Utilizing pre-trained protein models and a specie feature to improve post-translation modification site prediction.

Brent van Dodewaard (2599295)
First supervisor(s): Laura Hoekstra & Reza Haydarlou

Vrije Universiteit Amsterdam
B.Dodewaard@gmail.com

Abstract. The abstract should briefly summarize the contents of the paper in 15–250 words.

Keywords: First keyword · Second keyword · Another keyword.

Table of Contents

1 Introduction

1.1 An introduction to PTM prediction

Post-translational modifications (PTMs) are modifications of proteins, which occur by modifying groups being added to one or more amino acids after the translation of said proteins. PTMs can affect both the structural and biological activity of proteins, and thus play a key role in a lot of biological processes [1]. PTMs occur on almost all proteins, with many being modified by multiple PTMs [2]. Because of their biological importance, PTMs, and their dysregulation, are linked to a wide array of diseases, such as cancer, Alzheimer's and diabetes [3]. Knowing which amino acids are modified by which PTM-types can help in understanding the pathology of these diseases, but can also be directly used as disease markers and molecular targets for drug development. While the amount of experimentally verified PTM sites has greatly increased in the last few years, most protein sequences remain unannotated. Experimental identification of PTM sites is still both time-consuming and expensive, which is why computational models which can predict the sites of PTMs are of high value [4].

PTM prediction is usually structured as a binary classification problem, in which a separate model is trained for each PTM-type. PTM prediction is performed on a per-residue basis instead of a per-protein basis. This means we aim to predict if a certain amino acid is modified by a PTM, and not if a certain PTM occurs at all on a certain protein. Given a certain amino acid sequence surrounding a PTM site, we want to predict if the middle amino acid is either modified by a PTM, or not. A simple visual representation of PTM prediction can be found in Figure 1.

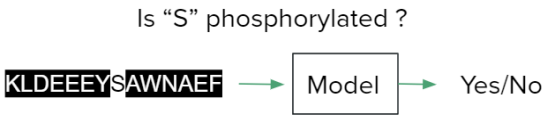

Fig. 1. A visual representation of the PTM prediction task. A prediction model predicts if a certain amino acid is modified by a certain PTM-type, given the amino acids which surround that amino acid.

In the past, this was commonly performed using traditional machine-learning algorithms, like random forest models [5] and support vector machines [6], on hand-crafted amino acid features [4]. However, over the last few years, we have seen more and more end-to-end deep neural network (DNN) models reach state-of- the-art (SOTA) performance, while often only using the amino acid sequence as an input. This does not mean using additional features could not be beneficial when using DNNs, as has been shown in several papers [7, 8]. It is known that patterns associated with PTM sites can depend on the specie in which an amino acid sequence is found, which is further demonstrated by the fact that for some

PTM-types, specie-specific PTM prediction models show increased performance when compared to general models [9, 10]. However, using the specie in which a certain sample is found as a feature, remains unexplored.

A interesting recent advancement in the field of protein prediction tasks is the development of various pre-trained protein language models (LM), such as protVec [11], SeqVec [12] and ProtTrans. These models use architectures typically designed for natural language processing, and can be used to generate embeddings of amino acid sequences. Embeddings are real-valued vector representations of in this case amino acids, which are used to represent the characteristics of them in their given sequence, such that amino acids that are closer in the vector space also share more characteristics. These models are trained using self-supervision on up to billions of amino acids, and are thus able to capture some biophysical features of protein sequences. These embeddings have successfully been used in various protein prediction tasks to reach SOTA performance, such as secondary structure and subcellular localization prediction [13]. Using embeddings from such a protein LM for PTM prediction remains unexplored.

While a lot of research is performed in the domain of PTM prediction, a public dataset which includes all available experimental data, can directly be used for both training and testing, and includes the most commonly occurring PTM-types, does not currently exist. This makes research into PTM prediction methods which are robust across different PTM-types less accessible. It also makes fair comparisons between methods in different papers unnecessary difficult.

In conclusion, advancements in the quality of PTM prediction models are of utmost importance. And while a lot of research is performed in this domain, three areas with room for improvement have been identified: 1. The creation of a complete public benchmark dataset, which includes the most-commonly occurring PTM-types. 2. Using amino acid sequence embeddings from a pre-trained protein LM as input for a PTM prediction model. and 3. Using the specie from which a particular sample is from, as a feature for a PTM prediction model.

1.2 Research Context: Internship Provider

A recent research paper from the Center for Integrative Bioinformatics (IBIVU) [14], showed that the presence of certain PTM's on proteins was the best predictor of a protein being associated with extracellular vesicles. They showed that while the presence of PTMs is a very strong predictive feature whenever a PTM site was experimentally verified, this was not the case when acquired from the SOTA PTM prediction model MusiteDeep. This finding lead to interest from the IBIVU in improving the prediction quality of ~~these~~ PTM prediction models, with the hypothesis that they could also be useful in other protein prediction tasks. This motivated carrying out this particular research paper.

1.3 Research Goals

The first goal of this research is to create a benchmark dataset, which can be used for both training and evaluation of a PTM prediction model on the most often occurring PTM-types. This dataset will be made public, with the intent of making future research into PTM prediction more accessible.

The second goal of this research, is to optimize the performance of a self-designed PTM prediction model, by comparing several feature and architecture design choices, some of which are novel. The two scientific contributions here are: 1. Evaluating the usage of the species in which a PTM is found as an additional feature for a PTM prediction model, and 2. Evaluating using ProtTrans embeddings as the input for a PTM prediction model.

This gives us the following research questions:

1. *What are the optimal choices for acquiring and processing PTM-samples in order to create a dataset which can be used for creating robust prediction models across the most often occurring PTM types?*
2. *What are the optimal design choices when creating prediction models which are to be used on the PTM-types as found in our dataset?*
   (a) *How does including the species in which a sample was found as an additional feature influence PTM prediction performance?*
   (b) *How does using embeddings as generated by ProtTrans an input for a PTM prediction model influence prediction performance?*
3. *How well do our models perform on both our own and an independent dataset, and how does this compare to an existing model from the literature?*

1.4 Research Approach

In this paper, a non-redundant dataset which can be used for both training and testing PTM prediction models for 13 of the most commonly occurring PTM-types was constructed. After this, the usefulness of ProtTrans embeddings for the various PTM-types in our dataset was shown using t-SNE visualizations. Sequence logos were also generated of the various PTM-types, which were used to show the difference in sequence patterns between species. In order to design an optimal deep learning model architecture, amino acid representations, model architectures and data sampling techniques were compared using a validation set on three representative PTM-types, which represented PTM-type with a low-, medium-, and high amount of samples. Using the optimal design as found during these validation experiments, models for all 13 PTM-types will be trained and evaluated on our test set. In addition, the usage of ProtTrans embeddings and species as an input feature will be evaluated on the test set. Finally, the performance of the trained models will be compared with an independent prediction model called MusiteDeep on the constructed test set for all 13 PTM-types. For phosphorylation, the prediction performance will also be compared on a third party independent test set as provided by MusiteDeep.

2 Related work

In this section, related work to our research, and how it relates, will be presented.

2.1 Other PTM prediction models

Duolin Wang et al presents a novel DNN PTM prediction architecture named MusiteDeep [15], using only the amino acid sequence as the input. They trained separate prediction models for 13 of the most common PTM-types, reaching SOTA performance on all of them. Even though since then MusiteDeep has been outperformed on individual PTM-types by newer solutions [16–18], no other paper has presented improved performance over a wide range of PTM-types with a single architecture. In addition, MusiteDeep has a publicly available webserver which can be used to make PTM site predictions for large amounts of samples. Because of this, MusiteDeep will be used to compare the models designed in this paper ~~to.~~

An interesting recent PTM prediction architecture, called Adapt-Kcr, is presented in Li et al [16]. Using a rather simple embedding and model architecture, they are able to perform better in lysine crotonylation prediction on an independent test-dataset than other existing models, which use a lot more complex sequence encoding methods and model architectures. They also show the excellent transfer-ability of their architecture to other PTM-types, such as phosphorylation, in an experiment. Their model architecture consists of a single CNN-layer, a single bi-LSTM layer, a single attention layer, and finally a single fully connected hidden layer. As for regularization, they made use of dropout and batch normalization. Due to the relative simplicity of their design, their prediction performance, and the fact that their design does well across multiple PTM types, the design of the DNN PTM prediction architecture used in this study will largely be influenced by the design of Adapt-Kcr.

2.2 Peptide representation

MusiteDeep, Adapt-Kcr, and other recent research papers reaching SOTA performance using DNNs, all create a vector representation of the amino acids in their model~~,~~ using either self-attention [7, 15], a supervised language model (LM) [8], or a pre-trained LM [19]. A recent paper by Li et Al present a novel peptide representation method that they call adaptive embedding. They use a simple embedding layer, with separate weights for amino acid-type and residue-location. Using a simple DNN model, they are able to beat the SOTA models in both lysine crotonylation and phosphorylation classification. Their proposed embedding method shows promising results in their own experimental results when compared to an one-hot embedding. However, they fail to compare their new design with a single-layered embedding. Such an comparison would show the added value of the residue-location embedding, which is arguably the novel design decision in the adaptive embedding. Because of this, in this study, the

prediction performance of an adaptive embedding layer will be compared with both one-hot encoded amino acids and a single-layered embedding layer.

Elnaggar et al recently presented a range of publicly available protein language models (LM's) called ProtTrans, which can be used to generate vector representation of amino acid sequences [20]. These LM's were pre-trained on up to 393 billion amino acids, using 5616 GPUs. Using rather simple DNN architectures, they were able to reach SOTA performance in various protein- and residue-level prediction tasks. They also show that their protein LM's show improved performance when compared to other existing large protein LM's. No PTM prediction model up to now has made use of ProtTrans, or any of the other existing large pre-trained LM's. Due to the success of using ProtTrans embeddings for other per-residue protein prediction tasks, we will also experiment with utilizing these embeddings for PTM prediction.

2.3 PTM data-imbalance

Dou et al provides a comprehensive review of the data-imbalance problem in PTM prediction [21]. Data imbalance is a common problem in PTM predicion, where the amount of negative samples is often much higher than the amount of positive samples. When not taken into account ~~in the taken approach~~, this usually leads to a predictive bias for the majority class, in this case the negative samples. However, researchers are more interested in the model performance on true PTM sites, thus yielding sub-optimal models. The simplest solution, and one that a lot of papers still utilize, is discarding the redundant negative samples. However, specialized techniques to make use of all data have a clear positive effect on model performance. Common techniques ~~used which are~~ applicable to deep learning are: random undersampling, oversampling, class specific loss-weights and more advanced variations of them. Dou et al emphasizes how important the usage of data-imbalance techniques is when implementing PTM prediction models. Because of this, A comparison will be made between the basic versions of the most often used techniques, namely: sample balancing, random undersampling, oversampling and class specific loss-weights.

More advanced data imbalance techniques which have been utilized by deep learning PTM prediction models are the synthetic minority oversampling technique (SMOTE)~~,~~ and using a bootstrap sampling ensemble. SMOTE works by creating new synthetic samples for the minority class by finding two samples that are similar using a nearest neighbor algorithm, and then creating a new sample that is somewhere in between these two samples. SMOTE, and a variant of it, has been shown to exhibit superior performance to random undersampling for PTM prediciton [22]. A bootstrap sampling ensemble is an ensemble of models which all utilize a different set ~~of~~ of the majority class, while all using the same samples from the minority class. Such as bootstrap sampling method has also been used MusiteDeep. While implementing and comparing SMOTE and a bootstrap sampling ensemble is outside of the scope of this study, it is important to realise more advanced data imbalance sampling methods which have been shown to be effective for PTM prediction exist.

2.4 Features beyond Sequence

While DNN solutions for PTM prediction often only use the amino acid sequence as an input, this does not have to be the case. Additional (hand-crafted) features can provide improved performance. Features that have been shown to lead to improved generalization in PTM DNN models are: protein-protein interactions [7], additional sequence-based features (CKSAAP, PWAA), and physicochemical features (~~AAindex, CTD, EBGW~~) [8]. While not utilizing a DNN, Song et al analysed 7 different types of functional features to use in adjacency to sequence based-features [23]. They conclude the most impactful functional feature are: protein-protein interactions, pathway features, cellular component features, native disorder features, and secondary structure information. Another type of feature used to improve prediction performance is the inclusion of evolutionary information. One such approach is using a nearest neighbour algorithm to identify if the sequence surrounding a PTM site is closer to known positive sites or known negative sites. Such an approach is used in Chen et al [24], in which they use such a feature to improve ubiquitylation site prediction. Another type of evolutionary feature used is the position-specific scoring matrix (PSSM), which measures how often different amino acids occur in that location for protein sequences which share common ancestors, with the idea that biologically relevant amino acids are "conserved" among ancestors. The usefulness of PSSM as a feature is highly dependent on the PTM-type, as some PTM types show only very weak conservation among common ancestors [25]. For other PTM-types, such as succinylation, it can be used as the sole input for a prediction model [26].

2.5 PTMs: differences between species

A lot of PTM prediction models use samples from across all species to build a single generic model. While research is not available for all PTM-types, it has been shown for various PTM-types that they show key differences between different species. Some types of PTM's show very different rates of occurrence, or are even restricted to certain types of species, with some existing research suggesting this might be dependent on the set of enzymes which can catalyst the formation of PTMs as present in a species [25]. Such is the case for O-linked glycosylation, which is widespread in Gram-negative bacteria but non-existing in Gram-positive bacteria due to the absence of the enzyme Oligosaccharyltransferase [27]. Another key difference ~~which has been shown between species for several PTM-types,~~ is the pattern of the sequence surrounding the PTM sites. Key differences in these patterns between species have been shown for ubiquitination [9], methylation [10] and acetylation [28]. These differences can influence the importance of different features based on the species of interest. This has been shown to be the case for methylation, where the optimal physicochemical features used differed for all three different species analysed [10]. Using a single model for proteins from multiple species can lead to interference of feature importance between different species. It is thus no surprise that for various PTM-types, specie specific predictors have been developed, which show improved predictive

performance to their specie-generic counterpart [9, 10, 28]. Only using samples from one or several species, drastically reduces the amount of available training data, which is especially difficult for PTM-types with not a lot of available date. Additionally, just because some of the PTM-characteristics between species differ, this does not mean that patterns used for predictions are not transferable between species at all. Thus, in this research, we aim to research the addition of the species in which a sample was found as an additional input feature. Theoretically, this would allow a DNN to recognize specie specific patterns and negate the interference of feature importance differences while still utilizing all samples as available for a PTM-type.

3 Methodology

In this section, a description will be given how the benchmark dataset used in this research paper was created. After this, we will discuss the architectural design of the proposed model, how the models were trained, and how the performance of the models was evaluated.

3.1 Benchmark dataset

A benchmark dataset was created by further processing the experimentally verified PTM annotations as made available in dbPTM 2021, which can be regarded as the most complete source for PTM annotations [1], containing a total of 2,235,000 entries of 76 different PTM-types [29]. In our dataset, the decision was made to include the 13 PTM-types as used in the latest MusiteDeep paper, to allow for comparison ~~between the two~~. The included PTM types, and the amount of samples as extracted from dbPTM 2021, are shown in Table 1. PTM-types which name include an appended capital letter at the end of their name, have been split into separate PTM-types depending on the amino acid of the affected site, where the capital letter stands for the affected amino acid according to FASTA amino acid codes [30].

Table 1. The names of the different PTM-types included in our benchmark dataset, and the amount of positive samples as retrieved from dbPTM 2021.

Name	Amount of pos. samples
Phosphorylation-S/T	1,407,443
Ubiquitination	348,307
Phosphorylation-Y	119,697
N-linked Glycosylation	27,361
O-linked Glycosylation	16,692
Methylation-R	8547
Methylation-K	7034
S-palmitoylation-C	6358
Sumoylation	5889
Hydroxylation-P	1791
Pyrrolidone carboxylic acid	965
Hydroxylation-K	229

The performed processing steps to acquire our benchmark dataset from the acquired data are presented in Figure 2, and will now be further described.

dbPTM only provides a total of 20 amino acids surrounding each PTM site, which is not optimal for our use-case, as modern PTM prediction DNN's commonly use a longer sequence ~~than that~~. Using the provided UniProt accession number by dbPTM, the full protein sequence was acquired using the UniProt API. If the amino acid sequence between dbPTM and UniProt did not match,

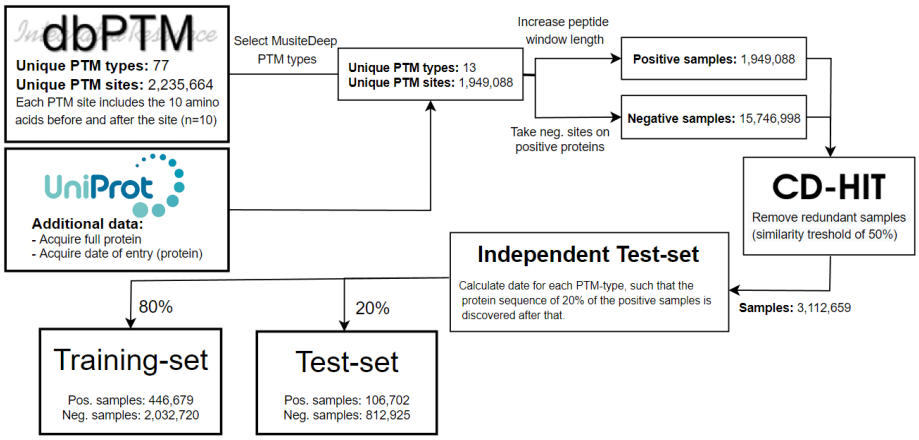

Fig. 2. The processing steps used to acquire our benchmark training- and test dataset, starting from the samples acquired from dbPTM 2021.

the sample was discarded. This affected 3.75% of all samples. The acquired full protein sequence was finally used to extend the length of the amino acid sequence surrounding each sample to 32 amino acids. A sequence length of 33 was the highest amount as used by any other model as found in the literature [31].

The samples as acquired from dbPTM only include positive experimentally verified PTM sites. It is not valid to take any amino acid which could be affected by a certain PTM site, but is not included in our positive samples, as a negative sample. There is no guarantee that a specific amino acid has been inspected for that specific PTM-type, and this would thus lead to a large amount of false negatives. Rather, the amino acids which can be modified by a certain PTM site, and are from a protein which includes at least one positive sample for that PTM-type, are taken as the negative samples.

To avoid data leakage between the training, validation, and test set, sequences which show too much similarity to each other were filtered out of the dataset. This was performed using CD-HIT [32], with a sequence identity threshold of 50% to remove any redundant sequences. This results in a dataset where given any two samples in the dataset, which are lined-up such that the most amount of amino acids possible match, no pair shares more than 50% of its amino acids.

Finally, for each PTM-type, the date was calculated such that the date of entry for 80% of the samples is before that date, and the date of entry for the remaining 20% is after that date. Everything before this date, and thus 80% of the samples, are taken as the training set, while every sample after this date is taken as the test set. The test set is consciously chosen to consist of newer samples. Both the quality of the annotations, and the selection of samples of newer data should serve as a better representation of future data.

The resulting training dataset contains a total of 446,679 positive and 2,032,720 negative samples, extracted from a total of 132.358 unique proteins and 869

unique species. The distribution of the species from which the training samples are extracted can be found in Figure 3, and a figure presenting the date of protein sequencing can be found in Figure 4. The average unique protein in our training dataset contains a total of 18.7 different PTM's, with the maximum being 1184. The imbalance ratio between the negative and positive samples ranges between 3.0x and 25.6x depending on the PTM-type. The resulting test dataset contains a total of 106,702 positive samples, and 812,925 negative samples.

The different amount of samples for each PTM-type, and the date that was used to split the training and test set, can be found in Table 2. The final dataset is stored in separate CSV-files for each PTM, and contains the following data-columns for each sample:

- ***UniProtID***: The entry name (or ID) used in the UniProtKB database, which is not necessarily stable from release to release.
- ***UniProtAC***: the accession number (AC) as assigned to each sequence upon inclusion into UniProtKB, which is stable from release to release.
- ***PTM location***: The location of the potential PTM site in the full protein sequence (1-based indexing)
- ***PTM Type***: The full name of the PTM-type,
- ***UniProtSequence***: The amino acid sequence of length 33, with in the middle the potential PTM modified amino acid.
- ***modifiedAA***: The potentially modified amino acid.
- ***DateCreated***: The date that the protein was first entered in UniProtKB.
- ***DateSeqModified***: The date the protein was last modified in UniProtKB.
- ***Label***: The label of the sample, which can either be a 1 (positive sample) or a 0 (negative sample).

Table 2. The amount of samples in the training and test set of our final dataset. The date that is used to separate the training from the test data is also given.

PTM-Type	Training		Test		Date of split
	Pos.	Neg.	Pos.	Neg.	
Phosphorylation-S/T	226,557	997,002	108,131	758,124	2011-12
Acetylation	49,328	267,460	12,747	94,357	2010-04
Ubiquitination	45,631	125,073	13,552	43,707	2010-07
Phosphorylation-Y	45,088	185,016	14,467	76,850	2010-10
N-linked glycosylation	11,014	39,373	2642	9961	2009-11
O-linked glycosylation	5589	71,271	1536	30,048	2010-05
Methylation-R	3942	43,425	699	13,922	2008-11
Sumoylation	3422	21,442	591	5685	2008-1
S-palmitoylation-C	2485	9894	667	3809	2010-04
Methylation-K	1785	28,335	2942	17,852	2008-09
Pyrrolidone carboxylic acid	247	1395	48	373	2008-09
Hydroxylation-P	138	642	77	292	2012-01
Hydroxylation-K	50	248	6	89	2007-01

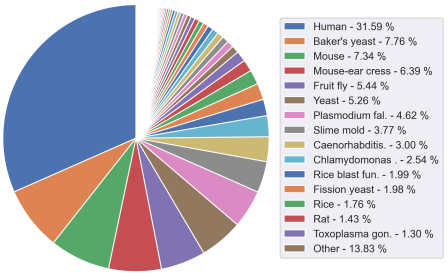

Fig. 3. The distribution of species from which the PTM-samples were sampled in our training-set

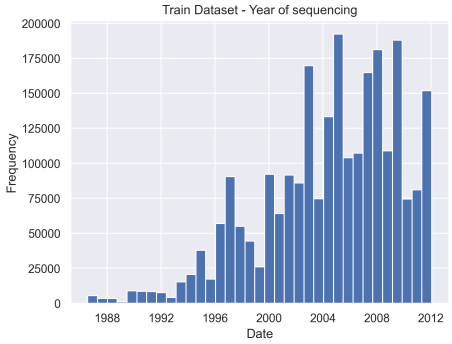

Fig. 4. The distribution of dates when the protein sequences of our training set were sequenced

3.2 Model Architecture

The design of our DNN, while self-implemented, is based on work by Li et al, as described in section 2.1. Our model architecture is visually presented in Figure 5a, and will now be further described.

Given a certain amino acid sequence, with in the middle the residue for which we want to predict if a PTM is present, the model architecture uses a one-of-K encoding of the sequence as the input to the model. Since there are 20 commonly occurring amino acids, and a dash ("-") is used whenever no amino acid is present, a one-of-21 encoding is used. Since our sequences are of length 33, this results in an input with a dimension of 33x21 for each sample. For each of the 33 amino acids, a "1" is used at a certain position to indicate the presence of a certain amino acid, while the other positions are left at "0".

In order for learned behaviour to generalize across both amino acid type and position in the sequence, the adaptive embedding module as presented in Li et al will be used. During training, for both each different amino acid type and position in the sequence, a vector representation of size 32 is learned. This is implemented using a look-up table as found in Pytorch's *Embedding* layer [33]. During forward propagation, the embedding for a certain amino acid, and the embedding for a certain location are summed together. This results in a single vector which is able to represent both the amino acid type~~,~~ and the location at which it occurs.

After obtaining a representation for a certain peptide, a single convolutional layer, with 256 filters of size 10, is used to extract high-level features of the sequence, and detect very simple patterns in the data. This is followed by a max pooling layer, with a stride of 2, which is used to decrease the complexity of the sequence, and thus decreasing both overfitting and the computational costs of the model. A bi-directional long short-term memory (bi-LSTM) layer, with a hidden size of 32, is then used to identify long- and variable length relations in the sequence. To identify the key information in the output features of the bi-LSTM,

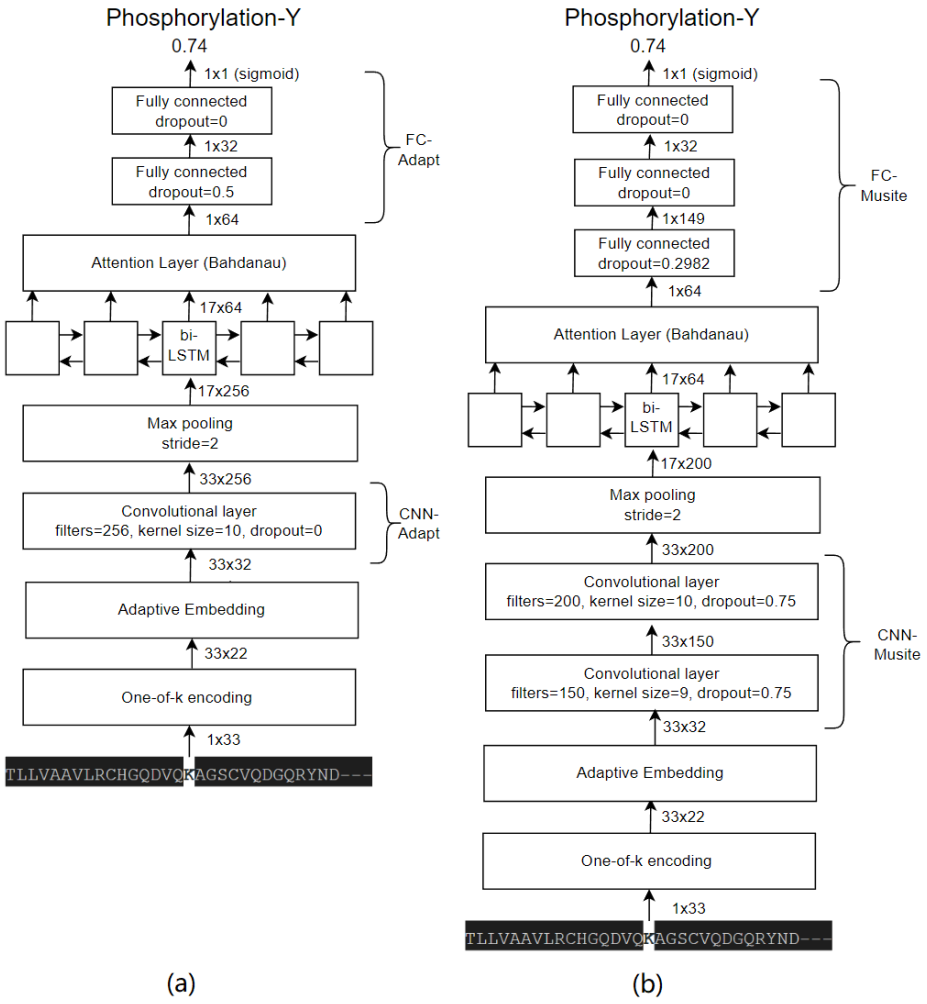

Fig. 5. (a) The model architecture as inspired by Adapt-Kcr, and as used in this study. (b) Our model architecture, where the CNN- and FC-layers have been replaced by slightly more complex modules which have been named FC-musite and CNN-Musite.

a attention mechanism called Bahdanau attention is used, with a hidden unit size of 10 [34]. After this, a fully connected layer of size 32, with a single output unit is used. For this layer, a regularization method called dropout is used [35], with a rate of 0.5. Finally, the final output prediction is acquired by using a sigmoid activation function. When making predictions, a decision threshold of 0.5 is used. Given model output x, if $x > 0.5$, a positive prediction is made, while if $x \leq 0.5$, a negative prediction is made.

3.3 Model Architecture: increasing complexity

While the model architecture as found in Li et al, and as described in the previous section, is able to outperform existing models using relatively few layers and hidden units, this does not mean that the model won't be able to perform better if it was more complex. To test this, there will be experimented with replacing the convolutional and fully connected layers with more complex modules. The original architecture can be found in Figure 5a, with its convolutional module being marked "CNN-Adapt", and its FC module being marked "FC-adapt". The more complex architecture, with its modules being inspired by the more complex modules as used by MusiteDeep, can be found in Figure 5b, with its convolutional module being marked "CNN-Musite", and its FC module being marked "FC-Musite"."CNN-Musite" utilizes two convolutional layers with filter sizes of 150 and 200, whereas "CNN-Adapt" utilizes only one convolutional layer with a size of 256. "FC-Musite" utilizes two hidden FC-layers of size 149 and 32, whereas "FC-Adapt" utilizes only one hidden FC-layer of size 32. Increasing the complexity of our model is expected to have a sizable negative impact on the training time of our model, which is why next to comparing the predictive performance, the average training time will also be compared between the architectures.

3.4 Language model Embeddings: ProtTrans

Elnaggar et al present a corpora of pre-trained protein LM's which allow embeddings to be extracted and used as input for prediction tasks. In the initial comparisons between the models as presented in Elnaggar et al, ProtT5-XL-UniRef50 is always the best performing model among their corpera of models [20]. Unfortunately, ProtT5-XL-UniRef50 requires a GPU with at least 32GB VRAM, which we do not have access too. For this research, the model ProtBert-BFD was chosen. This model is architecturally based on the BERT language model [36], and trained on the protein database *Big Fat Database* (BFD) [37]. Comparisons in the literature show an increase in accuracy between 2% and 46% [38] when using ProtT5-XL-UniRef50 as apposed to ProtBert-BFD to generate embeddings for protein prediction tasks. The embeddings are obtained by extracting the last hidden state of the model for each amino acid, which are of size 1024 per amino acid. Since in this case, a peptide length of 33 is used as input for the model, this results in a final embedding size of 33x1024 for each sample.

3.5 Data imbalance

In order to effectively train a model using our imbalanced dataset, 4 different techniques to reduce the bias of our model towards the majority class have been implemented. The first of which has been given the name *balanced dataset*, where before training starts, the redundant negative samples are discarded. This results in a balanced dataset. However, all information that was in the discarded negative samples is lost. The second technique is *random undersampling*, where during each epoch, majority samples are randomly deleted to balance the dataset. Each epoch still includes the same amount of total samples, but a new set of negative samples is selected each epoch. While now over several epochs, all negative samples are used, the random selection of negative samples can also lead to decreased training stability. The third technique is *random oversampling*, where during each epoch, the minority samples are repeated to balance the dataset. This means all samples are used during each epoch, with the downside that the same positive samples are repeated multiple times during each epoch, potentially unnecessarily increasing training time. The last technique is the use of a *weighted loss function*, where the weight of negative samples is set to $\frac{1}{R}$, where R is the ratio between the amount of negative and positive samples. This results in a loss as if the dataset was balanced, while being able to use all samples each epoch, and without unnecessarily having to repeat samples each epoch. The downside being that the low amount of positive samples per batch ($\frac{1}{R} \cdot \text{batch\_size}$) could reduce the stability of training. The four different data imbalance techniques will be compared in both prediction performance and training time, where different trade-offs between the two are expected between the different techniques. Furthermore, the techniques to utilize additional negative samples are expected to be more beneficial for PTM-types where the amount of total samples is already low. In general, increasing sample counts leads to more improvement in prediction quality when the initial sample count is low.

3.6 Species feature

The specie in which a PTM-sample was found, was used as an additional feature to the input sequence. The specie for each sample was determined by extracting the specie name from the UniProtID (e.g. ZSC32_HUMAN). In order to prevent overfitting, only the 20 most often occurring species received their own label, with the other species being labelled as "other". The specie feature was represented using a one-of-K encoding. There will be experimented with including the species feature in two different places in the model architecture. The first option is appending the species vector to the representation of each amino acid. This will allow the CNN, bi-LSTM, and attention layer access to the species. However the species vector is now unnecessarily repeatedly added 33 times, one time for each amino acid. This unnecessarily increased complexity of the model could lead to increased overfitting. The second option is only adding the species vector after the sequence layers, so appending it just before the first fully connected layer. While this means most of the model will not be able to determine specie-specific

relations, a single specie-vector will only have to be added to the model a single time, reducing overfitting risk.

3.7 Model training

The model architecture was implemented using PyTorch 1.10.2. All models were trained using the AdamW stochastic optimizer [39] (B1=0.9, B2=0.999), with a batch size of 512. The models were trained for a maximum of 200 epochs, while also using an early stopping strategy to reduce overfitting. That is, if the validation loss of model that is being trained does not increase for 10 epochs in a row, training will be stopped. A binary cross-entropy loss function is used, which is given in Equation 1, where n is the batch size, y_i is the model output for the i-th element, and $\hat{y}_i$ is the label for the i-th element.

$$\text{loss} = -\frac{1}{\text{n}} \sum_{i=1}^{n} y_i \cdot \log \hat{y}_i + (1 - y_i) \cdot \log (1 - \hat{y}_i) \tag{1}$$

Since the amount of samples available differs greatly between PTM-types, the most important hyperparameters will be optimized separately for each PTM-type. A framework called Optuna [40] will be used to find the optimal values for the learning rate and weight decay. Optuna uses a form of bayesian optimization to optimally select new hyperparameter values each trial. It can also prune unpromising trials, which can greatly reduce total run-time. To increase the reliability of a single trial when performing hyperparameter tuning, repeated (5x) 5-fold cross validation (CV) was used for PTM-types with less than 10k total samples, 5-fold CV was used for PTM-types with total samples in between 10k and 100k, and a 80%/20% training/validation split was only used for PTM-types with more than 100k total samples.

3.8 Evaluation metrics

The imbalance between negative and positive samples in our dataset makes the choice of evaluation metrics particularly important. Common evaluation metrics, such as accuracy, can show a deceivingly positive image when there is an imbalance between classes [41]. The performance of our models will primarily be evaluated using the area under the Receiver Operating Characteristics curve (AUC ROC). The performance of our final models on the test set will additionally be reported using 4 different common evaluation metrics. The following 3 are calculated using the confusion matrix, which is shown in Table 3: specificity, precision and sensitivity.

Table 3. The confusion matrix, where FP stands for False Positive, TN stand for True Negative, and FN stands for False Negative.

		Predicted	
		Yes	No
Actual	Yes	TP	FN
	No	FP	TN

MCC, specificity, precision and sensitivity are defined as follows:

$$\text{MCC} = \frac{TP \cdot TN - FP \cdot FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}} \tag{2}$$

$$\text{Specificity} = \frac{TN}{TN + FP} \tag{3}$$

$$\text{Precision} = \frac{TP}{TP + FP} \tag{4}$$

$$\text{Sensitivity} = \frac{TP}{TP + FN} \tag{5}$$

The AUC ROC represents the area under the graph which plots the TP-rate (sensitivity) versus the TN-rate (specificity) for all possible decision thresholds. A more intuitive understanding of the AUC ROC, is that it represents the probability that a randomly selected positive sample is ranked higher than a randomly selected negative sample [42]. The area under the precision-recall curve (AUC PR), is obtained by taking the area under the curve which plots the precision versus the recall for possible decision thresholds. AUC ROC and AUC PR measure the performance of the model over all possible decision thresholds, and are thus not dependent on the decision threshold used for the use-case at hand.

AUC ROC is the most commonly used metric for evaluation in PTM prediction research. AUC ROC is consistent when it comes to the imbalance in the dataset, and always has a baseline of 0.5. This means it can be used to compare models between different datasets, such as the validation and test set. While Matthew's correlation coefficient (MCC) also has this characteristic, AUC ROC has been shown to be a more discrimitative metric than MCC for evaluation on imbalanced datasets [43]. This makes it so that using AUC ROC, as opposed to MCC, makes it more likely to identify when a model is performing better than a different model.

3.9 Experimental setup

Three different types of experiments will be performed in order to design, evaluate, and compare our models. The three different types are: 1. Experiments on a validation set to decide the optimal architecture and features used in our final model, 2. Experiments on our own test set to validate the generalizability of the

chosen approach, and 3. Experiments on both our own and an independent test set in order to compare the effectiveness of our models to existing approaches.

Unless otherwise specified, hyperparameter tuning was performed separately for each experiment, as is described in Section 3.7. The experimental setup for each of the three different types of experiments will now be described.

Architectural Design Experiments that are used to decide the optimal architecture and features used in our model, are performed on only 3 out of the 13 PTM-types as included in our dataset. This is out of necessity due to both our computational and time-budget. The experiments are specifically performed on the types: hydroxylation-P, O-linked glycosylation and phosphorylation-Y. These are chosen to represent the PTM-types with a low-, medium- and high amount of available samples. Unless otherwise indicated, these experiments are performed using the architecture and adaptive embedding representation as described in Section 3.2, while training on a balanced dataset, and without the use of the species feature. These experiments are only evaluated on a validation set, by performing 5 times repeated 5-fold CV on the training set. The following three experiments will be executed to determine the optimal architectural design: 1. A comparison between three types of peptide representation, namely: *one-of-K*, *an embedding layer*, and the *adaptive embedding* layer, 2. A comparison between the 4 types of sampling methods, namely: *balanced*, *random undersampling*, *oversampling* and a *weighted loss function*, and 3. Measure the effect of increasing the complexity of the CNN- and FC-module of our architecture as explained in Section 3.3. The following four combinations of modules will be compared: *(CNN-Adapt, FC-Adapt)*, *(CNN-Musite, FC-Adapt)*, *(CNN-Adapt, FC-Musite)*, and *(CNN-Musite, FC-Musite)*.

As was mentioned in Section 3.5 and Section 3.3, the type of sampling method used, and the complexity of the DNN-layers, are both expected to potentially have a major effect on the training time for each model. Because of this, for the experiments regarding the sampling method and complexity of the DNN-layers, the average time that it took to train the model will also be reported.

Model Evaluation The experiments that are performed to validate the generalizability of our chosen approach, are performed for all 13 PTM-types, on both a validation set using 5-times repeated 5-fold CV, and on the separate test set, for which it was described in Section 3.1 how it was obtained. These experiments will be performed while using the architectural design as was found to be optimal in the experiments as described in the previous section. In addition to this, two other experiments will be performed on both the validation and test set: 1. The inclusion of the specie in which a PTM-sample was found as an additional input feature, and 2. Using an embedding as generated using the ProtTrans LM as an exclusive input to the model. Due to the size of the ProtTrans embeddings, and the accompanying increase in computational power needed, they will only be performed on the 6 PTM-types with the least total samples. The theoretical improvement in performance while using these embeddings, which would

be caused by easier to learn generalizability and a regularizing effect, should be most obvious in these PTM-types with the least amount of samples. Due to the computational budget, the hyperparameters of the models will not be retuned again for the experiments regarding the specie feature and the ProtTrans embedding.

Comparison with independent model In order to compare the effectiveness of our models to existing approaches, the performance of our model will be compared to the performance of MusiteDeep on both our own test set, and the test set as provided by MusiteDeep. MusiteDeep predictions were acquired using their publicly available webserver. Since their webserver limits the maximum amount of samples submitted, a random subset of 10.000 samples was selected for each PTM-type, which were also used for testing our own model. MusiteDeep has only made their test set publicly available for phosphorylation-Y and phosphorylation-ST, which is why testing using the independent test set of MusiteDeep was limited to these two PTM-types. In order to prevent an overlap in data, the date that was used to split samples between the training and test set in our own dataset was changed to 2010 for all PTM-types. This is the same training and test split that MusiteDeep uses.

The test set as constructed in this research paper, and the test set as provided by MusiteDeep, have different distributions when it comes to the species from which the samples have been extracted, which can be found in Appendix 9.4. In order to make a fair ~~direct~~ comparison between the prediction performance on the two test sets possible, the evaluation results will also be separately reported for the different species which occur most often in the two test sets. For phosphorylation-Y the 3 most often occurring species in the test set of MusiteDeep are: mouse-ear cress, rat and mouse, and the 3 most often occurring species in our test set are: human, mouse and baker's yeast. For phosphorylation-Y the evaluation metrics will thus be reported separately for the species mouse-ear cress, rat, mouse, human and baker's yeast. For phosphorylation-ST the 3 most often occurring species in the test set of MusiteDeep are: mouse-ear cress, mouse and rat, and the 3 most often occurring species in our test set are: human, mouse-ear cress and yeast. For phosphorylation-ST the evaluation metrics will thus be reported separately for the species mouse-ear cress, mouse, rat, human and yeast.

Another major difference between how their test set and our test set is generated, is that for the test set constructed in this paper, sequence filtering was performed both between the training and test set and for samples within the training and test set, while MusiteDeep does not. In order to investigate the effect of this difference between the test sets, two additional experiments were performed. For the first experiment, additional sequence filtering, both between the training sets and test set and within the test set, was introduced with a threshold of 50% to the test set of MusiteDeep. For the second experiment, the sequence filtering was removed from our own test set.

3.10 Experiments: Statistical analysis

In order to compare the different model designs as evaluated on the validation set, statistical analysis will be performed. For each different experiment, and for each amino acid, it will be calculated if the best performing design has a statistically significant different mean prediction quality than the other design options. While such an analysis between models which are evaluated using repeated k-fold CV is still often performed using the paired Student's t test, its application would be highly flawed. Using the Student's t test here would result in a high type I error, where a significant difference could be perceived while there is in fact not one. This is due to a violation of the key assumption of samples being independent of each other, which is caused by test-data being re-used between the several repeated folds [44]. In Bouckaert et al, a corrected repeated k-fold CV t-test is presented, where they correct the flawed paired t-test by taking the dependency between folds into account [45]. They also empirically show the validity of the this corrected t-test. The formula for this corrected r-times k-fold CV t-test can be found in Equation 6, where x_{ij} is the difference in the observation for that sample, n_2 is the amount of samples used for testing, n_1 is the amount of samples used for training, and $\hat{\sigma}$ is the estimated standard deviation.

$$t = \frac{\frac{1}{k.r}\sum_{i=1}^{k}\sum_{j=1}^{r} x_{ij}}{\sqrt{\left(\frac{1}{k \cdot r} + \frac{n_2}{n_1}\right)\hat{\sigma}^2}} \tag{6}$$

Using the corrected t-test, the null-hypothesis will be tested that for each experiment performed, the best performing model has the same sample distribution as the worse performing models. If this were to be rejected, the conclusion can be made that the difference in mean prediction quality of the models is statistically significant. Four different values will be used to indicate the level of significance in the results, namely α=0.1, α=0.05, α=0.01, and α=0.001. In order to test the assumption of normality, which is required for the paired t-test, the null-hypothesis that the samples are normally distributed was tested using the Shapiro-Wilk test for all CV samples.(α=0.05).

4 Data analysis

4.1 Sequence Logos: specie differences

As was discussed in section 2.5, it has been shown for some PTM-types that PTM patterns surrounding PTM sites show differences between species. However, the same analysis has not been performed for a lot of other PTM-types. In an attempt to analyse if different species show different patterns for all our PTM-types, sequence logos were generated for both positive and negatives sequence samples. These samples were split into human, animal, and non-animal samples. Sequence logos are graphical representations of the over-represented patterns between multiple sequences [46]. The height of each symbol at each position shows the amount over-representation of that amino acid in your sequences against the usual background distribution of amino acids as found in your dataset. As the background distribution, the amino acid distribution for that particular PTM-type in our dataset was taken. The amount of over-representation is calculated as the difference between the maximum possible entropy and the entropy of the observed symbol distribution, measured in bits. The equation used to calculate this can be found in equation 7, where R_i is the information content, A is the set of all amino acids, p_a is the observed probability of amino acid a, and q_a is background probability of amino acid a.

$$R_i = S_{max} - S_{obs} = \log_2 20 - \left(- \sum_{n=1}^{A} p_a \log_2 \frac{p_a}{q_a} \right) \tag{7}$$

The sequence logo's were generated using the Python package Logomaker [47]. The resulting sequence logos can be found in Appendix 9.2. A general trend can be seen where positive sites show strong over-representation of various amino acids, while no patterns can be found for negative sites. Various PTM-types show a difference in ~~between~~ patterns found in their humans, animals, and non-animal positive samples. The PTM-types which show the biggest difference between their logo's for non-animals and animals are methylation-k, methylation-R, hydroxylation-P, S-palmitoylation-C and SUMOylation, and can be found in Figure 4.1.

While not dependent on the species, N-linked glycolysation shows a very clear pattern for positive PTM-samples, which is shown in Figure 7. Either serine or threonine is present at a distance of 2 from the affected PTM site when a sample is positive. This is a known pattern, with only a few known exceptions [48].

4.2 ProtTrans embeddings: t-SNE Visualization

In an attempt to analyse the usefulness of the ProtTrans embeddings for PTM prediction, a technique called t-SNE dimensionality reduction was used in order the visualize the embeddings on a two-dimensional plane. t-SNE is a non-linear dimensionality reduction technique which is especially effective for high-dimensional data such as embeddings [49]. The t-SNE implementation of sk-learn

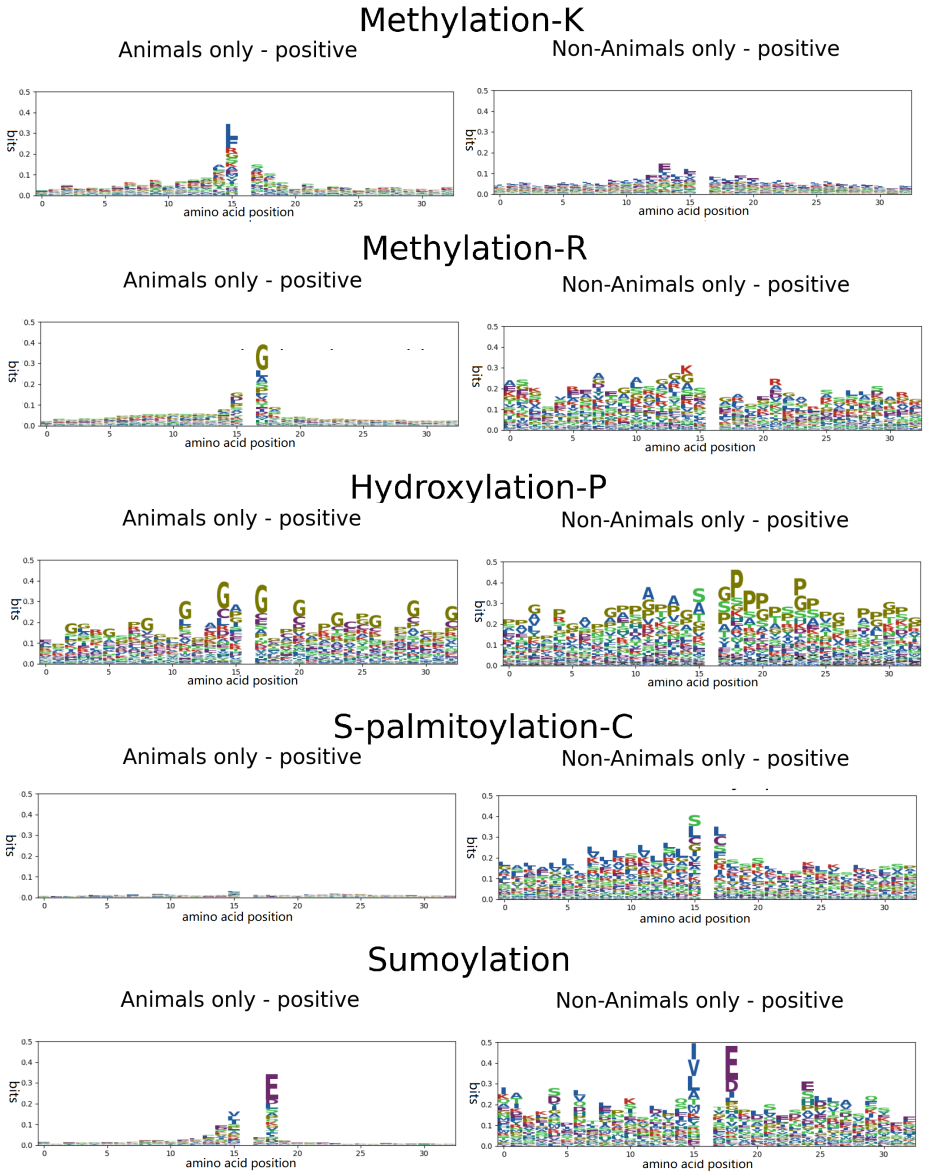

Fig. 6. Sequence logos of positive animal and non-animal samples for the PTM-types which showed the most difference between the two logos.

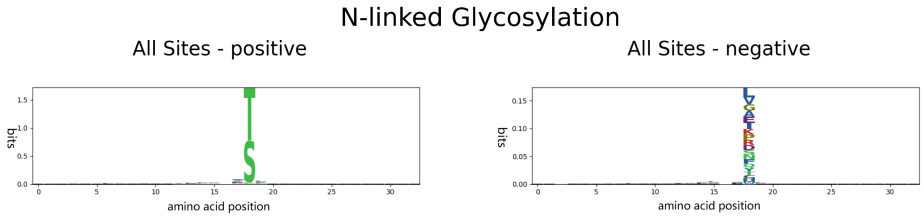

Fig. 7. Sequence logos for positive and negatives samples for N-linked glycosylation, which shows a very strong sequence logo for positive sites.

was used for this [50], using the default parameters. Before applying t-SNE, The ProtTrans embeddings of size 33x1024 were first reduced to a one-dimensional vector of size 1024 by taking the mean over the sequence dimension. The resulting visualizations showed that for several of the PTM-types, the positive and negative samples were separable to a degree based on just the dimensionality-reduced embeddings. The PTM-types which showed the highest degree of separability can be found in Figure 8, while a figure including the t-SNE visualizations of all PTM-types can be found in Appendix 9.1. While an absence of separability would not show that the embeddings are not useful for us, the presence does show that the embeddings capture some features of the peptide which are useful for determining the presence of a PTM.

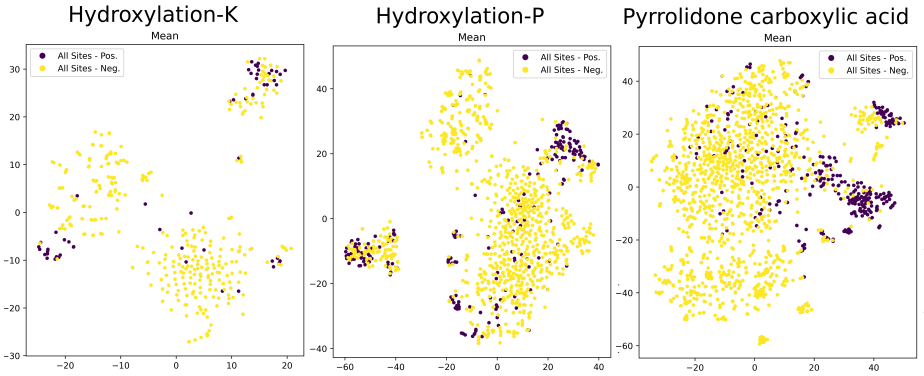

Fig. 8. t-SNE visualizations of the ProTrans embeddings for the PTM-Types hydroxylation-K, hydroxylation-P, and pyrroliodine carboxylic acid. These PTM-types showed the biggest amount of separation between the negative and positive samples. The embeddings for all other types can be found in Appendix 9.1

5 Results: Experiments

5.1 Validation set

The following experiment results, which were performed on the validation set, were utilized to determine the optimal architecture and features as used in our final models.

Amino acid representation The resulting average AUC ROC while using a one-hot representation, an embedding-layer, and the adaptive embedding for the three representative PTM-types while using CV can be found in Figure 9. The adaptive embedding was the best performing representation for all PTM-types, with an improvement in average AUC ROC over the one-hot representation of 0.024, 0.009, and 0.008 for the PTM-types hydroxylation-P, O-linked glycosylation and Phosphorylation-Y. The resulting average AUC ROC's for the adaptive embedding are significantly different from the Embedding-Layer for all PTM-types, and significantly different from the one-hot representation for 2 out of 3 PTM-types.

Data imbalance techniques The resulting average AUC ROC while using a balanced dataset, random undersampling, oversampling, and a weighted loss-function for the representative PTM-types while using CV can be found in Figure 10. For hydroxylation-P, oversampling showed the best results, with an increased average AUC ROC of 0.020 over the balanced dataset. The increased AUC ROC of oversampling over a balanced dataset decreased to 0.01 for O-linked glycosylation, and disappeared for Phosphorylation-Y. hydroxylation-P was the only amino acid for which a statistical significant difference between oversampling and a balanced dataset could be established. Interestingly, while random undersampling and a weighted loss-functions showed slightly increased performance when compared to the balanced dataset for hydroxylation-P, they actually showed decreased performance for O-linked glycosylation. The average training time for each sampling method can be found in Table 4. Oversampling resulted in the longest training time, with the average training time being 1.3x, 4.8x, and 3.1x longer than when using a balanced dataset.

Model complexity The resulting average AUC ROC while utilizing DNN modules with different complexities for the representative PTM-types while using CV can be found in Figure 11. As detailed in Section 3.3, CNN-Musite is a more complex replacement for CNN-Adapt, and FC-Musite is a more complex replacement for FC-Adapt. For both the CNN and the FC module, a general trend was found where increasing the complexity of both the CNN and FC module resulted in an slightly improved average AUC ROC. The only exception here being O-linked glycosylation, where utilizing the modules (CNN-Musite, FC-Musite) instead of (CNN-Musite, FC-Adapt) let to decreased performance. For O-linked glycosylation, a significant difference in average AUC ROC between the more complex

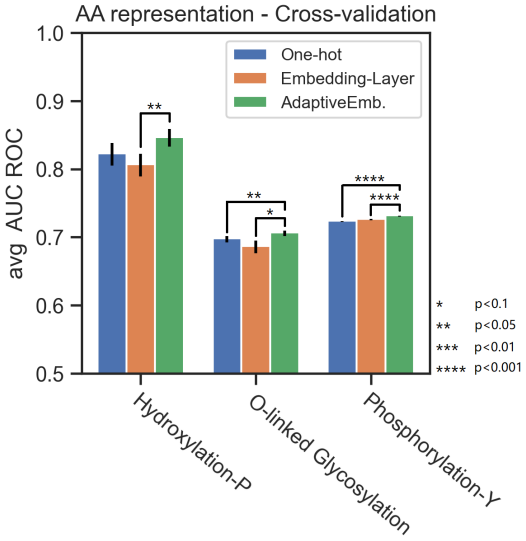

Fig. 9. CV results for the type of amino acid representation, measured in average AUC ROC. The black bar indicates the sample standard deviation. The presence of a significant difference between the best and other results are indicated by the amount of asterisks.

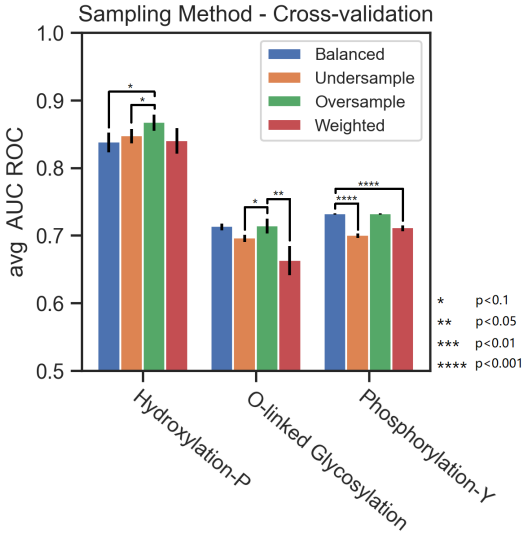

Fig. 10. CV results for the type of sampling method used, measured in average AUC ROC. The black bar indicates the sample standard deviation. The presence of a significant difference between the best and other results are indicated by the amount of asterisks.

Table 4. The average training time for the different sampling methods on the three different PTM-types tested

Amino acid	Sampling Method	Avg. train time
Hydroxylation-P	Balanced	19.43s (+-3.75)
	Undersample	20.64s (+-10.86)
	Oversample	26.14s (+-7.54)
	Weighted	27.79s (+-4.75)
O-linked glycosylation	Balanced	31.72s (+-13.09)
	Undersample	39.81s (+-11.99)
	Oversample	151.93s (+-71.1)
	Weighted	125.22s (+-48.62)
Phosphorylation-Y	Balanced	277.00s (+-43.82)
	Undersample	194.24s (+-33.89)
	Oversample	862.01s (+-277.08)
	Weighted	652.04s (+-314.33)

(CNN-Musite, FC-adapt) and the standard (CNN-Adapt, FC-adapt) was able to be established. For Phosphorylation-Y, a significant difference in average AUC ROC was able to be established between (CNN-Musite, FC-Musite) and the two less complex designs (CNN-Adapt, FC-adapt) and (CNN-Adapt, FC-Musite.). The average training time for each architecture can be found in Table 5. The average training time increases as the complexity of the architecture increases, with the most complex model having a training time that is 1.1x, 1.7x, and 4.5x longer than the simplest architecture.

ProtTrans Embeddings The resulting average AUC ROC while utilizing ProtTrans embeddings on the 6 PTM-types with the least amount of samples and evaluated using CV can be found in Figure 12. ProtTrans embeddings showed an increased average AUC ROC when compared to the Adaptive embedding for 4 out of 6 PTM-types. Only for 1 out of 6 PTM-types, methylation-K, it could be shown that the ProtTrans embeddings had a statistically significant improved outcome, where it increased the average AUC ROC by 0.017. On the other hand, for 1 PTM-type the opposite was true. SUMOylation showed statistically significant decreased performance when utilizing ProtTrans Embeddings when compared to the adaptive embedding.

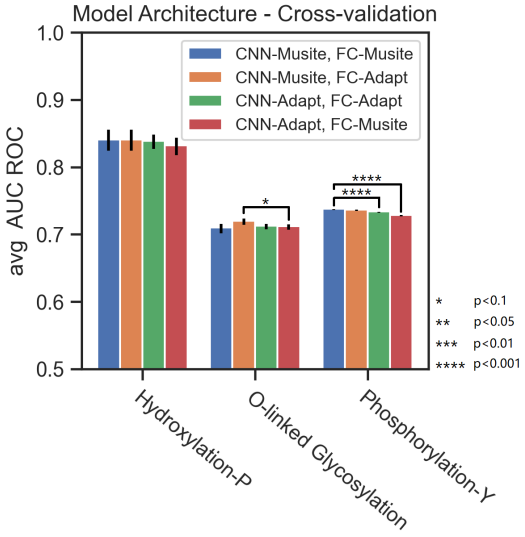

Fig. 11. CV results for different variations of the model architecture, measured in average AUC ROC. The black bar indicates the sample standard deviation. The presence of a significant difference between the best and other results are indicated by the amount of asterisks.

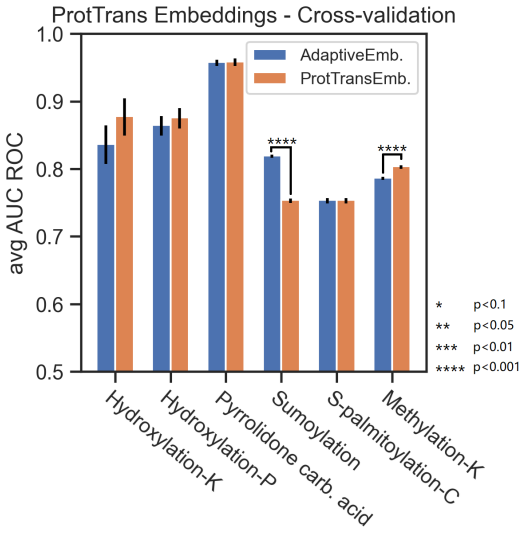

Fig. 12. CV results for using ProtTrans embeddings as input for the model, measured in average AUC ROC. The black bar indicates the sample standard deviation. The presence of a significant difference between the best and other results are indicated by the amount of asterisks.

Table 5. The average training time for the model architectures using the different CNN- and FC-modules on the three different PTM-types tested.

Amino acid	CNN-Type	FC-Type	Avg. Train time
Hydroxylation-P	Adapt	Adapt	25.25 (+-6.93)
	Adapt	Musite	20.09 (+-4.26)
	Musite	Adapt	23.45 (+-4.52)
	Musite	Musite	24.42 (+-8.50)
O-linked glycosylation	Adapt	Adapt	29.71 (+-14.14)
	Adapt	Musite	29.71 (+-14.14)
	Musite	Adapt	38.26 (+-24.71)
	Musite	Musite	50.56 (+-9.53)
Phosphorylation-Y	Adapt	Adapt	159.60 (+-4.53)
	Adapt	Musite	292.42 (+-69.56)
	Musite	Adapt	645.26 (+-50.39)
	Musite	Musite	711.53 (+-27.41)

Species feature The resulting average AUC ROC when including the species as a feature in either the CNN-layer or the FC-layer for all 13 PTM-types and evaluated using CV can be found in Figure 13. 12 of the 13 PTM-types showed an increased average AUC ROC when the specie feature was utilized as opposed to when it was not. For 10 out of 13 PTM-types, a statistical significance difference could be shown for the increased performance when utilizing species as a feature. Including the specie feature in the CNN-layer instead of the FC-layer showed slightly increased performance for all PTM-types but the two with the least amount of samples. 7 PTM-types showed a significant increased performance when including the species feature in the CNN-layer when compared to the FC-layer. 1 PTM-type, hydroxylation-P, showed a significant decreased performance when including the species feature in the CNN-layer when compared to the FC-layer.

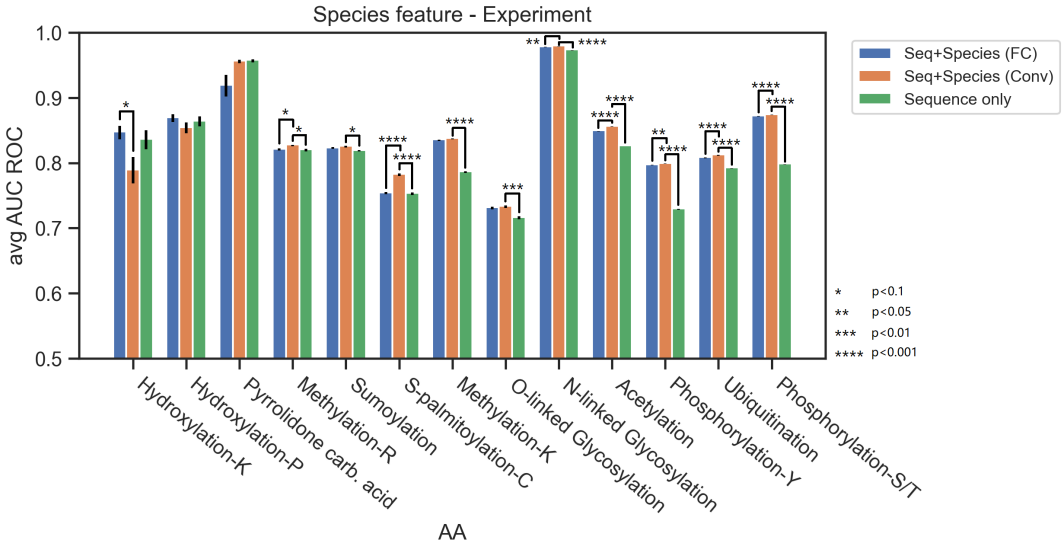

Fig. 13. CV results for the usage of the species in which a sample was found as an additional input feature, measured in average AUC ROC. Both adding species as a feature in the CNN-layer, and in the FC-layer was tested. The black bar indicates the sample standard deviation. The presence of a significant difference between the best and other results are indicated by the amount of asterisks.

5.2 Test set

Final Model Based on our validation results, our final models utilize an adaptive embedding layer, a CNN-Musite module, a FC-Adapt module, and uses species as an additional feature. For PTM-types with up to 51.000 total samples, oversampling was used, while the dataset was balanced for PTM-types with more than 51.000 samples. The resulting AUC ROC of our final models for all PTM-types evaluated on our test set, and compared with the AUC ROC's on the validation set, can be found in Figure 14. Depending on the PTM-type, the average AUC ROC varies between 0.734 and 0.980 for the validation set, and between 0.746 and 0.975 for the test set. On average, the AUC ROC on the test set was 0.013 lower than when performing CV.

Species feature The resulting AUC ROC while including and excluding species as a feature, evaluated on our test set, can be found in Figure 15. 12 of the 13 PTM-types showed improved results when the species was included as a feature, with an average increase in AUC ROC of 0.018. The PTM-types which showed the most improvement were methylation-K (+0.056), phosphorylation-Y (+0.056) and phosphorylation-ST (+0.038).

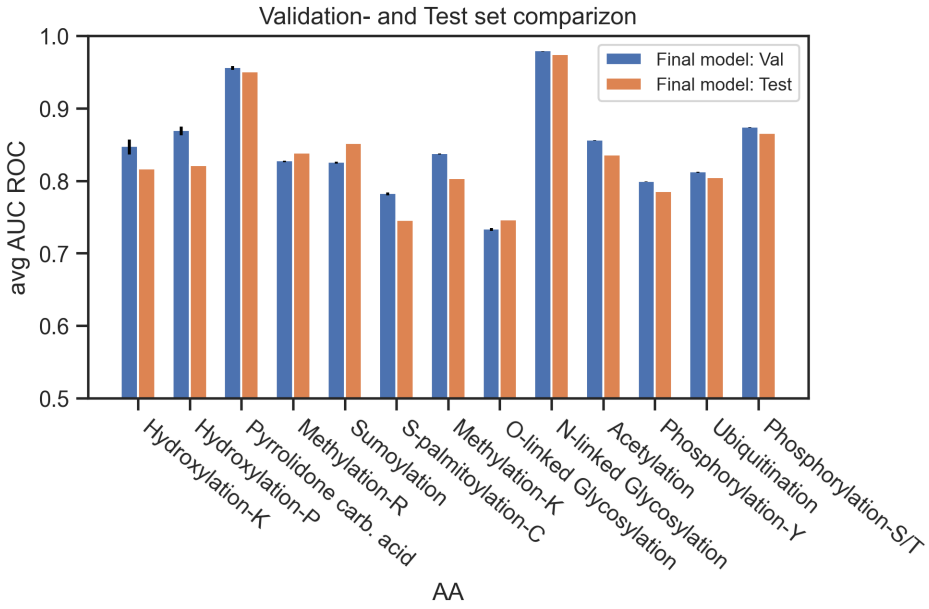

Fig. 14. Evaluation results on our test set for our final optimized model on all 13 PTM-types, where the results on the validation set (using CV) and the results on the test set are compared. The black bar indicates the sample standard deviation.

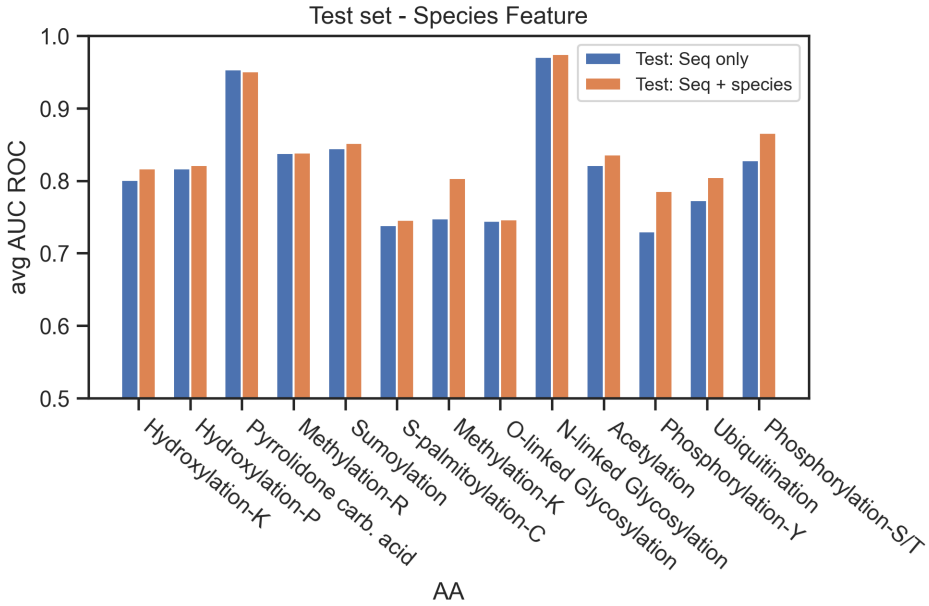

Fig. 15. Evaluation results on our test set for the usage of the species in which a sample was found as an feature, measured in AUC ROC, compared to the test set results when not using species as a feature. The species feature was either added to the CNN-layer or the FC-layer, depending on the best performing option in the validation tests.

ProtTrans Embeddings The resulting AUC ROC while utilizing ProtTrans embeddings on the 6 PTM-types with the least amount of samples and evaluated on our test set Figure 16. 3 out of 6 PTM-types showed improved results when using the ProtTrans embeddings. The PTM-types which showed the most improvement were hydroxylation-K (+0.025) and methylation-K (+0.014).

5.3 Comparison: MusiteDeep

In Figure 17, the resulting average AUC ROC of the Musite Webserver on our test set can be found, compared with the results of our own model. Our model achieves improved results when compared to MusiteDeep for 12 out of 13 PTM-types. On average, our model achieves an AUC ROC that is 0.093 higher than MusiteDeep. A table containing all evaluation metrics as described in Section 3.8, for both our models, and the models from MusiteDeep, can be found in Appendix 9.3.

In Table 6, the overall and specie-specific results of both our model and MusiteDeep on our phosphorylation-ST test set can be found. Our model outperformed MusiteDeep both overall, with an AUC ROC that is 0.118 higher, and on all 5 species. In Table 7, the overall and specie-specific results of both

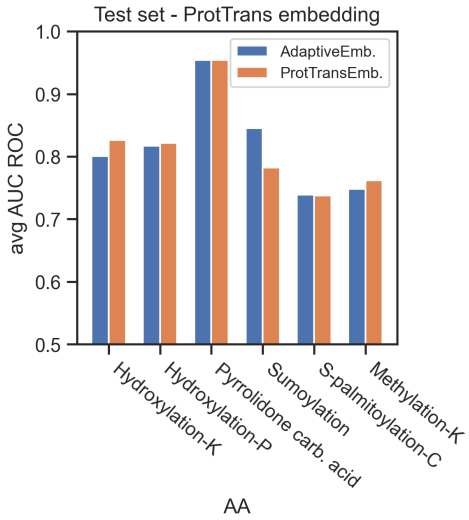

Fig. 16. Evaluation results on our test set for using ProtTrans embeddings as input for the model, measured in AUC ROC, compared to using the adaptive embedding layer.

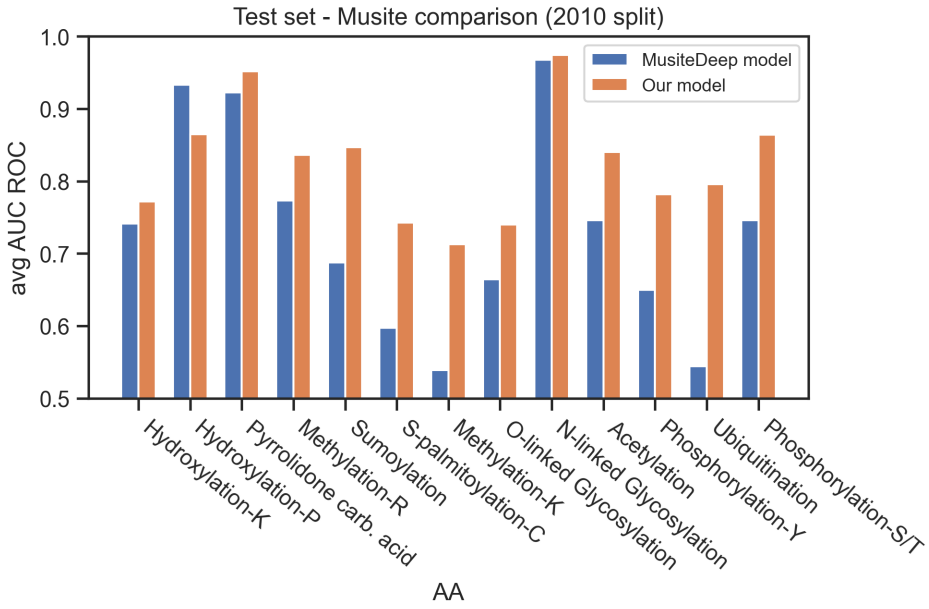

Fig. 17. This figure compares the evaluation results, measured in AUC ROC, of the models as presented in this study with evaluation results of the MusiteDeep webserver for all 13 PTM types as included in our test dataset.

our model and MusiteDeep on the phosphorylation-ST test set of MusiteDeep can be found. While on our own test set, our model outperformed MusiteDeep, the opposite is true for the MusiteDeep test set. MusiteDeep outperformed our model both overall, with an AUC ROC that is 0.172 higher, and for 4 of the 5 specie specific results. Our model showed a higher AUC ROC on the specie mouse-ear cress, which is also the only plant out of the 5 species

In Table 8, the overall and specie-specific results of both our model and MusiteDeep on our phosphorylation-ST test set can be found. Our model outperformed MusiteDeep both overall, with an AUC ROC that is 0.132 higher, and on all 5 species. In Table 9, the overall and specie-specific results of both our model and MusiteDeep on the phosphorylation-ST test set of MusiteDeep can be found. While on our own test set, our model outperformed MusiteDeep, the opposite is true for the MusiteDeep test set. MusiteDeep outperformed our model both overall, with an AUC ROC that is 0.314 higher, and on all 5 species.

Table 6. Phosphorylation-ST Specie specific test set results on **our test set**, measured in AUC ROC. An X indicates the test set did not contain samples for that specie.

	Overall	Mouse-ear cress	Mouse	Rat	Human	Yeast
Our model	**0.864**	**0.792**	**0.720**	X	**0.818**	**0.877**
MusiteDeep	0.746	0.652	0.646	X	0.739	0.816

Table 7. Phosphorylation-ST Specie specific test set results on the **MusiteDeep test set**, measured in AUC ROC.

	Overall	Mouse-ear cress	Mouse	Rat	Human	Yeast
Our model	0.717	**0.866**	0.830	0.825	0.723	0.830
MusiteDeep	**0.889**	0.8213	**0.909**	**0.924**	**0.906**	**0.885**

Table 8. Phosphorylation-Y Specie specific test set results on **our test set**, measured in AUC ROC.

	Overall	Mouse-ear cress	Rat	Mouse	Human	Baker's yeast
Our model	**0.782**	**0.766**	**0.740**	**0.700**	**0.747**	**0.867**
MusiteDeep	0.650	0.652	0.708	0.646	0.738	0.746

Table 9. Phosphorylation-Y Specie specific test set results on the **MusiteDeep test set**, measured in AUC ROC. An X indicates the test set did not contain samples for that specie.

	Overall	Mouse-ear cress	Rat	Mouse	Human	Baker's yeast
Our model	0.641	0.608	0.635	0.715	0.638	X
MusiteDeep	**0.955**	**0.957**	**0.986**	**0.975**	**0.954**	X

AUC ROC results for the MusiteDeep test set where we applied sequence filtering between both the training sets and the test set, but also between samples in the test set, can be found in Table 11. After applying sequence filtering, the difference in AUC ROC between our model and MusiteDeep became smaller, but MusiteDeep still outperformed out model for both PTM-types. Interestingly, introducing sequence filtering improved prediction performance for our own model for both PTM-types, and also improved prediction performance on phosphorylation-Y for MusiteDeep, while this is expected to reduce the score.

AUC ROC results for our test set where any sequence filtering on the test set was removed can be found in Table 11. After removing sequence filtering, the difference in AUC ROC between our model and MusiteDeep became smaller, but our model still outperforms MusiteDeep for both PTM-types. Interestingly, removing sequence filtering lowered prediction performance for our own model for both PTM-types, while this is expected to increase prediction performance.

Table 10. Test set evaluation results on **our test set**, where we removed the sequence identity filtering between the training and test set, measured in AUC ROC.

PTM	Our model	MusiteDeep
Phos-S/T	**0.801**	0.710
Phos-Y	**0.718**	0.645

Table 11. test set evaluation results on the **MusiteDeep test set** with extra sequence identity filtering applied, measured in AUC ROC.

PTM	Our model	MusiteDeep
Phos-S/T	0.789	**0.799**
Phos-Y	0.712	**0.994**

6 Discussion

In this study, a non-redundant training- and test set was constructed for 13 of the most common occurring PTM-types. The samples in our dataset for the different PTM-types were analyzed using t-SNE visualization of ProtTrans embeddings to show if these embeddings contained information which could lead to improved prediction performance, which was found for several PTM-types. Using sequence logos, differences in patterns between species were also anylyzed for the different PTM-types. These were found for several PTM-types, suggesting that using the specie in which as samples was found as an additional input feature could lead to improved performance. The architecture of a PTM prediction model found in the literature, named Adapt-Kcr, was adopted as the basis of our own PTM prediction model. Using 3 representative PTM-types, experiments on a validation set were performed in order to further optimize the design of this architecture. The results showed the effectiveness of the Adaptive embedding layer of Adapt-Kcr, and that increasing the amount of layers in the convolutional and fully-connected layers of Adapt-Kcr slightly increased performance. The experiments also showed that oversampling could increase the performance for PTM-types with a low amount of samples, while random undersampling and a weighted loss function could actually degrade performance for PTM-types with a high amount of samples. Using the optimized model architecture, the trained models for all 13 PTM-types were evaluated on both the validation and test set, showing similar performance on both sets. Using the species as an input feature showed increased performance for 12 out of the 13 PTM-types, on both the validation and test set. Using the ProtTrans embeddings showed increased performance for 3 out of the 6 PTM-types, with the increased performance being only statistically significant for methylation-K. Lastly, the prediction performance of our models was compared with MusiteDeep on both our own test set and the test set of MusiteDeep. While our model showed vastly superior performance on our test set, the reverse was true for the test set of MusiteDeep. Additional performed tests showed that this discrepancy between prediction performance between the two test sets was not caused by a difference in specie distribution, or a difference in sequence redundancy filtering.

6.1 ProtTrans embeddings

The results for using ProTrans embeddings varied quite a lot between PTM-types. Using the amount of separability of samples in t-SNE visualizations, three PTM-types were identified which were most likely to benefit from using ProTrans embeddings. While two of the three identified PTM-types did show increased performance, the third one did not, showing that sample separability of the t-SNE visualization is not a guarantee of improved performance. Vice versa, methylation-k did not show a lot of separability while showing improved performance when using ProtTrans embeddings, showing that a lack of separability is also not a guarantee of ProtTrans embeddings being useless for a PTM-type. On our validation set, we were able to show with a high amount of

statistical certainty that ProtTrans embeddings showed improved performance for methylation-K, while showing decreased performance for SUMOylation. This was confirmed by the results on our test set. This shows that the usefulness of the ProtTrans embeddings depends on the PTM-type, which could depend on if the biophysical properties ~~which are relevant~~ are captured by the embedding.

6.2 Species as a feature

By inspecting sequence logo differences between species, the PTM-types which showed the most difference in PTM site patterns between species were identified, and are thus expected to benefit the most when utilizing species as a feature. However, 4 out of the 5 PTM-types which showed the biggest difference between species showed a relatively small increase in AUC ROC when compared to other PTM-types. This indicates differences in sequence logos are not a great indicator for the potential of specie-specific prediction. This is also consistent with the fact that some of the PTM-types that do not show any sequence logo pattern at all, still show relatively strong prediction performance. As this shows that the patterns used by our model are more complex than that can be captured by using a sequence logo. The results on our validation set showed improved performance for 12 out of the 13 PTM-types when using species as an feature, which were statistically significant in 10 out of the 13 cases. This was also confirmed by the results on our test set, which showed an average increase of 0.018 AUC ROC when using the species feature. The uplift in performance differed quite a lot between PTM-types, varying between 0.001 and 0.056. This shows that while the amount of difference between PTM-patterns depends on the PTM-type, almost all PTM-types showed some difference between species. A general trend could be seen where PTM-types with more samples showed a higher uplift in performance. This is not surprising, as learning specie-specific patterns needs enough samples for each specie occurring in the dataset.

6.3 Comparison with MusiteDeep

On our own test set, our model showed vastly superior performance to MusiteDeep. The results for MusiteDeep here are very inconsistent with the results on the same PTM-types as found in their research paper. On the other hand MusiteDeep vastly outperforms our model on their test set. This is inconsistent with the results that Adapt-Kcr showed on the MusiteDeep test set for phosphorylation-ST, since our used architecture is based on Adapt-Kcr. These results suggest that the samples found in the two datasets differ to such a degree, that prediction performance does not transfer well between test sets. One of the suspected differences between the two datasets which might cause such a discrepancy, was the difference in sample specie distribution between the two datasets. However, specie specific evaluation results showed the same inconsistent results between the two, which invalidates this theory. A second suspected difference between the datasets which might cause such a discrepancy is how redundant sequence filtering was performed. For our own dataset, we performed

both sequence filtering between the samples in training- and test-dataset, but also in between samples in both the training and test-dataset. For the MusiteDeep dataset, no sequence filtering was performed at all, although they show that prediction performance holds up relatively well in their research paper when sequence filtering is performed. The effect of this difference in redundant sequence filtering was analyzed by removing the sequence filtering on our own test set, while adding the same type of sequence filtering as was originally applied to our own test set to the MusiteDeep test set. While this reduced the difference in performance for both test set, the discrepancy between test sets remained, indicating that it is also not caused by the difference in sequence filtering.

As a source for the PTM-samples in our dataset, dbPTM was used, while for MusiteDeep samples were collected from UniProtKB/Swiss-Prot. The total amount of PTM-samples are much higher for dbPTM than UniProtKB. However, why this is the case is found in the difference between data collection methods between the two databases. In UniProtKB/Swiss-Prot, the number of false positives is kept to a minimum by using an expert-driven analysis pipeline, where individual publications are manually curated by experts. They analyze the level of experimental reporting, the relevance of the scientific articles and methods used, and the precision of the found results. They also enforce a minimum PTM localization probability (LP) of 0.99 before including found sites in their database [51]. The PTM localization probability is a metric used to quantify the probability that a found PTM site is a true positive, with the cutoff being set to an acceptable false localization rate, which is generally set to 0.75 for publication [21]. Meanwhile, dbPTM utilizes no specific data quality requirements for samples to be included to their dataset, with samples being collected from both other databases, and all PTM research papers as found in PubMed [29]. The resulting difference in data quality originating from the different data collections methods between dbPTM and UniProt might thus explain the discrepancy in test-results between the two test sets.

6.4 Limitations

There are several limitations in how ProtTrans embeddings were implemented in this study. One of the smaller models made available by ProtTrans (protBert-BFD) was used due to hardware limitations. Using their biggest model (ProtT5-XL), has been shown to lead to ~~anywhere~~ between 2% to 46% increase in accuracy depending on the protein prediction task. The deep-learning architecture used in this model was also not optimized for using such a high-dimensionality embedding as an input, and the hyperparameters used were not tuned separately when using the ProtTrans embeddings. Another potential improvement is the finetuning of the ProtTrans model for our PTM prediction tasks. While the ProtTrans research paper initially showed that fine-tuning was not beneficial for prediction tasks, more recent research has shown that with a modified method finetuning can actually be used to increase prediction task performance [52].

Regarding the species feature, a limitation of this study is that a relative simple encoding scheme was used for all PTM-types, where the 20 most often

occurring species in the dataset were encoded using a one-hot encoding. It is likely that a lot of difference in PTM-patterns are not specie specific, but shared among species that are close in biological classification. Such as has been shown for ubiquitylation, where differences in PTM patterns seem to especially exists between the groups plants, animals, and fungi [9]. Depending on the PTM-type, it might work better to represent the specie feature on different levels of taxonomy. For example, on the level of the domain of the species, using a feature to represent if the sample comes from a specie that is a Bacteria, Archaea or Eukarya. This might especially be effective for the PTM-types with a low amount samples, which show relatively low benefit to using the species feature in our results. Using a higher taxonomy rank as the input feature would result in less taxonomic groups, and thus more samples for each group. Another limitation is that while this study shows that using species as a feature leads to increased performance for almost all PTM-types, a comparison with specie-specific models has not been made. Specie-specific models have been shown to work very well for several PTM-types. The optimal approach between utilizing species as a feature, or using separate specie-specific models, might depend on the total amount of samples available for a particular PTM, as specie-specific models need a certain amount of samples for each specie to be effective.

In this study, the choice was made to perform sequence filtering both between the training and test set, but also within the training and test set. This has the advantage that CV can be performed on the training set without data leakage occurring, since no redundant sequences exist between the training and validation set. Sequence filtering is expected to lower the performance of the model on the test set, as it reduces data leakage between the training and test set. However, when performing tests on our test set where sequence filtering is removed, the reverse effect is seen, and prediction performance of both our own model and MusiteDeep degrades. This suggests that the identity filtering performed within the test set created a biased dataset, with easier to predict samples being over-represented. Since we applied the same identity filtering between samples in the training set, this could lead to a biased training dataset, and ultimately a biased model. While more difficult to predict samples are actually desirable in the training set, they might thus be underrepresented. Test results on the MusiteDeep test set where we added both sequence filtering between the training sets and the test set, and sequence filtering within the test set, showed similar results. The prediction performance of our model increased for both PTM-types.

The data quality of the source of our collected PTM-samples was highlighted as a concern by comparing the data collection methods of dbPTM and MusiteDeep. No data quality requirements or minimum false positive rate is used for the samples as included in dbPTM. This might results in poor data quality of our dataset, which limits both the prediction performance and generalizability to other datasets. Another limitation in data quality, which is shared by all PTM datasets, and not this one in particular, is the presence of false negatives among the negatives samples in the dataset. Creating negative PTM-samples is difficult, because experimental negative results are rare. Thus, negative samples are

synthesized by assuming that amino acids for which no PTM has been found, but for which at least one PTM has been found on the same protein, are negative sites. This process could potentially be improved by limiting the negative set to amino acids for which their mass has been measured precisely by mass spectroscopy, and whos experimentally measured mass is within the theoretical molecular weight of the peptide [53].

6.5 Recommendations

Based on the results and limitation of this study, various recommendations can be made.

While the results for using ProtTrans embeddings for PTM prediction varied between PTM-types, its value was shown for at least one PTM-type. While using these embeddings can not currently be advised for all PTM-types, more research into them is needed, as the implementation of ProtTrans embeddings in this study has various limitations. For future work, the model ProtT5-XL-UniRef50 should be used, while utilizing a DNN architecture specifically optimized for using these embeddings. Furthermore, finetuning could be employed to further increase performance for the PTM prediction task.

The results in this study show that using the specie in which a sample is found, can be used as an effective feature to improve prediction performance across a wide array of PTM-types. Since the feature is very easy to obtain, inclusion of this feature in future PTM prediction models is recommended. In future work, the species feature could potentially be further improved by grouping species together depending on the PTM-type, for example based on its taxonomy. It also remains to be compared with the other approach for improving prediction performance for specific species, namely specie-specific models.

Lastly, a discrepancy was identified between samples of the same PTM-type between datasets, such that prediction results did not transfer well between the two datasets. The sequence identity filtering which was performed within the training and test set in this study was identified as a contributor, which results in a biased training and test set. This leads to the recommendation that such sequence filtering within the training and test set should not be performed in future works. However, even when doing tests without this sequence filtering applied, the discrepancy between datasets remained. We suggest that evaluation of PTM prediction models should be performed on multiple independent test sets, as performance on a single test set does not guarantee performance on other test sets. A lack of data quality and the potential presence of false positives in our dataset were identified as a possible explanation for the discrepancy of model performance between datasets. Because of this, the recommendation is made for future research to make use of PTM data-sources which use data collection methods which guarantee the quality of collected data, such as UniProtKB/Swiss-Prot. Furthermore, filtering dbPTM to a subset which only exhibit high data-quality could potentially be used to identify if a difference in data quality is indeed the explanation of our discrepancy between database results.

7 Conclusion

In this study, a non-redundant dataset was constructed for 13 of the most common occurring PTM-types. Data analysis showed the potential of both ProtTrans embeddings and a specie feature to improve the performance of PTM predictions models. An architecture called Adapt-Kcr was adapted from the literature to create PTM prediction models for all PTM-types as found in this constructed dataset, which was further optimized using cross-validation on the training set. The use of ProtTrans embeddings was evaluated on both the validation set and the test set. The results varied between PTM-types, with 3 out of the 6 PTM-types showing increased performance, where Methylation-K showed a statistically significant improved AUC ROC of 0.017. The use of a specie feature was also evaluated on both the validation set and the test. While the amount increased prediction performance varied between PTM-types, it improved the prediction performance on both the validation- and test set for 12 out of the 13 PTM-types, with an average increase in AUC ROC on the test set of 0.018. This increase in performance was statistically significant for 11 out of 13 PTM-types on the validation set. The final prediction models showed an AUC ROC of between 0.734 and 0.980 on the test set, with an average AUC ROC of 0.825. Comparisons between our models and the SOTA architecture MusiteDeep shows that while our models show much better results on our test dataset, the reverse is true on the test set of MusiteDeep. Additional tests where performed to determine if this was caused by a difference in specie distribution or sequence filtering, which were both not the case.

Based on the effectiveness on a wide array of PTM-types, the recommendation is made for species to be included as a feature in future PTM prediction models. Creating taxonomic groups of species which share PTM-pattern difference might increase the effectiveness of this even further. Furthermore, the recommendation is made for future work to compare our approach to specie specific models, which could determine if and when which of the two approaches are more effective. While the effectiveness of ProtTrans embeddings was shown for atleast one PTM-type, more research is needed to determine if and how useful they are for PTM prediction. Specifically, the recommendation is made for future research to make use of the biggest model as presented in ProtTrans, optimize an architecture specifically for use with these embeddings, and to finetune the model for PTM prediction. The sequence filtering performed within the training and test set was identified as a source of bias in our dataset, which resulted in biased test results, and degraded performance on an independent test-set. In future works, sequence filtering should be limited between the training and test set, and thus not performed within the sets. Lastly, the discrepancy between test-sets show the importance of evaluating PTM prediction models on multiple independent test sets. In this case, a lack of data quality was identified as a possible explanation for the discrepancy. Future research should make use of PTM data-sources which use data collection methods that guarantee the quality of the collected data.

8 References

1. Ramazi, S., Zahiri, J.: Post-translational modifications in proteins: resources, tools and prediction methods. Database **2021**, baab012 (10 2021). https://doi.org/10.1093/database/baab012, https://doi.org/10.1093/database/baab012
2. De Brevern, A.G., Rebehmed, J.: Current status of PTMs structural databases: applications, limitations and prospects. Amino Acids **1**, 3. https://doi.org/10.1007/s00726-021-03119-z, https://doi.org/10.1007/s00726-021-03119-z
3. Karve, T.M., Cheema, A.K.: Small Changes Huge Impact: The Role of Protein Posttranslational Modifications in Cellular Homeostasis and Disease. Journal of Amino Acids **2011**, 1–13 (7 2011). https://doi.org/10.4061/2011/207691, /pmc/articles/PMC3268018/ /pmc/articles/PMC3268018/?report=abstract https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3268018/
4. He, W., Wei, L., Zou, Q.: Research progress in protein posttranslational modification site prediction. Briefings in Functional Genomics **18**(4), 220–229 (7 2019). https://doi.org/10.1093/BFGP/ELY039, https://academic.oup.com/bfg/article/18/4/220/5253850
5. Hamby, S.E., Hirst, J.D.: Prediction of glycosylation sites using random forests. BMC Bioinformatics **9**(1), 1–13 (11 2008). https://doi.org/10.1186/1471-2105-9-500/FIGURES/5, https://link.springer.com/articles/10.1186/1471-2105-9-500 https://link.springer.com/article/10.1186/1471-2105-9-500
6. Li, S., Li, H., Li, M., Shyr, Y., Xie, L., Li, Y.: Improved Prediction of Lysine Acetylation by Support Vector Machines. Protein & Peptide Letters **16**(8), 977–983 (8 2009). https://doi.org/10.2174/092986609788923338
7. Yang, H., Wang, M., Liu, X., Zhao, X.M., Li, A.: PhosIDN: An integrated deep neural network for improving protein phosphorylation site prediction by combining sequence and protein-protein interaction information. Bioinformatics **37**(24), 4668–4676 (12 2021). https://doi.org/10.1093/BIOINFORMATICS/BTAB551
8. Lv, H., Dao, F.Y., Guan, Z.X., Yang, H., Li, Y.W., Lin, H.: Deep-Kcr: accurate detection of lysine crotonylation sites using deep learning method. Briefings in Bioinformatics **22**(4), 1–10 (7 2021). https://doi.org/10.1093/BIB/BBAA255, https://academic.oup.com/bib/article/22/4/bbaa255/5937175
9. Wang, H., Wang, Z., Li, Z., Lee, T.Y.: Incorporating Deep Learning With Word Embedding to Identify Plant Ubiquitylation Sites. Frontiers in Cell and Developmental Biology **8**, 942 (9 2020). https://doi.org/10.3389/FCELL.2020.572195/BIBTEX
10. Wen, P.P., Shi, S.P., Xu, H.D., Wang, L.N., Qiu, J.D.: Accurate in silico prediction of species-specific methylation sites based on information gain feature optimization. Bioinformatics **32**(20), 3107–3115 (10 2016). https://doi.org/10.1093/BIOINFORMATICS/BTW377, https://academic.oup.com/bioinformatics/article/32/20/3107/2196502
11. Heinzinger, M., Elnaggar, A., Wang, Y., Dallago, C., Nechaev, D., Matthes, F., Rost, B.: Modeling aspects of the language of life through transfer-learning protein sequences. BMC Bioinformatics **20**(1), 1–17 (12 2019). https://doi.org/10.1186/S12859-019-3220-8/FIGURES/5, https://bmcbioinformatics.biomedcentral.com/articles/10.1186/s12859-019-3220-8

12. Asgari, E., Mofrad, M.R.: Continuous Distributed Representation of Biological Sequences for Deep Proteomics and Genomics. PLOS ONE **10**(11), e0141287 (11 2015). https://doi.org/10.1371/JOURNAL.PONE.0141287, https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0141287
13. Thumuluri, V., Jos´, J., Juan, J., Armenteros, A., Johansen, A.R., Nielsen, H., Winther, O.: DeepLoc 2.0: multi-label subcellular localization prediction using protein language models. Nucleic Acids Research **50**(W1), W228–W234 (7 2022). https://doi.org/10.1093/NAR/GKAC278, https://academic.oup.com/nar/article/50/W1/W228/6576357
14. Using machine learning to. Tech. rep., http://www.exocarta.org,
15. Wang, D., Liu, D., Yuchi, J., He, F., Jiang, Y., Cai, S., Li, J., Xu, D.: MusiteDeep: a deep-learning based webserver for protein post-translational modification site prediction and visualization. Nucleic Acids Research **48**(W1), W140–W146 (7 2020). https://doi.org/10.1093/NAR/GKAA275, https://academic.oup.com/nar/article/48/W1/W140/5824154
16. Li, Z., Fang, J., Wang, S., Zhang, L., Chen, Y., Pian, C.: Adapt-Kcr: a novel deep learning framework for accurate prediction of lysine crotonylation sites based on learning embedding features and attention architecture. Briefings in Bioinformatics (2 2022). https://doi.org/10.1093/BIB/BBAC037, https://academic.oup.com/bib/advance-article/doi/10.1093/bib/bbac037/6533505
17. Luo, F., Wang, M., Liu, Y., Zhao, X.M., Li, A.: DeepPhos: prediction of protein phosphorylation sites with deep learning. Bioinformatics **35**(16), 2766–2773 (8 2019). https://doi.org/10.1093/BIOINFORMATICS/BTY1051, https://academic.oup.com/bioinformatics/article/35/16/2766/5270665
18. He, F., Li, J., Wang, R., Zhao, X., Han, Y.: An Ensemble Deep Learning based Predictor for Simultaneously Identifying Protein Ubiquitylation and SUMOylation Sites. BMC Bioinformatics **22**(1), 1–15 (12 2021). https://doi.org/10.1186/S12859-021-04445-5/TABLES/1, https://link.springer.com/articles/10.1186/s12859-021-04445-5 https://link.springer.com/article/10.1186/s12859-021-04445-5
19. Qiao, Y., Zhu, X., Gong, H.: BERT-Kcr: prediction of lysine crotonylation sites by a transfer learning method with pre-trained BERT models. Bioinformatics **38**(3), 648–654 (1 2022). https://doi.org/10.1093/BIOINFORMATICS/BTAB712, https://academic.oup.com/bioinformatics/article/38/3/648/6395351
20. Elnaggar, A., Heinzinger, M., Dallago, C., Rihawi, G., Wang, Y., Jones, L., Gibbs, T., Feher, T., Angerer, C., Steinegger, M., Bhowmik, D., Rost, B.: ProtTrans: Towards Cracking the Language of Life's Code Through Self-Supervised Deep Learning and High Performance Computing. bioRxiv **14**(8) (7 2020), https://arxiv.org/abs/2007.06225v3
21. Dou, L., Yang, F., Xu, L., Zou, Q.: A comprehensive review of the imbalance classification of protein post-translational modifications. Briefings in Bioinformatics **22**(5), 1–18 (9 2021). https://doi.org/10.1093/BIB/BBAB089, https://academic-oup-com.vu-nl.idm.oclc.org/bib/article/22/5/bbab089/6217722
22. Song, L., Xu, Y., Wang, M., Leng, Y.: PreCar_DeepA deep learning framework for prediction of protein carbonylation sites based on Borderline-SMOTE strategy. Chemometrics and Intelligent Laboratory Systems **218**, 104428 (11 2021). https://doi.org/10.1016/J.CHEMOLAB.2021.104428
23. Song, J., Wang, H., Wang, J., Leier, A., Marquez-Lago, T., Yang, B., Zhang, Z., Akutsu, T., Webb, G.I., Daly, R.J.: PhosphoPredict: A bioinformatics tool for prediction of human kinase-specific phosphorylation sub-

strates and sites by integrating heterogeneous feature selection. Scientific Reports 2017 7:1 **7**(1), 1–19 (7 2017). https://doi.org/10.1038/s41598-017-07199-4, https://www.nature.com/articles/s41598-017-07199-4

24. Chen, X., Qiu, J.D., Shi, S.P., Suo, S.B., Huang, S.Y., Liang, R.P.: Incorporating key position and amino acid residue features to identify general and species-specific Ubiquitin conjugation sites. Bioinformatics **29**(13), 1614–1622 (7 2013). https://doi.org/10.1093/BIOINFORMATICS/BTT196, https://academic.oup.com/bioinformatics/article/29/13/1614/185116
25. Beltrao, P., Bork, P., Krogan, N.J., Van Noort, V.: Evolution and functional cross-talk of protein post-translational modifications. Molecular Systems Biology **9**(1), 714 (1 2013). https://doi.org/10.1002/MSB.201304521, https://onlinelibrary.wiley.com/doi/full/10.1002/msb.201304521 https://onlinelibrary.wiley.com/doi/abs/10.1002/msb.201304521 https://www.embopress.org/doi/10.1002/msb.201304521
26. Dehzangi, A., López, Y., Lal, S.P., Taherzadeh, G., Michaelson, J., Sattar, A., Tsunoda, T., Sharma, A.: PSSM-Suc: Accurately predicting succinylation using position specific scoring matrix into bigram for feature extraction. Journal of Theoretical Biology **425**, 97–102 (7 2017). https://doi.org/10.1016/J.JTBI.2017.05.005
27. Dell, A., Galadari, A., Sastre, F., Hitchen, P.: Similarities and Differences in the Glycosylation Mechanisms in Prokaryotes and Eukaryotes. International Journal of Microbiology **2010** (2010). https://doi.org/10.1155/2010/148178, /pmc/articles/PMC3068309/ /pmc/articles/PMC3068309/?report=abstract https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3068309/
28. Li, Y., Wang, M., Wang, H., Tan, H., Zhang, Z., Webb, G.I., Song, J.: Accurate in silico identification of species-specific acetylation sites by integrating protein sequence-derived and functional features. Scientific Reports 2014 4:1 **4**(1), 1–12 (7 2014). https://doi.org/10.1038/srep05765, https://www.nature.com/articles/srep05765
29. Li, Z., Li, S., Luo, M., Jhong, J.H., Li, W., Yao, L., Pang, Y., Wang, Z., Wang, R., Ma, R., Yu, J., Huang, Y., Zhu, X., Cheng, Q., Feng, H., Zhang, J., Wang, C., Hsu, J.B.K., Chang, W.C., Wei, F.X., Huang, H.D., Lee, T.Y.: dbPTM in 2022: an updated database for exploring regulatory networks and functional associations of protein post-translational modifications. Nucleic Acids Research **50**(D1), D471–D479 (1 2022). https://doi.org/10.1093/NAR/GKAB1017, https://academic.oup.com/nar/article/50/D1/D471/6426061
30. BLAST TOPICS, https://blast.ncbi.nlm.nih.gov/Blast.cgi?CMD=WebPAGE$_TYPE = BlastDocsDOC_TYPE = BlastHelp$
31. Meng, L., Chan, W.S., Huang, L., Liu, L., Chen, X., Zhang, W., Wang, F., Cheng, K., Sun, H., Wong, K.C.: Mini-review: Recent advances in post-translational modification site prediction based on deep learning. Computational and Structural Biotechnology Journal **20**, 3522–3532 (1 2022). https://doi.org/10.1016/J.CSBJ.2022.06.045, http://www.ncbi.nlm.nih.gov/pubmed/35860402 http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC928
32. Li, W., Godzik, A.: Cd-hit: a fast program for clustering and comparing large sets of protein or nucleotide sequences. Bioinformatics (Oxford, England) **22**(13), 1658–1659 (7 2006). https://doi.org/10.1093/BIOINFORMATICS/BTL158, https://pubmed.ncbi.nlm.nih.gov/16731699/
33. Embedding — PyTorch 1.12 documentation, https://pytorch.org/docs/stable/generated/torch.nn.Embedding.html
34. Bahdanau, D., Cho, K., Bengio, Y.: NEURAL MACHINE TRANSLATION BY JOINTLY LEARNING TO ALIGN AND TRANSLATE

35. Srivastava, N., Hinton, G., Krizhevsky, A., Salakhutdinov, R.: Dropout: A Simple Way to Prevent Neural Networks from Overfitting. Journal of Machine Learning Research **15**, 1929–1958 (2014)
36. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. NAACL HLT 2019 - 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference **1**, 4171–4186 (10 2018). https://doi.org/10.48550/arxiv.1810.04805, https://arxiv.org/abs/1810.04805v2
37. Steinegger, M., Söding, J.: Clustering huge protein sequence sets in linear time. Nature Communications 2018 9:1 **9**(1), 1–8 (6 2018). https://doi.org/10.1038/s41467-018-04964-5, https://www.nature.com/articles/s41467-018-04964-5
38. Heinzinger, M., Littmann, M., Sillitoe, I., Bordin, N., Orengo, C., Rost, .B.: Contrastive learning on protein embeddings enlightens midnight zone at lightning speed. bioRxiv p. 2021.11.14.468528 (11 2021). https://doi.org/10.1101/2021.11.14.468528, https://www.biorxiv.org/content/10.1101/2021.11.14.468528v1 https://www.biorxiv.org/content/10.1101/2021.11.14.468528v1.abstract
39. Loshchilov, I., Hutter, F.: Decoupled Weight Decay Regularization. 7th International Conference on Learning Representations, ICLR 2019 (11 2017). https://doi.org/10.48550/arxiv.1711.05101, https://arxiv.org/abs/1711.05101v3
40. Akiba, T., Sano, S., Yanase, T., Ohta, T., Koyama, M.: Optuna: A Next-generation Hyperparameter Optimization Framework. Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining pp. 2623–2631 (7 2019). https://doi.org/10.48550/arxiv.1907.10902, https://arxiv.org/abs/1907.10902v1
41. Hossin, M., Sulaiman: A REVIEW ON EVALUATION METRICS FOR DATA CLASSIFICATION EVALUATIONS. IJDKP) International Journal of Data Mining & Knowledge Management Process (IJDKP) **5**(2) (2020). https://doi.org/10.5121/ijdkp.2015.5201
42. Fawcett, T.: An introduction to ROC analysis. Pattern Recognition Letters **27**(8), 861–874 (6 2006). https://doi.org/10.1016/J.PATREC.2005.10.010
43. Halimu, C., Kasem, A., Newaz, S.H.: Empirical comparison of area under ROC curve (AUC) and Mathew correlation coefficient (MCC) for evaluating machine learning algorithms on imbalanced datasets for binary classification. ACM International Conference Proceeding Series pp. 1–6 (1 2019). https://doi.org/10.1145/3310986.3311023, https://doi.org/10.1145/3310986.3311023
44. Nadeau, C., Bengio, Y.: Inference for the Generalization Error. Machine Learning 2003 52:3 **52**(3), 239–281 (9 2003). https://doi.org/10.1023/A:1024068626366, https://link.springer.com/article/10.1023/A:1024068626366
45. Bouckaert, R.R., Frank, E.: Evaluating the replicability of significance tests for comparing learning algorithms. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) **3056**, 3–12 (2004). https://doi.org/10.1007/978-3-540-24775-3_3/COVER, https://link.springer.com/chapter/10.1007/978-3-540-24775-3$_3$
46. Crooks, G.E., Hon, G., Chandonia, J.M., Brenner, S.E.: WebLogo: a sequence logo generator. Genome research **14**(6), 1188–1190 (6 2004). https://doi.org/10.1101/GR.849004, https://pubmed.ncbi.nlm.nih.gov/15173120/
47. Tareen, A., Kinney, J.B.: Logomaker: Beautiful sequence logos in Python (4 2020). https://doi.org/10.1093/BIOINFORMATICS/BTZ921
48. Lowenthal, M.S., Davis, K.S., Formolo, T., Kilpatrick, L.E., Phinney, K.W.: Identification of novel N-glycosylation sites at non-canonical protein consensus motifs. Journal of proteome research **15**(7),

2087 (7 2016). https://doi.org/10.1021/ACS.JPROTEOME.5B00733, /pmc/articles/PMC5100817/ /pmc/articles/PMC5100817/?report=abstract https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5100817/

49. Van Der Maaten, L., Hinton, G.: Visualizing Data using t-SNE. Journal of Machine Learning Research **9**, 2579–2605 (2008)
50. sklearn.manifold.TSNE — scikit-learn 1.1.2 documentation, https://scikit-learn.org/stable/modules/generated/sklearn.manifold.TSNE.html
51. Breuza, L., Poux, S., Estreicher, A., Famiglietti, M.L., Magrane, M., Tognolli, M., Bridge, A., Baratin, D., Redaschi, N.: The UniProtKB guide to the human proteome. Database: The Journal of Biological Databases and Curation **2016** (2016). https://doi.org/10.1093/DATABASE/BAV120, /pmc/articles/PMC4761109/ /pmc/articles/PMC4761109/?report=abstract https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4761109/
52. Salem, M., Arshadi, A.K., Yuan, J.S.: AMPDeep: Hemolytic Activity Prediction of Antimicrobial Peptides using Transfer Learning (2022). https://doi.org/10.21203/rs.3.rs-1615895/v1, https://doi.org/10.21203/rs.3.rs-1615895/v1
53. Farriol-Mathis, N., Garavelli, J.S., Boeckmann, B., Duvaud, S., Gasteiger, E., Gateau, A., Veuthey, A.L., Bairoch, A.: Annotation of post-translational modifications in the Swiss-Prot knowledge base. PROTEOMICS **4**(6), 1537–1550 (6 2004). https://doi.org/10.1002/PMIC.200300764, https://onlinelibrary-wiley-com.vu-nl.idm.oclc.org/doi/full/10.1002/pmic.200300764 https://onlinelibrary-wiley-com.vu-nl.idm.oclc.org/doi/abs/10.1002/pmic.200300764 https://analyticalsciencejournals-onlinelibrary-wiley-com.vu-nl.idm.oclc.org/doi/10.1002/pmic.200300764

9 Appendix

9.1 ProtTrans: t-SNE embeddings

The t-SNE visualizations of the ProtTrans embeddings for all PTM-types, generated as described in Section 4.2, can be found in Figure 18.

9.2 Sequence Logos

The generated sequence logos for all PTM-types, generated as described in Section 4.1, can be found in Figure 19, 20 and 21.

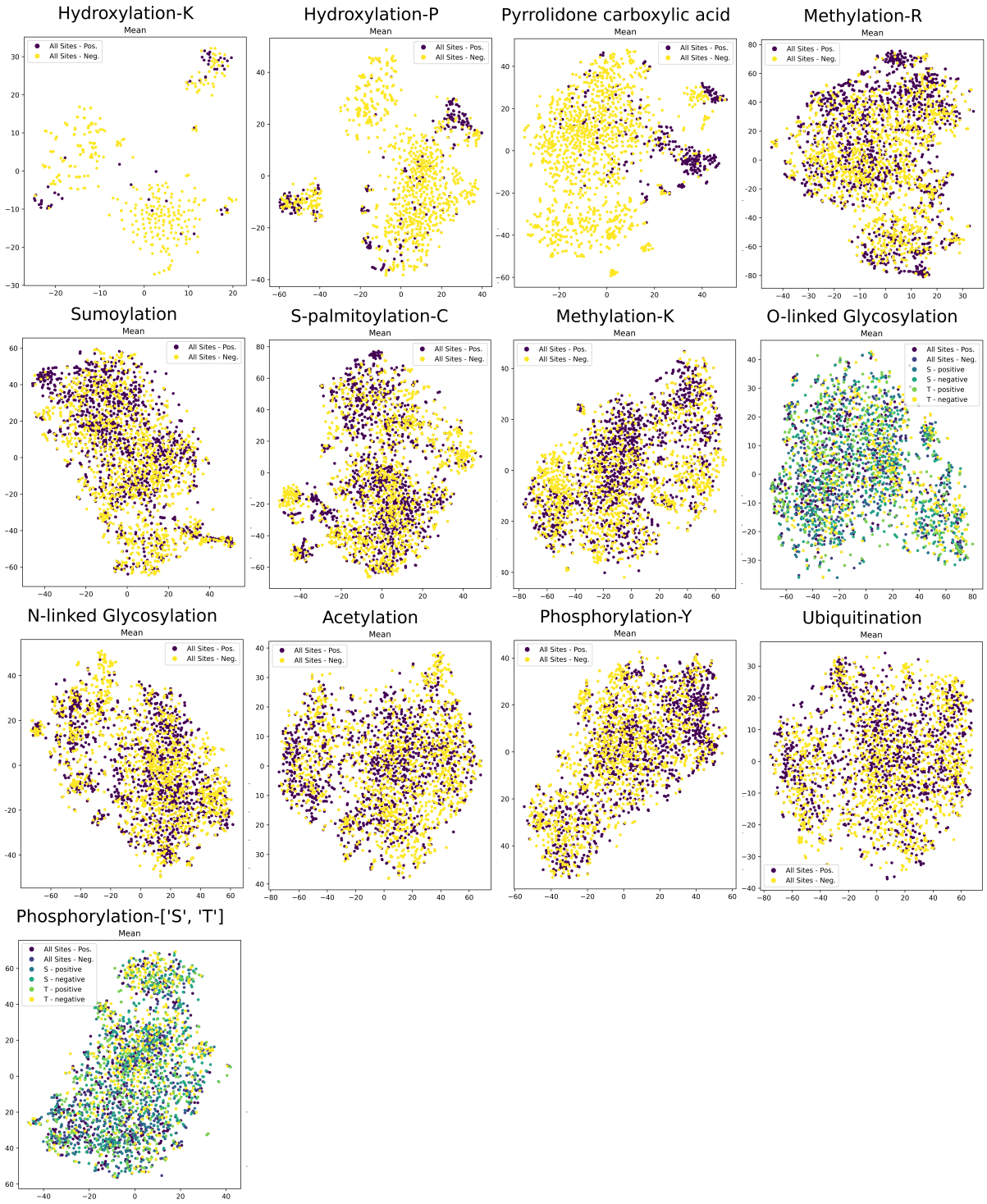

Fig. 18. This figure visually presents the processing steps used to acquire our benchmark training- and test dataset, starting from the samples acquired from dbPTM 2021.

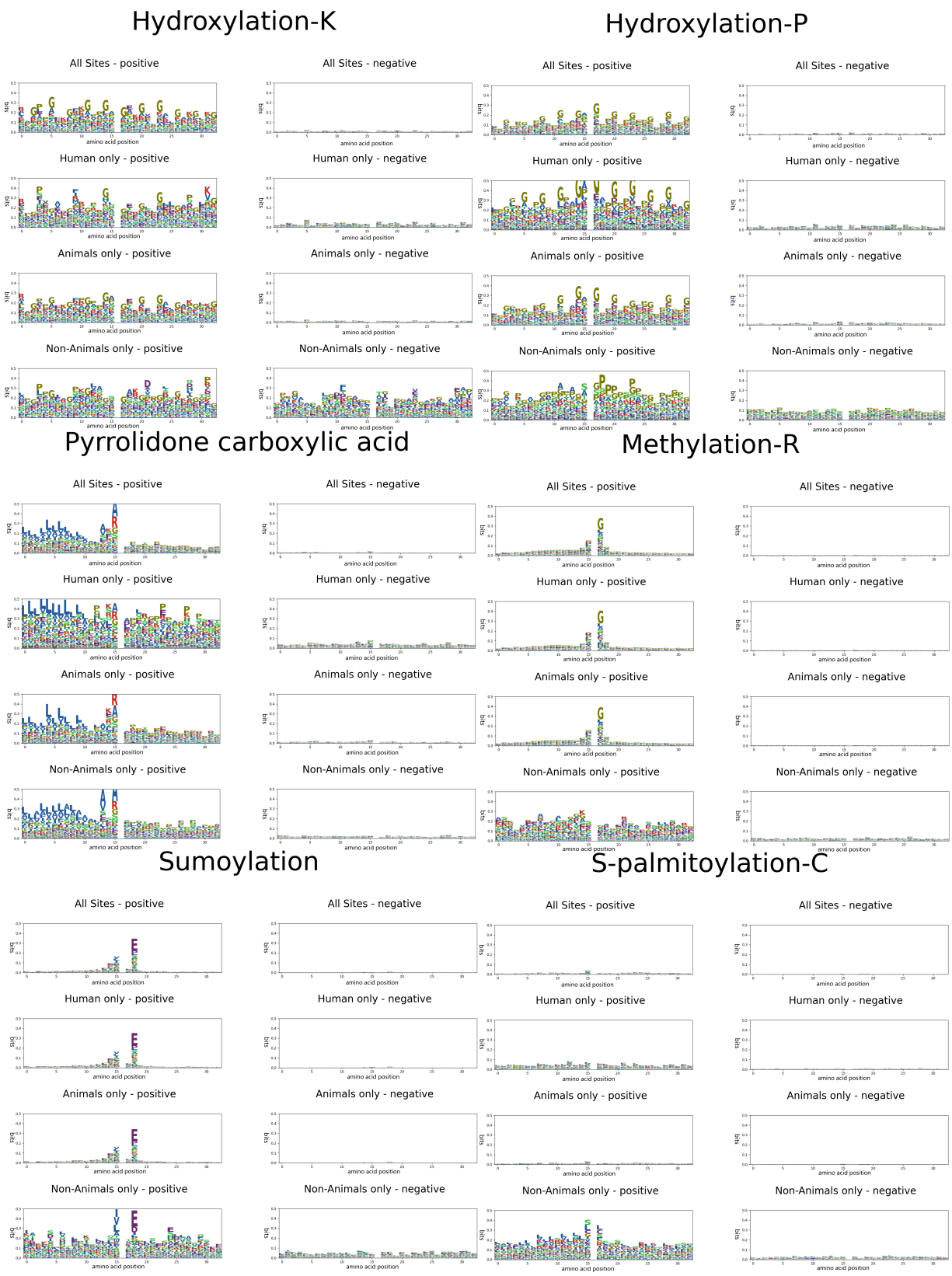

Fig. 19. This figure visually presents the processing steps used to acquire our benchmark training- and test dataset, starting from the samples acquired from dbPTM 2021.

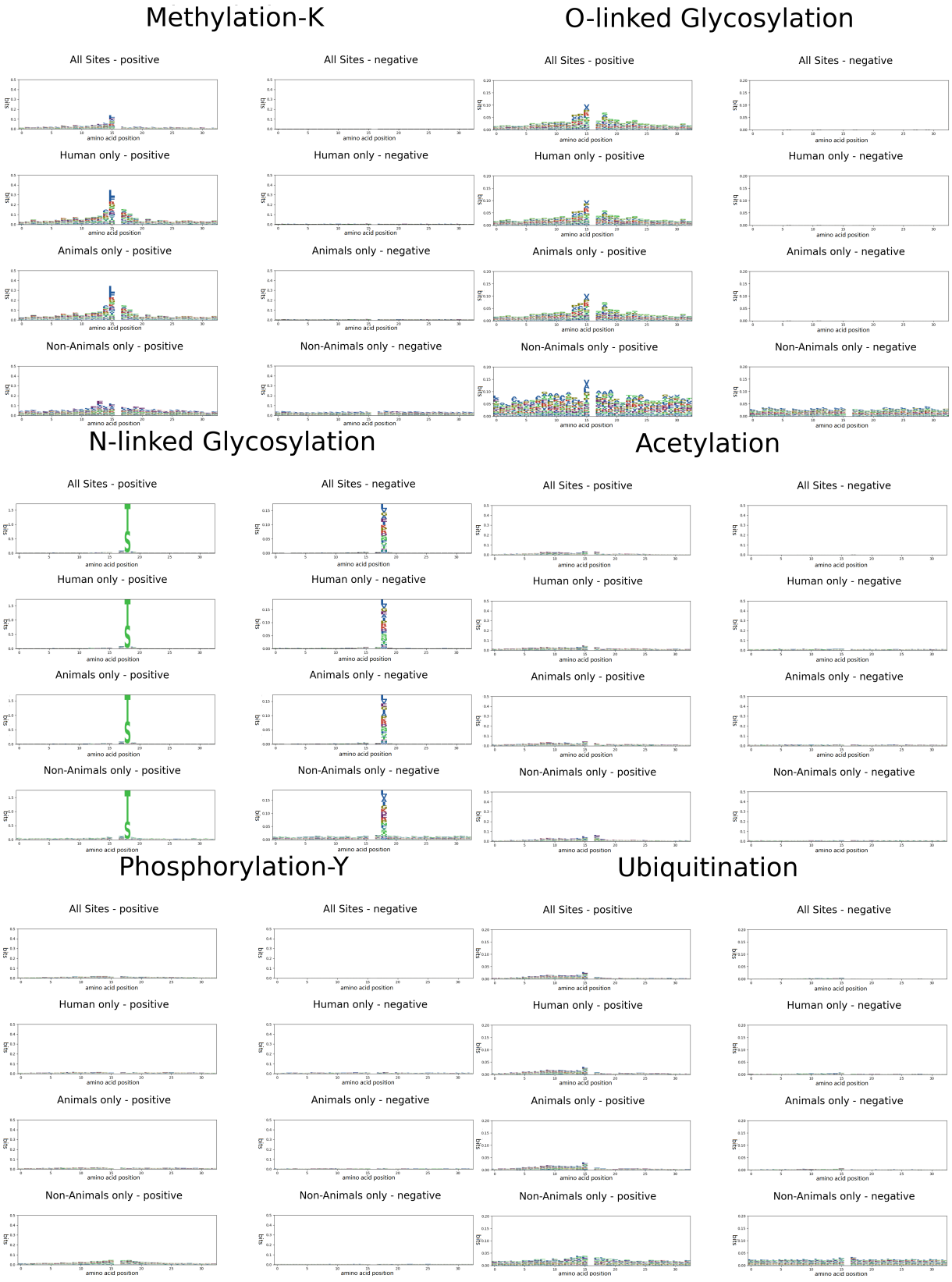

Fig. 20. This figure visually presents the processing steps used to acquire our benchmark training- and test dataset, starting from the samples acquired from dbPTM 2021.

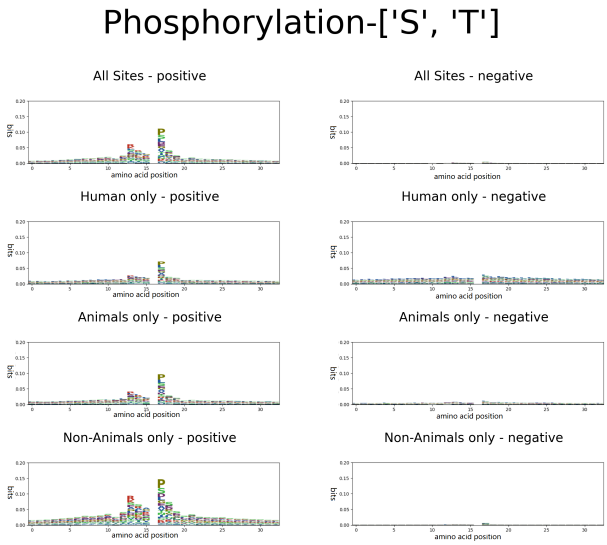

Fig. 21. This figure visually presents the processing steps used to acquire our benchmark training- and test dataset, starting from the samples acquired from dbPTM 2021.

9.3 Evaluation metrics: test sets

The compared evaluation metrics of both our model and MusiteDeep on our test set can be found in Table 12. The compared evaluation metrics of both our model and MusiteDeep on the test set of MusiteDeep can be found in Table 13.

Table 12. Generated by Spread-LaTeX

PTM	Model	AUC ROC	AUC PR	Sensitivity	Specificity	Precision
Hydroxylation-K	Our Model	**0,772**	0,346	**0.480**	0.832	0.175
	MusiteDeep	0,742	**0,393**	0,333	**0,978**	**0,500**
Hydroxylation-P	Our Model	0,865	0,593	**0.730**	0.795	0.53
	MusiteDeep	**0,933**	**0,851**	0,675	**0,966**	**0,839**
pyr eegg acid	Our Model	**0,952**	0,805	**0.837**	0.952	**0.706**
	MusiteDeep	0,923	**0,871**	0,833	**0,973**	0,800
Methylation-R	Our Model	**0,836**	0,324	**0.701**	0.815	0.165
	MusiteDeep	0,773	**0,416**	0,435	**0,966**	**0,390**
Sumoylation	Our Model	**0,847**	**0,507**	**0.75**	0.767	0.255
	MusiteDeep	0,687	0,313	0,327	**0,961**	**0,465**
S-palmitoylation	Our Model	**0,743**	**0,327**	**0.703**	0.642	0.257
	MusiteDeep	0,597	0,246	0,112	**0,961**	**0,333**
Methylation-K	Our Model	**0,713**	**0,266**	**0.697**	0.619	**0.227**
	MusiteDeep	0,539	0,160	0,021	**0,985**	0,186
O-linked Glycosylation	Our Model	**0,740**	**0,148**	**0.622**	0.726	0.097
	MusiteDeep	0,664	0,128	0,223	**0,932**	**0,131**
N-linked Glycosylation	Our Model	**0,974**	**0,872**	**0.993**	0.916	0.760
	MusiteDeep	0,968	0,835	0,989	**0,924**	**0,777**
Acetylation	Our Model	**0,840**	**0,410**	**0.876**	0.619	0.236
	MusiteDeep	0,746	0,298	0,246	**0,937**	**0,342**
Phosphorylation-Y	Our Model	**0,782**	**0,412**	**0.771**	0.647	0,290
	MusiteDeep	0,650	0,272	0,224	**0,908**	**0,314**
Ubiquitination	Our Model	**0,796**	**0,533**	**0.831**	0.605	**0.391**
	MusiteDeep	0,544	0,262	0,149	**0,864**	0,251
Phosphorylation-ST	Our Model	**0,864**	**0,521**	**0.752**	**0.814**	**0.352**
	MusiteDeep	0,746	0,345	0,618	0,760	0,257

Table 13. Generated by Spread-LaTeX

PTM	model	AUC ROC	AUC PR	Sensitivity	Specificity	Precision
Phosphorylation-Y	Our model	0.577	0.109	0.517	0.575	0.099
	MusiteDeep	**0.955**	**0.820**	**0.880**	**0.920**	**0.499**
Phosphorylation-ST	Our model	0.717	0.125	0.714	0.576	0.060
	MusiteDeep	**0.889**	**0.324**	**0.845**	**0.793**	**0.135**

9.4 Test sets: specie distribution

The relative frequency of species represented for phosphorylation-ST for our test can be found in Figure 22, and can be found for the test set of MusiteDeep in Figure 23. The relative frequency of species represented for phosphorylation-Y for our test can be found in Figure 24, and can be found for the test set of MusiteDeep in Figure 25.

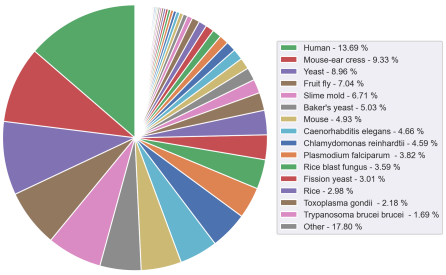

Fig. 22. The distribution of species from which the PTM-samples were sampled in our test set for phosphorylation-ST

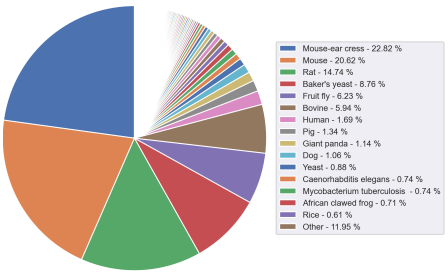

Fig. 23. The distribution of species from which the PTM-samples were sampled in the test set of MusiteDeep for phosphorylation-ST

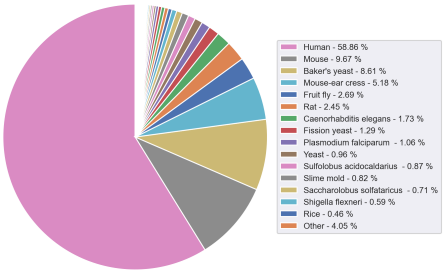

Fig. 24. The distribution of species from which the PTM-samples were sampled in our test set for phosphorylation-ST

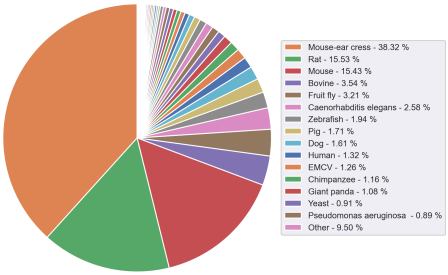

Fig. 25. The distribution of species from which the PTM-samples were sampled in the test set of MusiteDeep for phosphorylation-ST