acs.jproteome.8b00448.

(12) Bittremieux, W., Laukens, K., Noble, W. S., Dorrestein, P. C. Large-Scale Tandem Mass Spectrum Clustering Using Fast Nearest Neighbor Searching. *Rapid Communications in Mass Spectrometry* **2021**, e9153, DOI: 10.1002/rcm.9153.

(13) LeCun, Y., Bengio, Y., Hinton, G. Deep Learning. *Nature* **2015**, *521*, 436–444, DOI: 10.1038/nature14539.

(14) Tran, N. H., Zhang, X., Xin, L., Shan, B., et al. De Novo Peptide Sequencing by Deep Learning. *Proceedings of the National Academy of Sciences* **2017**, *114*, 8247–8252, DOI: 10.1073/pnas.1705691114.

(15) Tran, N. H., Qiao, R., Xin, L., Chen, X., et al. Deep Learning Enables de Novo Peptide Sequencing from Data-Independent-Acquisition Mass Spectrometry. *Nature Methods* **2018**, *16*, 63–66, DOI: 10.1038/s41592-018-0260-3.

(16) Gessulat, S., Schmidt, T., Zolg, D. P., Samaras, P., et al. Prosit: Proteome-Wide Prediction of Peptide Tandem Mass Spectra by Deep Learning. *Nature Methods* **2019**, *16*, 509–518, DOI: 10.1038/s41592-019-0426-7.

(17) Tiwary, S., Levy, R., Gutenbrunner, P., Salinas Soto, F., et al. High-Quality MS/MS Spectrum Prediction for Data-Dependent and Data-Independent Acquisition Data Analysis. *Nature Methods* **2019**, *16*, 519–525, DOI: 10.1038/s41592-019-0427-6.

(18) Mikolov, T., Chen, K., Corrado, G., Dean, J. Efficient Estimation of Word Representations in Vector Space https://arxiv.org/abs/1301.3781v3.

(19) Chen, T., Kornblith, S., Norouzi, M., Hinton, G. A Simple Framework for Contrastive Learning of Visual Representations http://arxiv.org/abs/2002.05709.

(20) Ester, M., Kriegel, H.-P., Sander, J., Xu, X. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining - KDD'96*, AAAI Press: Portland, OR, USA, 1996, pp 226–231.

(21) Hadsell, R., Chopra, S., LeCun, Y. In *Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - CVPR '06*, IEEE: New York, NY, USA, 2006; Vol. 2, pp 1735–1742, DOI: 10.1109/CVPR.2006.100.

(22) McInnes, L., Healy, J., Melville, J. UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction http://arxiv.org/abs/1802.03426.

(23) Hijazi, M., Smith, R., Rajeeve, V., Bessant, C., et al. Reconstructing Kinase Network Topologies from Phosphoproteomics Data Reveals Cancer-Associated Rewiring. *Nature Biotechnology* **2020**, *38*, 493–502, DOI: 10.1038/s41587-019-0391-9.

(24) Creasy, D. M., Cottrell, J. S. Unimod: Protein Modifications for Mass Spectrometry. *PROTEOMICS* **2004**, *4*, 1534–1536, DOI: 10.1002/pmic.200300744.

(25) Frank, A. M., Monroe, M. E., Shah, A. R., Carver, J. J., et al. Spectral Archives: Extending Spectral Libraries to Analyze Both Identified and Unidentified Spectra. *Nature Methods* **2011**, *8*, 587–591, DOI: 10.1038/nmeth.1609.

(26) Bittremieux, W., Meysman, P., Noble, W. S., Laukens, K. Fast Open Modification Spectral Library Searching through Approximate Nearest Neighbor Indexing. *Journal of Proteome Research* **2018**, *17*, 3463–3474, DOI: 10.1021/acs.jproteome.8b00359.

(27) Bittremieux, W., Laukens, K., Noble, W. S. Extremely Fast and Accurate Open Modification Spectral Library Searching of High-Resolution Mass Spectra Using Feature Hashing and Graphics Processing Units. *Journal of Proteome Research* **2019**, *18*, 3792–3799, DOI: 10.1021/acs.jproteome.9b00291.

(28) Johnson, J., Douze, M., Jégou, H. Billion-Scale Similarity Search with GPUs http://arxiv.org/abs/1702.08734.

(29) He, K., Fu, Y., Zeng, W.-F., Luo, L., et al. A theoretical foundation of the target-decoy search strategy for false discovery rate control in proteomics. *arXiv* **2015**, https://arxiv.org/abs/1501.00537.

(30) Wolski, W. E., Farrow, M., Emde, A.-K., Lehrach, H., et al. Analytical Model of Peptide Mass Cluster Centres with Applications. *Proteome Science* **2006**, *4*, 18, DOI: 10.1186/1477-5956-4-18.

(31) Hofmann, T., Schölkopf, B., Smola, A. J. Kernel Methods in Machine Learning. *The Annals of Statistics* **2008**, *36*, 1171–1220, DOI: 10.1214/009053607000000677.

(32) Schreiber, J., Bilmes, J., Noble, W. S. Apricot: Submodular Selection for Data Summarization in Python http://arxiv.org/abs/1906.03543.

(33) Martens, L., Chambers, M., Sturm, M., Kessner, D., et al. mzML—a Community Standard for Mass Spectrometry Data. *Molecular & Cellular Proteomics* **2011**, *10*, R110.000133–R110.000133, DOI: 10.1074/mcp.R110.000133.

(34) Kim, S., Pevzner, P. A. MS-GF+ Makes Progress towards a Universal Database Search Tool for Proteomics. *Nature Communications* **2014**, *5*, 5277, DOI: 10.1038/ncomms6277.

(35) Breuza, L., Poux, S., Estreicher, A., Famiglietti, M. L., et al. The UniProtKB Guide to the Human Proteome. *Database* **2016**, *2016*, bav120, DOI: 10.1093/database/bav120.

(36) Zolg, D. P., Wilhelm, M., Schnatbaum, K., Zerweck, J., et al. Building ProteomeTools Based on a Complete Synthetic Human Proteome. *Nature Methods* **2017**, DOI: 10.1038/nmeth.4153.

(37) Huttlin, E. L., Ting, L., Bruckner, R. J., Gebreab, F., et al. The BioPlex Network: A Systematic Exploration of the Human Interactome. *Cell* **2015**, *162*, 425–440, DOI: 10.1016/j.cell.2015.06.043.

(38) Griss, J., Jones, A. R., Sachsenberg, T., Walzer, M., et al. The mzTab Data Exchange Format: Communicating Mass-Spectrometry-Based Proteomics and Metabolomics Experimental Results to a Wider Audience. *Molecular & Cellular Proteomics* **2014**, *13*, 2765–2775, DOI: 10.1074/mcp.O113.036681.

(39) Simonyan, K., Zisserman, A. Very Deep Convolutional Networks for Large-Scale Image Recognition http://arxiv.org/abs/1409.1556.

(40) Klambauer, G., Unterthiner, T., Mayr, A., Hochreiter, S. Self-Normalizing Neural Networks http://arxiv.org/abs/1706.02515.

(41) LeCun, Y. A., Bottou, L., Orr, G. B., Müller, K.-R. In *Neural Networks: Tricks of the Trade*, Montavon, G., Orr, G. B., Müller, K.-R., Eds.; Lecture Notes in Computer Science, Vol. 7700; Springer Berlin Heidelberg: Berlin, Heidelberg, 2012, pp 9–48.

(42) Glorot, X., Bengio, Y. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, ed. by Teh, Y. W., Titterington, M., JMLR Workshop and Conference Proceedings: Chia Laguna Resort, Sardinia, Italy, 2010; Vol. 9, pp 249–256.

(43) Liu, L., Jiang, H., He, P., Chen, W., et al. On the Variance of the Adaptive Learning Rate and Beyond http://arxiv.org/abs/1908.03265.

(44) Jones, A. R., Eisenacher, M., Mayer, G., Kohlbacher, O., et al. The mzIdentML Data Standard for Mass Spectrometry-Based Proteomics Results. *Molecular & Cellular Proteomics* **2012**, *11*, M111.014381–M111.014381, DOI: 10.1074/mcp.M111.014381.

(45) Fondrie, W. E., Bittremieux, W., Noble, W. S. ppx: Programmatic Access to Proteomics Data Repositories. *Journal of Proteome Research* **2021**, *20*, 4621–4624, DOI: 10.1021/acs.jproteome.1c00454.

(46) Hulstaert, N., Shofstahl, J., Sachsenberg, T., Walzer, M., et al. ThermoRawFileParser: Modular, Scalable, and Cross-Platform RAW File Conversion. *Journal of Proteome Research* **2020**, *19*, 537–542, DOI: 10.1021/acs.jproteome.9b00328.

(47) Perkins, D. N., Pappin, D. J. C., Creasy, D. M., Cottrell, J. S. Probability-Based Protein Identification by Searching Sequence Databases Using Mass Spectrometry Data. *Electrophoresis* **1999**, *20*, 3551–3567, DOI: 10.1002/(SICI)1522-2683(19991201)20:18<3551::AID-ELPS3551>3.0.CO;2-2.

(48) Rosenberg, A., Hirschberg, J. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL) - CoNLL-EMNLP 2007*, Association for Computational Linguistics: Prague, Czech Republic, 2007, pp 410–420.

(49) Lam, H., Deutsch, E. W., Eddes, J. S., Eng, J. K., et al. Development and Validation of a Spectral Library Searching Method for Peptide Identification from MS/MS. *PROTEOMICS* **2007**, *7*, 655–667, DOI: 10.1002/pmic.200600625.

(50) Deutsch, E. W., Mendoza, L., Shteynberg, D., Farrah, T., et al. A Guided Tour of the Trans-Proteomic Pipeline. *PROTEOMICS* **2010**, *10*, 1150–1159, DOI: 10.1002/pmic.200900375.

(51) Lam, H., Deutsch, E. W., Aebersold, R. Artificial Decoy Spectral Libraries for False Discovery Rate Estimation in Spectral Library Searching in Proteomics. *Journal of Proteome Research* **2010**, *9*, 605–610, DOI: 10.1021/pr900947u.

(52) Fu, Y., Qian, X. Transferred Subgroup False Discovery Rate for Rare Post-Translational Modifications Detected by Mass Spectrometry. *Molecular & Cellular Proteomics* **2014**, *13*, 1359–1368, DOI: 10.1074/mcp.O113.030189.

(53) Levitsky, L. I., Klein, J. A., Ivanov, M. V., Gorshkov, M. Pyteomics 4.0: Five Years of Development of a Python Proteomics Framework. *Journal of Proteome Research* **2019**, *18*, 709–714, DOI: 10.1021/acs.jproteome.8b00717.

(54) Bittremieux, W. spectrum_utils: A Python Package for Mass Spectrometry Data Processing and Visualization. *Analytical Chemistry* **2020**, *92*, 659–661, DOI: 10.1021/acs.analchem.9b04884.

(55) Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, et al. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems, Software available from tensorflow.org, 2015.

(56) Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., et al. Scikit-Learn: Machine Learning in Python. *Journal of Machine Learning Research* **2011**, *12*, 2825–2830.

(57) Müllner, D. Fastcluster: Fast Hierarchical, Agglomerative Clustering Routines for R and Python. *Journal of Statistical Software* **2013**, *53*, DOI: 10.18637/jss.v053.i09.

(58) Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., et al. Array Programming with NumPy. *Nature* **2020**, *585*, 357–362, DOI: 10.1038/s41586-020-2649-2.

(59) SciPy 1.0 Contributors, Virtanen, P., Gommers, R., Oliphant, T. E., et al. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods* **2020**, DOI: 10.1038/s41592-019-0686-2.

(60) Lam, S. K., Pitrou, A., Seibert, S. In *Proceedings of the Second Workshop on the LLVM Compiler Infrastructure in HPC - LLVM '15*, ACM Press: Austin, TX, USA, 2015, pp 1–6, DOI: 10.1145/2833157.2833162.

(61) McKinney, W. In *Proceedings of the 9th Python in Science Conference*, ed. by van der Walt, S., Millman, J., Austin, Texas, USA, 2010, pp 51–56.

(62) Thomas, K., Benjamin, R.-K., Fernando, P., Brian, G., et al. In *Positioning and Power in Academic Publishing: Players, Agents and Agendas*; IOS Press: 2016, pp 87–90.

(63) Hunter, J. D. Matplotlib: A 2D Graphics Environment. *Computing in Science & Engineering* **2007**, *9*, 90–95, DOI: 10.1109/MCSE.2007.55.

(64) Waskom, M., the seaborn development team mwaskom/seaborn, https://doi.org/10.5281/zenodo.592845, version latest, 2020, DOI: 10.5281/zenodo.592845.