DETECTION OF UNEXPECTED
POST-TRANSLATIONAL MODIFICATIONS IN
VERY LARGE VOLUMES OF PUBLIC
PROTEOMICS DATA

Ralf Gabriels

Supervisors: Prof. Dr. Lennart Martens, Dr. Andrea Argentini
Faculty of Medicine and Health Sciences, Department of Biochemistry

A dissertation submitted to Ghent University in partial fulfilment of the requirements for the degree of
Master of Science in Biomedical Sciences.

Academic year: 2016 - 2017
ACKNOWLEDGEMENTS

For this thesis project, I would like to thank every team member of the CompOmics group: Prof. Dr. Lennart Martens, Dr. Andrea Argentini, Dr. Elien Vandermarliere, Dr. Sami Naji, Dr. Sven Degroeve, Dr. Paola Masuzzo, Dr. Pieter-Jan Volders, Dr. Şule Yılmaz, Adriaan Sticker, Ana Silvia C. Silva, Davy Maddelein, Demet Turan, Elvis Ndah, Genet Abay Shiferaw, Gwendolien Sergeant, Kenneth Verheggen, Niels Hulstaert, Pathmanaban Ramasamy, Robbin Bouwmeester, Surya Gupta, Tessa Vermeire and Tim Van Den Bossche. Throughout the year, their help, support and feedback has been indispensable!

Special gratitude goes out to Adriaan Sticker and Robin Bouwmeester for answering all of my most-randomly asked questions. Many thanks to Davy Maddelein and Niels Hulstaert, for fixing the numerous computer problems I encountered. I would also like to thank Dr. Şule Yılmaz for providing an adapted version of the Spectrum Similarity tool.

I am grateful to Dr. Sven Degroeve, for guiding me through the world of machine learning and to Ana Silvia C. Silva for giving me close-up guidance with MS²PIP and, by extension, with everything else. I would especially like to thank Dr. Andrea Argentini for assisting me during both my research internship and this thesis project.

Of course, none of this would have been possible without Prof. Dr. Lennart Martens, who gave me the incredible opportunity to work in his research group and who provided me with the most interesting thesis subject.

I would also like to mention Jeroen Piens and Jonas Van Cleemput, for making me not the only student Biomedical Sciences who was seemingly crazy enough to take on a thesis project in bioinformatics.

Last but not least, I am extremely grateful to my family and friends; and most importantly to Evi Duthoo, who figuratively, and almost literally, always stands by my side.
# TABLE OF CONTENTS

1 Introduction ........................................................................................................................................ 3

1.1 Central Dogma of molecular biology ............................................................................................ 3

1.2 Mass spectrometry-based proteomics ............................................................................................. 3

1.2.1 Different methods in mass spectrometry-based proteomics ......................................................... 3

1.2.2 Tandem mass spectrometry ........................................................................................................ 5

1.2.3 Fragmentation spectra ................................................................................................................. 5

1.2.4 Inferring the peptide sequence from a fragmentation spectrum .................................................. 7

1.2.4.1 Sequence search engines ........................................................................................................ 7

1.2.4.2 Spectral search engines ............................................................................................................ 7

1.3 Problem description ........................................................................................................................ 9

1.3.1 Online databases are filled with unidentified spectra ................................................................. 9

1.3.2 Amino acid variations and PTMs are among the culprits .......................................................... 9

1.3.3 Importance of PTM identification ............................................................................................... 11

1.3.4 Existing methods to identify modified peptides .......................................................................... 11

1.3.5 MS²PIP for phosphorylated peptides ......................................................................................... 12

2 Materials and methods ..................................................................................................................... 14

2.1 Machine learning .......................................................................................................................... 14

2.1.1 Introduction .................................................................................................................................. 14

2.1.2 Decision tree models .................................................................................................................. 15

2.1.3 Random Forest ............................................................................................................................ 16

2.1.4 Extreme Gradient Boosting (XGBoost) .................................................................................... 16

2.2 MS²PIP ........................................................................................................................................... 16

2.2.1 Goal of MS²PIP .......................................................................................................................... 16

2.2.2 MS²PIP Server and MS²PIPc ....................................................................................................... 17

2.2.3 MS²PIPc feature selection .......................................................................................................... 17

2.2.4 MS²PIPc target selection ............................................................................................................ 17

2.2.5 Performance evaluation .............................................................................................................. 18

2.3 Other tools ...................................................................................................................................... 18
2.3.1 Spectral library manipulation with SpectraST ..............................................18
2.3.2 Spectrum Similarity .......................................................................................18
2.4 Datasets ..............................................................................................................19
  2.4.1 PRIDE Cluster dataset ..................................................................................19
  2.4.2 Synthetic peptide library .............................................................................19
  2.4.3 NIST human spectral library .......................................................................19
  2.4.4 Spectral library for external validation .........................................................20
2.5 Workflow ............................................................................................................20
  2.5.1 Evaluation on MS²PIP Server .........................................................................20
    2.5.1.1 Parsing PRIDE Cluster spectral library ..................................................20
    2.5.1.2 Getting predictions from the MS²PIP Server .........................................21
    2.5.1.3 Calculating spectrum correlations with Spectrum Similarity ...............21
  2.5.2 MS²PIPc .........................................................................................................21
    2.5.2.1 Re-evaluation on MS²PIPc .................................................................21
    2.5.2.2 Search for another spectral library .......................................................21
    2.5.2.3 Feature addition .....................................................................................22
    2.5.2.4 Necessary training dataset size .............................................................23
    2.5.2.5 Training, testing and evaluating in one go ..........................................23
    2.5.2.6 Parameter tuning ....................................................................................24
    2.5.2.7 Final model evaluation ..........................................................................25
3 Results ....................................................................................................................26
  3.1 Spectral library properties ..............................................................................26
  3.2 Performance of MS²PIP Server on phosphorylated peptides .......................29
  3.3 Performance of MS²PIPc on phosphorylated peptides ....................................31
  3.4 Necessary training dataset size for MS²PIPc ....................................................34
  3.5 Parameter tuning .............................................................................................35
  3.6 Feature addition ..............................................................................................37
  3.7 Evaluation of the final models .........................................................................40
4 Discussion ............................................................................................................42
5 Conclusion .......................................................................................................................... 43
6 References .......................................................................................................................... 44
ABSTRACT

**Background:** As complete proteomes can be analysed in one shotgun proteomics experiment, public online proteomics databases are growing by the month. Unfortunately, more than two thirds of the submitted spectra are never identified, with post-translational modifications (PTMs) as one of the reasons. As most commonly used search engines do not take the mass-to-charge ratio \((m/z)\) shifts and altered fragmentation patterns caused by PTMs into account, these spectra are easily missed. To improve the identification rate of modified peptides, we built a predictor for modified peptide spectra. Such a tool already exists in the form of MS²PIP, but its capabilities on modified peptides had not been explored yet. As phosphorylation is one of the most abundant and most studied PTM, it was used as a case study.

**Methods:** Machine learning models were trained and evaluated on phosphopeptide data. The minimum training dataset size was determined and parameter optimisation was carried out. Besides retraining the models on phosphopeptide data, phosphopeptide-specific features were added.

**Results:** MS2PIP showed more difficulties in predicting phosphopeptide spectra than unmodified peptide spectra. The phosphorylation models outperformed the default models, but feature addition only slightly influenced the results. Serine and threonine phosphorylations seem to have a larger effect on the fragmentation patterns than tyrosine phosphorylations.

**Conclusions:** While we successfully trained models for phosphopeptides, this was only the start of the project. Other modifications need to be tested and the models need to be implemented into a search space-efficient search engine, before we can tackle the issue of unidentified spectra.
SAMENVATTING


Om het aantal identificaties te vergroten, hebben we een tool gebouwd die gemondepte peptidespectra kan voorspellen. Zo’n tool bestond reeds, in de vorm van MS²PIP, maar de capaciteiten met betrekking tot PTM’s waren nog niet onderzocht. Aangezien fosforlyatie één van de meest onderzochte en meest voorkomende modificaties is, gebruiken we deze PTM als case study.

Methoden: Machine leren modellen werden getraind en geëvalueerd met fosfopeptide data.

De minimum datasetgrootte om performante modellen trainen werd bepaald en hyperparameter-optimalisatie werd uitgevoerd. Naast het hertrainen van de modellen op fosfopeptide data, werden fosfopeptide-specifieke features toegevoegd.

Resultaten: MS²PIP had meer moeite bij het voorspellen van fosfopeptidespectra dan bij ongemodificeerde peptidespectra. De forfipeptide modellen toonden betere resultaten dan de standaard MS²PIP modellen. Het toevoegen van features had slechts een kleine invloed op de modellen. Serine en threonine phosphorylaties bleken de fragmentatie patronen meer te beïnvloeden dan tyrosine phosphorylaties.

Besluiten: Desondanks we succesvol modellen hebben kunnen trainen voor fosfopeptidespectra, is dit slechts een klein deel van het project. Andere modificaties moeten getest worden en de modellen moeten geïmplementeerd worden in een search space-efficiente zoekmachine alvorens we het probleem van ongeïdentificeerde spectra kunnen aanpakken.
1 INTRODUCTION

1.1 CENTRAL DOGMA OF MOLECULAR BIOLOGY

When studying life at its molecular core, three levels of sequence information can be described: DNA, RNA and protein. The informational flow between these levels was described as the Central Dogma of Molecular Biology by Francis Crick in 1958\cite{1}. The Central Dogma defines three general transfers of information: DNA to DNA (replication), DNA to RNA (transcription) and RNA to protein (translation). Between each transfer, sequence information gets more complex in several ways. One example is alternative splicing, where one DNA sequence results in different RNA transcripts. Another example is the occurrence of post-translational modifications (PTMs): small (or in some cases larger) molecules that are added to the protein after its transcription. Moreover, DNA and RNA comprise out of only four different molecular building blocks, whereas proteins are built out of twenty different amino acids. With all these factors in mind, it is easily understood that proteins are more difficult to analyse than DNA or even RNA.

To further complicate matters, protein concentrations have an extremely high dynamic range. In human blood serum, albumin has a concentration that’s 10^{10} times higher than the concentration of interleukin 6.\cite{2} As typical proteomics instruments have a dynamic range of 10^2-10^4, low abundance proteins are only sporadically found in a routine experiment.

Whereas DNA and RNA can be sequenced in a relatively unambiguous way, unravelling a protein sequence brings in a lot more challenges. Over the years, mass spectrometry has proven to be the method of choice in the field of proteomics. In these proteomics experiments, researchers try not to identify a single protein, but they try to identify all proteins present in a sample.

1.2 MASS SPECTROMETRY-BASED PROTEOMICS

1.2.1 Different methods in mass spectrometry-based proteomics

In mass spectrometry-based proteomics, complex samples of proteins are identified and quantified by first cutting the proteins into shorter peptides, and then analysing these peptides by tandem mass spectrometry (MS/MS or MS\(^2\)). The peptides are ionised and fragmented upon which their mass-to-charge ratio (\(m/z\)) is measured. The way by which certain peptides are selected for fragmentation depends on the method in use.

In data-dependent acquisition (DDA) there are two forms of peptide selection. In one method, named shotgun proteomics, as many peptides as possible are selected one-by-one for
fragmentation and analysis. The number of peptides analysed in this method is limited by the speed by which the peptides are fed to the mass spectrometer and the pace by which the mass spectrometer can analyse them. As most mass spectrometers select the peptides with the highest intensity, easily ionised peptides and peptides with higher concentrations are much more likely to be picked out for fragmentation and analysis. Lower stoichiometric peptides are therefore easily missed in shotgun proteomics.

In the second DDA method, targeted proteomics, the mass spectrometer is set in selected reaction monitoring (SRM) mode. An SRM-set mass spectrometer only fragments and analyses certain predefined peptides. Both DDA methods are complementary in advantages. Shotgun proteomics is ideal for identifying as much peptides as possible, while targeted proteomics is mainly used to quantify certain peptides in a reproducible way.[3]

Data-independent acquisition (DIA) methods aim to combine the advantages of both DDA methods. In DIA experiments, all peptides within a predetermined window of mass-to-charge ratios are analysed. This results in not one fragmentation spectrum per peptide, but in a mixture of all fragmentation spectra per m/z window. As such, the output data of DIA experiments is more complex to analyse.[4] Figure 1 compares DDA and DIA methods.

![Figure 1: Schematic overview of data-dependent acquisition (DDA, top) and data-independent acquisition (DIA, bottom). In DDA peptides are fragmented and analysed one-by-one, while in DIA multiple peptides are fragmented and analysed in one run.](image)

Of the methods described above, shotgun proteomics is the most mature. Therefore, most public proteomics datasets have originated from this technique. As of July 2017, 2309 shotgun proteomics datasets were present in the PRIDE Archive, making up for 59% of the database.[5] Because of the abundance of publicly available shotgun proteomics datasets and the method’s maturity, this thesis project will further only handle shotgun proteomics and its resulting data.
1.2.2 Tandem mass spectrometry
As mentioned above, proteins are cut into short peptides by digesting them with a proteolytic enzyme, such as trypsin. These peptides are then separated by liquid chromatography (LC). To analyse peptides in a mass spectrometer, they must first be ionized. This can be done with a laser (matrix-assisted laser desorption/ionisation, MALDI) or by using a highly electrically charged spray (electrospray ionisation, ESI). The latter uses a liquid sample input, which means the LC column can be coupled directly to the mass spectrometer. For this reason, ESI is the method of choice for most proteomics experiments.

In the mass spectrometer, every peptide’s mass-over-charge ratio (m/z) is measured. Using a form of collision induced dissociation (CID), electron capture dissociation (ECD), or electron transfer dissociation (ETD), peptides of interest are cleaved into fragment ions. Then, the m/z of these fragment ions is also measured. Because there are two consecutive measurements of the m/z (first the m/z of the precursor ions and then the m/z of the fragment ions), the method is called tandem mass spectrometry, MS/MS or MS².

1.2.3 Fragmentation spectra
To identify proteins, we need to interpret the output from the mass spectrometer. For every peptide that was fragmented, there is a resulting fragmentation spectrum. In this spectrum, every peak represents a fragment ion. The height of the peak indicates the ion’s intensity and the position of the peak on the x-axis represents the ion’s m/z.

![Figure 2: Different fragmentation leads to different fragment ions.]

Fortunately, peptide fragmentation is not a random process. Depending on the fragmentation technique used, peptides are cleaved on one of three distinct positions. Every cleavage position leads to a different fragment ion. A peptide can for instance be cleaved between the central carbon and the carboxyl group, which lead to an a-ion on the N-terminus and an x-ion on the C-terminal end. Similarly, a cleavage between the carboxyl group and the nitrogen atom leads to a b-ion and a y-ion; and a cleavage between the nitrogen atom and the central carbon leads to a c-ion and a z-ion (Figure 2). To summarise, ions on the left of the cleavage position
are called the a-, b- and c-ions, while the ones on the right are called the x-, y- and z-ions. CID and HCD (Higher-energy collisional dissociation) lead mostly to b- and y-ions. [6]

Every fragment ion is numbered by the location of the cleavage in the amino acid sequence. A-, b- and c-ions are numbered starting from the N-terminus and x-, y- and z-ions are numbered starting from the C-terminus. For instance, if an eight-amino acid long peptide is cleaved by CID between the first and second amino acid, the resulting fragment ions are referred to as b₁ and y₇.

As every one of the twenty amino acids has a different side chain, every one of them has a distinct molecular mass. Using this information, we can predict for a given peptide sequence the theoretical MS/MS fragmentation spectrum. The distance between each consecutive ion peak equals the weight of the added amino acid, divided by the ion’s charge. In the spectrum shown in Figure 3, the most left peak (which is almost not visible in the empirical spectrum) represents the y₁-ion, and consists of a single serine residue. It has a molecular weight of 105 Da and a net charge of +1. The distance to the third peak, y₂, is exactly 131 m/z and represents the added isoleucine. In this fashion, we can continue to deduce the peptide sequence by moving up the “fragmentation ladder”.

![Figure 3: Theoretical and empirical fragmentation spectrum from the peptide sequence “THESIS”](image)

Unfortunately, as both b- and y-ions are present in the spectrum, inferring the exact peptide sequence becomes rather complex. Moreover, in this example we mentioned the added isoleucine. But this could also have been a leucine, as both amino acids have exactly the same molecular weight. Further complicating matters are noise peaks, multiply charged fragment ions, double fragmentation (internal fragment ions), and so on.[6] Also, as seen in the first peak in Figure 3, some fragment ion peaks are (almost or completely) not present in the spectrum.
1.2.4 Inferring the peptide sequence from a fragmentation spectrum

Because inferring a peptide sequence is so complex, it is easily understood that this is done using computational tools. The peptide sequence can be inferred de novo or by using sequential or spectral databases.\footnote{7}

1.2.4.1 Sequence search engines

In most routine proteomics experiments, sequence database search engines are used. Some examples of these tools are SEQUEST\footnote{8}, Mascot\footnote{9}, X! Tandem\footnote{10} and Andromeda\footnote{11}. As the name implies, sequence database search engines use sequence data to identify fragmentation spectra. Protein coding sequences are digested in silico to obtain all peptides that could be expected in the sample. For instance, if trypsin is used, proteins are cut into peptides after every lysine and arginine. By calculating the masses of all possible fragment ions, theoretical fragmentation spectra can be obtained for every peptide. These theoretical spectra are simplified versions of the actual spectrum, as all peaks have an equal intensity (for instance the bottom spectrum in Figure 3). Experimental fragmentation spectra are matched against the theoretical spectra and are given a score. Such a match is called a peptide-spectrum match (PSM).

To discriminate correct matches from false positives, the target-decoy approach is used.\footnote{12} For this method, the theoretical spectra are also matched to decoy spectra. These decoy spectra are theoretical spectra, but for peptides from which we are sure they are not in the sample. They can be generated by shuffling or reversing the amino acid order of the actual sequences, or by generating them at random. The target-decoy approach assumes that the matching scores from false positive PSMs follow the same distribution as the decoy PSMs. In this way, we can set for a given assumed false discovery rate (FDR) a score cut-off. PSMs with a score above the cut-off are assumed to be correct, PSMs with a score below the cut-off are assumed to be incorrect.

1.2.4.2 Spectral search engines

As more and more MS\(^2\) spectra become available, it makes more sense to use this data for peptide identifications. With this in mind, spectral search engines such as SpectraST\footnote{13}, X!Hunter\footnote{14} and BiblioSpec\footnote{15} were created. These search engines try to find matches for the experimental spectra in an annotated library of empirical fragmentation spectra.

Some peptides are always found in shotgun experiments, while others are never found. Nevertheless, sequence search engines do look for every theoretically possible peptide. By doing so, a lot of computing time is wasted. Spectral search engines, on the other hand, only look for peptides that are present in the spectral library. Therefore, all peptides that are never
found in shotgun experiments are excluded from the search space. As spectral search engines have a much smaller search space, the necessary computing time is reduced drastically.[13] Furthermore, SpectraST reports to have a simpler matching algorithm, which lead to faster search times per spectrum.[13]

Another big advantage of spectral search engines is the fact that they do take the peak intensity into account. As mentioned before, theoretical spectra all have the same peak intensity. Therefore, sequence search engines do not discriminate peaks that are very prominent or peaks that almost not visible between noise peaks. However, it is known that peak intensities are strongly dependent on the peptide sequence and its charge.[16] By using this information, spectral search engines can discriminate good and bad hits with a much higher probability. As a result, more spectra can be identified at the same FDR by spectral search engines than by sequence search engines.[13]

Relevant to this thesis project is the fact that spectral search engines can search for modified peptides in the same way as for normal peptides. The only requirement is that the modified peptide is present in the library, which at this moment poses a problem. NIST (National Institute of Standards and Technology) only provides a specialised PTM spectral library for iTRAQ-4 phosphorylations.[17] The normal NIST spectral libraries do contain some modified peptides, such as oxidation, acetylation and carbamidomethylation, but these modifications are only a tip of the iceberg. A comprehensive spectral library for, for instance, phosphorylated peptides, could make their identification a lot easier.
1.3 PROBLEM DESCRIPTION

1.3.1 Online databases are filled with unidentified spectra

Since every shotgun proteomics experiment can analyse a complete proteome in one run, online proteomics databases are growing exponentially. During the period of mid-2015, an average of 150 datasets were submitted per month to the PRIDE Archive (Figure 4). [18]

![Figure 4: Number of submitted datasets per month to the PRIDE Archive. Adapted from Vizcaíno et al.[18]](image)

Unfortunately, around 75% of the MS/MS spectra remain unidentified.[19] Most of these spectra seem to be of good quality and have originated from peptides.[20] As we are talking about hundreds of full-proteome datasets, a lot of potential biologically significant information is being missed.

1.3.2 Amino acid variations and PTMs are among the culprits

Two phenomena are known to cause unidentified spectra: amino acid variations and post-translational modifications. As they are not present in the sequence databases that search engines use, both result in unidentified peptides. Amino acid variations can be the result of a technical sequencing error or of a biological polymorphism. As polymorphisms can lead to dysfunctional proteins, a biologically relevant peptide can be missed if it is not identified. Fortunately, it is expected that most of the unidentified peptides are the result of PTMs.[20,21]
If a peptide is modified, the \( m/z \) of all fragment ion peaks that contain the PTM are shifted by the mass of the added atoms. These mass shifts occur with all PTMs. Another, less defined effect of peptide modifications is an altered fragmentation pattern. Modifications can alter the physiochemical properties of the peptide in such a way that it’s fragmentation pattern will change. This means that some fragment ions will have a higher or lower intensity than their unmodified counterparts. With some modifications, such as phosphorylation, a neutral loss can occur. In this event, the PTM dissociates from the peptide as a neutral molecule, before fragmentation. Because this molecule has no charge, it cannot be observed in the spectrum. What we can observe is a very prominent peak that represents the precursor ion, without the PTM. Often its intensity is so dominant, that it obscures the fragment ion peaks.\(^{[22]}\) Examples of a mass shift, an altered fragmentation pattern and neutral loss are shown in Figure 5.

![Figure 5: On top the fragmentation spectrum from an unmodified peptide. On the bottom the fragmentation spectrum of the same peptide sequence, with a serine phosphorylation. Mass shift is indicated with grey arrows. A dominant neutral loss peak is present in the modified spectrum. Also, a change in fragmentation spectrum can be observed. Adapted from Na et al.\(^{[23]}\)](image)

Altered fragmentation patterns are not the only difficulties that arise when analysing modified peptides with mass spectrometry. Most PTMs have a very low stoichiometry, meaning that only a small portion of the peptides is modified. Often a sample will contain a few modified proteins among a pool of their unmodified counterparts. Moreover, to identify a normal protein, any peptide identification will do. To identify a modified protein, however, the exact peptide carrying the PTM needs to be identified. Also, when a peptide sequence has multiple candidate modification sites, localising the exact position of the modification is not straightforward.\(^{[24]}\)
1.3.3 Importance of PTM identification

Even though we know most of the unidentified spectra have originated from modified peptides, it would be foolish to ignore them. Next to alternative splicing, most of our proteome’s diversity is the result of post-translational modifications. They play a key role in cell signalling, metabolism, structure and more. It is not surprising that PTMs need to be studied if we want to unravel diseases such as cancer and Alzheimer’s.[25,26]

Khoury et al. analysed the PTM annotations in the SwissProt database to shed some light on PTM statistics.[27] As can be seen in Figure 6, phosphorylation is by far the most studied PTM. This makes sense, as it is one of the most important regulators of protein activity. Next to N-linked glycosylation, phosphorylation is also expected to be the most abundant PTM.

1.3.4 Existing methods to identify modified peptides

Over the years, a host of bioinformatics tools were developed to identify protein modifications. The classic method uses normal sequence search engines, such as Mascot and SEQUEST. First, the user needs to predefine possible PTMs. Then, the spectrum is matched to the theoretical spectrum from both the unmodified and from all the possible modified versions.[22] Afterwards, the delta score is calculated, which is the difference between the matching scores of the two best versions. The bigger the delta score, the more unambiguous the best match is.[28] The classic method has two big disadvantages, the first being combinatorial explosion.
Due to all possible modifications, the search space and necessary computational time expand dramatically. The second drawback of the classic method is the fact that PTMs need to be specified beforehand. Thus, unexpected modifications will not be found. Additionally, as there are more candidate solutions, these searches tend to have a lower specificity.[29]

As an alternative to the classic method, open modification searches (OMS) were developed.[29] Here, the algorithms try to match the spectrum to the (theoretical or empirical) spectrum from the unmodified version, but allows mass shifts to occur. Most OMS algorithms work in two steps. In the first step, the search space is reduced. Tools such as InsPecT[30] accomplish this by looking for sequence tags in the spectra. Sequence tags are sets of ion peaks that are known to have originated from certain short sequences. With this information, all peptides that are certainly not present in the sample, can be excluded during the open search. Other OMS tools, such as ModifiComb[31], first perform a normal search for unmodified peptides. The rationale behind ModifiComb is that of most modified peptides, their unmodified counterpart is also present in the sample. During the second search, the search space is restricted to all possible modified versions of the previously found peptides. A very recent OMS implementation is MSFragger.[32]

Methods such as pMatch[21], QuickMod[33] and MzMod[34], combine OMS methods with spectral library searching. Modified peptide spectra are also matched to their unmodified versions, but instead of using theoretical spectra, empirical spectra are being used. In this way, the reduced search space is retained and combined with the improved sensitivity of spectral library searching.

1.3.5 MS²PIP for phosphorylated peptides

In this thesis project, a different approach will be taken to identify modified peptide spectra. Instead of using spectral libraries for direct spectrum matching, we will train a machine learning model on these libraries. This model can then be used to predict modified peptide spectra, which in their turn can be used for peptide identification. Because this model combines all knowledge of existing spectral libraries, empirical-like spectra can be predicted for peptides from which currently no empirical spectrum is available. In this way, the improved sensitivity of spectral library searching is retained and we can overcome its biggest disadvantage: our method does not need every peptide to have a known empirical spectrum available.

Fortunately, such a machine learning approach already exists in the form of MS²PIP or MS² Peak Intensity Prediction.[35] At a certain level, it can already handle PTMs, but this capability has not been fully explored yet. First, its current performance on modified peptides will be assessed. Then MS²PIP will be improved upon where possible. As described above,
phosphorylation is the most studied and one of the most abundant PTMs. Therefore, in this project, phosphorylation will be used as a case study.
2 MATERIALS AND METHODS

2.1 MACHINE LEARNING

2.1.1 Introduction

Machine learning is a branch in computational statistics and a form of artificial intelligence. Using lots of data around a complex problem, machine learning algorithms are able to learn how to solve the problem themselves. Depending on the type of data that is given to the algorithm, machine learning can be divided into supervised and unsupervised learning. In supervised learning, the algorithm is trained on fully labelled data. This means that the problem is already solved for all data samples. When applying unsupervised learning, the algorithm uses unlabelled data to find previously unknown relationships. In this project, we use supervised learning.

In supervised learning, data is fitted to a user-defined model. We assume that this model approximates the model that created the dataset. The data is usually presented as a table, in which the rows are samples and the columns are the actual known data characteristics, called the “features”. Selecting the appropriate features for a given problem is a very important part of machine learning. Features are only good if they help the algorithm to solve the problem. Redundant or irrelevant features should be avoided as they add noise to the data. The solution of the problem can be presented as an extra column in the table. Either the solution is a numerical value and is called the “target” or the solution is categorical, in which case it’s called the “label”. Predictors for numerical targets are regression models, predictors for categorical labels are called classification models.

Figure 7: Overfitting. The red line represents an overfitted model. It fits the training data perfectly, but is far too specific to match unseen external data. The green line shows a good model. It is less complex than the overfitted model and fits both the training data and unseen external data.
Too much noise in the dataset can lead to overfitting, a big issue in machine learning. It arises when the model is fitted too specifically on the training data and performs badly on external unseen data. There is always some noise in a dataset, so measures must be taken to prevent overfitting. As can be seen in Figure 7, a complex model tends to overfit the data. Therefore, we need to control the model's complexity. This is called regularization and is done by tuning certain model hyperparameters.

As mentioned before, machine learning can solve complex problems by using lots of data. As more and more proteomics data are becoming publicly available and there are plenty of complex proteomics problems to solve, it makes sense to apply machine learning tools to the field of proteomics.[36]

2.1.2 Decision tree models

A decision tree is a machine learning model that can be represented as a tree-shaped, flowchart-like structure (Figure 8). At every node, a feature is tested with a binary question. For instance: “Is feature A larger than 1,62?” Depending on the answer, true or false, one of the two branches is followed to the next test node. The last “node” of a decision tree is called a leaf and represents a certain outcome: a prediction of the label.

Decision trees are easily interpreted manually. For instance, important features are always queried in the first nodes of the tree, as they are the most decisive. A big disadvantage of decision trees is that they often result in high variance models and are susceptible to overfitting. Therefore, ensemble decision tree models were created. Ensemble methods combine multiple models into one better performing model. Ensemble decision tree models, more specifically, combine multiple decision trees into one model.

![Simple decision tree. Nodes are shown in blue, leaves are shown in grey.](image)
2.1.3 **Random Forest**

One popular ensemble tree method is Random Forests\(^{[37]}\). In its predecessor, Bagging (Bootstrap Aggregating), trees are made on subsets of the training data, selected randomly with replacement. The Random Forest method takes it one step further and also samples the features used in every tree. Thanks to the data sampling, these methods have a built-in cross validation system. Every tree’s performance can be evaluated on the data that was not sampled. This data acts as unseen external data for that specific tree. The error rate on not-sampled data is called the out-of-bag error.

2.1.4 **Extreme Gradient Boosting (XGBoost)**

Extreme Gradient Boosting\(^{[38]}\) is an advanced implementation of the Gradient Boosting technique\(^{[39]}\). Boosting is, just as Random Forests, an ensemble decision tree method. The different trees are, however, not based on different subsets of training data or features. In boosting, trees are created consecutively and are based on the performance of the previous tree. After the first tree is build, data points are given a weight based on their assignment by the first tree. Data points that were assigned correctly are given a lower weight, data points that were assigned incorrectly are given a higher weight. The second tree is then constructed, taking those weights into account. Each individual tree is called a weak learner, as their individual performance is generally low. This is not a big surprise, given that these single trees often contain only one or two nodes. However, when combined the trees form a strong and more complex model. An advantage of XGBoost over other boosting methods is its capability to run in parallel processing. This means that bigger datasets can be analysed much faster, if multiple computer processing cores are available.

2.2 **MS\(^2\)PIP**

2.2.1 **Goal of MS\(^2\)PIP**

MS\(^2\)PIP, or MS\(^2\) Peak Intensity Prediction\(^{[35]}\) is a tool that, as the name implies, can predict the peak intensity of MS\(^2\) spectra. In 1.2.3 we explained that a theoretical peptide spectrum can be easily predicted by calculating the fragment ion’s m/z. Sequence search engines use this theoretical spectrum for peptide identification. Of course, the big disadvantage here is that in theoretical spectra all peaks have the same intensity. Spectral library search engines overcome this problem by using experimental spectra. The big disadvantage here is that the search engines sensitivity depends on the coverage of the spectral library. However, thanks to MS\(^2\)PIP, we can predict the peak intensities and have the best of both worlds.
2.2.2 **MS²PIP Server and MS²PIPc**

During the project, we worked with two distinct versions of MS²PIP. The existing, online accessible, MS²PIP Server\(^{40}\) was used for early validation. All further steps were carried out on the newer MS²PIPc. The older MS²PIP Server has a few key differences in comparison to MS²PIPc. MS²PIP Server groups peptides by their length and charge and builds individual models for every group. The newer MS²PIPc on the other hand, builds one model for all peptides. Secondly, a different machine learning model is used. MS²PIP Server works with Random Forests\(^{37}\), while MS²PIPc uses Extreme Gradient Boosting (XGBoost)\(^{38}\). Thirdly, MS²PIP is written completely in the Python programming language. To speed up certain time-consuming tasks, MS²PIPc also uses C code, hence the new name.

2.2.3 **MS²PIPc feature selection**

MS²PIPc takes as input a PEPREC (peptide record) file, which is a list with peptide sequences, their modifications, charge state and identifier. For training and evaluation, MS²PIPc also takes an MGF file as input. MGF is the abbreviation of Mascot Generic Format\(^{41}\). The file contains for every spectrum some metadata and a list with m/z – peak intensity pairs.

MS²PIPc treats every fragment ion couple (b- and y-ion) as one data point. As such, the dataset provided to the XGBoost algorithm contains one row for every ion couple. All features are derived and calculated from the peptide sequence. Firstly, the peptide length and its charge are added. Secondly, all amino acids are given a percentage of their presence in both of the fragment ions. For instance, an ion with sequence AADK would have 0.5 for cysteine and 0.25 for the other two amino acids. Thirdly, to give more meaning to the sequential information, physicochemical characteristics of the amino acids are added. These characteristics are m/z, basicity, helicity, hydrophobicity and isoelectric point. For every ion couple, the mean, minimum and maximum of these characteristics is calculated over the whole peptide, and over the b- and y-ion. Fourthly, the presence of basic or acidic amino acids and proline is added as a feature. Lastly, the physicochemical characteristics and presence of one of the above mentioned amino acids is given for the first, second, last and second-to-last amino acids of the peptide; and for the amino acids on both sides of the cleavage.

2.2.4 **MS²PIPc target selection**

The targets of MS²PIP are, of course, the fragment ion peak intensities. To extract these from the MGF file, MS²PIPc starts by normalising the spectra. The individual peaks are first divided by the spectrum’s total ion current (TIC), which is the sum of all peak intensities. Then the binary logarithm is taken from the resulting value. The first step is standard normalisation. The
second step emphasizes the lower peaks. This is useful as most spectra contain a few very large peaks that dominate the spectrum.

Then the spectrum is searched for b- and y-ions. This is done by calculation the theoretical spectrum and looking for the highest peak in a certain window around every ion’s m/z. At this point the normal version of MS²PIPc already takes modifications into account. It adapts the fragment ion’s m/z to the modification by adding or subtracting the relevant mass. As an example, this is 79.966331 Da for phosphorylations.

2.2.5 Performance evaluation
To estimate its performance, MS²PIP uses the Pearson correlation by default. During the project, a few tests were carried out concerning the best metric for spectrum correlation with MS²PIP. In these tests, it appeared that the root mean square error (RMSE) is a more accurate metric. The root mean square error is calculated, as the name implies, by taking the square root of the average of all squared errors (Equation 1). For every fragment ion, the error is calculated as the predicted peak intensity ($\hat{y}$) subtracted from the empirical (or target) peak intensity ($y$). For every peptide, the RMSE is then computed using those errors. The result is a distribution of RMSE values for every peptide. As the RMSE is a measurement of the model’s error, a lower RMSE indicates a better model.

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n} (\hat{y}_i - y_i)^2}{n}}$$

Equation 1: Calculation of the Root-mean-square error, metric for a spectrum prediction

2.3 OTHER TOOLS
2.3.1 Spectral library manipulation with SpectraST
As mentioned in 1.2.4.2, SpectraST is a spectral search engine. Besides its Search Mode, it also includes a Create Mode. This mode allows importing and manipulating spectral libraries. Therefore, we used SpectraST to filter spectral libraries for modified peptides, nonmodified peptides and phosphorylated peptides.

2.3.2 Spectrum Similarity
Spectrum Similarity is tool to assess the similarity between fragmentation spectra with a user-specified metric. We used the tool to calculate the correlations between empirical spectra and spectra predicted by MS²PIP Server. As MS²PIPc has correlation calculation built-in, we did not need Spectrum Similarity for the second part of the project.
2.4 DATASETS

2.4.1 PRIDE Cluster dataset
In the PRIDE Cluster project, all complete and publicly available shotgun proteomics datasets from the PRIDE Archive were combined into a single resource.\cite{19} This was done by clustering all MS\textsuperscript{2} spectra by similarity and combining their identifications. If in a cluster of at least 3 identified spectra, 70\% of those spectra pointed to the same peptide sequence, the cluster was regarded as reliable. By doing this, 9.1 million previously unidentified spectra could be linked to a reliable cluster and were, in that way, identified. From the reliable clusters consensus spectra were made and combined into a spectral library. As the PRIDE Cluster dataset combines all datasets in the PRIDE Archive, it seemed to be a perfect candidate spectral library to train MS\textsuperscript{2}PIP on. However, during the project we ran into one big disadvantage: CID and HCD spectra are combined into one dataset and their type is not annotated. As MS\textsuperscript{2}PIP treats both types of spectra differently, this caused problems while training and evaluating.

2.4.2 Synthetic peptide library
Because of the above-mentioned problems with the PRIDE Cluster dataset, another dataset was needed. Marx et al.\cite{44} synthesised and analysed more than 100,000 peptides and their phosphorylated counterparts. They used this synthetic peptide library to evaluate the identification of phosphopeptides with HCD and ETD an evaluated the Andromeda and Mascot search engines. The spectrum files and the search engines’ output files were made publicly available on PRIDE. To use the HCD dataset as a spectral library, the target-decoy approach was carried out on the Andromeda output file. This file already contained annotations on which peptide-spectrum-match (PSM) originated from an actual peptide sequence or from a decoy sequence. As such, the false discovery rate (FDR) was easily calculated at every Andromeda matching score. All PSMs with a score above the 0.01 FDR threshold were included in the library. If a spectrum matched to multiple peptide sequences, the highest scoring PSM was selected.

2.4.3 NIST human spectral library
MS\textsuperscript{2}PIPc is by default trained on the human CID and HCD spectral libraries provided by NIST.\cite{17} Therefore, we used the same dataset to compare our results to. As the human NIST spectral library contains multiple spectra for the same peptide, the spectra with the highest total ion current (which is the sum of all peaks) were selected. By doing this, redundancy in the dataset is removed. Redundancy can introduce unnecessary noise into the dataset, which can lead to overfitting.
2.4.4 Spectral library for external validation

To evaluate the final models on an actual unseen external dataset, we used the HCD data from an experiment where a cell sample was enriched for phosphorylated peptides using Ti-IMAC (Titanium Immobilized Metal Ion Affinity Chromatography).\[45\] For this dataset, the target-decoy approach was already carried out. To select only the best PSMs, we chose for every spectrum the PSM with the best search engine rank and took for every peptide sequence the spectrum with the lowest q-value. The q-value is the lowest FDR at which the PSM would be accepted. The result was a spectral library of 9708 peptides out of which 83% were phosphopeptides. Other modifications were carbamidomethylation and oxidation. Both are present in approximately 10% of the peptides. This is to be expected, as both modifications occur during sample preparation.

2.5 WORKFLOW

2.5.1 Evaluation on MS²PIP Server

2.5.1.1 Parsing PRIDE Cluster spectral library

MS²PIPC was still in full development at the start of the project, so we decided to start by validating MS²PIP’s current performance on phosphopeptides with the fully functional MS²PIP Server. To do this, we downloaded the PRIDE Cluster human spectral library and used SpectraST’s create mode to split up the library into three sublibraries: one with nonmodified peptides (NonMod), one with modified peptides excluding phosphorylation (ModNonPhospho) and one with peptides that are at least one phosphorylated once (Phospho). For some graphs, we made a fourth selection of peptides, which only contain one phosphorylation and, as such, do not contain any other modifications. These peptides are also present in the Phospho sublibrary. All sublibraries were exported in SpectraST’s default file types. SpectraST divides spectral libraries into a file containing all peptide identification (.pepidx) and a file containing all spectral information (.sptxt). Both files are also available in binary formats, which is faster for a computer to read. As we parsed the files line-by-line in Python code, we did not use the binary files. We parsed the peptide identification files and did a quick analysis of the sublibraries. The distribution of peptide length, charge and number of modifications and the prevalence of modifications were plotted. Later on, these plots were also made for the synthetic phosphopeptide library and the NIST library.
2.5.1.2 Getting predictions from the MS²PIP Server

Next, all peptide sequences were sent to the MS²PIP Server, directly from a Python script, through MS²PIP’s REST API. REST is an abbreviation for Representational State Transfer, which is a way of communicating to a server. API, or Application Programming Interface, is a general term for communication channels between computer programs. As such, REST is a type of API. To send the peptide sequences to MS²PIP, they first need to be put in a specific notation, namely JSON (JavaScript Object Notation). Together with the sequences, information about the peptides’ charge and modifications is also present in the JSON notation.

MS²PIP Server has models for both CID and HCD spectra, as such we parsed all peptide sequences to both versions. For every peptide, the predicted b- and y-ion intensities and their m/z values are sent back to the Python script, again through REST, in JSON format. We added these results to the original dataset.

To evaluate MS²PIP’s performance, we need to compare the predictions to the empirical spectra. For this, the .sptxt files were also parsed and were combined with the previous dataset. Both empirical and predicted spectra were exported as MGF files.

2.5.1.3 Calculating spectrum correlations with Spectrum Similarity

By providing MGF files for empirical and predicted spectra to Spectrum Similarity, pairwise Pearson correlation coefficients between the spectra could be calculated. The resulting correlations were parsed into the Python script and combined with the peptide identifications and the spectra. The Pearson correlation coefficients were then plotted in a two-dimensional histogram. Also, the effects of peptide length and number of modifications were plotted.

2.5.2 MS²PIPc

2.5.2.1 Re-evaluation on MS²PIPc

After the initial evaluation on MS²PIP Server, MS²PIPc was, although still being worked on, ready for use. Because there are a few differences between the MS²PIP versions, we repeated the same tests as we did on the MS²PIP Server. The result was, again, plotted in a two-dimensional histogram.

2.5.2.2 Search for another spectral library

However, after a few days, a newer version of MS²PIPc was available. This version introduced the use of a configuration file, where the user can adjust a number of settings. Among those settings is the fragmentation error. This setting defines how broad the window needs to be around the m/z value where MS²PIP searches for a fragment peak. In higher resolution spectra, such as the ones originating from HCD, this window can be smaller, leading to a
higher accuracy. In CID spectra, however, the measurement of a peak’s m/z value is not so accurate and may be further away from the theoretical m/z. Therefore, when using CID spectra, using a small window leads to a lot of peaks being missed. This results in a poor performance of the model. Because of this, the mixture of CID and HCD spectra in the PRIDE Cluster spectral library really became an issue. As such, we started using the HCD data from the synthetic phosphopeptide library. To train MS\textsuperscript{2}PIPc on this dataset, we first processed the files as described in 2.4.2.

### 2.5.2.3 Feature addition

To get a better performing model for phosphopeptides, we began by training the model on the synthetic phosphopeptide library, without altering any code. Then we started adding features. As MS\textsuperscript{2}PIP uses the physicochemical properties of the amino acids to create features, the most elegant solution would be to add the properties of phosphorylated peptides to the model. Unfortunately, data on the basicity, helicity and hydrophobicity of phosphorylated peptides could not be found. Only the mass and isoelectric point of phosphorylated peptides were available. Therefore, we added features that count the number of phosphorylated peptides on the b- and y-ion. In total, this included six added features: for every fragment ion type (b and y) the number of phosphorylated tyrosine residues, serine residues and threonine residues.

As fragment peak intensities depend, among other factors, on the amino acids flanking the fragmentation point, we also added features that define whether there is a phosphorylated peptide at that location. A zero indicates no phosphorylation of that type present, a one means that type of phosphorylation is present. Again, we did this for both b- and y-ions and for the three amino acids that could carry a phosphorylation, resulting in another six added features.

*Table 1: Added features for phosphorylation. In total 12 features were added, each one handling one of the three phosphorylatable amino acids, either counting the phosphorylations on the fragment ions or indicating whether or not a phosphorylation flanks the fragmentation on each one of the ions.*

<table>
<thead>
<tr>
<th>Tyrosine phosphorylation</th>
<th>Count on fragment ion</th>
<th>b-ion</th>
<th>y-ion</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Flanking fragmentation</td>
<td>b-ion</td>
<td>y-ion</td>
</tr>
<tr>
<td>Serine phosphorylation</td>
<td>Count on fragment ion</td>
<td>b-ion</td>
<td>y-ion</td>
</tr>
<tr>
<td></td>
<td>Flanking fragmentation</td>
<td>b-ion</td>
<td>y-ion</td>
</tr>
<tr>
<td>Threonine phosphorylation</td>
<td>Count on fragment ion</td>
<td>b-ion</td>
<td>y-ion</td>
</tr>
<tr>
<td></td>
<td>Flanking fragmentation</td>
<td>b-ion</td>
<td>y-ion</td>
</tr>
</tbody>
</table>
As features were added, we compared the performance of eight different models. Each of the models had one of the four following variations of added features: all added features, only the count feature added, only the fragmentation flanking feature added and no added features. Four of the models were trained on the synthetic phosphopeptide dataset containing only phosphopeptides and the four other models were trained on the synthetic phosphopeptide dataset containing both phosphopeptides and unmodified peptides. The models’ performances were plotted in a point plot. To analyse which features have the biggest influence on the model, we also plotted the features’ XGBoost F-scores. The F-score is simply the number of times a feature is used in a node. The more a feature is used, the bigger the F-score and the more important the feature is to the model.

2.5.2.4 Necessary training dataset size

At this point in the project, we were still using the PRIDE Cluster library as unseen external data for evaluation. As the CID spectra present in the library complicated evaluating the model’s performance, we decided to stop using the PRIDE Cluster data all together. Therefore, we took out a tenth of the synthetic phosphopeptide dataset to use as external unseen data. To be sure that 90% of the original dataset would still be enough to create a performant model, we trained models on dataset sizes ranging from 2000 up to 500 000 fragment ion pairs. The models were evaluated on 10% of the original datasets that was taken out prior to training dataset selection. The test was carried out on three different spectral (sub)libraries. The synthetic phosphopeptide library was split up into a sublibrary containing phosphorylated peptides and a sublibrary containing non-phosphorylated peptides. And for good measure, we also added the NIST spectral library to the test.

2.5.2.5 Training, testing and evaluating in one go

Up until now, every test was carried out within the full MS²PIPc programme. With every change, we had to re-extract the features, retrain the model, recompile some C files, test the model, and export the results to another Python script for analysis. The recompilation is necessary as the decision trees are converted to C-code for faster processing. C-code always needs to be compiled, which means the code is being translated from human readable programming scripts to machine readable instructions. Python code, on the other hand, does not need compilation, as it is translated to machine instructions by an interpreter while the code is being executed. Using C-code in MS²PIPc makes sense, as it greatly improves processing speed when using the programme for predictions with the same model every time. But if we want to train and test a lot of new models, compiling slows down the process.
To work more efficiently, we took the necessary Python code out of MS²PIPC, so we could train, test and evaluate within one Python script.

2.5.2.6 Parameter tuning

Like most machine learning models, XGBoost has hyperparameters that need to be tuned in order to create an accurate model. Most hyperparameters are used to find the balance between underfitting and overfitting. We tuned four important XGBoost hyperparameters: eta, gamma, max_depth and min_child_weight.

- **Eta** is the learning rate. It defines how much more weight is given to incorrectly assigned data points than to correctly assigned data points at each boosting step. Setting eta lower leads to more robust models, but needs more boosting iterations to get a good performance.
- **Gamma** specifies how much loss reduction is needed for a node to be split. To put it simply, a higher gamma lets XGBoost split a node only if it is of enough value to the model. As such, setting a high gamma can prevent overfitting.
- **Max_depth** is, as the name says, the maximum depth a three can be. As XGBoost is an ensemble tree method, the individual trees can be very short and only contain a few nodes. In some models the trees can even exist out of a single node. These trees are called stumps. Reducing tree depth prevents overfitting.
- **Min_child_weight** defines the minimum sum of the weights of all leaves a node needs to be split. It prevents that nodes are made that lead to leaves with only a few data samples in them. As such, a higher value prevents overfitting.

To tune these parameters, a grid search was carried out. In this process, the best combination of hyperparameter values is searched out or a range of given values. This is done in an exhaustive manner, meaning all possible combinations are tested. The grid search was performed on two datasets (synthetic phosphopeptide library with both phospho- and nonmodified peptides and the same library with only phosphopeptides) and four variations of the added features (no additional features, only adding the phosphopeptide counting features, only adding the fragmentation-flanking phosphopeptide feature and adding all additional features). The grid search included a range of values for all above mentioned hyperparameters, except for eta. To reduce computing time, this parameter was set to 1. In this way, 10 rounds of boosting seemed to be sufficient.

Every model, originating from different tests in the grid search, was evaluated by a three-fold cross-validation. In cross-validation, the dataset is split into a number of folds. Each iteration, one fold is left out of the training dataset. The model is then tested on the fold that was left out.
The process repeats until all folds have been tested on. Then, the scores on the testing and the training dataset from different folds can be averaged. By performing cross-validation, the model can be tested on a virtual unseen external dataset. In this way, overfitting can be assessed without having to take out another part of the dataset.

2.5.2.7 Final model evaluation

The final models were evaluated in two ways. First, the models were evaluated on the 10% of the dataset that was taken out prior to training. Hence, the models trained on data from the synthetic phosphopeptide sublibrary containing only phosphopeptides were trained on the 10% that was taken out of that sublibrary and the models trained on data from the synthetic phosphopeptide sublibrary containing both phosphopeptides and unmodified peptides was trained on the 10% that was taken out of that sublibrary. Secondly, we used the data from the phosphopeptide-enriched experiment (see 2.4.4) as unseen external data. Logically, in this second test all models were evaluated on the same dataset. For a comparison with the default version of MS²PIPc, we also predicted the three evaluation datasets with the default MS²PIPc HCD models. The RMSE-scores were calculated for every peptide and the distributions were plotted in box plots.
3 RESULTS

3.1 SPECTRAL LIBRARY PROPERTIES

As explained in the workflow, we started by plotting the properties of the PRIDE Cluster spectral library. Later on in the project, the properties of the other spectral libraries were also added. These plots were made mostly to check that the datasets were divided correctly into sublibraries and whether there are any big differences between and within the spectral libraries that were used. When comparing MS²PIP’s performance on phosphopeptides versus nonmodified peptides, it is important to know that there are no other factors influencing the results.

Spectral libraries shown are PRIDE Cluster with only unmodified peptides (Cluster, NonMod); PRIDE Cluster with modified peptides, but no phosphopeptides (Cluster, ModNonPhospho); PRIDE Cluster, peptides with at least one phosphorylation (Cluster, Phospho); Synthetic phosphopeptide library, only phosphopeptides (Synthetic, only phospho); Synthetic phosphopeptide library, both phosphor and unmodified peptides (Synthetic, both phospho and unmodified); the NIST spectral library, which was not specifically filtered for modifications (NIST).

![Graph showing charge distribution of peptides across different spectral libraries.](image)

**Figure 9:** Percentages of peptides with a given precursor peptide charge within a spectral library. Peptides with charges up to +16 were also present, but in negligible amounts. Therefore, these charges were not added to the graph. Percentages were calculated within the respective library.

Most noteworthy in the charge distribution (Figure 9) is the relatively larger amount of charge +3 peptides in the Phospho PRIDE Cluster sublibrary, compared to the other spectral libraries. However, this difference is not dramatic and no explicit causes could be found. This effect is not seen in the synthetic phosphopeptide library.
Figure 10: Estimated distribution of peptide length within each spectral library. Plotted with Kernel Density Estimation from the Python Seaborn package\cite{46}, using a Gaussian kernel with bandwidth 2.

Peptide lengths are all averaged between 14 and 18 amino acids. Within the PRIDE Cluster data, phosphorylated peptides tend to be on average three amino acids longer than unmodified peptides. Perhaps a longer peptide increases the chance that one of the three phosphorylatable amino acids is present.

Figure 11: Amino acid composition within each spectral library. All peptide sequences from their respective spectral library were combined and the relative occurrence of each amino acid was calculated.
In Figure 11, the amino acid composition of the different spectral libraries is depicted. Notable is the larger amount of cysteine and methionine in the PRIDE Cluster ModNonPhospho sublibrary. This is probably due to an enrichment of oxidised peptides in this sublibrary. In the PRIDE Cluster Phospho sublibrary, the same effect is seen for serine, which can be phosphorylated. Threonine and tyrosine, the other two phosphorylatable amino acids, do not show this enrichment in phosphopeptide-containing libraries. Also enriched in the PRIDE Cluster Phospho sublibrary is proline. This makes sense, as the large family of proline-directed protein kinases targets serine and threonine residues preceded by proline.\textsuperscript{[47]}

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{peptide_modifications.png}
\caption{Percentage of peptides with a given number of modifications within each sublibrary.}
\end{figure}

The number of modifications per peptide (Figure 12) was plotted mainly to check that the PRIDE Cluster NonMod sublibrary had no modified peptides, and that the other PRIDE Cluster sublibraries consisted out of peptides with at least one modification. The same principle applies to the Synthetic Phospho sublibrary, that should not contain any unmodified peptides. The PRIDE Cluster sublibraries have on average more modifications per peptide than the other libraries. This is most likely due to the fact that the PRIDE Cluster library contains more types of modifications (Figure 13) and, as such, contains more types of amino acids that can be modified. The chance that a modifiable amino acid occurs in a peptide is therefore higher in this library.
Figure 13: Prevalence of modifications within each sublibrary. For every sublibrary, modifications were counted and percentages were calculated to the total number of modifications in the library. Modifications were sorted on the y-axis by mean percentage over all libraries.

Figure 13 shows the prevalence of modifications in the different sublibraries. All three types of phosphorylation are among the highest in the ranking. Carbamidomethylation and oxidation are also present in high numbers. This also makes sense, as these modifications are added frequently during sample preparation. The other modifications, among which a host of analytical labels, are seen mostly in the PRIDE Cluster library.

3.2 PERFORMANCE OF MS²PIP SERVER ON PHOSPHORYLATED PEPTIDES

After MS²PIP Server was tested and its spectrum predictions were compared to the original spectra using the Spectrum Similarity tool, the Pearson correlations were plotted in a two-dimensional histogram (Figure 14). In the histograms, we can (especially on the plots the CID model and charge +2 data) clearly see that the Pearson correlation drops as soon as phosphorylated peptides are introduced. Other modifications, however, do not influence the performance as much and their plots are comparable to the plots from unmodified peptides. This leads us to believe that phosphorylations do not simply shift a peptide’s m/z (which MS²PIP takes into account), but also introduces a different fragmentation pattern. This altered fragmentation results in different peak intensities compared to the unmodified spectra. A new model, specifically for phosphorylations, could increase MS²PIP’s performance on phosphopeptides.
Figure 14: Two-dimensional histograms of the performance of MS²PIP Server (CID and HCD models), tested with PRIDE Cluster data. Every pixel represents one x-y bin of the peptides’ molecular weight (MW) and Pearson correlation. The more peptides in a given x-y bin, the darker its colour. Pearson correlation was calculated for each peptide between the spectral library spectrum and the MS²PIP Server predicted spectrum, with the Spectrum Similarity tool. Sublibraries were split by charge, as each charge has its own models in MS²PIP. Peptides with only one phosphorylation were also shown in a fourth plot. These peptides do not contain any other modifications than the single phosphorylation.
3.3 PERFORMANCE OF MS²PIPc ON PHOSPHORYLATED PEPTIDES

As we were starting to use a new version of MS²PIP, the same analysis was repeated (Figure 15). The CID model performance is comparable to that of MS²PIP Server. Performance is good on unmodified peptides and on modified peptides without any phosphorylations, while phosphopeptides decrease the model’s performance. Peptides with charge +2 are predicted better than peptides with charge +3. However, when observing the HCD model’s performance, something is clearly wrong. A large group of peptides is clustered at a very low Pearson correlation. These peptide spectra most likely originated from CID experiments. As explained in 2.5.2.2, the newer MS²PIPc version enables the use of a smaller peak searching window for HCD spectra. With this lower fragmentation error set, peaks from the lower-resolution CID spectra cannot be found by MS²PIP.

To investigate this theory properly, we plotted the performance of the CID and HCD models on unmodified spectra against each other, with different fragmentation errors (Figure 16). We can see the predictions of the presumed CID spectra improving for both models, as the fragmentation error becomes bigger. The presumed HCD spectra, on the other hand, are predicted well at all fragmentation errors. As expected, the HCD model scores best on the presumed HCD spectra and the CID model scores (at a fragmentation error of 0.8) best on the presumed CID spectra.

Furthermore, when plotting a histogram of the ratio of peaks in a peptide that are not found by MS²PIPc for both high scoring and low scoring peptide clusters (Figure 17), the two groups of peptides have distinct distributions. High scoring peptides, which we presume to have originated from HCD experiments, tend to have all their peaks found by MS²PIPc. Low scoring peptides, on the other hand, which we presume to be from CID experiments, have on average only half of their peaks found by MS²PIPc. Figure 16 and Figure 17 thereby confirm our theory that MS²PIPc has trouble finding peaks from CID spectra, if a small fragmentation error is set.

While posing a problem for this project, this data shows us that we could use MS²PIP as a tool to identify the fragmentation method that generated a given peptide spectrum. By predicting the spectrum with both the CID and the HCD model and comparing both predictions to the actual spectrum, we could identify the spectrum’s fragmentation method. If we also incorporate a different fragmentation error with both tests, the CID and HCD spectra seem to be separated reasonably well. Of course, this specific use of MS²PIP needs further investigation.
Figure 15: Two-dimensional histograms of the performance of MS2PIPC (CID and HCD models), tested with PRIDE Cluster data. The plots were created with the same methods as the plots in Figure 14. The CID model uses a fragmentation error of 0.8 m/z, the HCD model uses a fragmentation error of 0.02 m/z, both are routinely used for each technique.
Figure 16: Two-dimensional histograms of the MS²PIPc CID and HCD models, tested only with unmodified PRIDE Cluster data and different fragmentation errors. The performance of the CID model is plotted on the x-axis, against the performance of the HCD model (for the same peptide) on the y-axis. Three different fragmentation errors were used, for both models. Fragmentation error 0.02 (left) is routinely used for HCD spectra, fragmentation error 0.8 (right) is routinely used for CID spectra. Fragmentation error 0.1 (middle) was added as an intermediate.

Figure 17: Histogram of the ratio of unfound peaks. Peptides from the PRIDE Cluster unmodified sublibrary were separated by their Pearson correlation from the HCD model with a fragmentation error of 0.02. Peptides with a correlation higher than 0.6 were considered ‘High scoring spectra’ and originating from HCD experiments, peptides with a correlation lower of equal to 0.6 were considered ‘Low scoring spectra’ and originating from CID experiments. For every peptide, the ratio of b- and y-ions that MS²PIPc did not found in the empirical spectrum, and thus were assigned an intensity of zero, was calculated. From these ratio’s, a histogram was plotted for both peptide groups.

Because we cannot rely on MS²PIP’s separation of CID and HCD spectra with high confidence, and because the mixture of both types of spectra makes analysing the results quite difficult, the need for another spectral library for both training and evaluating MS²PIPc grows. Therefore, we started using the synthetic phosphopeptide library. Our conclusion that MS²PIP has problems predicting phosphopeptides, however, still holds.
3.4 NECESSARY TRAINING DATASET SIZE FOR MS²PIPC

To be sure that the synthetic phosphopeptide spectral library provides enough data points to train an MS²PIPC on, we tested the models’ performance in function of the training set size. Figure 18 shows us that for all three of the used datasets approximately 200 000 fragment ion pairs seems sufficient to train a well performing model. However, performance still steadily grows as the training set gets larger, even past 200 000 fragment ion couples. This emphasizes that the general characteristic of machine learning, that more data is better, is also true in the case of MS²PIPC. These plots also show us that, when tested on an unseen portion of the same dataset, better performing models are created with the synthetic peptide library then with the NIST library.

Figure 18: MS²PIPC performance on different training dataset sizes. Models were trained on training dataset sizes ranging from 2000 up to 500 000 fragment ion pairs. Three datasets were tested: synthetic peptide library with phosphopeptides (Phospho, left graphs), synthetic peptide library with unmodified peptides (NonMod, center graphs) and the NIST dataset (right graphs). Models were evaluated on 10 percent of the complete datasets, which was taken out prior to training dataset selection. As evaluation metrics, both Pearson correlation (top graphs) and root mean square error (RMSE, bottom graphs) were used. A higher Pearson correlation indicates a better performance, and a lower RMSE indicates a better performance. MS²PIPC makes different models for b- and y-ions and as such, the performance of these models is shown separate. Blue dots represent the b-ion models, green dots represent the y-ion models.
3.5 PARAMETER TUNING

The grid search returned similar results for all datasets and all variations on feature addition. Gamma values between 0 and 1 led to the best results, across all other parameter values. For some models a gamma of 0 gave the best result, while some other models were at their best with a gamma of 1. When max_depth and min_child_weight were set to values that allowed overfitting, a gamma of 1 slightly reduced the overfitting. Therefore, we chose to set gamma to 1 for further testing.

Max_depth and min_child_weight need to balance each other: at least one of them needs to be more conservative to prevent overfitting. The models' performance gets better as the trees are allowed to be deeper. However, a higher min_child_weight is needed to prevent overfitting (Figure 19). When we turn the strategy around and use a limited tree depth and test different minimum child weights, overfitting is also prevented, but the model's performance is not as good as it is with the first strategy. Therefore, we chose to allow slightly deeper trees and mainly use min_child_weight to prevent overfitting. After more finetuning (Figure 20) a min_child_weight value of 570 was chosen.

Figure 19: Example of XGBoost hyperparameter tuning for max_depth and min_child_weight on the synthetic phosphopeptide dataset, with gamma 1 and 10 rounds of boosting. These plots are the result of the dataset with both phosphopeptides and unmodified peptides and all added features included. Plots for other grid search tests show similar results. The RMSE values shown, are the averages over the three cross-validation folds. The standard deviations are shown as vertical lines on the plot. For every fold, the model is evaluated on both the current test fold and on the current training folds. On the left, the results of the test set are shown, on the right, the results on the training set are shown. The x-axis indicates the tested value of max_depth and the different values of min_child_weight are shown as individual lines. If for a certain combination of hyperparameters, the RMSE is very low on the training set, but not on the test set, the model is overfitting.
Figure 20: Finetuning of min_child_weight. These plots are similar to the plots in Figure 19, only here the x-axis shows the values of min_child_weight and the different lines represent different max_depth values. The plots for max_depth values larger than 10 are all superimposed, as trees evidently did not grow much deeper than 50 nodes.

To reduce computing time during initial parameter optimisation, we limited the number of boosting rounds to 10. As this results in a limited number of trees, deeper trees are needed to obtain a good performing model. However, to make the final model we allowed more boosting rounds and could, as such, reduce the three depth to further prevent overfitting. Figure 21 shows us that the performance on the testing set converges at approximately 75 rounds of boosting and for max_depth values of 15 and higher. On the training set however, performance keeps rising, even with max_depth values larger than 15. This shows us that max_depth values of more than 15 do not improve the model and only allow for more overfitting. Therefore, we selected a max_depth of 15 and found 75 rounds of boosting sufficient.

Figure 21: Example of XGBoost hyperparameter tuning for max_depth and the number of boosting rounds. This plot is made with the same methods as Figure 19 and Figure 20, only now the x-axis shows the number of boosting rounds and the different lines represent different max_depth values. Gamma was set to 1 and min_child_weight was set to 570.
3.6 FEATURE ADDITION

Figure 22 shows us the importances of the features we added. Clearly, the added features are not among the top features for the model. This is not surprising, keeping in mind that we are predicting peak intensities, which are influenced by a lot more than just the presence of a phosphorylation.

![Swarm plot of feature importances.](image)

Figure 22: Swarm plot of feature importances. The F-score indicates the number of times a feature was used in a node. The higher the F-score, the more important the feature is. Added features for phosphorylation are coloured in orange. Four models are depicted, for both ions a model was trained on the synthetic phosphopeptide sublibrary containing only phosphopeptides (OnlyPhospho) and on the synthetic phosphopeptide sublibrary containing both phosphopeptides and unmodified peptides (Both).

To get a better idea of which of the added features is more valuable to the model, we plotted the importances of only the added features (Figure 23). Even more interesting is to see which features get more important when unmodified peptides are also added to the training dataset. We expect that with such a “mixed” training dataset, it becomes more important for the model to differentiate fragment ions with phosphorylations from fragment ions without phosphorylations. In the top graph of Figure 23, we can observe that the counting features for serine and threonine phosphorylation indeed get more important, depending on the fragment ions for which the model is trained. The count of tyrosine phosphorylations is an exception. Its importance does not increase as much and sometimes even decreases.

All features indicating whether a phosphorylation is flanking the fragmentation position (except for serine and threonine phosphorylations on the b-ion in the b-ion model) increase in importance when the training dataset also contains unmodified peptides (Figure 23, bottom). Most remarkable here is that for these added features, the ones for tyrosine phosphorylation are not used by any of the models. In concordance with the behaviour of counting features for
tyrosine phosphorylation, this may indicate that tyrosine phosphorylation does not influence fragmentation patterns as much as serine and threonine phosphorylations do.

From these plots, we can also conclude that the counting features (F-scores between 19 and 90) get a much higher importance than the features indicating a phosphorylation flanking the fragmentation point (F-scores between 0 and 19).

Figure 23: Feature importances for added features that count the number of phosphorylation on each fragment ion (top) and for added features that indicate whether a phosphorylation flanks the fragmentation position (bottom). A comparison is made between models trained on only phosphopeptide data and models trained on both phosphopeptide and unmodified data. These plots show us which features become more important when the model is trained on mixed datasets (containing both phosphopeptides as unmodified peptides).
Figure 24: Evaluation and comparison of the performance of models with four variations of feature addition. Four types of models were evaluated: without added features, with the features indicating phosphorylation flanking the fragmentation position (Frag), with the features counting the number of phosphorylations on the fragment ions (Count) and with both types of features added (Count & Frag). For every of the four types, two models were trained: either on only phosphopeptides (triangular markers and lighter colours) or on both phosphopeptides and unmodified peptides (round markers and darker colours). The different models for b- and y-ions and their results are also shown separately: b-ion in shades of blue and y-ion in shades of red. The models were evaluated on the 10% of the synthetic phosphopeptide library that was taken out prior to training (left graph) and on the external unseen dataset described in 2.4.4 (right graph).

The performance validation of models trained with variations on feature addition (Figure 24) show different results between internal validation (on synthetic phosphopeptide data) and external validation. With internal validation, the count feature clearly has the most positive effect. Unexpectedly, adding both types of features results in slightly less performant models then when only the count feature is added. This might indicate redundancy between the two types of features. When looking at the external validation, making a conclusion gets more complex. Here, the results do not show the same trend for both training datasets. The differences are also less pronounced in comparison with the internal validation. It must be noted however, that the differences for all plots in Figure 24 are very minor.
3.7 EVALUATION OF THE FINAL MODELS

In Figure 25 and Figure 26 the final phosphorylation models are compared to the default MS²PIPc models. A few conclusions can be made from these plots:

- First and most importantly, the phosphopeptide models clearly outperform the default MS²PIPc models, when evaluated on phosphopeptide datasets. This can be seen for the evaluation on both synthetic phosphopeptide datasets, as well as for the external evaluation dataset.
- While the performance between models with different feature additions was very similar (see 3.6), the phosphorylation models do outperform the default MS²PIPc models. This implies that most of the strength on phosphopeptides comes from retraining the model on a phosphopeptide spectral library.
- The difference in performance between models trained on both phosphorylated and unmodified peptides and models trained on only phosphorylated peptides is very minimal and varies between b- and y-ion models.
- The b-ion models (left graphs) always perform better than the y-ion models (right graphs). This is also seen for the default MS²PIPc models.
- The scores for the evaluation on the synthetic phosphopeptide library (top graphs) are consistently better than the scores for the unseen external data (bottom graphs). We expected this, as the models were trained on data originating from the same dataset. Nevertheless, this effect is also seen for the default MS²PIPc models. This may be a result of the synthetic phosphopeptide dataset being less noisy and containing less uncommon peptide sequences than the external dataset. Another explanation might be that the external dataset contains spectra of lower quality.
Figure 25: Evaluation of the final models and comparison with the default MS²PIPC models. The two phosphorylation models use the hyperparameters that resulted out of the grid search described in 3.5 and include all added features, as described in 3.6. One phosphorylation model was trained on the synthetic phosphopeptide sublibrary with both phosphorylated and unmodified peptides, the other phosphorylation model was trained on only phosphopeptide data. The data used for validation is for both phosphopeptide models and the default MS²PIPC model the 10% of the data that was taken out of the synthetic phosphopeptide library prior to training. For the phosphopeptide models, the two sublibraries used for evaluation correspond to the two sublibraries used for training. The box plots show the distribution of RMSE scores for every peptide spectrum prediction. As the models for b- and y-ions show different results, their results are depicted in different boxplots.

Figure 26: External validation of the phosphopeptide models and comparison with the default MS²PIPC model. These plots are similar to the plots in Figure 25, with the only difference the dataset that was used for evaluation. For these plots, the models were evaluated on the external unseen spectral library described in 2.4.4.
4 DISCUSSION

By retraining MS\(^2\)PIPc on a phosphopeptide spectral library, adding phosphopeptide-specific features and performing hyperparameter optimisation, our model was able to outperform the default MS\(^2\)PIPc HCD models on an unseen external dataset. This is, however only a first, small step towards the use of MS\(^2\)PIP for the identification of unexpected post-translational modifications.

At this point we have separate models for normal peptide spectra and phosphopeptides spectra. Hence, when using the phosphopeptide models, we inherently expect phosphopeptides to occur in the sample. In an ideal situation, MS\(^2\)PIPc would be trained on vast amounts of unmodified and variously modified peptide spectra. If those future MS\(^2\)PIPc models can integrate features for all known modifications, the models would in theory be able to predict altered fragmentation spectra for all of these modifications. With this case study on phosphopeptides in mind, this theory seems plausible. At the beginning of the project, we showed that phosphopeptides make it more difficult to be predicted than other modifications. Still, we were able to create a well-predicting model for phosphopeptides.

The next step, after creating a universal modifications-friendly MS\(^2\)PIP, is the implementation into a search engine. At this point, the connection of MS\(^2\)PIP to the in-house developed search engine, Omega, is in the works. For the search engine to be able to truly identify modifications unexpectedly, some sort of Open Modification Search method would have to be developed into the search engine. As explained in 1.3.4, various OMS methods already exist and more are being developed.

The result would be a pseudo-spectral open modification search engine. The search engine could collect all previously obtained spectral data into one machine learning model, reducing redundancy and noise and combining all knowledge of fragmentation patterns. It would retain spectral search engines’ improved sensitivity, but overcome their biggest disadvantage: all theoretical peptides could now be searched for, instead of just the ones present in the library. Also, all theoretically possible modified peptides could be predicted. Of course, the possibility to predict every modified variation of any peptide spectrum, explodes the search space dramatically. So, we would need a way to reduce the search space again, which may be found in existing OMS methods. All of this is still a long way ahead, but forms an interesting future for peptide identification by mass spectrometry-based proteomics.
5 CONCLUSION

The PRIDE Cluster spectral library seemed to be very promising, but due to the mixture of unannotated CID and HCD spectra, the library was unusable for this project. Otherwise it would have been an ideal resource for MS$^2$PIP, as it combines all datasets submitted to the PRIDE Archive in a non-redundant manner. Hopefully, in a next iteration of the PRIDE Cluster project, these annotations will be added. This problem did help us discover that MS$^2$PIP could be used as a tool to discriminate CID from HCD spectra. However, this potential needs further research, mainly to assess its accuracy and precision.

We were able to show that both MS$^2$PIPc and MS$^2$PIP Server have difficulties to predict phosphopeptide spectra, even more so than other modified spectra. This confirms that phosphorylation does not only introduce a mass shift in the spectrum, but actually alters the fragmentation pattern and thereby the peak intensities. Retraining MS$^2$PIP on phosphopeptides and adding some phosphorylation-specific features therefore seemed like a good idea.

As the training set size has a big influence on a machine learning model’s performance, it was important to know how big our spectral libraries needed to be. 200 000 fragment ion couples seemed to be sufficient, although more data kept improving the predictions. This means that if we assume an average peptide length of 11 amino acids, 20 000 peptide spectra would suffice. The synthetic peptide library contains more than 90 000 spectra (with approximately half of them phosphopeptides), which meant we had plenty of data.

Parameter tuning was a very time-consuming part of the analysis. We selected values for gamma, max_depth, min_child_weight and the number of boosting rounds, based on a number of simultaneous grid searches with cross-validation. However, we are confident that with a more thorough search, combining all hyperparameters in a simultaneous grid search or in a random search, more precise hyperparameters can be found. Nevertheless, we were able to train models that outperform the default MS$^2$PIPc models on phosphopeptide spectra.

This project is only a small, but nonetheless essential step towards identifying unexpected post-translational modifications. We showed that even one of the more difficult modifications to predict can be modelled, if only enough spectral data is available. With continuously growing online proteomics databases, such as the PRIDE Archive, enough data will soon, if not already, be available for all known modifications.
6 REFERENCES


5. PRIDE Archive. (2017). Available at: https://www.ebi.ac.uk/pride/archive/.


