Journal of Proteomics 247 (2021) 104316

Contents lists available at ScienceDirect

Journal of Proteomics

journal homepage: www.elsevier.com/locate/jprot

Deep learning for peptide identification from metaproteomics datasets

Shichao Feng [a], Ryan Sterzenbach [b], Xuan Guo [a,*]

[a] Department of Computer Science and Engineering, University of North Texas, TX, USA
[b] Department of Biomedical Engineering, University of North Texas, TX, USA

ARTICLE INFO

Keywords:
Peptide identification
Deep learning
Tandem mass spectrometry
CNN

ABSTRACT

Metaproteomics is becoming widely used in microbiome research for gaining insights into the functional state of the microbial community. Current metaproteomics studies are generally based on high-throughput tandem mass spectrometry (MS/MS) coupled with liquid chromatography. In this paper, we proposed a deep-learning-based algorithm, named DeepFilter, for improving peptide identifications from a collection of tandem mass spectra. The key advantage of the DeepFilter is that it does not need ad hoc training or fine-tuning as in existing filtering tools. DeepFilter is freely available under the GNU GPL license at https://github.com/Biocomputing-Research-Group/DeepFilter.

Significance: The identification of peptides and proteins from MS data involves the computational procedure of searching MS/MS spectra against a predefined protein sequence database and assigning top-scored peptides to spectra. Existing computational tools are still far from being able to extract all the information out of MS/MS data sets acquired from metaproteome samples. Systematical experiment results demonstrate that the DeepFilter identified up to 12% and 9% more peptide-spectrum-matches and proteins, respectively, compared with existing filtering algorithms, including Percolator, Q-ranker, PeptideProphet, and iProphet, on marine and soil microbial metaproteome samples with false discovery rate at 1%. The taxonomic analysis shows that DeepFilter found up to 7%, 10%, and 14% more species from marine, soil, and human gut samples compared with existing filtering algorithms. Therefore, DeepFilter was believed to generalize properly to new, previously unseen peptide-spectrum-matches and can be readily applied in peptide identification from metaproteomics data.

1. Introduction

Metaproteomics focuses on the entire protein complement recovered directly from complex microbial communities like aqueous ecosystems, terrestrial systems, and eukaryotic host microbiomes [1,2,3,4]. Understanding the functionality of microbial communities is essential. For example, the gut microbiome was known to play a crucial role in health by benefiting the immune system and helping control digestion [5,6,7]. The microbial activities can be inferred from the total proteins of its constituent microorganisms. Mass spectrometry (MS)-based metaproteomics has emerged as a discovery method for analyzing the proteome of a microbial community in a high-throughput fashion. In shotgun MS-based metaproteomics, proteins are digested into peptides using high-performance liquid chromatography (HPLC), then ionized, isolated, fragmented, and detected in the mass analyzer as they elute from the HPLC. The central component of computational metaproteomics data analysis is database searching. This is where measured tandem mass spectra (MS/MS), of unknown microbial peptides, are compared with theoretical tandem mass spectra predicted from a database of proteins encoded in metagenomes. Peptide-spectrum match (PSM) scores are calculated by the comparisons between each MS/MS and in-silico digested peptides from the protein database. The peptide in the top-scoring PSM is used as a candidate for the query MS/MS. The candidate PSMs are filtered with a score threshold to generate a set of confident PSMs at a designed false discovery rate (FDR).

In the database matching procedure, it is crucial to choose an appropriate PSM scoring function which plays two important roles. First, the scoring function is used to rank candidate peptides for a single spectrum, producing a top-scoring PSM for each spectrum. Second, the scoring function is used to rank the PSMs from different spectra. The second ranking task is intrinsically more complicated than the first ranking task due to the variance of spectra. A perfect scoring function for ranking top-scoring PSM per spectrum may not be a perfect scoring function for ranking PSMs from different spectra because PSM scoring may not be well-calibrated from one spectrum to the other. To solve this issue, a variety of approaches have been developed to learn PSM scoring

* Corresponding author at: Dept of Computer Science and Engineering, University of North Texas, 3940 N. Elm Street Ste. F290, Denton, TX 76207-7102, USA.
E-mail address: xuan.guo@unt.edu (X. Guo).

https://doi.org/10.1016/j.jprot.2021.104316
Received 14 March 2021; Received in revised form 2 June 2021; Accepted 18 June 2021
Available online 8 July 2021

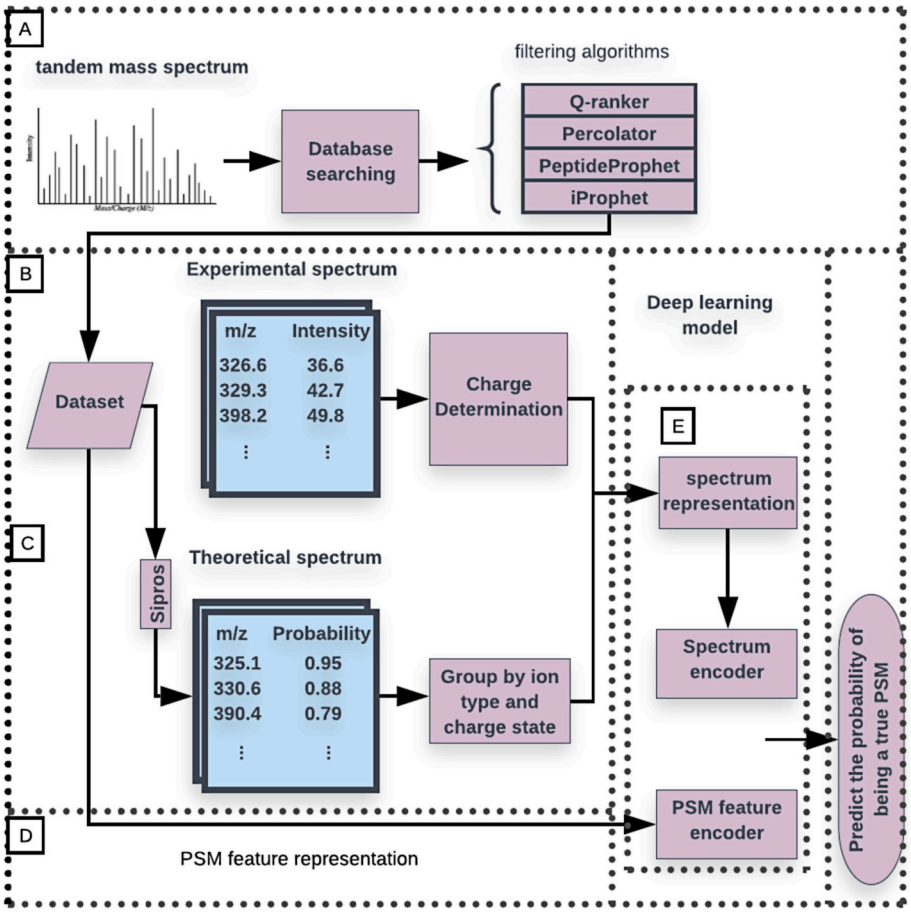


Fig. 1. The workflow of building DeepFilter.

functions for the second ranking task after the initial PSM scoring. These approaches can be categorized into two types. The first type is based on statistical modeling [8]. For instance, PeptideProphet [9] determines the confidence of identified PSMs by a probability-based model using Bayes' Law. Other statistical modeling methods, such as linear discriminant analysis [10] and Bayes classifier [11] were also used as discriminant functions. Ivanov et al. [12] designed a multiple-parameter scoring scheme to find PSM outliers to estimate the distributions of PSMs with information from the experimental spectra. iProphet [13] implemented five models based on the number of sibling searches, replicate spectra, sibling experiments, sibling ions, and sibling modifications to filter PSMs. The second type of PSM filtering algorithms discriminate true PSMs from false ones based on machine learning [14,15,16,17,18,19,20]. Percolator [16], Q-ranker [19], and CRanker [20] belong to the second type. They train Supporting Vector Machines to classify PSMs. Other machine learning models, such as decision tree [14], random forest [15], Bayesian network models [17], and logistic regression [18] were also applied to re-rank or re-score PSMs with different strategies in constructing training data sets and feature extraction.

Although the above-mentioned methods improved the number of identified PSMs for single organism proteome, there are still more than 50% of spectra without correctly assigned peptides in MS-based metaproteomics [21,22,23]. One reason for that is the large metaproteomic protein databases, which may contain millions of predicted proteins spanning thousands of organisms in complex communities [24,25]. Also, the scores of random matches generally follow a probabilistic distribution with a small tail towards high scores. A spectrum should have a high score for a correctly matched peptide. As a result, when the databases of candidate peptides increase in size, the probability of an incorrect random match that scores higher than the correct match increases as well. Therefore, a more sensitive ranking strategy is needed for ranking PSMs from different spectra with the properties of spectra and peptide sequences being taken into account. Another drawback of the existing PSM filtering algorithms is that they often do not generalize well across different metaproteome samples and experimental conditions, such as different instrument platforms, etc. For solving this issue, ad hoc training is required when the samples and experimental conditions change. Therefore, it will be hard to justify the confidence of results when training data is from the one needs to be inferred.

In this study, we propose a deep learning model, called DeepFilter, to re-rank PSM candidates after the database search for shotgun metaproteomics. DeepFilter has two key contributions. First, it can learn the mapping patterns between spectra and peptide sequences and combine them with the features known to relate to the PSM score distribution. These automatically extracted features enabled DeepFilter to produce substantially higher numbers of PSM, peptide, and protein identifications in complex metaproteomics data sets than the existing algorithms benchmarked here. Second, DeepFilter eliminates the ad hoc training and can be applied to analyze different metaproteome samples without fine-tuning and still obtain substantial improvements over existing tools. The rest of the paper is organized as follows. In section 2, we elaborate on the architecture of DeepFilter and the whole workflow, including training data set construction, spectrum peak charge detection, and feature extraction. In section 3, systematic experiments on five real-world metaproteomes and single organism proteome were used to demonstrate that our method outperformed the other state-of-the-art approaches. In section 4, we visualized the learned features in DeepFilter by using class activation mappings. In section 5, we concluded that DeepFilter not only achieves higher identification performance but also can be generalized to different metaproteomic studies.

Table 1
The total numbers of MS/MS of nine metaproteome data sets.

	Marine 1	Marine 2	Marine 3	Soil 1	Soil 2	Soil 3	P1	P2	HG
# of spectra	138,682	143,344	127,075	391,249	489,785	409,202	356,160	351,658	668,162

1. P1 and P2 are two mock metaproteome samples.
2. HG is the human gut metaproteome sample.

Table 2
The numbers of postive and negative PSMs in training data sets.

	Marine 1	Marine 2	Marine 3	Soil 1
# of positive PSMs	79,057	77,916	79,496	252,829
# of negative PSMs	160,562	153,692	132,051	1,702,530

2. Materials and methods

The workflow of building DeepFilter is shown in Fig. 1, which includes five steps: training data set construction (Part A), charge detection for experimental spectra (Part B), theoretical isotopic envelope generation (Part C), 11 features extraction (Part D), and feature/spectrum encoding (Part E). In the following sections, We will explain the details of each component. DeepFilter is freely available under the GNU GPL license at https://github.com/Biocomputing-Research-Group/DeepFilter, where step-by-step installation and usage were provided. In short, DeepFilter needs the mass spectra data in ms2 format and the database searching results by Comet in pin format, and generates re-ranked PSMs in a tab-separated values file. Note that DeepFilter needs 20 GB GPU memory for training.

2.1. Training data construction

There are nine data sets used in our experiments. The summary of spectrum numbers is in Table 1. Marine 1, 2, and 3 are metaproteomes of marine microbial communities [26]. Soil 1, 2, and 3 are metaproteomes of soil microbial communities [27]. P1 and P2 are metaproteomoes of mock community [28]. HG is the metaproteome of human hut microbial community [29]. Marine data sets and one of soil data sets were used to construct our training data set, and others are for benchmarking.

Since the metaproteome data sets do not have ground-truth PSMs, we used existing algorithms to generate a set of PSMs as positive data points. We used Comet [30] to collect a set of top-scoring PSM candidates for each spectrum, and re-scored these PSMs by different filters, including Percolator [16], Q-ranker [19], PeptideProphet [13], and iProphet [13]. We chose the one that identified the highest number of identified PSMs with FDR controlled by the target-decoy search [31]. In our experiments, Percolator performed best and was chosen for generating positive PSMs. We generated a training data set for each Marine data set. Take Marine 1 for example. After the Comet search and the Percolator filtering, top-5 scoring PSM candidates for each spectrum were collected. The PSM candidates with posterior error probabilities (calculated by Percolator) larger than 0.93 were removed. A total of 243,928 PSM candidates were left. For the top-ranked PSM candidates, if they were target PSMs from the matched protein database, we labeled them as positive PSMs. For the rest PSMs, including all decoy PSMs from the decoy protein database and non-top-ranked target PSMs, we labeled them as negative PSMs. The number of positive and negative PSMs for four training data sets are shown in the Table 2.

2.2. Charge detection for experimental mass spectra

To provide more information to the spectrum encoder, each experimental spectrum was deconvoluted with charge states assigned to each fragment peak. Not all the MS data comes with charge information. Here, we developed a charge detection method based on Patterson routine algorithm [32]. We assigned charges up to +3 and, for the fragment ions with charges more than +3, we put them into one group without further categorization. The equation of charge detection is shown in Equ. 1, where ΔM represents the mathematical inverse of charge state which is being evaluated, M_i represents the m/z of nearby fragment peaks, and $f(M_i)$ is the intensity of corresponding peak. The details of determining the charge state for each fragment peak is described in Algorithm 1.

Algorithm 1: Charge detection

Input: MS2 file
Output: File containing fragment peaks with predicted charge states

```
for scan in MS2 file do
    Initialization:
    Group_{charge=i}: the list to store the fragment peak by their detected charge, i means the detected charge for
      the m/z. If i equals to OtherStates, it means that detected charge state is not +1, +2, +3.
    for peak in scan do
        Score: the list to store calculation score for each evaluated charge
        for charge in range (1 to 3) do
            PattersonRoutineFunction(peak, charge) ∈ Score
        if ∃ S ∈ Score which is the highest then
            cs= the corresponding detected charge state of S
            peak → Group_{charge=cs}
        else
            peak → Group_{charge=OtherStates}
```

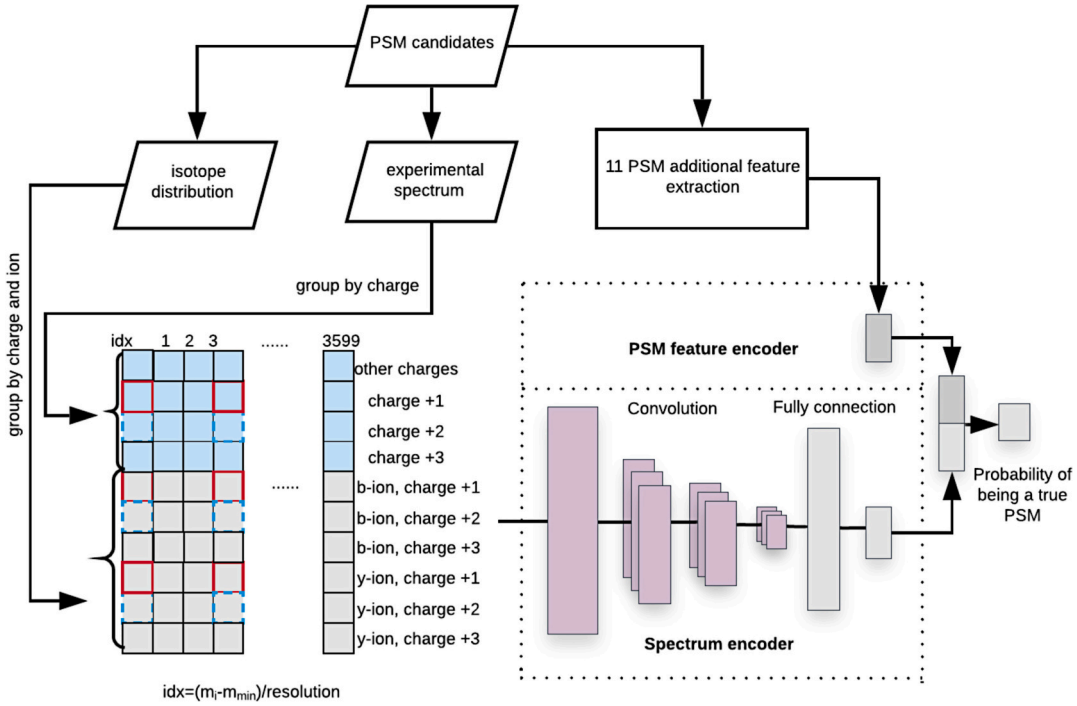


Fig. 2. The architecture of DeepFilter model.

$$P(\Delta M) = \sum_{i=1}^{k} f\left(M_i - \frac{\Delta M}{2}\right) * f\left(M_i + \frac{\Delta M}{2}\right) \quad (1)$$

2.3. Isotope envelope generation for peptide sequences

In addition to using the most abundant theoretical peaks of peptide sequences as in Comet and other database searching tools, we also generated the theoretical spectrum with isotope envelopes for each fragment ions. Here, we modified our open-source tool, Sipros [33], to obtain the isotope envelopes for the peptide sequences in the training PSMs. For each ion, we sorted the isotopes in descending order based on their abundances and keep the isotopes until the cumulative isotopic abundance no less than 98%. We then clustered fragment ions into six groups by considering 3 charge states, i.e., +1, +2, and + 3, and two ion types, i.e., b-ion and y-ion.

2.4. Input representations of PSMs and engineered features

Each input PSM was converted into a matrix, where the peaks in the experimental spectrum and theoretical spectrum were discretized based on their *m/z* values, and grouped based on their charge states and ion types. This PSM matrix was fed into a spectrum encoder based on the CNN (Convolutional Neural Network) model, which uses convolution kernels to construct a shared weight architecture. Inspired by the existing filtering algorithms [16,19,9], we extracted 11 features for each input PSM and encoded them by a PSM feature encoder based on a fully connected layer. The details of these two input representations are described as follows.

2.4.1. Spectrum representation

Our spectrum representation is a matrix constructed by peaks. The column index indicates the *m/z* value, and the row index indicates the ion types and the charge states. An example is shown in Fig. 2. We used 0.5 Da as a *resolution* parameter and considered the *m/z* values ranged from 100 Da to 1900 Da. We then constructed an 10 × 3600 matrix, where the first 4 rows are for the fragment ions in charge +1, +2, +3, and above in the experimental spectrum, and the rest 6 rows are for the predicted b-ions and y-ions in charge +1, +2, +3 from the peptide sequence, respectively. The column index was calculated as $index = (m_i - m_{min})/resolution$, where m_i is the *m/z* value of *i*th peak, m_{min} is the minimum *m/z* value considered, which is 100 Da. If the *m/z* value of a peak is 421 Da, then its intensity value is filled in the cell in 642^{nd} column. The intensities in the experimental spectrum were used to fill the first four rows, and the abundances of theoretical spectrum were used to fill the rest six rows. An L_2 normalization was applied to the input matrix before the CNN model calculation.

Table 3
11 PSM features used in DeepFilter.

Feature Index	Feature Name	Feature Description
1	*Xcorr*	Cross correlation between theoretical and observed spectra
2	ΔC_n	Fractional difference between current and second best XCorr
3	ΔC_n^L	Fractional difference between current and the fifth best XCorr
4	*Mass*	The observed mass $[M + H]^+$
5	ΔM	The difference in calculated and observed mass
6	*abs*(ΔM)	The absolute value of the difference in calculated and observed mass
7	*pepLen*	The length of the matched peptide, in residues
8	*enzInt*	Number of missed internal enzymatic (tryptic) sites
9–11	*charge* 1–3	Three Boolean features indicating the charge state

2.4.2. PSM feature representation

In addition to the features from CNN model, DeepFilter also used another 11 features extracted from the initial PSM score, the observed spectrum, and the peptide sequence for each input PSM. These features are shown in Table 3.

2.5. DeepFilter model architecture

The architecture of our DeepFilter model is in Fig. 2. It has two encoders, i.e., spectrum encoder and PSM feature encoder. The

representations from the spectrum encoder and PSM feature encoder were concatenated together into a 1024-dimension vector and fed into a fully connected layer with the softmax activation function. The output is the probability from 0 to 1 to indicate how likely a PSM candidate is a true match. We used a modified cross-entropy loss multiplied by the weights that indicate the probability of PSM being a true match. The detail of each encoder and loss function are described as follows.

2.5.1. Spectrum encoder

The spectrum encoder consists of four dilated convolutional layers and two fully connected layers. To grab features between experimental and theoretical representations within the same charge state, we set the dilation rate to be 3 as highlighted in the red boxes of the kernels of CNN in Fig. 2. For the four dilated convolutional layers, we used 16 kernels for each layer, and in each convolutional layer, we used different kernel sizes, which are (3,7), (2,5), (2,6), (2,6), respectively. We used max-pooling with (1,2) kernel size after each convolution operation. To speed up the calculation and avoid the over-fitting issue, we applied batch-normalization for each convolutional layer and added a dropout layer after the last convolutional layer, with the dropout rate being 0.5. In the first fully connected layer, the input dimension is 3504, and the hidden units are 1024. For the second fully connected layer, the input dimension is 1024, and the dimension of output vector is 512, which is activated by ReLU function, and used as the representation of spectrum for the next PSM classification task.

2.5.2. PSM feature encoder

The 11 PSM features were given to the PSM feature encoder made of a single fully connected layer. The input dimension for this layer is 11 with ReLU as the activation function. And the output is a 512-dimension vector. This vector was used as the representation of 11 PSM features.

2.5.3. Loss function

The scoring model is a binary classifier. We applied a modified cross-entropy loss function as in Equ. 2 by incorporating the posterior error probability (*pep*) calculated by Percolator. p_i is the predicted probability that *i*th PSM is a correct match, and pep_i is the posterior error probability. This modified loss function achieved a better number of identified PSMs than the classical cross-entropy loss did (data is not shown here).

$$Loss = -\sum[pep_i log p_i + (1 - pep_i) log(1 - p_i)] \quad (2)$$

2.5.4. Training DeepFilter

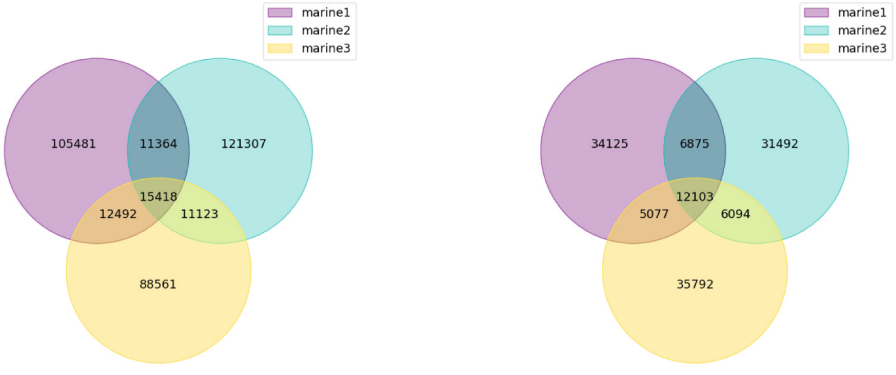
Here, we would like to emphasize some important techniques for training DeepFilter. DeepFilter was implemented using PyTorch version 1.4.0 and trained in a workstation with 8 GeForce RTX 2080 Ti GPUs. We randomly split the data sets into training and validation data sets with a ratio of 9 to 1. For the training data sets, we set the mini-batch size to 256. We applied backward propagation to get the gradient in each mini-batch and save the model as a checkpoint when the performance improved based on the accuracy calculated based on the validation data sets. We set the epochs as 150 to ensure the convergence and performed Adam optimizer, whose learning rate and weight decay were set to 1e-4.

3. Experiments and results

3.1. Experimental design

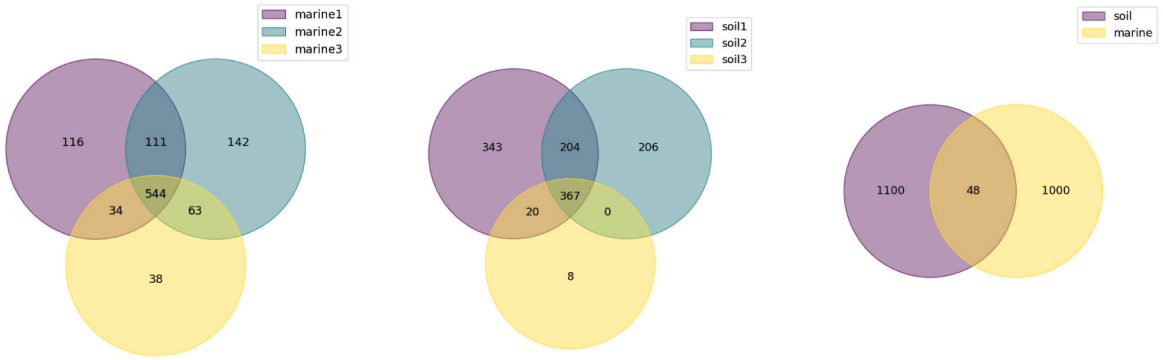
We evaluated the performance of DeepFilter using three metaproteome data sets from soil communities [27], three metaproteome data sets from marine communities [26], and one *E. coli* proteome data set. The summary of these data sets is in Table 1. These (meta)proteomes were all measured using the Multidimensional Protein Identification Technology (MudPIT) approach [34] on an LTQ Orbitrap Elite mass spectrometer (Thermo Scientific). Their matched metagenomes were used to construct a soil protein database with 3.4 million target proteins and a marine protein database with 392,000 target proteins. The mock community protein database contains 123,100 target proteins, and the human gut protein database has more than 4.9 million target proteins. The MS data and protein databases for marine and soil metaproteome are available from the ProteomeXchange Consortium via the PRIDE repository with the data set identifier of PXD007587. Details on these benchmarking data sets are described in our previous study [35]. The MS data and protein databases for human gut and mock community are provided through the PRIDE repository PXD006118 [28] and PXD013386 [29], respectively.

DeepFilter was compared with four state-of-the-art filtering algorithms, including Percolator [16], Q-ranker [19], PeptideProphet [8], and iProphet [13]. We did not compare with other filtering algorithms because they are either not available or outperformed by the tools mentioned above for metaproteome analysis. Percolator, Q-ranker, and PeptideProphet used the PSMs scored by Comet. iProphet used the PSMs



(a) The composition of peptide for PSMs in marine data sets (b) The composition of peptide for positive PSMs in marine data sets

Fig. 3. The composition of peptide for PSMs in marine data sets.



(a) The taxonomic composition overlap within marine metaproteome data sets

(b) The taxonomic composition overlap within soil metaproteome data sets

(c) The taxonomic composition overlap between marine and soil metaproteome data sets

Fig. 4. The taxonomic composition overlap between marine and soil data sets at species level.

scored by PeptideProphet. Because iProphet used the features at peptide and protein levels, which may cause the machine learning to share information among PSMs for discrimination and destroy the independence among the PSMs [36], we employed the iProphet without the features at the peptide and protein levels, including the number of sibling ions (NSI), the number of sibling modifications (NSM), and the number of sibling peptides (NSP). We also used iProphet by enabling above features and the results are in Supplementary Table 2 in the supplementary document. This version of iProphet achieved comparable results to DeepFilter at the PSM level but gave less protein identifications.

Benchmarking datasets were searched using Comet 2018.01 rev. 2. The database searching results were filtered by Percolator version 3.03.0, Q-ranker from Crux toolkit version 3.2, PeptideProphet, and iProphet from TPP v5.2.0 with default configuration settings, respectively. The following parameters were used: precursor mass tolerance set to 0.09 Da, fragment mass tolerance set to 0.01 Da, peptide mass range set from 700 Da to 7000 Da, Trypsin/P used for enzyme, and the allowed number of missing cleavages set to three. The protein FASTA files were from the PRIDE repository, where the studies provided both the mass spectrum data and the protein databases. The PSM filtering was executed on a desktop computer with one 2.3 GHz Intel(R) Xeon(R) Gold 5118 CPU and 32 GB memory.

The performance metrics include the number of identified target PSMs, peptides, and proteins with FDR controlled at different levels (where FDR was estimated by the target-decoy strategy) [37]. For each observed spectrum, only the top-scoring PSM was used for estimating FDR. The score threshold was adjusted to reach a user-defined FDR. The FDR is calculated as follows.

$$FDR = \frac{\#\ \text{Decoy PSMs/peptides/proteins}}{\#\ \text{Target PSMs/peptides/proteins}} \tag{3}$$

For peptide and protein level FDRs, we adjust the PSM level score threshold to control FDRs. For example, suppose the protein FDR is higher than the user-defined value. In that case, we will use a high score threshold to remove more decoy PSMs, which will reduce the more decoy proteins than target proteins, thus lower the protein FDR. We applied the same FDR control strategy for all the tested tools.

3.2. Performance comparison of DeepFilter on marine microbial complex

Rotation training and testing were applied here. More specifically, each marine data set was used to create a training data set for DeepFilter, and the remaining marine data sets were used to test the performance between DeepFilter and the other five existing filtering tools. Details on the execution of these algorithms are described in the supplementary document. Although these MS data sets were all from marine microbial communities, the metaprotome samples were extracted at the different times and dates, and the protein/peptide compositions were different among these marine data sets. Fig. 3 shows the compositions of peptides in marine data sets reported by Comet and the ones used in training DeepFilter. The three marine data sets share some peptides but only a small portion, which means that a significant amount of spectra in the test data sets were not seen in the training data sets. This experiment shows how well DeepFilter can be generalized to unseen data but similar to the data used in the training.

The filtering results are shown in Table 4, where the bold entry and underline entry represents the best and second best results, respectively. Table 4 demonstrates that DeepFilter achieves the highest identifications of PSMs, peptides, and proteins. The improvements of DeepFilter over the second best were 11.8% more PSMs, 10.3% more peptides, and 9.9% more proteins at 1% FDR on average. We also found that the DeepFilter model using Marine 3 as the training data set obtained a slightly better improvement compared to the ones using Marine 1 and Marine 2, which may be caused by the larger number of positive PSMs in Marine 3, as observed in Fig. 3(b). Our DeepFilter model trained on Soil 1 also obtained more identifications compared to baseline methods, although the improvement is not as significant as the DeepFilter models trained on the marine datasets. This may be caused by the difference between training and test data distributions. To show the discrepancy between Soil and Marine metaproteome samples, we did a taxonomic analysis by searching filtered proteins with FDR controlled at 1% against the NCBI database. We used Protein-Protein BLAST version 2.11.0+ with default parameters except only keeping one query result with the best *E*-value. The overlap of taxonomic compositions is shown in Fig. 4. The percentage of the shared identified species within marine and soil samples are 70% and 60%, separately on average. In contrast, only 5% of species are shared between soil and marine samples. Even with a low number of shared species, the DeepFilter model trained on Soil 1 still obtained more identifications compared to baseline methods, which shows how well DeepFilter can be generalized to unseen data. For marine samples, the overlap of identified PSMs, peptides, and proteins by the best DeepFilter model, Comet, and the second-best baseline are shown in Supplementary Fig. 4. Comet, DeepFilter, and other baseline methods share a significant portion of identifications. On average, 5353 PSMs, 3181 peptides, and 1035 proteins are identified only by DeepFilter, whereas 2974 PSMs, 1602 peptides, 173 proteins identified only by the second-best tool.

3.3. Performance comparison of DeepFilter on soil microbial complex

To further investigate the generalization ability of DeepFilter, we conducted the performance comparison of DeepFilter trained by marine data sets and one soil data set then tests on the rest marine and soil metaproteome data sets. Given the different microbe compositions

Table 5
Identification performance using three soil metaproteomes at FDR 1%.

	Baseline					DeepFilter			
	C	P	Q	PP	I	M1	M2	M3	S1
# PSM identification at PSM FDR 1%									
Soil 1	79,505	88,037	86,433	73,821	75,360	92,221	91,745	**94,011**	–
Soil 2	75,693	84,623	82,773	71,281	73,331	89,465	88,093	88,372	**91,384**
Soil 3	72,454	81,331	79,211	68,067	70,121	86,809	87,017	87,233	**88,015**
# Peptide identification at Peptide FDR 1%									
Soil 1	26,068	29,304	29,163	25,288	25,403	30,111	30,006	**30,923**	–
Soil 2	23,500	26,989	26,116	23,478	22,775	28,968	27,883	28,923	**29,338**
Soil 3	20,423	23,275	23,673	19,863	19,922	25,006	25,018	25,116	**25,392**
# Protein identification at Protein FDR 1%									
Soil 1	6938	7756	7684	6821	6819	8069	8011	**8184**	–
Soil 2	6913	7519	7498	6848	6879	8041	7727	8031	**8169**
Soil 3	5644	6183	6462	5473	5577	6976	6980	6998	**7029**

1 Baseline searching algorithms & filters: C, Comet only; P, Comet & Percolator; Q, Comet & Q-ranker; PP, Comet & PeptideProphet; I, Comet,PeptideProphet & iProphet;
2 DeepFilter models trained by M1 (Marine 1), M2 (Marine 2), M3 (Marine 3) and S1 (Soil 1);
3 The best entry was in bold and the next best from baseline methods was underlined.

Table 6
Performance comparison on human gut metaproteome at FDR 1%.

	PSM	Peptide	Protein
Comet	231,919	160,472	35,085
Percolator	249,371	171,183	36,183
Q-ranker	239,467	168,731	35,707
PeptideProphet	211,706	148,840	33,566
iProphet	211,706	148,840	33,566
DeepFilter	**264,875**	**182,698**	**37,644**

1 The best entry was in bold and the next best from baseline methods was underlined.

between marine and soil microbial communities, the numbers of common peptides/proteins will be even less than the ones we have in Fig. 3. DeepFilter was trained on one marine data set and was applied to soil data sets without ad hoc training or fine-tuning. If DeepFilter can still outperform other filtering tools, this means DeepFilter can be well generalized to unseen MS data even without ad hoc training or fine-tuning.

The results are shown in Table 5, where bold and underlined entries represent the best and the second best results. Among the six algorithms tested, DeepFilter always generated the highest number of identified PSMs, peptides, and proteins at 1% FDR. The improvements of DeepFilter over the second best were 6.5% more PSMs, 6.3% more peptides, and 6.9% more proteins at 1% FDR, by using marine metaproteome data sets for training. Therefore, DeepFilter was believed to have well modeled the matching between experimental spectra and peptide sequences. We also did an experiment by using soil 1 as training data set and found that DeepFilter improved the PSM/peptide/protein identifications, which gave us up to 8.2% more PSMs, 9.1% more peptides, and 8.8% proteins.

For soil samples, the overlap of identified PSMs, peptides, and proteins by the best DeepFilter, Comet, and the second-best baseline are shown in Supplementary Fig. 5. Similar to the marine data sets, Comet, DeepFilter, and other baseline methods share a significant portion of identifications. There were more identification results only reported by DeepFilter compared to other methods. On average, 920 proteins are identified only by DeepFilter at FDR 1%, whereas 234 proteins are identified only by the second-best tool.

3.4. Performance comparison of DeepFilter on human gut microbial complex

To investigate if DeepFilter performs well in the metaproteome with a large database, we tested DeepFilter on the human gut microbial complex with a protein database consisting of 5 million target proteins [29]. First, we used a small portion of mass spectra from the human gut metaproteome to test which DeepFilter model, trained by different data sets, performed best. We selected the one trained by Soil 1 data set, which achieved slightly better performance than other data sets. The identification results are shown in Table 6. Although DeepFilter did not achieve the same level improvements for human gut samples as in marine and soil samples, it still identified 6.2% more PSMs, 6.7% more peptides, and 4% more proteins compared to the best baseline method.

For the human gut sample, the overlap of identified PSMs, peptides, and proteins by the best DeepFilter, Comet, and the second-best baseline are shown in Supplementary Fig. 6. Similar to the marine and soil data sets, Comet, DeepFilter, and other baseline methods share a significant portion of identifications. 23,620 PSMs, 15,976 peptides, and 2914 proteins were identified by only DeepFilter at FDR 1%, whereas 4889 PSMs, 2845 peptides, and 480 proteins were identified only by the second-best tool.

Table 7
Identification performance using mock community metaproteome at FDR 1%.

	Baseline					DeepFilter
	C	P	Q	PP	I	
# PSM identification at PSM FDR 1%						
P1	95,098	101,563	98,461	93,669	93,669	**107,970**
P2	103,405	111,018	103,798	102,639	102,639	**118,029**
# Peptide identification at Peptide FDR 1%						
P1	26,773	28,706	27,334	25,642	25,642	**30,587**
P2	39,424	42,042	41,203	33,820	33,820	**44,901**
# Protein identification at Protein FDR 1%						
P1	7157	7670	7603	6884	6884	**8316**
P2	9417	10,134	9897	9557	9557	**10,838**

1 Baseline searching algorithms & filters: C, Comet only; P, Comet & Percolator; Q, Comet & Q-ranker; PP, Comet & PeptideProphet; I, Comet,PeptideProphet & iProphet;
2 The best entry was in bold and the next best from baseline methods was underlined.

3.5. Performance comparison of DeepFilter on mock community

We also tested the DeepFilter model using a mock community data set to see if DeepFilter is effective for a microbial complex with only a few species (30 species). We chose two of the "P" type communities from [28]. The "P" means the data sets have the same protein contents. Here, we labeled them as P1 and P2. The identification results for these two

Table 8
Computation time for nine metaproteomes (precise to second).

	Marine 1	Marine 2	Marine 3	Soil 1	Soil 2	Soil 3	HG	P1	P2
DeepFilter	204	218	210	578	476	432	1547	231	256
Percolator	126	134	127	359	277	266	902	145	157
Q-ranker	235	251	242	671	550	528	1639	247	260
PeptideProphet	67	68	182	159	142	136	473	96	108
iProphet	297	314	300	835	691	654	2217	429	319

1 P1 and P2 are two mock metaproteome samples;
2 HG is the human gut metaproteome sample.

Table 9
The number of species searched using protein identification results at FDR 1%.

	Comet	Percolator	Q-ranker	PeptideProphet	iProphet	DeepFilter
Marine 1	709	765	756	705	706	805
Marine 2	723	804	711	739	738	860
Marine 3	643	671	650	637	637	643
Soil 1	777	864	819	785	778	934
Soil 2	632	705	645	606	623	777
Soil 3	343	377	374	329	330	395
P1	30	30	30	30	30	30
P2	30	30	30	30	30	30
Human gut	2012	2148	2071	1987	1990	2454

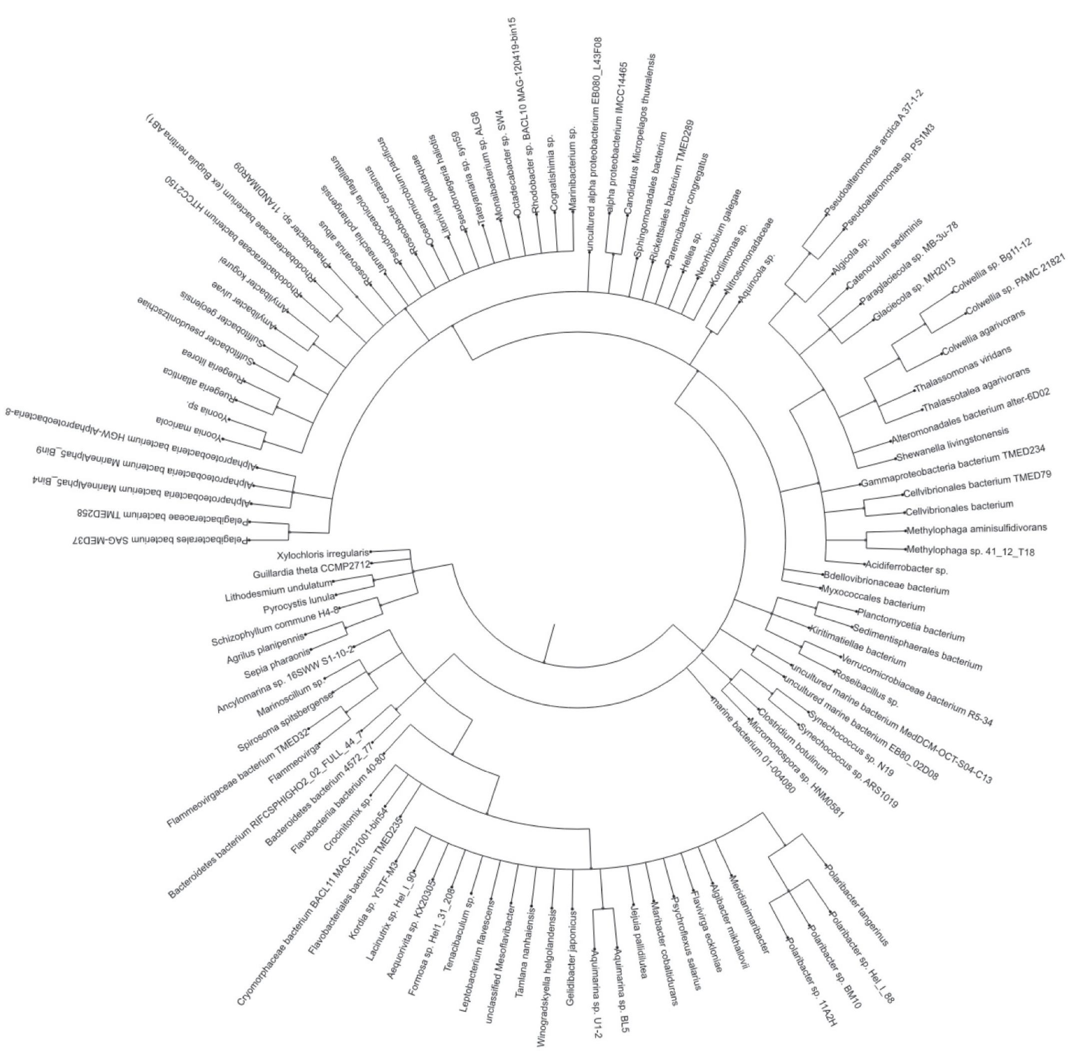


Fig. 5. Phylogenetic tree of the species only found by DeepFilter from the marine metaproteome samples.

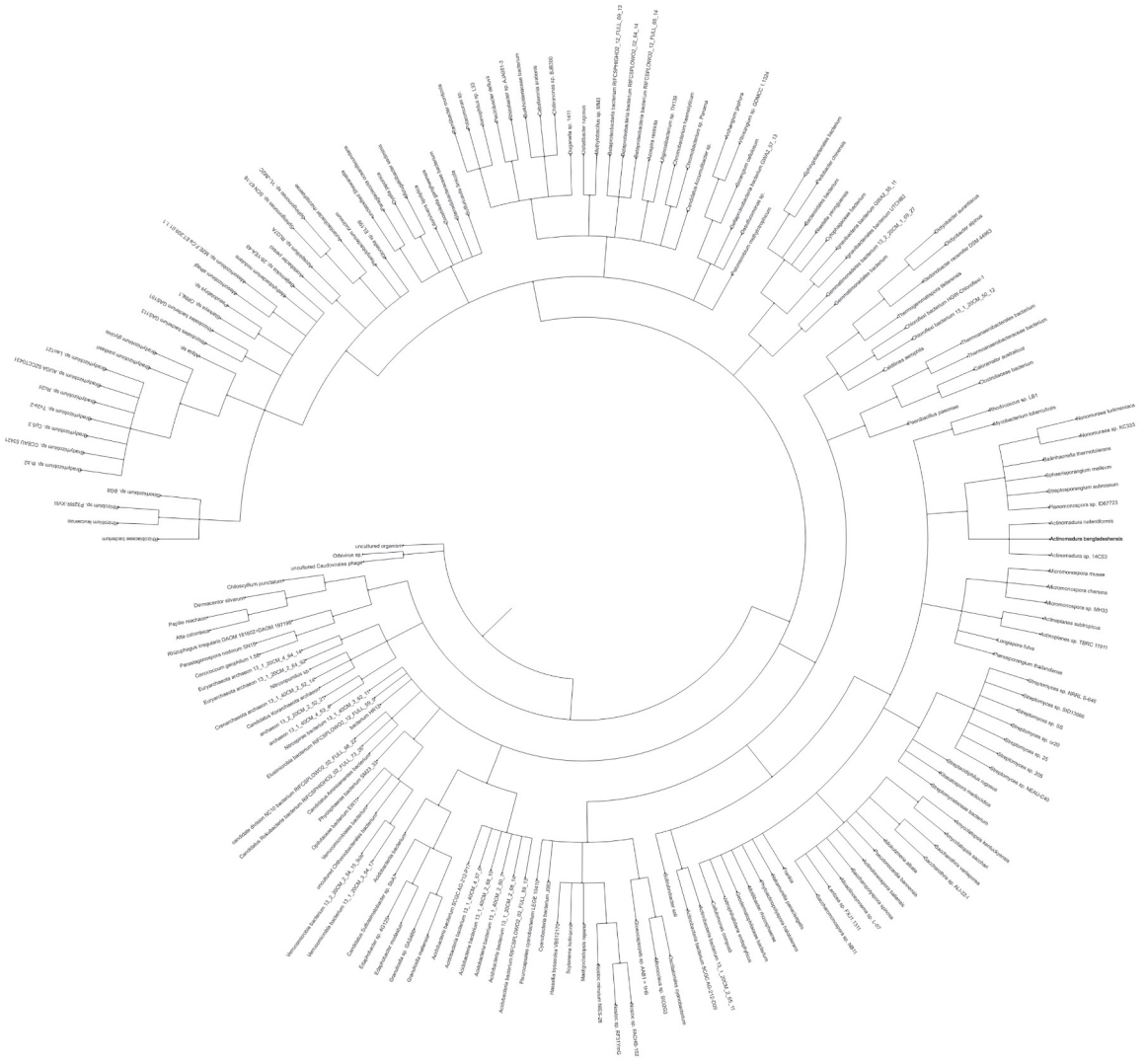


Fig. 6. Phylogenetic tree of the species only found by DeepFilter from the soil metaproteome samples.

data sets are shown in Table 7. Note that the results by the DeepFilter model were trained on another mock community data set, P3, from [28], and this DeepFilter model was slightly better than the models trained on marine and soil samples (Data not shown). DeepFilter identified 6.3% more PSMs, 6.7% more peptides, and 7.7% more proteins on average at FDR 1% than the second-best baseline method. Therefore, DeepFilter can also perform well in the metaproteome samples with a few species. For mock community samples, the overlap of identification results for PSM/peptide/protein levels by the best DeepFilter, Comet, and the second-best post-processing tools are shown in Supplementary Fig. 7. Up to 8311 PSMs, 3176 peptides, and 742 proteins were identified only by DeepFilter at FDR 1%, whereas up to 1254 PSMs, 363 peptides, and 51 proteins were identified only by the second-best tool.

3.6. Computation time

Table 8 presents the computation time when applying the DeepFilter and other filtering algorithms on different data sets. DeepFilter was running on a workstation with 8 GeForce RTX 2080 Ti GPUs, each with 12 GB memory. Baseline algorithms were executed on a desktop computer with one 2.3 GHz Intel Xeon Gold 5118 CPU and 32 GB memory. DeepFilter can finish the filtering in around 10 min with GPU acceleration.

4. Discussion

4.1. Analysis of the taxonomy information from protein identification results

To show the impact of DeepFilter on the taxonomy analysis, we used the protein-protein blast to search the protein identification results from different data sets against the NCBI database. The summary of the numbers of species identified is shown in Table 9. For the marine metaproteome data sets, up to 7% more species were found by DeepFilter compared to the second-best. DeepFilter found 10.21% more species for the soil metaproteome data sets. For the mock communities, all 30 species were found by the tested methods. For the human gut metaproteome, DeepFilter can discover more than 14% species than the

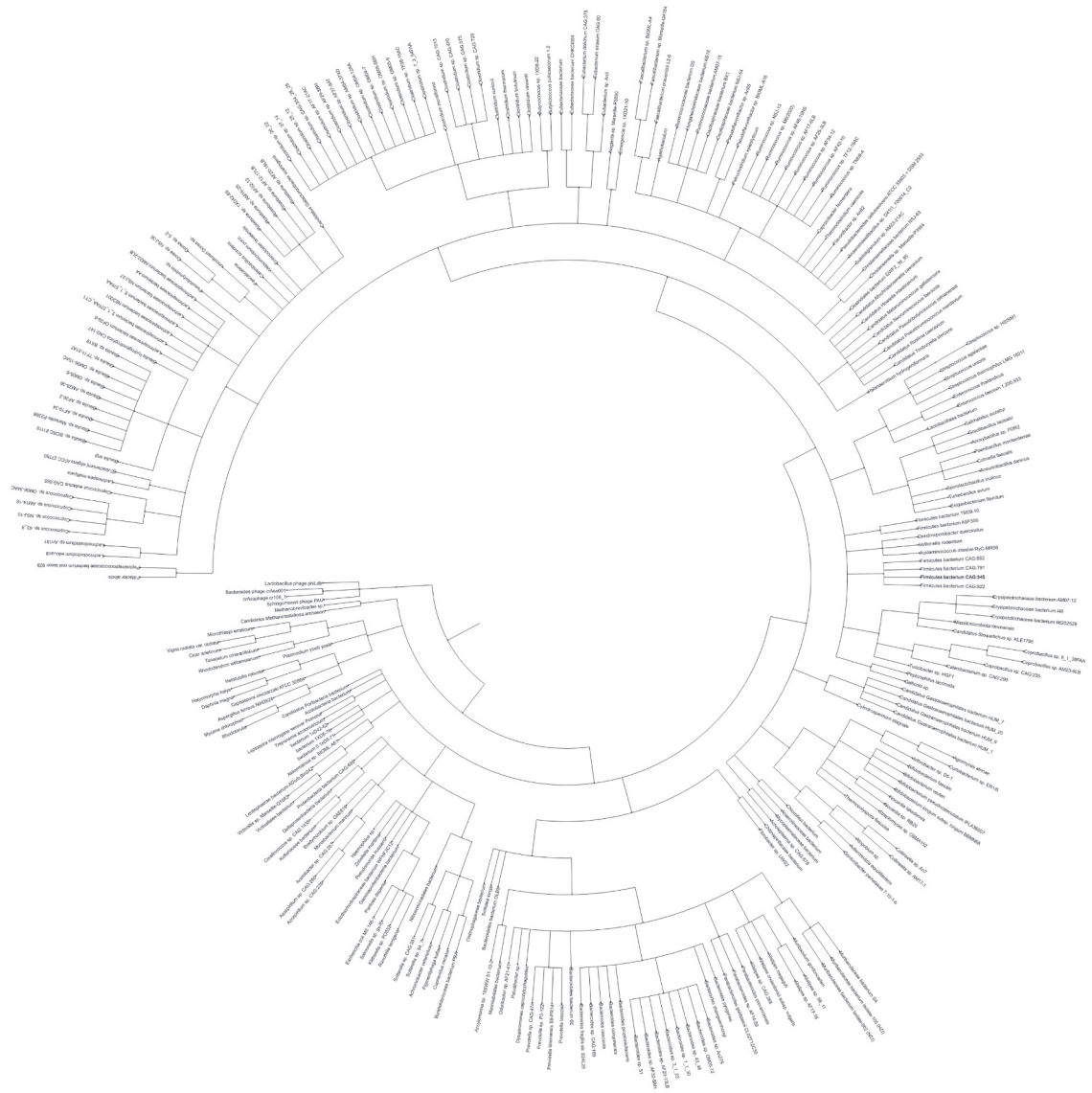
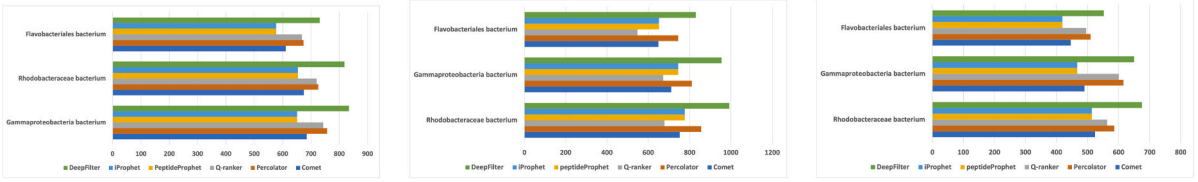


Fig. 7. Phylogenetic tree of the species only found by DeepFilter from the human gut metaproteome sample.



(a) The numbers of identified proteins at 1% FDR in the three out of five most abundant species for Marine 1.

(b) The numbers of identified proteins at 1% FDR in the three out of five most abundant species for Marine 2.

(c) The numbers of identified proteins at 1% FDR in the three out of five most abundant species for Marine 3.

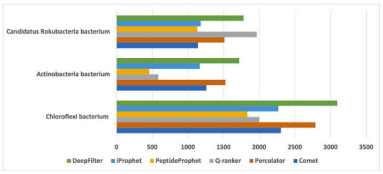
Fig. 8. The numbers of identified proteins at 1% FDR in the three out of five most abundant species for marine communities.

second-best one.

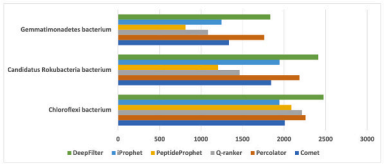
The species found only by DeepFilter for marine, soil, and human gut metaproteome samples have their lineage shown in Fig. 5, 6 and 7. These phylogenetic trees include 104 taxa for marine samples, 160 taxa for soil samples, and 306 taxa for the human gut samples. The taxa, which have the greatest number of identified proteins for marine, soil, human gut, and mock communities, are shown in Fig. 8, 9, 10 and 11. From these figures, we found that DeepFilter identified the largest or comparable numbers of proteins for each species.

4.2. *Analysis of the significance of the spectrum encoder*

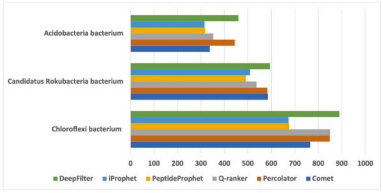
To evaluate the spectrum encoder's significance, we compared the performance of DeepFilter with the spectrum encoder disabled. The



(a) The numbers of identified proteins at 1% FDR in the three out of five most abundant species for Soil 1.



(b) The numbers of identified proteins at 1% FDR in the three out of five most abundant species for Soil 2.



(c) The numbers of identified proteins at 1% FDR in the three out of five most abundant species for Soil 3.

Fig. 9. The numbers of identified proteins at 1% FDR in the three out of five most abundant species for soil communities.

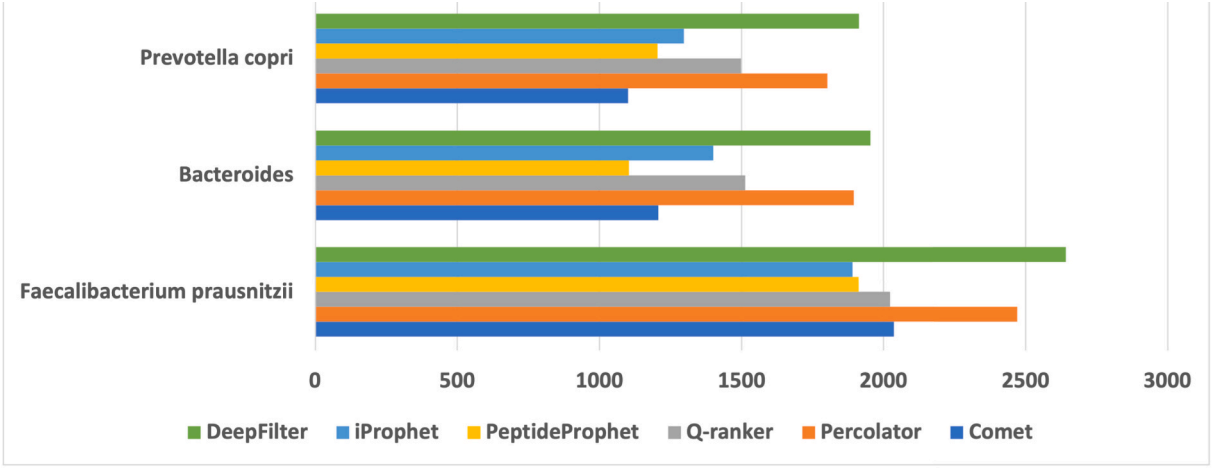
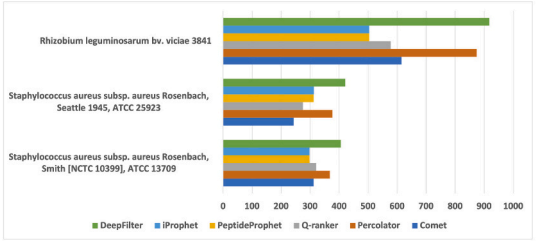
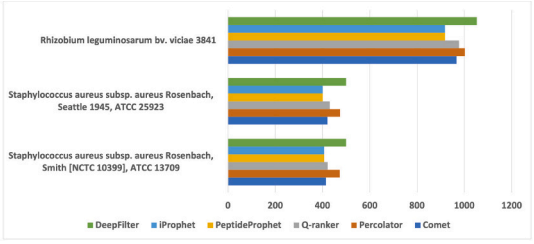


Fig. 10. The numbers of identified proteins at 1% FDR in the three out of five most abundant species for the human gut community.

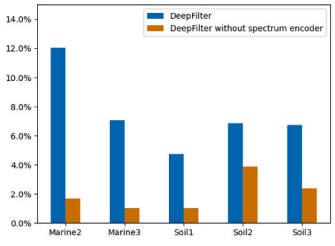


(a) The numbers of identified proteins at 1% FDR in the three out of five most abundant species for P1.

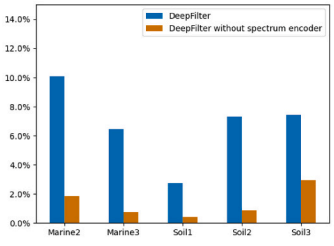


(b) The numbers of identified proteins at 1% FDR in the three out of five most abundant species for P2.

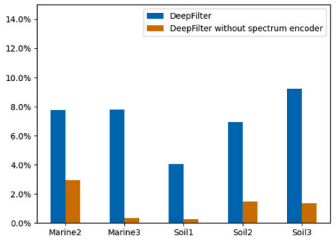
Fig. 11. The numbers of identified proteins at 1% FDR in the three out of five most abundant species for mock communities



(a) PSM level



(b) Peptide level



(c) Protein level

Fig. 12. Performance comparison of DeepFilter with/without the spectrum encoder.

models trained on Marine 2 were used here. Fig. 12 shows the improvements of DeepFilter over the second-best one from baseline methods at PSM FDR 1%. We can see that without the spectrum encoder, the PSM identifications of DeepFilter were slightly better than the existing tools but dropped significantly compared to the one with the spectrum encoder. The improvement decline was not even among different data sets. By looking at the number of unique peptides that are not shared by multiple protein sequences, fewer unique peptides give fewer protein identifications, which is the main reason for the uneven improvement decline.

4.3. Analysis of the features learned in DeepFilter

To mine the patterns and visualize the features learned by our DeepFilter, we adopted a class activation mapping (CAM) generation technique [38] to interpret the learning decision of DeepFilter. In the image analysis, CAM is used to show the input image regions that contribute to prediction process. In our experiments, we applied CAM in the spectrum representation to visualize the patterns that help predict correct PSMs.

Fig. 13 presents the CAMs for a target and a decoy PSM, respectively. The color mapping in these figures shows the weight from zero to one, as the legend indicates. The background color represents the learning weights from CNN for different regions. The red color regions make the PSMs more likely to be true (positive) PSMs if there are non-zero input values. In contrast, the blue color regions make the PSMs more likely to be false (negative) PSMs. The white points represent the peaks from the experimental and theoretical mass spectra. The CAMs have ten rows, each of which represents the peaks grouped by different charge states and different ion types as used in Section 2.4. The first four rows are for experimental spectra, and the rest six rows are for theoretical spectra. The actual spectra are shown in Supplementary Fig. 9. In Fig. 13, we can see that the isotopic envelopes of fragment ions from experimental spectra are mostly inside the red regions. Given that the isotopic envelopes indicate high-quality peaks, if the theoretical spectra also contain these ions, DeepFilter will tend to label the input PSM as a positive PSM; otherwise, DeepFilter will tend to label it as a negative PSM. For example, in the CAM of a target PSM in Fig. 13, there are several isotopic envelope mappings covered by the red region, which means there is a strong connection between the experimental spectrum and the theoretical spectrum. However, in the CAM of a decoy PSM in Fig. 13, the red region covers the parts where no isotopic envelope mappings exist. The above analysis showed that our DeepFilter models lean towards a true PSM if the matching fragment ions are of high intensity and have a detectable isotope pattern.

We examined the PSMs reported only by the DeepFilter in the hope of finding any pattern of these PSMs and why the DeepFilter performed better than the baseline methods. It turns out there are no obvious characteristics of these PSMs. We also visualized the score distribution of PSMs from the DeepFilter and the Percolator (Supplementary Fig. 8), from which we can found that both distributions of the PSMs from the Percolator and the DeepFilter are mixture distributions. For the PSMs reported only by the DeepFilter at 1% FDR, a large number of them have scores over 0.5 (Supplementary Fig. 8(c)).

4.4. Analysis of the identifications by DeepFilter in terms of false-discovery rate

In this study, we used the decoy-database approach to assess the confidence of identifications. Although the decoy-database approach is currently the gold standard in shotgun proteomics experiments, what might appear to be a good result could be, in fact, the product of overfitting [39]. Here, we used a modified decoy method, termed a semi-labeled decoy approach [39], to estimate if DeepFilter generated confident results. The semi-labeled decoy approach relies on labeled decoys and unlabeled decoys, where the latter serve as an internal error reference that helps to statistically deal with overfitting. In this

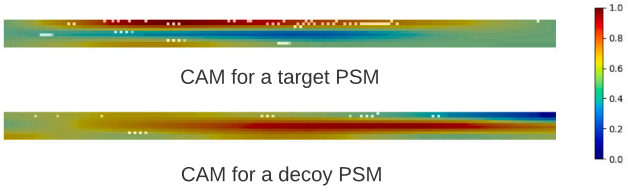


Fig. 13. Class activation mappings of one target and one decoy PSMs.

Table 4
Identification performance using marine metaproteomes at FDR 1%.

	Baseline					DeepFilter			
	C	P	Q	PP	I	M1	M2	M3	S1
# PSM identification at PSM FDR 1%									
Marine 1	34,425	37,951	36,472	33,061	33,358	–	41,423	**43,597**	41,170
Marine 2	31,822	34,741	33,899	30,670	30,846	38,927	–	**39,421**	37,165
Marine 3	38,490	41,714	40,832	37,072	37,304	**44,664**	44,273	–	44,073
# Peptide identification at Peptide FDR 1%									
Marine 1	21,334	23,597	23,007	20,961	20,961	–	25,012	**26,790**	25,387
Marine 2	22,004	24,150	23,589	21,597	21,696	26,582	–	**26,816**	25,767
Marine 3	25,085	27,522	26,674	24,653	24,661	**29,300**	29,007	–	29,127
# Protein identification at Protein FDR 1%									
Marine 1	6676	7312	7221	6458	6458	–	7687	**7956**	7781
Marine 2	7033	7715	7617	7039	5375	8313	–	**8740**	8109
Marine 3	7457	8209	8151	7354	7433	**8851**	8367	–	8690

1 Baseline searching algorithms & filters: C, Comet only; P, Comet & Percolator; Q, Comet & Q-ranker; PP, Comet & PeptideProphet; I, Comet,PeptideProphet & iProphet;
2 DeepFilter models trained by M1 (Marine 1), M2 (Marine 2), M3 (Marine 3) and S1 (Soil 1);
3 The best entry was in bold and the next best from baseline methods was underlined.

Table 10
The identified PSMs/peptides using semi-labeled decoys by the DeepFilter on marine 2 data set.

	Labeled Decoys	Unlabeled Decoys	Total Target	Total
PSM	207	192	39,421	39,820
Peptide	137	13,3	26,816	27,086

experiment, we generated two types of decoys, i.e., PR and MR sequences, for each target sequence. A PR peptide is generated by first swapping the two outermost amino acids, then treating pairs of the remaining amino acids as units and reversing their order. An MR peptide is generated by first swapping the two outermost amino acids, then dividing the remaining portion in half and reversing each of the halves separately. We randomly chose one of the decoys as labeled decoys. Table 4 shows the DeepFilter results on the Marine 2 data set. The results from the DeepFilter trained on Marine 3 appear to be consistent between the two types of decoys. For statistically estimating the overfitting issue, the number of unlabeled decoys identified follows the binomial distribution under the hypothesis that the results are not overfitted. Thus, the overfitting p-value can be approximated by $P = Pr(X > s) \approx \sum_{t=s+1}^{n} Bin(t, n, p)$, where X is a random variable indicating the number of identified unlabeled decoys, Bin is the binomial distribution function, s is the number of identified unlabeled decoys by DeepFilter, n is the total number of identifications, p is the expected fraction of unlabeled decoys (i.e., given FDR). By re-analyzing Table 10, we believe that the results from the DeepFilter can be taken with confidence ($P \gg 0.05$) without overfitting issue.

5. Conclusion

In this study, a CNN-based deep learning model, called DeepFilter, was designed to filter PSM candidates after database searching. It can automatically learn the features from experimental spectra and peptide sequences and combine with other engineered features to predict if PSMs are correct matches or not. Unlike the existing filtering tools, we did not apply a semi-supervised fashion or fine-tune the filter using a subset of the working data. Instead, we trained DeepFilter on a separate data set and tested its performance on other metaproteome data and single organism proteome data. The experimental results demonstrate that DeepFilter achieved the highest or comparable numbers of identified PSMs, peptides, and proteins. Therefore, DeepFilter was believed to generalize properly to new, previously unseen PSMs. In the future, we will further improve DeepFilter by training it on a composite data set with mass spectra from various microbes, such as those in the human intestines. We will also investigate its performance on other microbial communities that are available in Proteomics Identifications database [40].

Data availability statement

The raw MS data and protein databases are available from the PRIDE repository with the following data set identifier PXD007587, PXD006118, and PXD013386. The datasets generated during the current study are available in the GitHub repository at https://github.com/Biocomputing-Research-Group/DeepFilter.

Declaration of Competing Interest

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Acknowledgments

Research reported in this publication was supported by the National Library of Medicine of the National Institutes of Health under award number R15LM013460. The authors acknowledge the Talon 3 system at The University of North Texas for providing High Performance Computing resources that have contributed to the research results reported within this paper. URL: https://research.unt.edu/research-services/research-computing.

Appendix A. Supplementary data

Supplementary data to this article can be found online at https://doi.org/10.1016/j.jprot.2021.104316.

References

[1] R.D. Zwittink, D. van Zoeren-Grobben, R. Martin, R.A. van Lingen, L.J.G. Jebbink, S. Boeren, I.B. Renes, R.M. van Elburg, C. Belzer, J. Knol, Metaproteomics reveals functional differences in intestinal microbiota development of preterm infants, Mol. Cell. Proteomics 16 (9) (2017) 1610–1620.
[2] E. Timmins-Schiffman, D.H. May, M. Mikan, M. Riffle, C. Frazar, H. Harvey, W. S. Noble, B.L. Nunn, Critical decisions in metaproteomics: achieving high confidence protein annotations in a sea of unknowns, ISME j. 11 (2) (2017) 309–314.
[3] D. Liu, K.M. Keiblinger, A. Schindlbacher, U. Wegner, H. Sun, S. Fuchs, C. Lassek, K. Riedel, S. Zechmeister-Boltenstern, Microbial functionality as affected by experimental warming of a temperate mountain forest soil—a metaproteomics survey, Appl. Soil Ecol. 117 (2017) 196–202.
[4] A. Penzlin, M.S. Lindner, J. Doellinger, P.W. Dabrowski, A. Nitsche, B.Y. Renard, Pipasic: similarity and expression correction for strain-level identification and quantification in metaproteomics, Bioinformatics 30 (12) (2014) i149–i156.
[5] J. Alcock, C.C. Maley, C.A. Aktipis, Is eating behavior manipulated by the gastrointestinal microbiota? evolutionary pressures and potential mechanisms, Bioessays 36 (10) (2014) 940–949.
[6] E. Holmes, J.V. Li, J.R. Marchesi, J.K. Nicholson, Gut microbiota composition and activity in relation to host metabolic phenotype and disease risk, Cell Metab. 16 (5) (2012) 559–564.
[7] X. Zhang, W. Chen, Z. Ning, J. Mayne, D. Mack, A. Stintzi, R. Tian, D. Figeys, Deep metaproteomics approach for the study of human microbiomes, Anal. Chem. 89 (17) (2017) 9407–9415.
[8] A.I. Nesvizhskii, A. Keller, E. Kolker, R. Aebersold, A statistical model for identifying proteins by tandem mass spectrometry, Anal. Chem. 75 (17) (2003) 4646–4658.
[9] A. Keller, A.I. Nesvizhskii, E. Kolker, R. Aebersold, Empirical statistical model to estimate the accuracy of peptide identifications made by ms/ms and database search, Anal. Chem. 74 (20) (2002) 5383–5392.
[10] Y. Ding, H. Choi, A.I. Nesvizhskii, Adaptive discriminant function analysis and reranking of ms/ms database search results for improved peptide identification in shotgun proteomics, J. Proteome Res. 7 (11) (2008) 4878–4889.
[11] H. Choi, A.I. Nesvizhskii, Semisupervised model-based validation of peptide identifications in mass spectrometry-based proteomics, J. Proteome Res. 7 (01) (2008) 254–265.
[12] M.V. Ivanov, L.I. Levitsky, A.A. Lobas, T. Panic, U.A. Laskay, G. Mitulovic, R. Schmid, M.L. Pridatchenko, Y.O. Tsybin, M.V. Gorshkov, Empirical multidimensional space for scoring peptide spectrum matches in shotgun proteomics, J. Proteome Res. 13 (4) (2014) 1911–1920.
[13] D. Shteynberg, E.W. Deutsch, H. Lam, J.K. Eng, Z. Sun, N. Tasman, L. Mendoza, R. L. Moritz, R. Aebersold, A.I. Nesvizhskii, iprophet: multi-level integrative analysis of shotgun proteomic data improves peptide and protein identification rates and error estimates, Mol. Cell. Proteomics 10 (12) (2011).
[14] J.E. Elias, F.D. Gibbons, O.D. King, F.P. Roth, S.P. Gygi, Intensity-based protein identification by machine learning from a library of tandem mass spectra, Nat. Biotechnol. 22 (2) (2004) 214–219.
[15] P.J. Ulintz, J. Zhu, Z.S. Qin, P.C. Andrews, Improved classification of mass spectrometry database search results using newer machine learning approaches, Mol. Cell. Proteomics 5 (3) (2006) 497–509.
[16] L. Kall, J.D. Canterbury, J. Weston, W.S. Noble, M.J. MacCoss, Semi-supervised learning for peptide identification from shotgun proteomics datasets, Nat. Methods 4 (11) (2007) 923–925.
[17] A.A. Klammer, S.M. Reynolds, J.A. Bilmes, M.J. MacCoss, W.S. Noble, Modeling peptide fragmentation with dynamic bayesian networks for peptide identification, Bioinformatics 24 (13) (2008) i348–i356.
[18] G. Gonnelli, M. Stock, J. Verwaeren, D. Maddelein, B. De Baets, L. Martens, S. Degroeve, A decoy-free approach to the identification of peptides, J. Proteome Res. 14 (4) (2015) 1792–1798.
[19] M. Spivak, J. Weston, L. Bottou, L. Kall, W.S. Noble, Improvements to the percolator algorithm for peptide identification from shotgun proteomics data sets, J. Proteome Res. 8 (7) (2009) 3737–3745.
[20] X. Liang, Z. Xia, L. Jian, X. Niu, A. Link, An adaptive classification model for peptide identification, BMC Genomics 16 (S11) (2015) S1.
[21] T. Muth, D. Benndorf, U. Reichl, E. Rapp, L. Martens, Searching for a needle in a stack of needles: challenges in metaproteomics data analysis, Mol. BioSyst. 9 (4) (2013) 578–585.
[22] R. Heyer, K. Schallert, R. Zoun, B. Becher, G. Saake, D. Benndorf, Challenges and perspectives of metaproteomic data analysis, J. Biotechnol. 261 (2017) 24–36.

[23] Q. Yao, Z. Li, Y. Song, S.J. Wright, X. Guo, S.G. Tringe, M.M. Tfaily, L. Paša-Tolic, T.C. Hazen, B.L. Turner, et al., Community proteogenomics reveals the systemic impact of phosphorus availability on microbial functions in tropical soil, Nature Ecol. & Evol. 2 (3) (2018) 499–509.
[24] T.-H. Ahn, J. Chai, C. Pan, Sigma: strain-level inference of genomes from metagenomic analysis for biosurveillance, Bioinformatics 31 (2) (2015) 170–177.
[25] B. Haider, T.-H. Ahn, B. Bushnell, J. Chai, A. Copeland, C. Pan, Omega: an overlap-graph de novo assembler for metagenomics, Bioinformatics 30 (19) (2014) 2717–2722.
[26] S. Bryson, Z. Li, J. Pett-Ridge, R.L. Hettich, X. Mayali, C. Pan, R.S. Mueller, Proteomic stable isotope probing reveals taxonomically distinct patterns in amino acid assimilation by coastal marine bacterioplankton, Msystems 1 (2) (2016) e00027–15.
[27] C.N. Butterfield, Z. Li, P.F. Andeer, S. Spaulding, B.C. Thomas, A. Singh, R. L. Hettich, K.B. Suttle, A.J. Probst, S.G. Tringe, et al., Proteogenomic analyses indicate bacterial methylotrophy and archaeal heterotrophy are prevalent below the grass root zone, PeerJ 4 (2016), e2687.
[28] M. Kleiner, E. Thorson, C.E. Sharp, X. Dong, D. Liu, C. Li, M. Strous, Assessing species biomass contributions in microbial communities via metaproteomics, Nat. Commun. 8 (1) (2017) 1–14.
[29] S. Long, Y. Yang, C. Shen, Y. Wang, A. Deng, Q. Qin, L. Qiao, Metaproteomics characterizes human gut microbiome function in colorectal cancer, NPJ biofilms and microbiomes 6 (1) (2020) 1–10.
[30] J.K. Eng, T.A. Jahan, M.R. Hoopmann, Comet: an open-source ms/ms sequence database search tool, Proteomics 13 (1) (2013) 22–24.
[31] J.E. Elias, S.P. Gygi, Target-decoy search strategy for increased confidence in large-scale protein identifications by mass spectrometry, Nat. Methods 4 (3) (2007) 207–214.
[32] M.W. Senko, S.C. Beu, F.W. McLafferty, Automated assignment of charge states from resolved isotopic peaks for multiply charged ions, J. Am. Soc. Mass Spectrom. 6 (1) (1995) 52–56.
[33] D. Hyatt, C. Pan, Exhaustive database searching for amino acid mutations in proteomes, Bioinformatics 28 (14) (2012) 1895–1901.
[34] M.P. Washburn, D. Wolters, J.R. Yates, Large-scale analysis of the yeast proteome by multidimensional protein identification technology, Nat. Biotechnol. 19 (3) (2001) 242–247.
[35] X. Guo, Z. Li, Q. Yao, R.S. Mueller, J.K. Eng, D.L. Tabb, W.J. Hervey IV, C. Pan, Sipros ensemble improves database searching and filtering for complex metaproteomics, Bioinformatics 34 (5) (2018) 795–802.
[36] V. Granholm, W.S. Noble, L. Kall, On using samples of known protein content to assess the statistical calibration of scores assigned to peptide-spectrum matches in shotgun proteomics, J. Proteome Res. 10 (5) (2011) 2671–2678.
[37] K. Jeong, S. Kim, N. Bandeira, False discovery rates in spectral identification, BMC bioinformatics 13 (S16) (2012) S2.
[38] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, A. Torralba, Learning deep features for discriminative localization, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 2921–2929.
[39] R. Barboza, D. Cociorva, T. Xu, V.C. Barbosa, J. Perales, R.H. Valente, F.M. França, J.R. Yates, P.C. Carvalho, Can the false-discovery rate be misleading? Proteomics 11 (20) (2011) 4105–4108.
[40] Y. Perez-Riverol, A. Csordas, J. Bai, M. Bernal-Llinares, S. Hewapathirana, D. J. Kundu, A. Inuganti, J. Griss, G. Mayer, M. Eisenacher, et al., The pride database and related tools and resources in 2019: improving support for quantification data, Nucleic Acids Res. 47 (D1) (2019) D442–D450.