LIÈGE université Library

https://lib.uliege.be

https://matheo.uliege.be

Harnessing artificial intelligence to predict quantotypic property of peptides in targeted bottom-up proteomics

Auteur : Walraff, Jimmy
Promoteur(s) : Huynh-Thu, Vân Anh; Pierre, Nicolas
Faculté : Faculté des Sciences appliquées
Diplôme : Master : ingénieur civil en science des données, à finalité spécialisée
Année académique : 2024-2025
URI/URL : http://hdl.handle.net/2268.2/23220

Avertissement à l'attention des usagers :

Tous les documents placés en accès ouvert sur le site le site MatheO sont protégés par le droit d'auteur. Conformément aux principes énoncés par la "Budapest Open Access Initiative"(BOAI, 2002), l'utilisateur du site peut lire, télécharger, copier, transmettre, imprimer, chercher ou faire un lien vers le texte intégral de ces documents, les disséquer pour les indexer, s'en servir de données pour un logiciel, ou s'en servir à toute autre fin légale (ou prévue par la réglementation relative au droit d'auteur). Toute utilisation du document à des fins commerciales est strictement interdite.

Par ailleurs, l'utilisateur s'engage à respecter les droits moraux de l'auteur, principalement le droit à l'intégrité de l'oeuvre et le droit de paternité et ce dans toute utilisation que l'utilisateur entreprend. Ainsi, à titre d'exemple, lorsqu'il reproduira un document par extrait ou dans son intégralité, l'utilisateur citera de manière complète les sources telles que mentionnées ci-dessus. Toute utilisation non explicitement autorisée ci-avant (telle que par exemple, la modification du document ou son résumé) nécessite l'autorisation préalable et expresse des auteurs ou de leurs ayants droit.

UNIVERSITE DE LIEGE
FACULTE DES SCIENCES APPLIQUEES

Harnessing artificial intelligence to predict quantotypic property of peptides in targeted bottom-up proteomics

WALRAFF Jimmy

Thesis presented to obtain the degree of :
Master of Science in Data Science Engineering

Thesis supervisor :
HUYNH-THU Vân Anh
Thesis co-supervisor:
PIERRE Nicolas

Academic year: **2024 - 2025**

Acknowledgments

Je tiens à remercier tout particulièrement mes promoteurs, Vân Anh, Nicolas et Dominique, pour leur encadrement, leurs conseils avisés et leur grande disponibilité tout au long de ce travail. Leur soutien constant, tant sur le plan technique que dans la rédaction de ce mémoire, a été d'une aide précieuse. Ce travail n'aurait pas pu aboutir sans leur accompagnement, leur patience et leur implication à chaque étape du projet.

Je tiens également à remercier mes parents, qui, tout au long de mes années d'études, m'ont offert un foyer rempli d'amour, de bienveillance et de stabilité. Leur soutien inconditionnel a grandement contribué à ma réussite et m'a permis d'avancer sereinement dans mon parcours. Je souhaite aussi remercier ma copine, Brendy, pour son amour, sa patience et son soutien précieux au quotidien. Finalement, je tiens à remercier ma famille et mes amis, qui m'ont permis de m'évader du cadre du travail pour me changer les idées, partager des moments de détente et de rires. Grâce à toutes ces personnes, j'ai pu trouver un équilibre entre travail et bien-être, qui m'a permis de garder la motivation et la sérénité nécessaires à la réalisation de ce travail.

Statement of autorship

I, Jimmy Walraff, born on February 5, 2002, confirm that I am the author of the master's thesis titled "Harnessing artificial intelligence to predict quantotypic property of peptides in targeted bottom-up proteomics."

All the sources I used have been properly mentioned and referenced. The figures in this thesis were either created by me, made using my own code, or taken from other sources with correct citations.

I also confirm that I used ChatGPT to refine the language and improve the readability of my original text.

Abstract

Proteins are essential biomolecules that perform a wide range of functions in all living organisms. Proteomics, the large-scale study of proteins, aims to characterize their structure, function, and abundance in biological systems. Within this field, one major challenge is the accurate quantification of proteins through their peptides. In particular, there is a lack of tools capable of predicting the quantotypic property of peptides, a task which could reduce reliance on expensive and time-consuming experiments. This thesis explores the feasibility of using artificial intelligence to predict the quantotypic property of peptides based solely on their amino acid sequences. We first applied supervised learning techniques on a labeled dataset, testing classical machine learning models (Random Forest, XGBoost) and deep learning architectures (Multilayer Perceptron, Bidirectional Long Short-Term Memory). Both training from scratch and transfer learning strategies were evaluated. Transfer learning used pre-trained models on peptide-related tasks (AlphaPeptDeep) and protein-related tasks (Evolutionary Scale Modeling). Two transfer learning approaches were compared: fine-tuning and feature extraction. To deal with severe class imbalance in the data, cost-sensitive learning and Random Oversampling were applied. In the second part of the thesis, we explored self-training to leverage unlabeled data. We tested three pseudo-labeling strategies: threshold-based, proportion-based, and optimal thresholding. Additionally, we experimented with both hard and soft pseudo-labels, and introduced a dual loss function to further penalize the misclassification of true labels. Despite the modest results, this work represents a first step towards the development of predictive tools for the quantotypic property of peptides and provides valuable insights for future research in this field.

Contents

Chapter 1

Introduction

1.1 Context and general objective of the thesis

Proteins are essential biomolecules that perform a wide range of functions in all living organisms. They are composed of chains of amino acids, linked together by peptide bonds to form complex three-dimensional structures. The specific sequence of amino acids, dictated by genetic information, determines a protein's structure and function. Proteins play diverse roles, including catalyzing biochemical reactions (enzymes), providing structural support, facilitating molecular transport, and mediating cellular communication. Given their importance in biological systems, the study of proteins is crucial for understanding life at the molecular level. There are a total of 20 different amino acids incorporated into proteins (proteinogenic), represented by a letter in practice (see Figure 1.1).

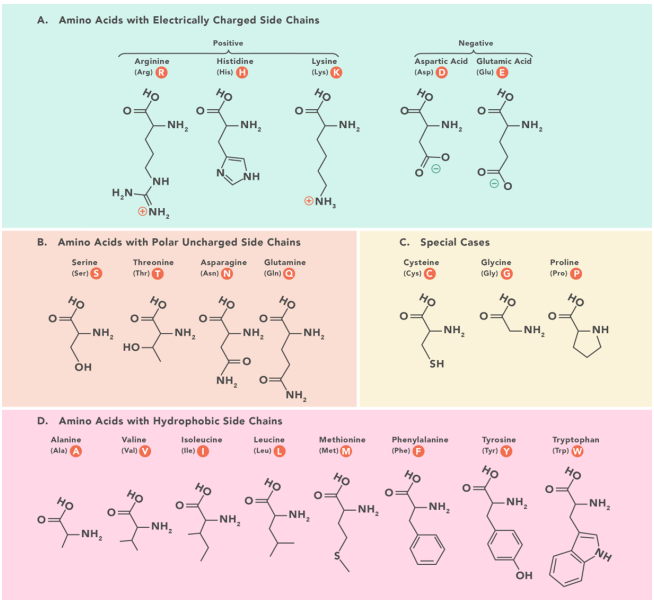
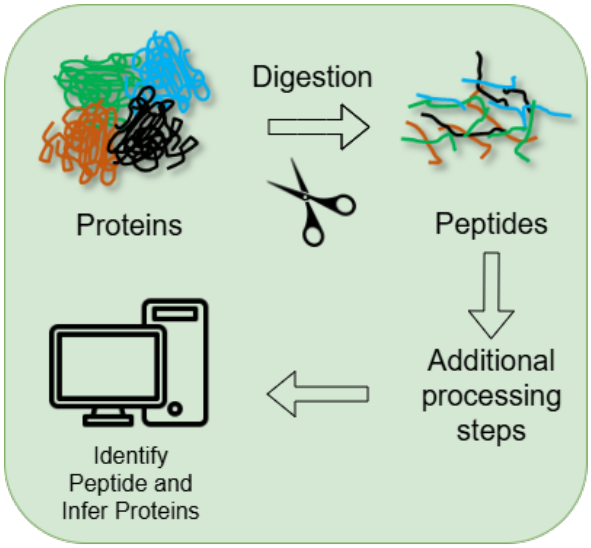


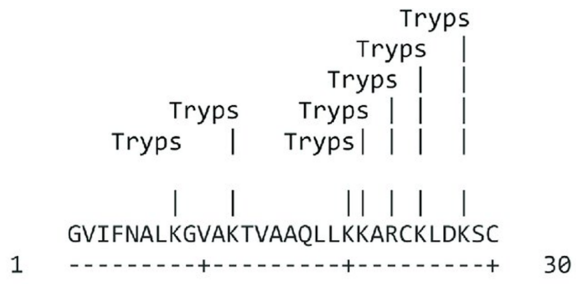
Figure 1.1: Proteinogenic amino acids. Reproduced from [1].

Proteomics is the large-scale study of proteins, encompassing their structure, function, interactions, and expression within a given biological system. One of the main challenges in proteomics is the accurate quantification of proteins, which is essential for understanding their biological roles and identifying potential biomarkers in various medical and biotechnological contexts.

Mass spectrometry-based proteomics (MS-based proteomics) is a powerful analytical technique used to identify and quantify proteins in biological samples. In MS-based proteomics, there exist two possible approaches: top-down and bottom-up. The top-down approach involves analyzing intact proteins directly in order to identify and characterize protein structure. On the other hand, bottom-up proteomics begins with the enzymatic digestion of proteins into smaller amino acids sequences, called peptides (see Figure 1.2a). This digestion is carried out by specific proteins, called enzymes, which catalyze the cleavage of proteins. One of the most widely used enzymes is trypsin, which cleaves proteins at the carboxyl side of lysine (K) and arginine (R) amino acids (see Figure 1.2b). After protein digestion into peptides, the resulting peptide mixture is typically separated by liquid chromatography (LC) and analyzed by tandem mass spectrometry (MS/MS) to quantify the peptide abundances, which should ideally be equivalent to the abundances of their corresponding proteins. More detailed information on these processes can be found in [2–4] for those interested.



(a) Bottom-Up proteomics workflow.



(b) Trypsin cleavage sites illustration.

Figure 1.2: Illustration of MS-based proteomics principles. (a) Bottom-up proteomics workflow. (b) Illustration of trypsin cleavage sites in a 30-amino acid sequence. The enzyme trypsin cleaves the peptide bond at every occurrence of lysine (K) or arginine (R).

Three types of peptides are considered in MS-based proteomics (see Figure 1.3):

- **Non-Proteotypic Peptides**: These are peptides that are not unique to a specific protein, they may be shared by several proteins. Consequently, their presence does not unambiguously confirm the expression of any single protein.
- **Proteotypic Peptides**: These are peptides present in only one protein (unique peptide) and which can be detected by MS. They confirm the presence of the corresponding protein in a biological sample.
- **Quantotypic Peptides**: A subset of proteotypic peptides whose abundance is stoichiometrically equivalent to that of their protein. In other words, measuring a protein or their quantotypic peptides is equivalent.

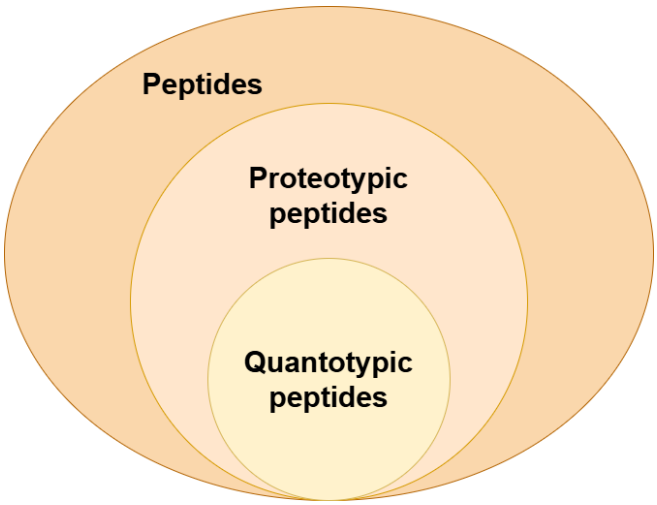
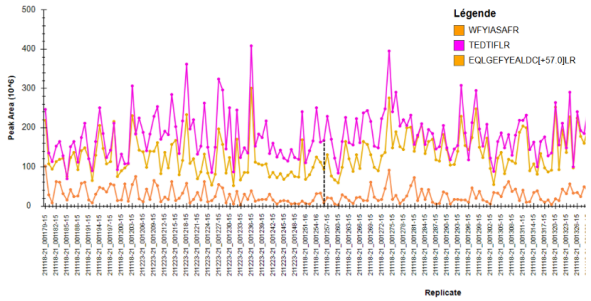


Figure 1.3: Diagram showing the hierarchy of peptides: quantotypic peptides form a subset of proteotypic peptides, which in turn are a subset of all peptides.

Targeted proteomics is an analytical strategy that focuses on the precise measurement of selected proteins of interest within biological samples. This approach monitors a set of predefined peptides. In targeted proteomics, selecting appropriate peptides (quantotypic peptides) for reliable protein quantification is crucial, as typically only a few peptides per protein are analyzed. However, the selection of quantotypic peptides is an overlooked topic in MS-based proteomics. To date, there is no reliable method to guide the a priori selection of quantotypic peptides. A key challenge is that not all peptides accurately reflect the parent protein's abundance. Traditional selection methods employ guiding rules based on peptide characteristics, such as amino acid composition. The different rules are split into three categories:

- **Ensure complete proteolysis:** For example, avoid peptide sequence with N-terminal (the beginning of the peptide sequence) proline cleavage (KP, RP).
- **Avoid peptides prone to chemical modifications:** For example, avoid peptides containing amino acids such as tryptophan (W) and methionine (M), which are susceptible to oxidation.
- **Avoid peptides prone to biological modifications:** For example, avoid peptides from the N-terminal (the beginning of the peptide sequence) and C-terminal (the end of the peptide sequence) regions, as these are more prone to degradation.

However, Chiva *et al.* [5] demonstrated that following or not following these rules do not affect the reliability of protein quantification. In other words, the usefulness of these rules has been called into question. Worboys *et al.* [6] evaluated the quantotypic property of peptides based on the following idea: if they are quantotypic, two peptides from the same protein should present a perfect correlation across samples. Based on this idea, they demonstrated that performing correlation matrix of peptides derived from the same protein can be used to estimate the quantotypic property of peptides. Peptides exhibiting high correlations within this matrix can be considered quantotypic, an example of such behaviour is displayed in Figure 1.4. While this method has proven effective, it requires experiments to measure the abundances of multiple peptides for each protein. In contrast, our work aims to develop an a priori method that enables peptide selection before experimental quantification, thereby avoiding costly and time-consuming experiments.



(a) Peptide signal intensities across samples.

sp|P02763|A1AG1_HUMAN			
	WFYIASAFR	TEDTIFLR	EQLGEFYEALDCLR
WFYIASAFR	1	0.409547	0.451792
TEDTIFLR	0.409547	1	0.842064
EQLGEFYEALDCLR	0.451792	0.842064	1

(b) Correlation matrix of peptide signals.

Figure 1.4: Comparison of peptide signals and their Spearman correlation for the A1AG1 protein, from the SPARE study. (a) Peptide signal intensities across samples. (b) Correlation matrix of peptide signals. Analysis of these correlations reveals that the peptide WFYIASAFR exhibits a low correlation with the other two peptides (0.4 and 0.45). In contrast, the other two peptides show a higher correlation (0.84 between them). Hence, the quantotypic property of the peptides TEDTIFLR and EQLGEFYEALDCLR are better than the peptide WFYIASAFR. In an ideal theoretical scenario, the correlation should be 1, but this is rarely observed in practice.

With the rapid advancements in artificial intelligence (AI), its applications have expanded across various fields, including proteomics, where machine learning and deep learning technologies have revolutionized multiple subfields. From protein structure prediction to mass spectrometry data analysis, AI-driven methods have significantly enhanced the accuracy and efficiency of proteomic research.

Several studies have explored computational and experimental strategies to improve peptide selection in mass spectrometry-based proteomics. These works primarily focus on identifying proteotypic peptides. Various approaches have been proposed such as leveraging machine learning models or empirical screening techniques [7–10]. CONSeQuence [11] is a machine learning tool designed to predict whether peptides are proteotypic. It first establishes a baseline by testing four distinct machine learning models to assess peptide detectability. Then, it employs a consensus method, referred to as CONS, that combines the predictions from these models using a simple voting system. In addition, it uses the physicochemical properties of peptides to make its predictions. This approach has been shown to outperform existing tools like ESP-Predictor, making CONSeQuence a more reliable option for identifying peptides that uniquely represent proteins in mass spectrometry-based proteomics. However, by focalising on the selection of proteotypic peptides, these approaches do not adress the question of quantotypic peptides. In this context, there is no guarantees that a proteotypic peptide will provide a reliable protein quantification. To address this limitation, there is a need to develop computational methods for predicting quantotypic peptides.

The integration of artificial intelligence into peptide selection for protein quantification could transform the field by enabling an a priori selection of peptides, reducing reliance on expensive and labor-intensive experimental techniques.

Objectives

This work aims to explore the use of artificial intelligence to **predict the quantotypic property of peptides solely based on their amino acid sequence**.

1.2 Dataset

In this thesis, two datasets were used to train models.

1.2.1 SPARE dataset

The first dataset used in this thesis originates from a study (SPARE [12]) conducted by the laboratory of translational gastroenterology at the University of Liege. This dataset comprises 230 peptides measured in the blood samples from 200 patients with Crohn's disease. Peptide signals and their correlations can be analyzed to evaluate the quantotypic property of each peptide, hence providing labels for each peptide.

1.2.2 Unique peptides of the human proteome

The second dataset consists of a list of unique peptides covering the entire human proteome, this gives us around 330 000 new peptides that can be exploited. This list was extracted using Skyline [13] by applying appropriate filters and selection criteria to retain peptides specific to the human proteome, ensure peptide uniqueness, and select proteins digested with trypsin as the enzyme. This dataset will be particularly useful for a self-training strategy, where pseudo-labels will be generated based on predictions from models trained on the first dataset. This approach aims to expand the training data and improve model generalization for peptide quantotypic property prediction.

1.3 Methodology overview

As previously mentioned, the objective of this thesis is to explore the use of artificial intelligence to predict the quantotypic property of peptides solely based on their sequence. In machine learning terminology, this amount to a binary classification task that aims to classify a peptide as either quantotypic or non-quantotypic, using its amino acid sequence as input. The methodology of this work can be structured into the following key steps:

- **Peptide labeling:**
  In order to perform supervised learning, we must use the first dataset (SPARE) to label peptides. To achieve this, we analyze the correlations among peptides belonging to the same protein by examining their peptide signals.
- **Supervised learning:**
  This step involves exploring various machine learning models, including Random Forest and XGBoost, alongside deep learning architectures such as Multi-Layer Perceptron (MLP) and Long Short-Term Memory (LSTM) networks, which will serve as baseline models for comparison. Additionally, transfer learning will be investigated, employing techniques such as fine-tuning and feature extraction.
- **Self-training:**
  The number of labeled peptides at our disposal being unfortunately low, this limits the performance of fully supervised learning techniques. We will hence investigate self-training approaches to leverage the (higher quantity of) unlabeled peptides.

1.4 Outline

This section outlines the structure of the remainder of this thesis, providing a description of each chapter.

- **Chapter 2: Machine learning**
  The Machine learning chapter presents an overview of the key artificial intelligence concepts used in this thesis. It explores fundamental principles such as Machine Learning, Supervised Learning, Transfer Learning, Self-Training, and various architectural frameworks relevant to this thesis.

- **Chapter 3: Dataset**
  The Dataset chapter provides an analysis of the SPARE dataset, details the methodology for labeling peptides, and introduces class imbalance along with the techniques used to handle it.
- **Chapter 4: Supervised learning**
  The Supervised learning chapter provides a detailed explanation of the methodology related to the first task of this thesis. The results are also presented and discussed after introducing the methodolody.
- **Chapter 5: Self-training**
  The Self-training chapter provides a detailed explanation of the methodology related to the second task of this thesis. After explain the methodology, the results are presented and discussed.
- **Chapter 6: Discussions**
  The Discussions chapter outlines several points that could explain the poor quality of the results obtained. Potential directions for future work are also mentioned.
- **Chapter 7: Conclusion**
  The Conclusion chapter provides a summary of this thesis.

The data and the Python implementation of the methods discussed in this thesis are available at https://github.com/Ziboce/TFE-proteomics.

Chapter 2

Machine Learning

Machine learning is a field of artificial intelligence that focuses on learning patterns from observed data to make predictions or decisions without being explicitly programmed. In machine learning, a model is a mathematical representation of such patterns in data. The process of optimizing the parameters of the model so that its predictions match the observed data is called training.

A machine learning model follows a learning algorithm, which defines how it optimizes its internal parameters based on the data it receives. Depending on the nature of the problem and the available data, different learning approaches can be used.

The learning process involves multiple stages, including feature extraction, where relevant attributes are selected from raw data, and optimization, where the model fine-tunes its parameters to minimize errors. A trained model is then evaluated on a separate test set to assess its generalization performance.

In machine learning, learning methods can be categorized into different families based on how models process and adapt to data. This chapter describe the families that are relevant for the thesis. It also introduces well-known machine learning algorithms based on decision trees and popular deep learning architectures [14].

2.1 Families of learning methods

2.1.1 Supervised learning

Supervised learning is a machine learning approach where a model is trained using labeled data, meaning that each input sample x_i is associated with a corresponding target output y_i. The objective is to learn a function f^* that minimizes a predefined loss function $\mathcal{L}$, which quantifies the difference between the predicted output $f(x_i)$ and the true label y_i.

Formally, given a training dataset $D = \{(x_i, y_i)\}_{i=1}^{n}$, where $x_i \in \mathbb{R}^d$ represents an input feature vector and y_i is the corresponding target, the goal is to learn a function $f : \mathbb{R}^d \to \mathbb{R}$ for regression or $f : \mathbb{R}^d \to \{0, 1, \ldots, K\}$ for classification, where K is the number of classes.

The function f belongs to a hypothesis space $\mathcal{H}$, which defines the set of all models the learning algorithm can explore. Learning consists of selecting the best $f^* \in \mathcal{H}$ that minimizes the loss function $\mathcal{L}$ over the training set.

$$f^* = \arg\min_{f \in \mathcal{H}} \sum_{i=1}^{n} \mathcal{L}(f(x_i), y_i)$$

Once trained, the model undergoes evaluation on unseen data to assess its ability to generalize. The capacity to generalize beyond the training data depends on the complexity of $\mathcal{H}$, where overly simple models may underfit, while excessively complex ones may overfit. Regularization techniques, validation strategies, and careful model selection are used to ensure that f^* effectively captures the underlying data distribution while maintaining good generalization to new inputs.

In supervised learning, the labeled data are typically obtained through human annotation, where experts manually assign correct labels to each input sample. This process ensures high-quality training data but can be time-consuming and costly, especially in domains requiring specialized knowledge [14].

In this thesis, the focus is on binary classification models, where the objective is to assign a discrete label to each input sample based on learned patterns (quantotypic vs non-quantotypic).

2.1.2 Unsupervised learning

Unsupervised learning is a machine learning approach where models are trained on unlabeled data, meaning that the input samples $x \in \mathbb{R}^d$ do not have associated target values. Instead of learning an explicit mapping from inputs to outputs as in supervised learning, the goal is to identify hidden structures, patterns, or relationships within the data.

Formally, given a dataset $\mathcal{D} = \{x_i\}_{i=1}^n$, where each x_i is an input feature vector, the objective is to learn a function $f : \mathbb{R}^d \to \mathbb{R}^m$ that transforms the data into a meaningful representation, where m can be equal to or smaller than d depending on the task.

Unsupervised learning methods are often associated with clustering and dimensionality reduction [14].

2.1.3 Self-training

Self-training serves as an intermediary approach between supervised learning and unsupervised learning, enabling the utilization of both labeled and unlabeled data to improve model performance [15]. The process begins with training a model on a labeled dataset and subsequently using this model to generate pseudo-labels for an unlabeled dataset. This technique is particularly beneficial in scenarios where labeled data is scarce, but a large amount of unlabeled data is available. By leveraging self-training, it becomes possible to label data without requiring human intervention, thereby expanding the training dataset and enhancing model generalization. The self-training process can be broken down into several key steps:

- **Step 1: Training an Initial Model Using Labeled Data**
  The process begins by splitting the available labeled dataset into a training set and a test set. A preliminary model is then trained using the training set, leveraging the limited labeled data to establish an initial predictive function.
- **Step 2: Predicting on Unlabeled Data**
  Once the initial model is trained, it is used to generate predictions on the unlabeled dataset. These predictions are referred to as pseudo-labels. Different strategies can be employed to determine which pseudo-labels to retain: for example, one approach considers all predicted labels as valid, while another filters pseudo-labels based on a confidence threshold, ensuring that only highly confident predictions contribute to subsequent training steps.

- **Step 3: Merging Labeled Data and Pseudo-Labeled Data for Re-Training**
  After filtering and selecting the most reliable pseudo-labels, they are merged with the original labeled dataset to form an expanded training dataset. The model is then retrained on this augmented dataset, allowing it to leverage additional training examples and potentially improve its performance.

- **Step 4: Evaluating the Model on Labeled Data**
  The updated model is tested on a held-out labeled dataset to assess its performance using appropriate evaluation metrics. This step ensures that the self-training process contributes to improving model generalization rather than reinforcing incorrect predictions.

- **Step 5: Iterative Refinement**
  The self-training cycle is repeated iteratively, refining the model with each iteration. The process continues until no more pseudo-labels meet the confidence threshold or until there are no remaining unlabeled samples to process.

All the steps described above can be visualized in the figure 2.1.

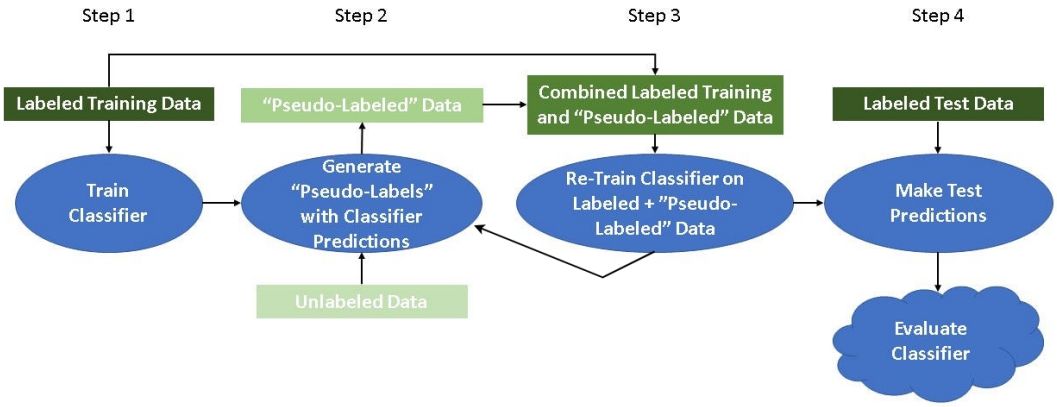


Figure 2.1: Illustration of the self-training process. The procces is composed of four iterative steps: (1) Training a classifier on labeled training data, (2) Generate pseudo-labels using predictions from the trained model, (3) Concatenate the pseudo-labeled data with the labeled dataset and retraining the classifier, and (4) Evaluation of the retrained model on labeled test data. The process is repeated until no more confident pseudo-label is generated or all unlabeled data has been used. Figure adapted from [16].

2.1.4 Transfer learning

Transfer learning is a machine learning technique where a model trained on one task is reused to improve performance on another related task. This approach is particularly useful when the second task has limited labeled data, as it allows leveraging knowledge learned from a pretrained model. The core idea of transfer learning is to repurpose learned representations rather than training a model from scratch [17]. There are two main transfer learning techniques: Feature Extraction and Fine-Tuning. In both approaches, the pretrained model serves as an encoder, providing learned representations that are used as input for a new classifier [18].

- **Feature extraction:**
  In feature extraction, the pretrained model is used as a fixed feature extractor. The lower layers of deep neural networks typically capture general patterns while the final layers are more task-specific. Instead of training the entire network, the pretrained model is kept frozen, and only a new classifier (e.g., fully connected layers) is trained on top of it using the new dataset. This approach is effective when the new dataset is small but sufficiently similar to the original dataset used for pretraining (see Figure 2.2).

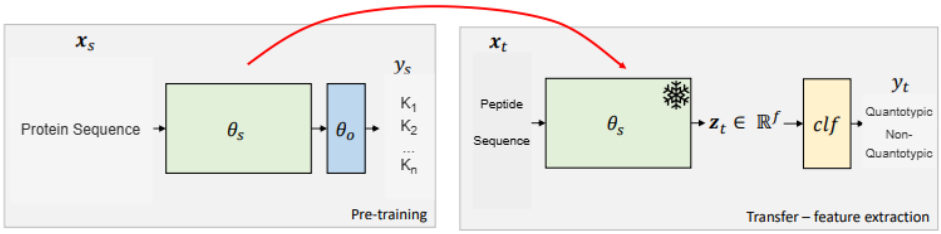


Figure 2.2: Illustration of supervised transfer learning using feature extraction. A first model is trained using labeled data on a specific task. θ_s refers to the encoder, while θ_o refers to the decoder on the original task. Only θ_s is reused as an encoder for feature extraction, and its parameters are frozen during training on the new task. The *clf* classifier is the new decoder for the target task, built on top of the frozen encoder θ_s. Figure adapted from *Romain Mormont's PhD thesis [18]*.

- **Fine tuning:**
  In fine-tuning, the pretrained model is also reused, but unlike feature extraction, some or all of its layers are updated during training on the new task. Typically, the final layers are unfrozen and fine-tuned to adapt to the specific characteristics of the new dataset, while earlier layers could remain frozen to keep their general feature extraction capabilities. Fine-tuning is beneficial when the new task is sufficiently different from the original task but still shares some underlying structure (see Figure 2.3).

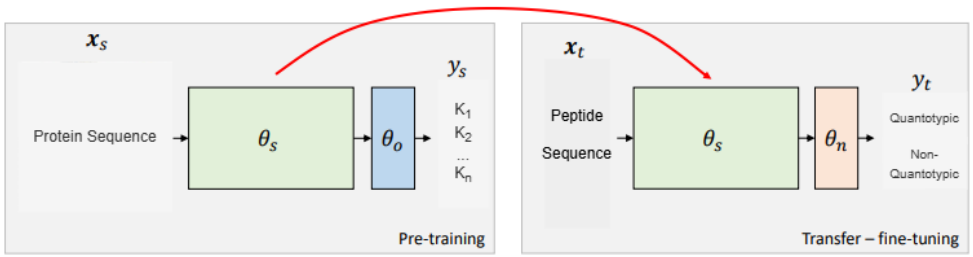


Figure 2.3: Illustration of supervised transfer learning using fine-tuning. A first model is trained using labeled data on a specific task. θ_s refers to the encoder, while θ_o refers to the decoder on the original task. Only θ_s is reused as an encoder for fine-tuning, and its parameters are updated during training on the new task. The θ_n classifier is the new decoder for the target task, built on top of the encoder θ_s. Figure adapted from *Romain Mormont's PhD thesis [18]*.

2.2 Classical machine learning models

2.2.1 Decision trees

A decision tree is a widely used machine learning model for classification tasks, structured as a tree where each internal node represents a decision based on a specific feature, branches correspond to possible

outcomes, and leaf nodes indicate the final predicted class. The core principle of a decision tree is to recursively split the dataset into increasingly homogeneous subsets, ensuring that samples within each subset belong to the same class as much as possible. The algorithm determines the best feature to split on using a splitting criterion, commonly Gini impurity or entropy (information gain), both of which measure how mixed the classes are in a given subset. The goal is to minimize impurity at each step, making the groups purer and improving classification accuracy [14].

The tree-building process starts at the root node, where the dataset is evaluated to find the most discriminative feature for the first split. The process continues recursively, with each subset being further divided based on the optimal feature at each step, until a stopping condition is met, such as reaching a maximum depth or having too few samples to justify further splitting. The final leaf nodes assign a class label based on the majority class of the samples they contain. An example of a decision tree is displayed on Figure 2.4.

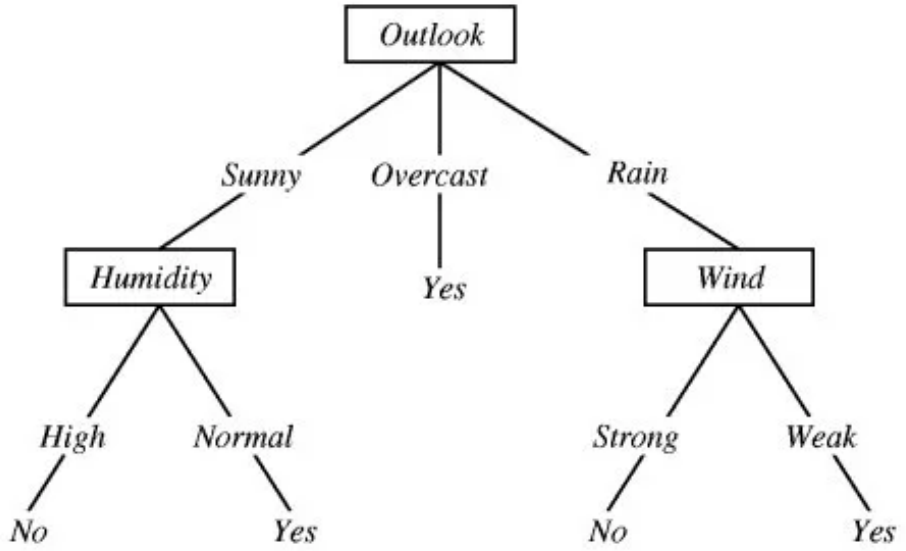


Figure 2.4: Famous example of a decision tree that determines whether one should play tennis based on various features, where the final decision is either "Yes" or "No". Figure reproduced from [19].

2.2.2 Ensemble method

While a single decision tree is easy to interpret and can capture complex decision boundaries, it often lacks robustness and generalization, making it prone to overfitting on training data. A powerful approach to mitigate these limitations is the use of ensemble methods, which combine multiple models to improve predictive performance, stability, and resilience to noise. The core idea behind ensemble learning is that by aggregating the outputs of multiple weak models, the overall prediction becomes more accurate and reliable than any individual model.

Ensemble methods typically fall into two main categories: bagging and boosting. Bagging (Bootstrap Aggregating) aims to reduce variance by training multiple models independently on different random subsets of the data and averaging their predictions. Random Forest, one of the most widely used bagging-based algorithms, constructs multiple decision trees using randomly selected features and samples, leading to a more stable and accurate model. In contrast, Boosting focuses on reducing bias by training models sequentially, where each new model learns to correct the mistakes of its predecessors. XGBoost enhances the decision-making process through an iterative boosting framework, progressively improving model accuracy by correcting errors from previous iterations [14].

2.2.2.1 Random forest

Random Forest is an extension of the bagging (Bootstrap Aggregating) method, incorporating both data and feature randomness to construct a collection of decision trees. Each decision tree is trained on a random subset of the training data, sampled with replacement (bootstrapping), ensuring diversity among the trees. Additionally, at each split within a tree, only a random subset of features is considered, which reduces correlation between individual trees and enhances model robustness. Once all trees have made their predictions, the final output is obtained by aggregating their results (using majority voting for classification) [14]. The algorithm is illustrated in Figure 2.5.

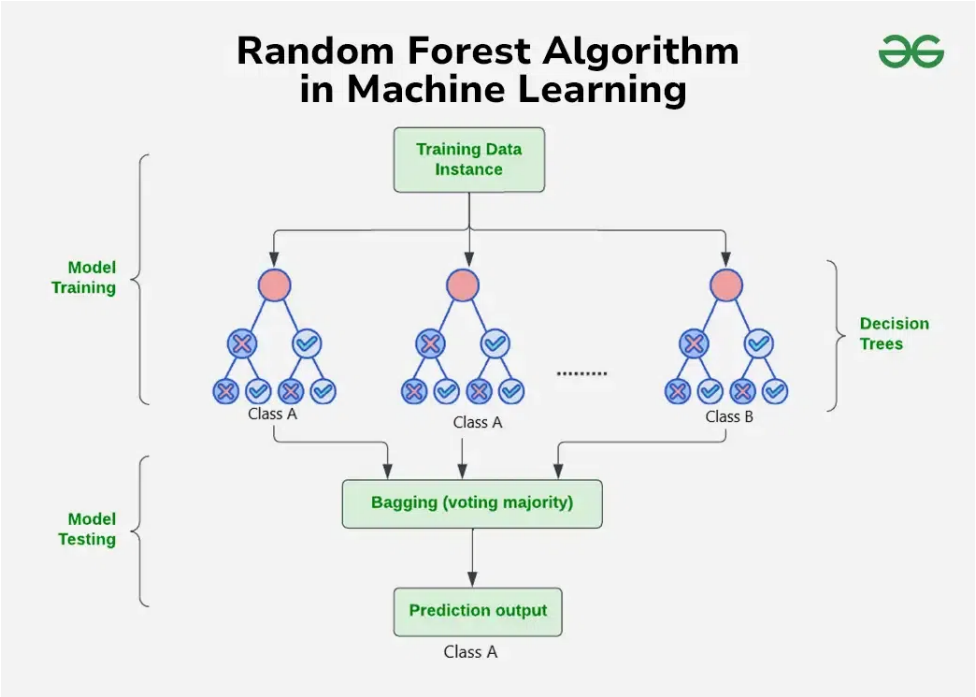


Figure 2.5: Random Forest Algorithm: The original dataset is divided into multiple subsets using bootstrapping, with each subset used to train an individual decision tree. The final prediction is obtained by aggregating the outputs of all trees. Figure reproduced from [20].

2.2.2.2 XGBoost

XGBoost (eXtreme Gradient Boosting) is an extension of the boosting method, designed to enhance both accuracy and computational efficiency in machine learning tasks. As an ensemble learning technique, XGBoost builds multiple weak learners (decision trees) and combines them to form a stronger predictive model. Unlike Random Forest, which grows trees independently in parallel (bagging), XGBoost constructs trees sequentially, where each new tree corrects the errors of its predecessors through gradient boosting. The core principle of gradient boosting is to iteratively train an ensemble of weak decision trees, where each tree is trained on the residual errors of the previous trees. This process enables XGBoost to minimize bias and improve model accuracy, as each tree is specifically designed to reduce the error of the previous one. At each iteration, the model adjusts predictions by optimizing an objective function, which consists of a loss function (to measure error) and a regularization term (to prevent overfitting) [14, 21]. The algorithm is illustrated in Figure 2.6.

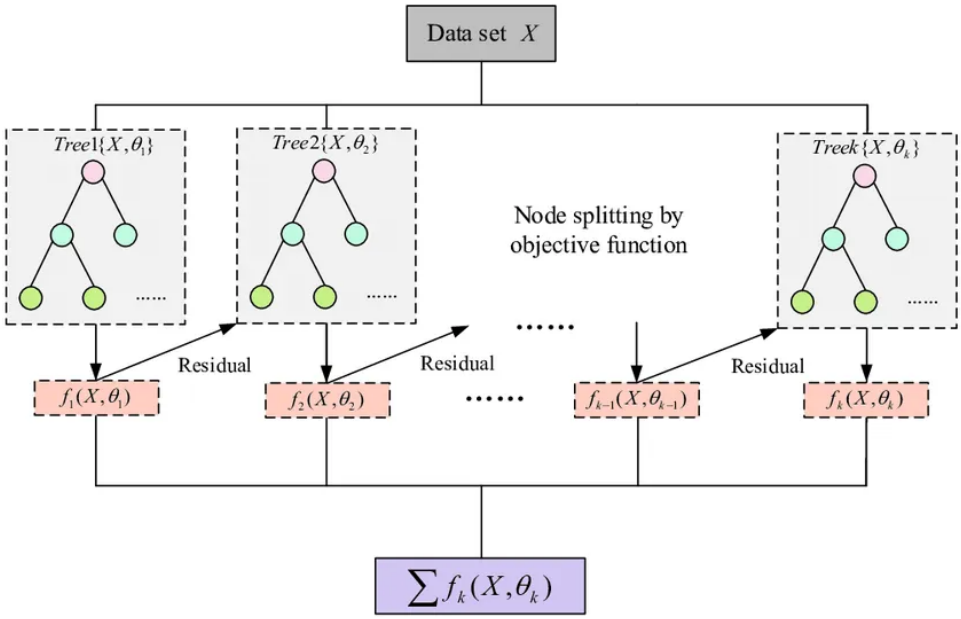


Figure 2.6: XGBoost Algorithm: the algorithm begins with an initial prediction, typically the average of the target variable. It then computes the residuals, which represent the difference between predicted and actual values. The first decision tree is trained to learn these residuals, finding the best splits that minimize the overall error. Subsequent trees are added iteratively, each focusing on correcting the remaining residual errors from the previous trees. The ensemble continuously updates predictions by aggregating the outputs of all trained trees. XGBoost optimizes this process using a loss function, which is minimized through gradient descent to enhance predictive accuracy. The process continues until a stopping criterion is met, such as reaching a maximum number of trees or achieving minimal improvement in the loss function. Figure reproduced from [22].

2.3 Deep learning architectures

2.3.1 Multi-Layer Perceptron

A Multi-Layer Perceptron (MLP) is a fundamental type of artificial neural network that learns patterns by passing data through multiple layers of interconnected neurons. It is composed of an input layer, one or more hidden layers, and an output layer. Each neuron in a given layer is fully connected to neurons in the next layer, forming a dense feedforward network [23, 24].

Each connection between neurons is associated with a weight that determines the importance of the input value, and each neuron has an additional bias term that helps shift activations. Given an input vector $X = (x_1, x_2, ..., x_n)$, the weighted sum of inputs for a neuron i is computed as:

$$a_i = f\Big(\sum_j w_{ij} x_j + b_i\Big)$$

where:

- w_{ij} represents the weight of the connection between neuron i and neuron j,
- b_i is the bias associated with neuron i,
- $f(\cdot)$ is an activation function. Activation functions introduce non-linearity, allowing the MLP to learn complex patterns.

The number of neurons in the input layer corresponds to the number of features in the dataset, while the output layer size depends on the task. For binary classification, a single neuron with a sigmoid activation function is often used, where the probability of class 1 is given by the output, and the probability of class 0 is $1 - output$. Such architecture is illustrated in Figure 2.7.

However, a key limitation of MLP is that neurons operate independently of each other within a given layer. This means that each neuron's output is computed solely based on its own weighted sum of inputs, without considering contextual information from other elements in a sequence.

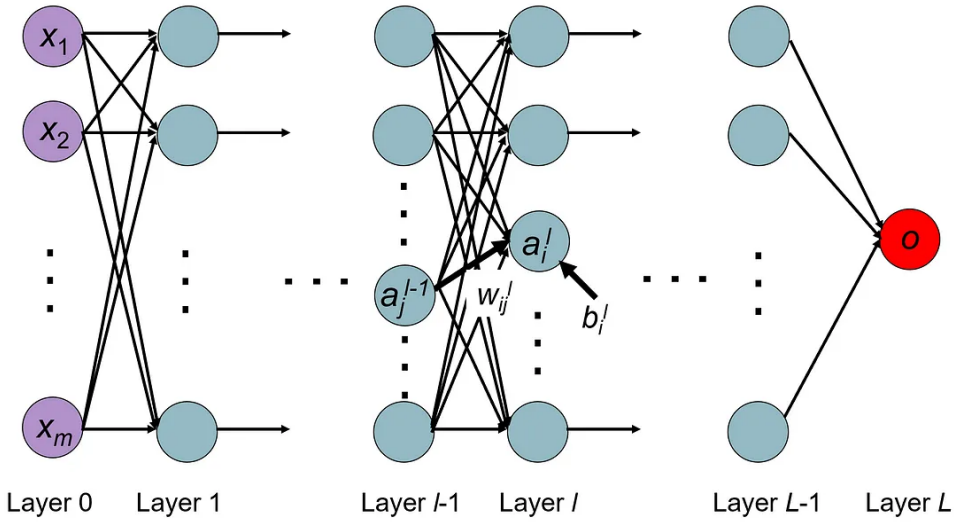


Figure 2.7: MLP Architecture: The input layer (Layer 0) consists of nodes (purple) representing features from the dataset. The output layer (Layer L) contains the final prediction node (red). The layers between the input and output layers are hidden layers, responsible for learning representations through weighted connections. Each node in a layer is fully connected to the nodes in the next layer. The activation of node i at layer l is denoted as a_i^l, while b_i^l represents the bias term of node i at layer l. The weight of the connection between node j in layer l-1 and node i in layer l is denoted as w_{ij}^l. Figure reproduced from [25].

2.3.2 Recurrent Neural Networks

A Recurrent Neural Network (RNN) is a neural network architecture designed to process sequences by incorporating memory. Unlike standard neural networks that process each input independently (like MLP), an RNN maintains an internal hidden state that retains information from previous time steps [24].

The two equations representing an RNN is:

$$h_t = f(W_{hh}h_{t-1} + W_{xh}x_t + b_h)$$
$$y_t = g(W_{hy}h_t + b_y)$$

where:

- h_t represents the hidden state at time step t,
- x_t is the input at time t,
- y_t is the output at time t,
- W_{hh}, W_{xh}, and W_{hy} are weight matrices learned during training,
- b_h, b_y are bias terms,

- f is the activation function applied to the hidden state,
- g is the activation function applied to the output.

The hidden state h_t acts as a short-term memory, storing information about previous inputs and allowing the model to make predictions based on contextual information.

At each time step t, the RNN receives an input x_t and updates its hidden state h_t. The updated hidden state is used to generate an output y_t, and it is also passed to the next time step to maintain memory of past inputs. This process continues for the entire sequence. An illustration of the architecture is depicted in Figure 2.8.

While RNN are effective at capturing local dependencies in short sequences, they suffer from several significant limitations. Training an RNN relies on Backpropagation Through Time (BPTT) to update the weights. However, when dealing with long sequences, two major issues arise:

- The **vanishing gradient problem**, where gradients shrink exponentially as they are propagated backward, making it difficult for the model to learn dependencies from earlier time steps.
- The **exploding gradient problem**, where gradients grow uncontrollably, leading to instability in training and excessively large weight updates.

As a result, standard RNN struggle to retain long-term dependencies effectively. Their recurrent structure inherently limits their memory, causing important information from earlier steps in a sequence to be "forgotten" as new inputs are processed.

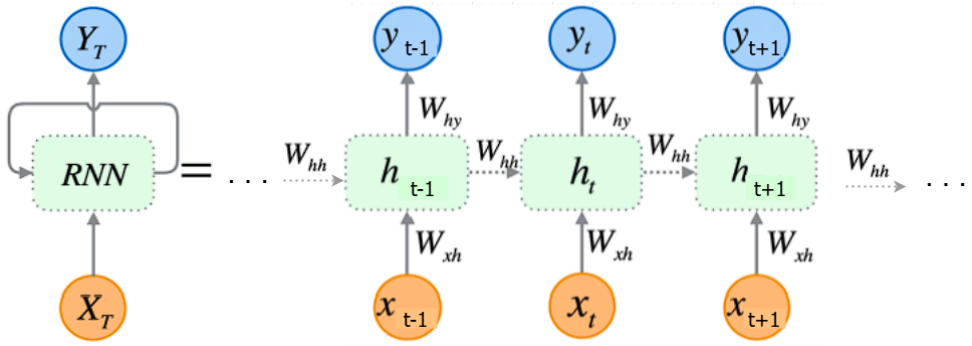


Figure 2.8: RNN architecture illustration. On the left, the compact representation of an RNN is shown, where the recurrent connection loops over time. On the right, the unfolded version demonstrates how the hidden state h_t is propagated across multiple time steps, influenced by both the current input x_t and the previous hidden state h_{t-1}. Each hidden state is connected to an output y_t through learned weights W_{hy}. Figure adapted from Gawde *et al.* [26].

2.3.3 Long Short-Term Memory

To overcome the limitations of standard RNN, an improved architecture called Long Short-Term Memory (LSTM) was developed. LSTM are specifically designed to mitigate the vanishing gradient problem and improve the model's ability to capture long-term dependencies by introducing a more structured memory mechanism [23, 24].

Unlike standard RNN, which rely solely on a hidden state h_t to pass information between time steps, LSTM introduce a dedicated cell state c_t that serves as a memory unit capable of carrying information over long distances in a sequence. The flow of information in an LSTM is controlled by three key gates:

- **Forget gate**: Determines how much information from the previous cell state should be retained or discarded.
- **Input gate**: Regulates how much new information from the current input should be added to the cell state.
- **Output gate**: Controls how much of the updated cell state should be passed to the next hidden state.

The mathematical formulation of an LSTM cell is given by:

$$
\begin{aligned}
f_t &= \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \\
i_t &= \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \\
o_t &= \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \\
\tilde{c}_t &= \tanh(W_c \cdot [h_{t-1}, x_t] + b_c) \\
c_t &= f_t \odot c_{t-1} + i_t \odot \tilde{c}_t \\
h_t &= o_t \odot \tanh(c_t)
\end{aligned}
$$

where:

- f_t, i_t, and o_t are the forget, input, and output gate activations, respectively,
- $\tilde{c}_t$ is the candidate cell state update, representing a filtered version of the input data that has been processed and prepared for potential integration into the current cell state,
- c_t is the cell state at time step t,
- h_t is the hidden state at time step t,
- W_f, W_i, W_o, W_c are the weight matrices learned during training, each corresponding to the forget gate, input gate, output gate, and candidate cell state update, respectively,
- b_f, b_i, b_o, b_c represents bias terms,
- $\sigma(\cdot)$ denotes the sigmoid activation function, and
- $\odot$ represents element-wise multiplication.

An illustration of the LSTM architecture is shown in Figure 2.9.

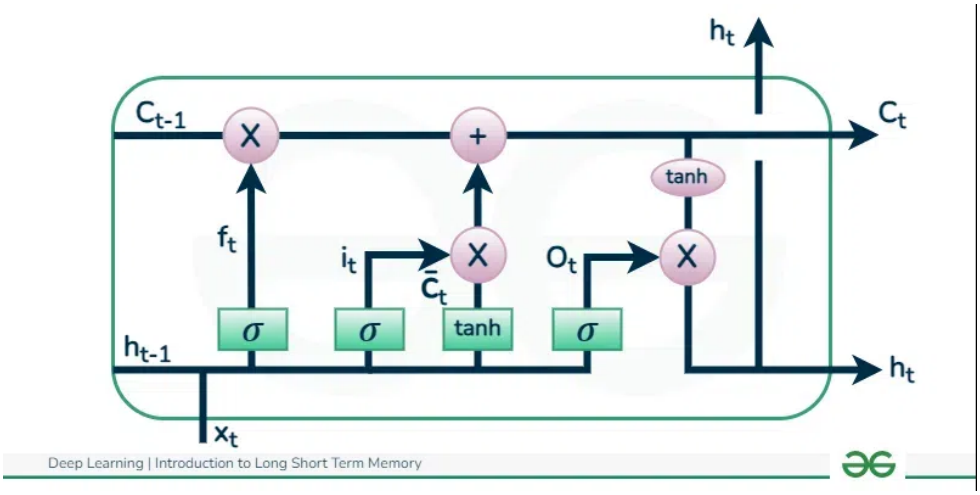


Figure 2.9: Illustration of a Long Short-Term Memory (LSTM) cell. The diagram shows the internal operations of an LSTM unit, including the three main gates: the forget gate (f_t), the input gate (i_t), and the output gate (o_t). The forget gate determines how much of the previous cell state C_{t-1} should be retained. The input gate regulates how much new information is added to the cell state, with a candidate update $\tilde{C}_t$ computed using a tanh activation function. The output gate controls how much of the updated cell state contributes to the hidden state h_t. The cell state C_t acts as a memory unit, enabling long-term information retention. The sigmoid (σ) and tanh functions ensure controlled gating and non-linearity. Figure reproduced from [27].

While LSTM significantly improve memory retention and mitigate vanishing gradients, they still process sequences in a unidirectional manner, meaning that only past information influences the current state. To address this limitation, an extension called Bidirectional LSTM (BiLSTM) was introduced [24].

A Bidirectional LSTM (BiLSTM) enhances the standard LSTM by processing sequences in both forward and backward directions. This is achieved by using two separate LSTM networks:

- A **forward LSTM**, which processes the sequence from $t = 1$ to $t = T$.
- A **backward LSTM**, which processes the sequence from $t = T$ to $t = 1$.

The outputs of these two networks are then concatenated to form the final representation:

$$h_t = \text{concat}(h_t^{\text{forward}}, h_t^{\text{backward}})$$

By incorporating future context alongside past information, BiLSTMs achieve superior performance in tasks where contextual dependencies span both directions.

2.4 Evaluation Metrics

This section presents the metrics used to evaluate the performance of the models in both the supervised learning setting and during the self-training process.

2.4.1 Accuracy

Accuracy is one of the most commonly used metrics for evaluating classification models. It measures the proportion of correctly classified instances over the total number of instances in the dataset. Mathematically, accuracy is defined as:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{2.1}$$

where:

- TP (True Positives): The number of correctly predicted positive samples.
- TN (True Negatives): The number of correctly predicted negative samples.
- FP (False Positives): The number of negative samples incorrectly classified as positive.
- FN (False Negatives): The number of positive samples incorrectly classified as negative.

While accuracy is a useful metric, it can be misleading in cases of class imbalance, where one class dominates the dataset.

2.4.2 Receiver Operating Characteristic - Area Under the Curve (ROC AUC)

The ROC AUC is a performance metric commonly used to evaluate classification models, particularly in binary classification tasks. It measures the ability of a model to distinguish between positive and negative classes by analyzing the trade-off between the true positive rate (sensitivity) and the false positive rate.

The ROC curve is a graphical representation that plots the true positive rate (recall) against the false positive rate at various classification thresholds. The ROC AUC (Area Under the Curve) computes the area under that curve and quantifies the overall performance of the model. An AUC value of 1.0 indicates a perfect classifier, while an AUC of 0.5 corresponds to a model unable to differentiate between classes, performing no better than random guessing. A higher AUC value reflects a model that is better at distinguishing between positive and negative cases across different thresholds.

ROC AUC is particularly useful when dealing with imbalanced datasets, as it evaluates the model's ability to rank positive instances higher than negative ones, regardless of the specific classification threshold.

2.4.3 Precision

Precision, also known as positive predictive value, evaluates the proportion of correctly predicted positive instances among all predicted positives. It measures how many of the samples classified as positive are actually positive. Mathematically, precision is defined as:

$$\text{Precision} = \frac{TP}{TP + FP} \tag{2.2}$$

where:

- TP (True Positives): The number of correctly predicted positive samples.
- FP (False Positives): The number of negative samples incorrectly classified as positive.

A high precision value indicates a low false positive rate, meaning that the model makes fewer incorrect positive predictions. However, precision alone does not provide a complete evaluation of a model's performance, as it does not consider false negatives. Therefore, it is often used alongside recall and F1-score for a more comprehensive assessment.

2.4.4 Recall

Recall, also known as sensitivity or true positive rate, measures the proportion of actual positive instances that are correctly identified by the model. It evaluates how well the model captures positive cases, making it particularly useful in scenarios where missing positive instances (false negatives) is costly. Mathematically, recall is defined as:

$$\text{Recall} = \frac{TP}{TP + FN} \tag{2.3}$$

where:

- TP (True Positives): The number of correctly predicted positive samples.
- FN (False Negatives): The number of positive samples incorrectly classified as negative.

A high recall indicates that the model correctly identifies most positive instances, but it does not take into account false positives.

2.4.5 F1-score

The F1-score is a metric that combines both precision and recall into a single value, providing a balanced measure of a model's performance. It is particularly useful when dealing with imbalanced datasets, where relying solely on precision or recall may be misleading. The F1-score is defined as the *harmonic mean* of precision and recall, ensuring that both metrics are given equal importance. Mathematically, it is expressed as:

$$F1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{2.4}$$

A high F1-score indicates that the model achieves a good balance between precision and recall, making it well-suited for applications where both false positives and false negatives carry significant consequences.

2.4.6 Precision-Recall - Area Under the Curve (PR AUC)

The PR AUC is a performance metric commonly used to evaluate classification models, particularly in cases where class imbalance is present. Unlike the ROC AUC, which considers both true positives and false positives across different thresholds, PR AUC focuses specifically on the trade-off between precision and recall.

The Precision-Recall (PR) curve is a graphical representation that plots precision against recall at various classification thresholds. It provides insights into the model's ability to correctly identify positive instances while minimizing false positives. PR AUC quantifies the overall area under this curve, with higher values indicating a model that maintains both high precision and high recall.

PR AUC is particularly useful when the positive class is rare, as it prioritizes the model's performance on the minority class rather than evaluating how well it distinguishes between both classes. In highly imbalanced datasets, PR AUC often provides a more realistic assessment of model performance than ROC AUC, as it better reflects the model's effectiveness in retrieving relevant positive instances [28].

Chapter 3

Dataset

This chapter presents the steps taken to prepare the dataset and address the class imbalance issue in the context of quantotypic peptide prediction. It first details the labeling process, including the filtering of peptides and the method used to assign quantotypic or non-quantotypic labels based on peptide signal correlations. The impact of the chosen correlation threshold on data distribution and class balance is also analyzed. The second part focuses on the class imbalance challenge. Since quantotypic peptides are overrepresented in the final dataset, various techniques are explored to mitigate this issue. These include oversampling, cost-sensitive learning, and a discussion of why standard oversampling methods are unsuitable for biological sequence data.

3.1 Dataset labelling and analysis

As a reminder, the first dataset (SPARE) consists of 230 peptides, each measured across 200 samples, all derived from human proteins. Our first task is to attribute a label (quantotypic or non-quantotypic) to each peptide. This labelling will be achieved by analyzing the correlations between the abundances of the peptides of a same protein. The first step in processing this dataset is to remove non-unique peptides, as a peptide must be unique (i.e., to correspond to only one protein) to be considered quantotypic. After filtering out non-unique peptides, 213 peptides remain in the dataset. We also must ensure that each protein is still represented by at least two peptides. If a protein is represented by only one peptide, it is removed from the dataset, along with its corresponding peptide. After applying this filter, the dataset is reduced to 206 peptides. The second step is to analyze the spearman correlation between peptide signals belonging to the same protein. Peptides from the same protein that exhibit a high correlation in their signal intensities are more likely to be quantotypic. An example of a peptide signal is illustrated in Figure 3.1, while the corresponding correlation matrix is shown in Figure 3.2.

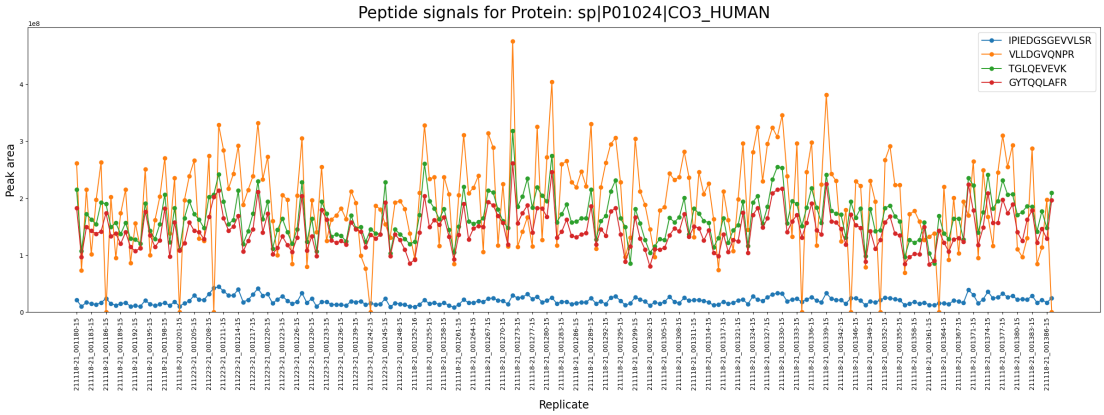


Figure 3.1: Peptide signals for the protein P01024.

Correlation matrix for Protein: sp|P01024|CO3_HUMAN

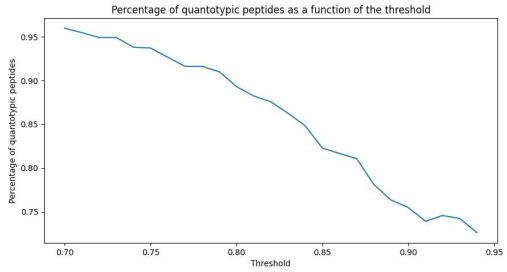
	IPIEDGSGEVVLSR	VLLDGVQNPR	TGLQEVEVK	GYTQQLAFR
IPIEDGSGEVVLSR	1.0	0.54007	0.79139	0.76862
VLLDGVQNPR	0.54007	1.0	0.60961	0.60896
TGLQEVEVK	0.79139	0.60961	1.0	0.94234
GYTQQLAFR	0.76862	0.60896	0.94234	1.0

Figure 3.2: Correlation matrix derived from the peptide signals for the protein P01024.

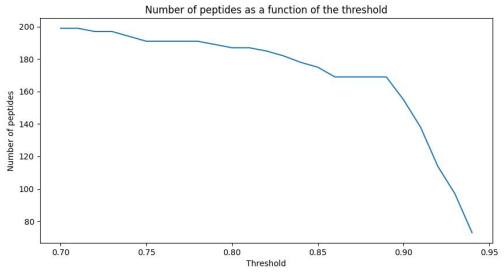
We now need to determine a threshold value on the correlation to classify peptides into two distinct categories: quantotypic and non-quantotypic. There is no predefined rule for selecting this threshold, and it is important to note that this choice will directly impact our dataset.

If we use a correlation matrix and label peptides based on a chosen threshold, it is possible that some correlation matrices may contain peptides exclusively from one category. For instance, if all peptides within a matrix exhibit high correlation values that exceed the threshold, they can be classified as quantotypic peptides. However, in cases where all peptides display low correlation, we cannot automatically conclude that they are all non-quantotypic. Consider a simple case with a correlation matrix of two peptides. If the correlation between them exceeds the threshold, they can both be categorized as quantotypic. However, if their correlation is below the threshold, it remains unclear which of the two peptides is responsible for the low correlation, making classification ambiguous. Both peptides must hence be discarded from the dataset. This issue can be extended with matrix correlation of bigger size.

To assist in selecting an appropriate threshold, I have decided to experiment with multiple threshold values and evaluate their impact on the quantotypic/non-quantotypic distribution (Illustrated in Figure 3.3a), as well as the overall number of peptides retained in the dataset (Illustrated in Figure 3.3b).



(a) Percentage of quantotypic peptides as a function of the threshold.



(b) Number of peptides as a function of the threshold.

Figure 3.3: Quantotypic peptide percentage and peptide count as a function of the threshold.

By analyzing these two graphs, we observe a logical trend: as the threshold increases, the number of defective correlation matrices (matrices in which all values fall below the threshold) also increases, leading to the removal of more peptides. Furthermore, we observe a class imbalance, where quantotypic peptides are overrepresented (percentage $> 50\%$). This imbalance can be explained by the fact that at the lowest threshold (0.7), nearly all peptides are labeled as quantotypic. When the threshold is raised, more peptides are classified as non-quantotypic, as shown by the decreasing trend. Even with the highest threshold (0.94), about 75% of the peptides are still labeled as quantotypic.

Additionally, around 0.9 in Figure 3.3b, we observe a sharp decline in the number of retained peptides, indicating a steep drop-off. Regarding the percentage of quantotypic peptides, the objective is to minimize

class imbalance as much as possible to prevent overfitting and simplify further processing.

Based on these observations, a threshold of 0.9 is selected. With this threshold, 155 peptides remain, of which 117 are classified as quantotypic and 38 as non-quantotypic. The final dataset distribution is as follows (Illustrated in Figure 3.4):

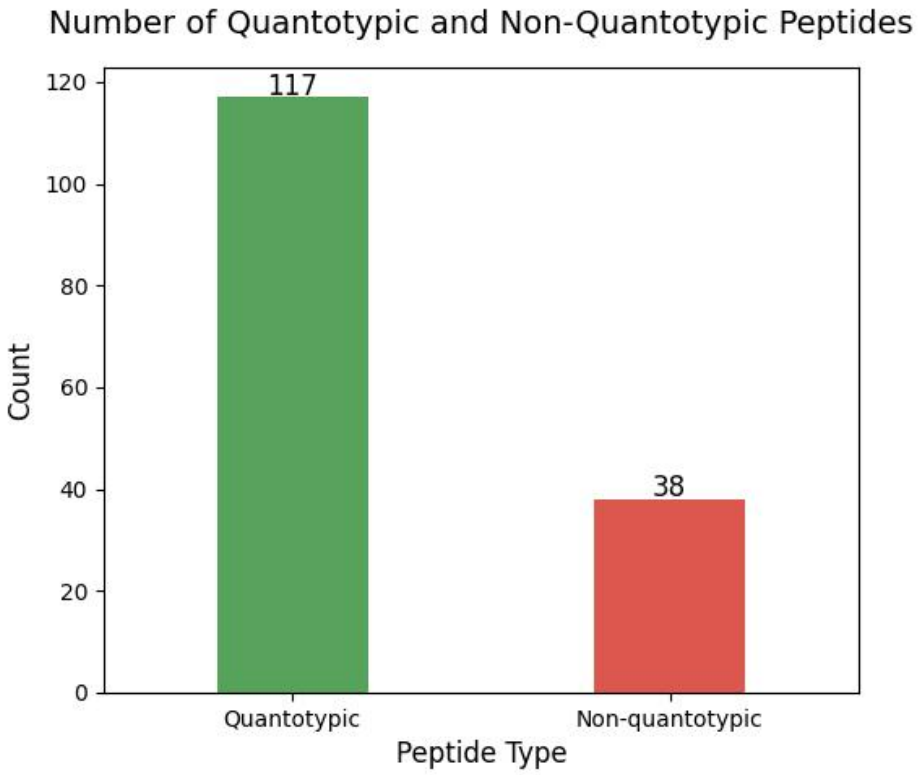


Figure 3.4: Distribution of quantotypic and non-quantotypic peptides in the final dataset.

3.2 Class imbalance

Class imbalance is a well-known challenge in artificial intelligence, occurring when one class (in this case, quantotypic peptides) is significantly overrepresented compared to the other (non-quantotypic peptides). Several techniques exist to address class imbalance:

- **Oversampling:** A technique that increases the representation of the minority class by generating synthetic samples or duplicating existing ones.
- **Undersampling:** A method that balances classes by reducing the number of samples in the majority class.
- **Cost-sensitive learning:** An approach that adjusts model weights to compensate for class imbalance by assigning higher costs to misclassification errors in the minority class.

In our case, the undersampling method will not be used. Applying this technique would result in a dataset with only 38 peptides per class, which is far too limited for deep learning. Given that we already have a small dataset, further reducing the number of samples would be counterproductive. For this reason, undersampling is not considered a viable approach.

3.2.1 Oversampling

Due to the proteomics context of this thesis, we cannot apply advanced oversampling techniques such as SMOTE (Synthetic Minority Over-sampling Technique), ADASYN (Adaptive Synthetic Sampling), or similar methods that generate synthetic data based on the existing dataset variables. While these techniques are commonly used in machine learning to balance class distributions, their application to peptide data poses significant challenges.

The fundamental issue with using SMOTE, ADASYN, or similar algorithms in proteomics is that they generate new synthetic samples by interpolating between existing data points. This approach works well in numerical datasets where features can be mathematically mixed without violating domain constraints (for further details, please refer to [29, 30]). However, in the context of peptide sequences, this could result in artificial combinations of amino acids that do not correspond to real biological peptides. This could mislead the model, negatively impacting its generalization and performance in real-world applications.

For these reasons, synthetic peptide generation via oversampling is not a viable option. Instead, the only feasible approach in our case is RandomOverSampler. Unlike interpolation-based methods, RandomOverSampler does not create artificial sequences; instead, it randomly selects and duplicates existing peptides from the minority class (non-quantotypic peptides). This ensures that all samples used for training remain biologically valid, maintaining the integrity of the dataset while addressing class imbalance. The impact of this oversampling approach on the dataset distribution is illustrated in Figure 3.5.

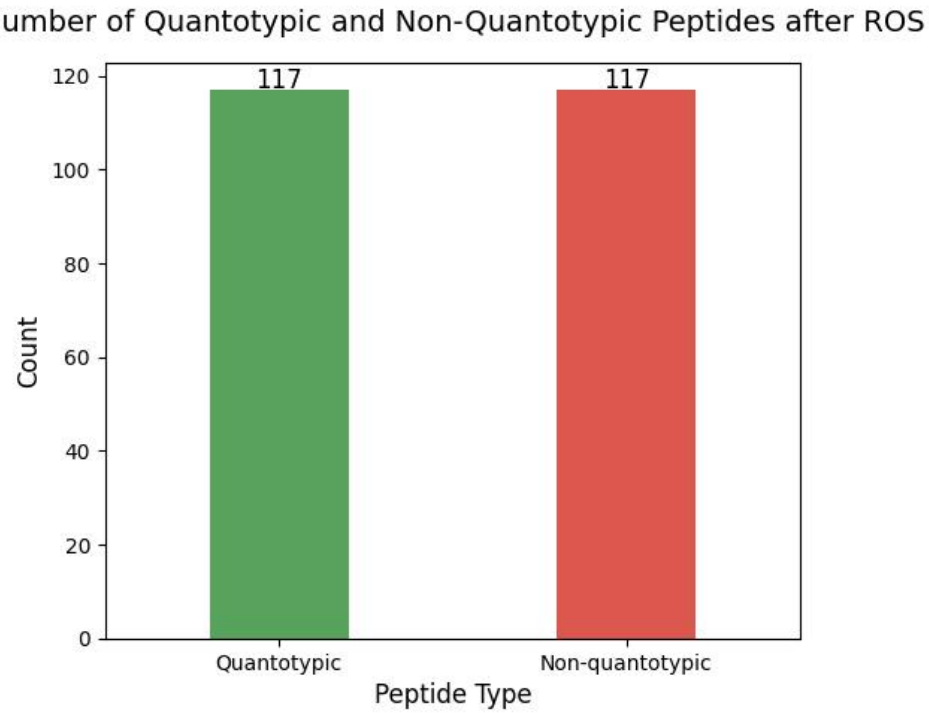


Figure 3.5: Distribution of quantotypic and non-quantotypic peptides in the dataset after applying RandomOverSampler (ROS).

3.2.2 Cost-sensitive learning

Cost-sensitive learning involves penalizing the misclassification of the minority class by assigning higher weights during model training. The idea behind this approach is to counteract the natural bias that models develop when trained on imbalanced datasets. Without such adjustments, a model may favor the majority

class, thereby overlooking the minority class. By increasing the penalty for misclassifying minority class instances, the model is encouraged to learn features specific to that class, hence improving overall performance.

When training models, one typically computes a loss function and assigns a specific weight to each class. In a binary cross-entropy implementation, the weight for the minority class (in our case, non-quantotypic) is explicitly set while keeping the majority class weight at 1. A common approach is to define the weight of the minority class as the ratio of the number of examples in the majority class to the number of examples in the minority class:

$$\text{Weight of minority class} = \frac{\text{Number of quantotypic examples}}{\text{Number of non-quantotypic examples}}$$

For the first part of this thesis which consists in leveraging artificial intelligence models using labeled data and referring to Figure 3.4, the weight of the minority class is 3.0789 (i.e., $\frac{117}{38}$) (for all techniques except Random Forest). In contrast, when using a Random Forest with the "balanced" option, the class weights are also determined based on class frequencies. Specifically, in scikit-learn, the "balanced" weight for each class i is given by:

$$w_i = \frac{n_{\text{samples}}}{n_{\text{classes}} \times n_{\text{samples}_i}},$$

where n_{samples} is the total number of samples, n_{classes} is the number of classes (2 in the binary case), and n_{samples_i} is the number of samples belonging to class i. This approach adjusts both weights instead of fixing one class weight to 1. Despite this, the proportional relationship between the two classes remains the same: a 1:3.0789 ratio between class 0 (quantotypic) and class 1 (non-quantotypic).

Chapter 4

Supervised learning

This chapter describes the supervised learning task carried out in this thesis, using the now labelled SPARE data for training and evaluating models. We start by building and testing models trained from scratch, such as a multi-layer perceptron (MLP), a bidirectional LSTM (BiLSTM), and traditional machine learning models like Random Forest and XGBoost. Then, we explore transfer learning by using pretrained encoders from models coming from AlphaPeptDeep and ESM, and applying two strategies: feature extraction and fine-tuning.

To make the evaluation reliable, we also introduce several important techniques: stratified cross-validation to split the data, grid search to tune hyperparameters, early stopping to prevent overfitting, and methods to handle class imbalance. Throughout this section, we compare the different models and techniques using performance metrics that do not depend on a threshold, such as PR AUC and ROC AUC.

The whole process, from data splitting to final evaluation, is explained in the following sections.

4.1 Methodology

4.1.1 Splitting strategy

Cross-validation is a statistical technique used to evaluate the performance of a machine learning model on unseen data, especially useful when working with limited datasets. By splitting the data into several subsets, or folds, the method involves training the model on a combination of these folds and validating it on the remaining portion. In the common approach known as k-fold cross-validation, the dataset is divided into k equal parts; the model is then trained k times, each time leaving out one distinct fold as the test set and using the remaining k-1 folds for training. The performance metrics obtained from each iteration are averaged to provide a more reliable estimate of the model's generalization capability [14].

In this work, cross-validation was employed due to the limited availability of data for supervised learning. Specifically, we used Stratified KFold which is a variant of k-fold cross-validation that preserves the distribution of quantotypic and non-quantotypic peptides across each fold, thereby ensuring that each subset faithfully represents the overall dataset. This approach mitigates the risk of obtaining folds that contain only peptides from the same class, an issue that could arise from random splitting. A 5-fold cross-validation strategy was adopted, meaning that our dataset of 155 data points was divided into five subsets of 31 data points each, with an approximate distribution of 23–24 quantotypic peptides and 7–8 non-quantotypic peptides per fold.

To correctly determine the best hyperparameter configuration, our approach involves training the model on a training set and evaluating it on a validation set, with the final performance measured on a

separate test set. We begin by applying cross-validation to the entire dataset. In each fold, the data is split into two parts: a test set and the remaining data. The remaining data is then further divided to form a validation set. Ultimately, this results in three distinct subsets: a test set comprising 20% of the original dataset, a validation set accounting for 16%, and a training set covering the remaining 64%.

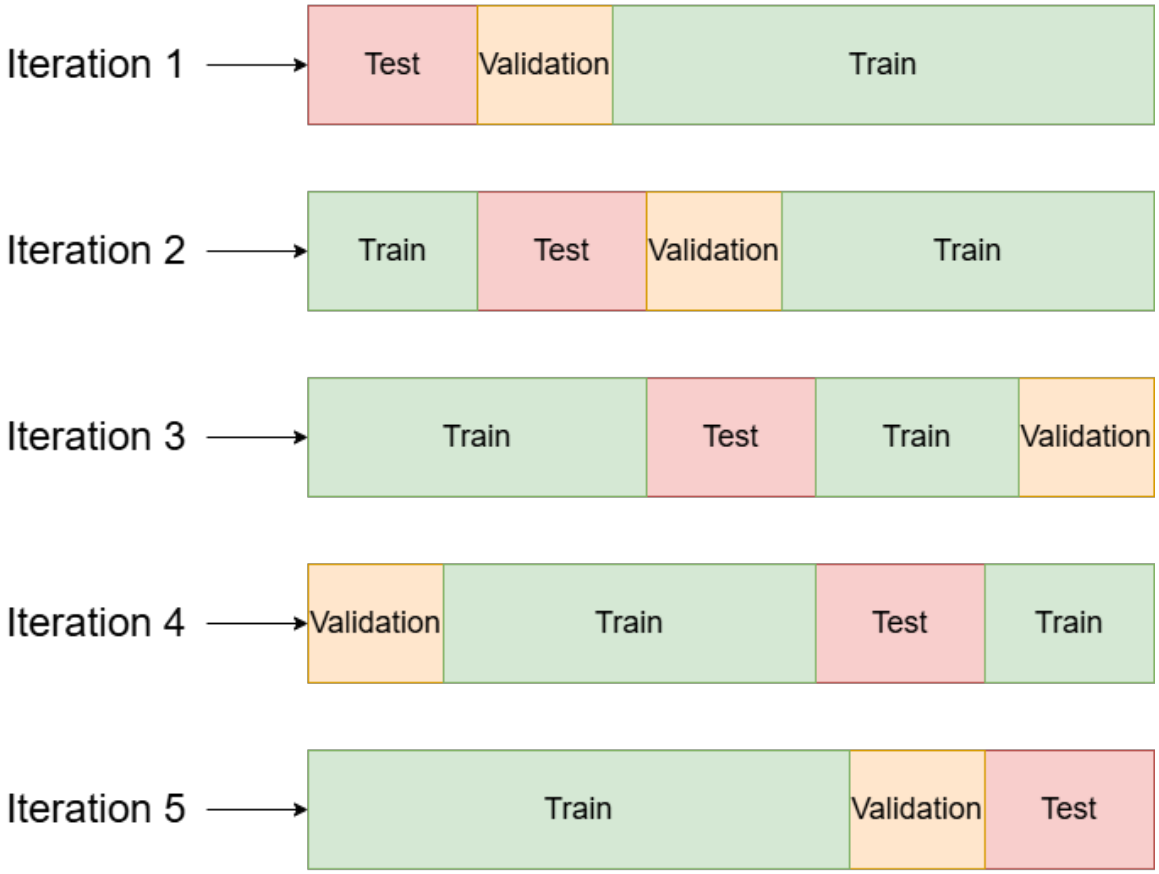


Figure 4.1: Splitting strategy: The dataset is initially divided using stratified cross-validation to ensure that each fold maintains the original class distribution. Another stratified split is performed, resulting in 3 sets in total: a test set, a validation set and a training set. In this process, 20% of the original data is allocated as the test set, while the remaining 80% is further divided so that the validation and training sets represent 16% and 64% of the original dataset, respectively.

4.1.2 Encoding technique

For models built from scratch, the amino acid sequence is encoded as follows: first, each of the 20 amino acids is replaced by a unique integer, starting from 1 (i.e., 'A' $\rightarrow$ 1, ..., 'Y' $\rightarrow$ 20). To ensure consistent input sizes, all sequences are padded to match the maximum sequence length observed in the dataset. Padding positions are filled with the value 0. For deep learning models (MLP and BiLSTM), these integers are then mapped to higher-dimensional vectors using PyTorch's built-in Embedding function. For RF and XGBoost models, the integer-encoded sequences are instead transformed using one-hot encoding.

4.1.3 Training configuration

We used the Adam optimizer to train all the deep learning models, using the binary cross entropy as loss function. We also set the batch size to 32.

4.1.3.1 Hyperparameter tuning

For each of the models tested, we conducted a GridSearch to tune the hyperparameters. GridSearch is a commonly used technique in machine learning that involves exhaustively evaluating every possible combination of hyperparameter values to determine the optimal configuration. Although this method can be computationally expensive, it ensures a thorough exploration of the parameter space, ultimately leading to more robust and reliable model performance. A detailed explanation of the different values tested can be found in the appendix A.

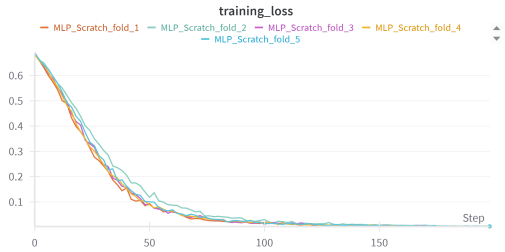
When performing a grid search over all the possible combinations of hyperparameters, we obtain one model per combination. It is then crucial to optimally select the best model among all. Metrics such as accuracy, precision, recall, and F1 rely on setting a threshold to convert the model's output probabilities into class labels. Since choosing the threshold is both crucial and tedious, these metrics are not ideal as criteria for model selection. In contrast, ROC AUC and PR AUC have the advantage of being threshold-free, meaning that no specific threshold value has to be chosen, which makes them more reliable performance metrics. In binary classification tasks with imbalanced data, although ROC AUC is popular, it can provide an overly positive view when negatives greatly outnumber positives because it takes true negatives into account, potentially masking poor performance on the positive class. Conversely, the Precision-Recall (PR) curve focuses solely on precision (the proportion of predicted positives that are actually correct) and recall (the proportion of true positives captured). This focus makes the PR AUC a more sensitive and honest measure of a model's ability to identify rare positive cases [28]. Thus, in our framework, we choose the optimal model as the one yielding the highest PR AUC score on the validation set.

4.1.3.2 Early stopping

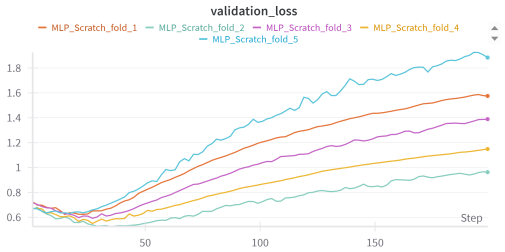
Early stopping is a regularization technique commonly used in deep learning. It involves monitoring the model's performance on a validation set during training and stopping the training process once the validation error begins to increase, even if the training error continues to decrease. This divergence indicates that the model is starting to overfit—i.e., it is learning noise and irrelevant details from the training data that do not generalize well to unseen data. By terminating training at this optimal point, early stopping helps to prevent overfitting and ensures better generalization performance [23].

In this work, early stopping is implemented by evaluating the validation loss every 5 epochs, and if the loss increases compared to the previous epochs, the training process is stopped.

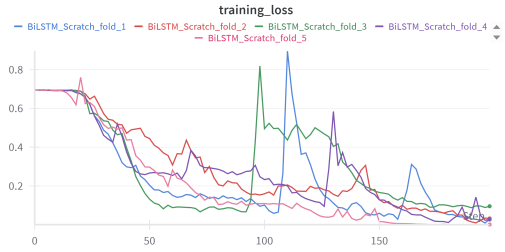
During training, I observed that only the MLP exhibited the desired behavior, which is why I applied early stopping exclusively to the MLP. For the other models, the validation loss increased immediately regardless of the hyperparameters tested. Consequently, I fixed the number of epochs for those models and added the epoch parameter to the grid search. This is surely due to the very small size of my dataset (only 155 data points). As we can see in Figure 4.2, only the MLP does have a little improvement at the very beginning of the training on the validation loss for each fold. For the other models, it either increases directly or sometimes one of the folds does have a validation loss that decreases. This is the reason why I decided to only apply Early stopping to MLP.



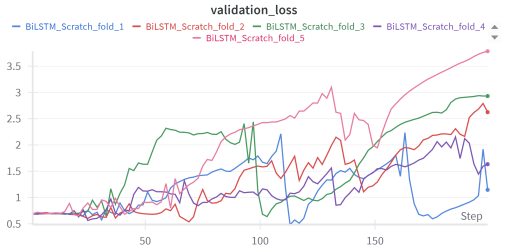
(a) MLP Train Loss



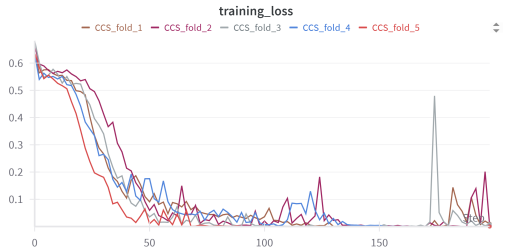
(b) MLP Validation Loss



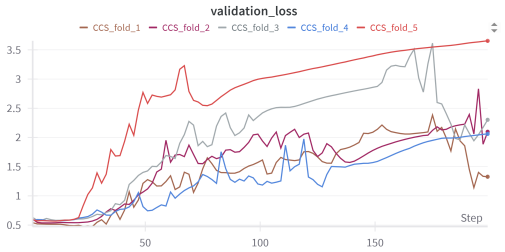
(c) BiLSTM Train Loss



(d) BiLSTM Validation Loss

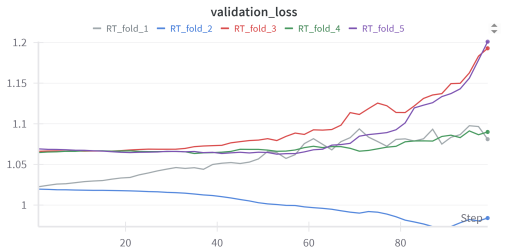


(e) CCS Train Loss



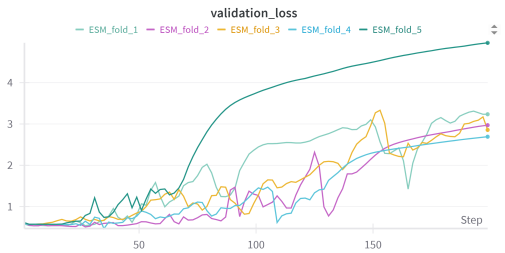
(f) CCS Validation Loss

(g) RT Train Loss



(h) RT Validation Loss

(i) ESM Train Loss



(j) ESM Validation Loss

Figure 4.2: Training and validation loss curves for the different deep learning models for the 5 folds. For each model, the left subfigure shows the training loss and the right subfigure shows the validation loss. The MLP and BiLSTM models were trained from scratch, while the ESM, RT, and CCS models were used in a transfer learning setting. Each plot correspond to a configuration with a batch size of 32 and a learning rate of 0.001.

4.1.4 Transfer learning baselines

4.1.4.1 AlphaPeptDeep

Zeng *et al.* present a modular deep learning framework, called AlphaPeptDeep [31], designed to predict peptide properties in proteomics from the amino acid sequences. It is built to be highly flexible, allowing people (even those new to AI) to easily reuse pre-trained models or integrate components into their own architectures. Currently, the framework includes three models. The first model predicts peptide retention time (RT), which is a measure of the time taken for a solute to pass through a chromatography column. The second model predicts collision cross section (CCS), a measure of the effective surface area of a peptide ion as it travels through a gas, providing insights into its three-dimensional structure. Finally, the third model predicts MS2 intensities, representing the abundance of fragment ions produced during tandem mass spectrometry experiments.

Given that pre-trained models for various tasks are available (on their github repository [32]), this framework is promising for transfer learning. These models can be fine-tuned to address new challenges, which significantly reduces the need for extensive datasets since the models have already captured essential patterns. We will further investigate whether these pre-learned patterns can be effectively used to predict the quantotypic property of peptides.

For the scope of this thesis, the MS2 model will not be used since, in addition to the amino acid sequence, it requires access to ion fragment data, whereas our aim is to predict the quantotypic characteristics of a peptide solely based on its sequence. Regarding the two remaining models, the CCS model requires an additional input column for "charge" which represents the net electrical charge of the peptide ion. In our dataset, this value will be set to 0 for each peptide to minimize its influence on the prediction, with the expectation that the model will learn to adjust accordingly. Similarly, the RT model requires two additional input columns: "mods" and "mods sites". These columns represent any chemical modifications present on the peptide and the specific sites of those modifications, respectively. Both will be set as empty strings for the same reason.

The models for predicting retention time (RT) and collision cross section (CCS) both combine a convolutional neural network (CNN) layer with two bidirectional LSTM layers and fully connected layers. Each of these models contains fewer than one million parameters, ensuring rapid and efficient training. To encode amino acid sequences, each amino acid is first mapped to a unique integer (1–26 for letters A–Z, with 0 reserved for padding). These integers are one-hot encoded into 27-dimensional vectors and then passed through a learnable embedding layer using torch.Embedding, which maps each input to a continuous N-dimensional vector.

In this work, the CCS or RT model will serve as an encoder to either a MLP/RF/XGBoost model. Each model consists of two main entities: an encoder and a decoder. We simply reuse the encoder component, which follows an architecture that first embeds the input, then processes it through a series of CNN layers, followed by LSTM layers, and finally applies attention mechanisms. In both cases, the initialization of the weights simply consists of reusing those learned by the encoder during training on the original task. We also test the two techniques used for transfer learning which are Feature Extraction and Fine Tuning.

4.1.4.2 Evolutionary Scale Modeling (ESM)

ESM is a set of protein language models developed by Meta's AI research team. It includes a variety of models designed for tasks such as structure prediction and protein design based on structural information [33]. The model that interests us is ESM2 [34], which is recognized as the state-of-the-art general-purpose protein language model, it can be used to predict structure or proteins properties. Al-

though this thesis focuses on predicting a property of peptides, it is important to remind that peptides are a subsets of proteins, and they share the same underlying language. The idea for using ESM2 is that the patterns and representations learned from large-scale protein data could potentially be transferred to peptides. This study will explore whether these learned patterns can be effectively applied to predict the quantotypic property of peptides.

ESM2 [34] is, a state-of-the-art protein language model based on a Transformer architecture and trained in a self-supervised manner using a sequence masking objective. In this approach, randomly selected amino acids within a protein sequence are masked, and the model is tasked with predicting the identity of these masked residues by analyzing the context provided by the remaining sequence. This training strategy forces the model to learn the dependencies and interactions between amino acids, capturing both local and long-range patterns that reflect the evolutionary and structural relationships inherent in protein sequences. Trained on millions of sequences, ESM2 effectively captures subtle evolutionary regularities and residue interactions. Moreover, there exist several ESM2 models ranging from 8 million to 15 billion parameters, and each increase in scale significantly enhances the model's ability to represent and understand underlying structures. This scaling results in an improved accuracy in predicting contacts and three-dimensional structures, establishing ESM2 as a state-of-the-art tool for protein structure prediction. With this understanding of the dependencies and interactions between amino acids, we will explore whether these patterns can be applied to property prediction.

In this work, the ESM will serve as an encoder to either a MLP/RF/XGBoost model. We use the version with 8 million parameters, which includes 6 representation layers. Features can be extracted from any of these layers; in our case, we choose to use the output from the 6th (final) representation layer. The initialization of the weights simply consists of reusing those learned by the encoder during training on the original task. We also test the two techniques used for transfer learning which are Feature Extraction and Fine Tuning.

4.1.5 Training variants

Table 4.1 summarizes the different variants tested for each model, where the approach to training is different, such as Early stopping, Feature extraction / Fine tuning, and class imbalance technique.

Table 4.1: Table summarizing the variants tested for each model. "Early Stopping" indicates that this technique was evaluated. "Transfer Learning" means that both fine-tuning (updating the encoder parameters) and feature extraction (freezing the encoder parameters) were tested. "Class Imbalance" refers to the application of oversampling, weighting or none of these.

Model	Early Stopping	Transfer Learning	Class Imbalance
Random Forest	✗	✗	✓
XGBoost	✗	✗	✓
MLP	✓	✗	✓
BiLSTM	✗	✗	✓
RT	✗	✓	✓
CCS	✗	✓	✓
ESM	✗	✓	✓

4.1.6 Overview of the methodology

Figure 4.3 illustrates the methodology applied for the supervised learning task. First, a model and its training variant are selected. Then, a cross-validation procedure is performed to ensure a robust evaluation,

with each fold kept fully independent. For each training fold, a grid search is conducted to identify the best combination of hyperparameters. The best model is subsequently evaluated on the corresponding test set. Finally, the results across all folds are aggregated by computing the mean and standard deviation of the evaluation metrics.

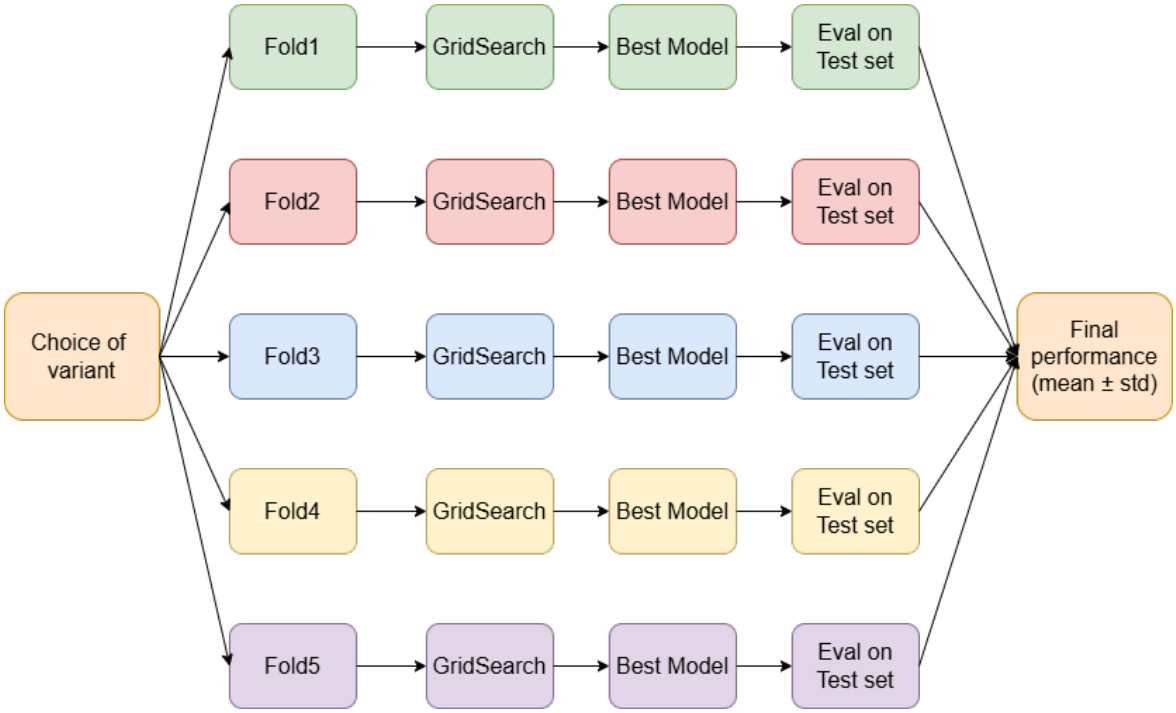


Figure 4.3: Illustration of the process for the supervised learning task. First, a model is chosen with its combination of variant. Then, cross-validation is performed, each fold is represented by a different color to remind their independence. GridSearch is done in order to find the best combination of hyperparameters, then we select the best model and then we evaluate its performance on the test set. Finally, the results from all folds are aggregated by calculating the mean and standard deviation (std).

4.2 Results

This section presents the results of the different models tested. It also compares the various variants explored, such as early stopping, transfer learning, and handling of class imbalance, and provides insights into their impact. When analyzing the effect of a specific variant, the results are grouped by that variant, and only the PR AUC and ROC AUC metrics are plotted, as they are the most relevant to us due to being threshold-independent. The plot is designed to avoid any aggregation between combinations of variants. For instance, if a model is trained using two different variants, we plot the PR AUC and ROC AUC scores with respect to the other variant on the x-axis. For each x-tick (each configuration of the other variant), we plot the performance of the variant of interest, allowing us to isolate and visualize its effect more clearly. All the numerical results can be found in Appendix B.1.

4.2.1 Early stopping effect

As a reminder, early stopping refers to stopping the training when the error on the validation set starts increasing while the error on the training set is still decreasing. This is a sign of overfitting, which we want to avoid. Stopping the training at the right moment helps prevent this problem. This method is only tested on the MLP as explained in the methodology (see 4.1.3.2).

By analyzing Figure 4.4, the MLP was tested with two different variants. As a result, the x-axis represents the three different class imbalance handling techniques with both PR AUC and ROC AUC. For each x-tick (each class imbalance method, where N=None, O=Oversampling and W=Weighting), we can compare the effect of early stopping. Regardless of the variant configuration, applying early stopping consistently leads to higher values for both PR AUC and ROC AUC. The best performance appears to be achieved when using the 'W' configuration, which corresponds to the weighting technique for handling class imbalance.

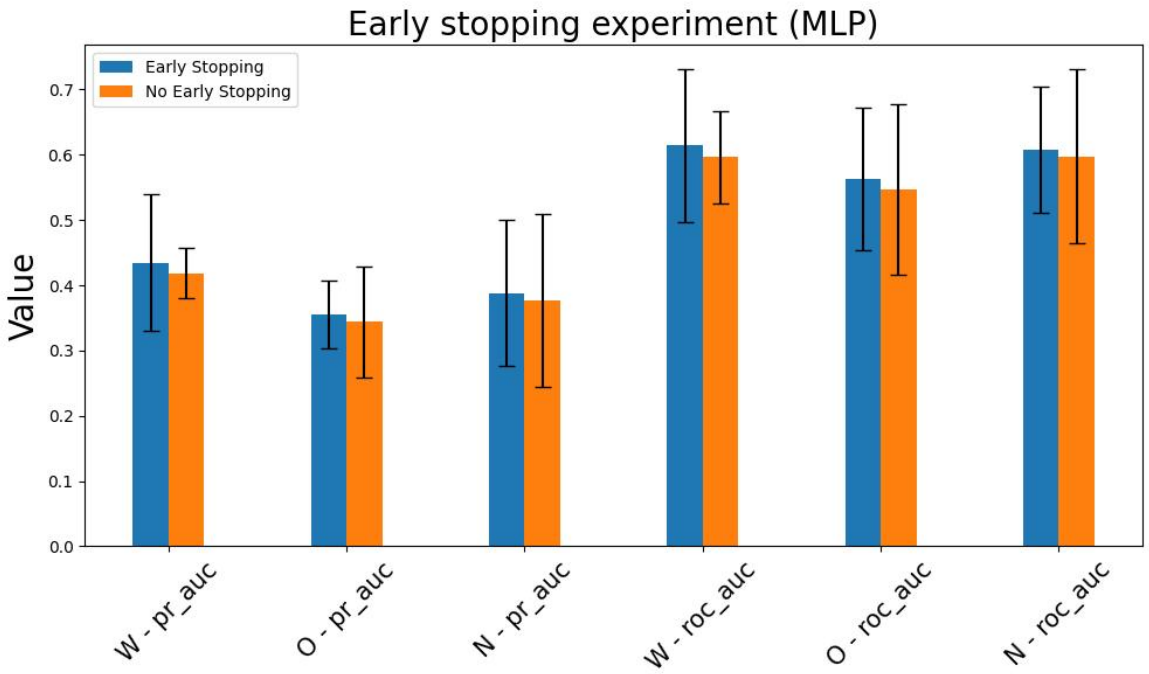


Figure 4.4: Effect of early stopping on the MLP model. The figure shows PR AUC and ROC AUC across different class imbalance handling techniques (N=None, O=Oversampling and W=Weighting). For each method, early stopping consistently improves performance, with the best results observed for the weighting strategy ('W').

4.2.2 Transfer learning effect

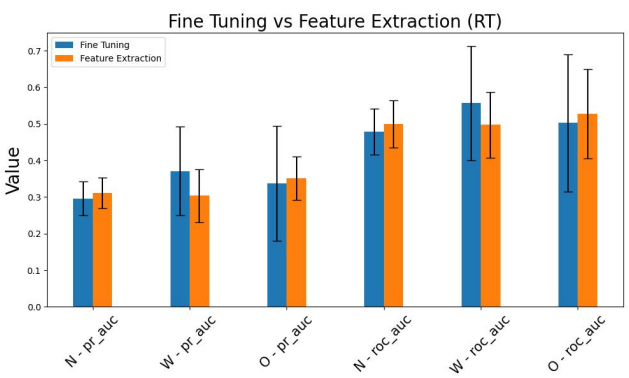
As a reminder, transfer learning involves using a model that was previously trained on one task and reapplying it to a different task. In our case, we use only the encoder part of the pre-trained models, which serves as input to a MLP. The encoder's weights can either be frozen, which corresponds to feature extraction, or updated during training, which corresponds to fine-tuning. The encoder is taken from either ESM or AlphaPeptDeep, specifically from the RT or CCS models. The graphs also present PR AUC and ROC AUC scores across different class imbalance handling techniques.

For the RT model (Figure 4.5a), feature extraction performs better when using either oversampling ('O') or no ('N') class imbalance handling. In contrast, when applying the weighting strategy ('W'), fine tuning yields better results. The best performance for both PR AUC and ROC AUC is achieved when combining fine tuning with the weighting technique.

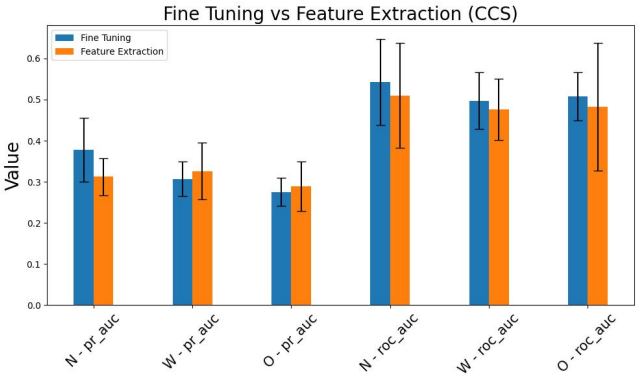
For the CCS model (Figure 4.5b), feature extraction performs better in terms of PR AUC when using either the weighting ('W') or oversampling ('O') techniques. When no class imbalance handling is applied ('N'), fine tuning outperforms feature extraction. For ROC AUC, fine tuning consistently yields better results regardless of the class imbalance technique used. Once again, the best overall performance is

observed when combining fine tuning with no class imbalance handling.

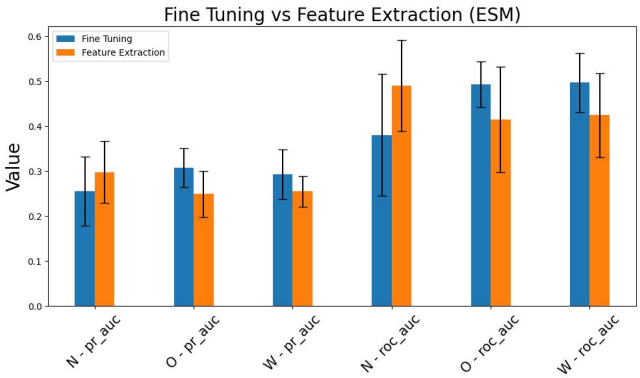
For the ESM model (Figure 4.5c), feature extraction performs better when no class imbalance technique is applied ('N'), for both PR AUC and ROC AUC. In contrast, fine tuning yields better performance when using either weighting ('W') or oversampling ('O') techniques. However, determining the best configuration is not straightforward, as the mean results are quite close. Regardless of the class imbalance technique used, both PR AUC and ROC AUC achieve similar mean values. Therefore, the standard deviation becomes a key factor for comparison. In this context, fine tuning combined with oversampling ('O') exhibits the lowest standard deviation, suggesting it provides the most consistent and reliable performance.



(a) RT + MLP



(b) CCS + MLP



(c) ESM + MLP

Figure 4.5: Effect of transfer learning on model performance across different encoders (RT, CCS, and ESM). For each encoder, the impact of updating the encoder weights during training (fine tuning) is compared to keeping them frozen (feature extraction), across various class imbalance handling techniques (N=None, O=Oversampling and W=Weighting). Although feature extraction occasionally outperforms fine tuning depending on the encoder and class imbalance configuration, the best overall performance is achieved with fine tuning.

4.2.3 Class imbalance effect

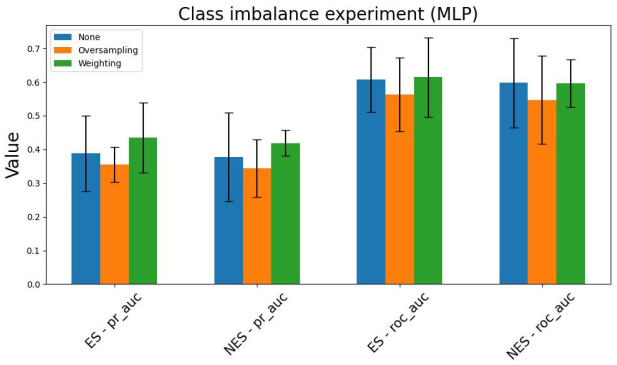
As a reminder, our dataset is affected by class imbalance, with the quantotypic class being overrepresented compared to the non-quantotypic class. This can cause issues during prediction, as the model might learn to favor the quantotypic class. Figure 4.6 shows the impact of the different strategies we used to address this imbalance (cost-sensitive learning, oversampling or none of these) across all models.

Among the 13 models tested, three show identical values for both PR AUC and ROC AUC. These models clearly fail to learn any useful patterns. They correspond to the models that use a transfer learning

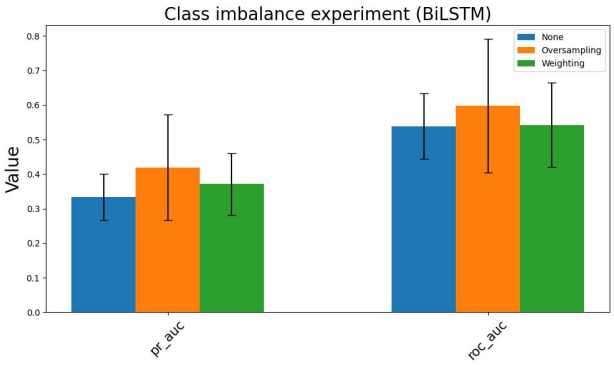
encoder followed by an XGB classifier (Figure 4.6k, Figure 4.6g and Figure 4.6i).

For the remaining 10 models, we first focus on those where one class imbalance handling technique consistently outperforms the others across both metrics and all variants. Two models perform best without any class imbalance handling: the CCS model with an MLP (Figure 4.6c), and the RT encoder followed by a Random Forest (RF) (Figure 4.6j). Two other models achieve better results with cost-sensitive learning (i.e., weighting): the MLP (Figure 4.6a) and XGB models (Figure 4.6m). Finally, four models perform best when using oversampling: the BiLSTM (Figure 4.6b), CCS + RF (Figure 4.6f), ESM + RF (Figure 4.6h), and RF models (Figure 4.6l).

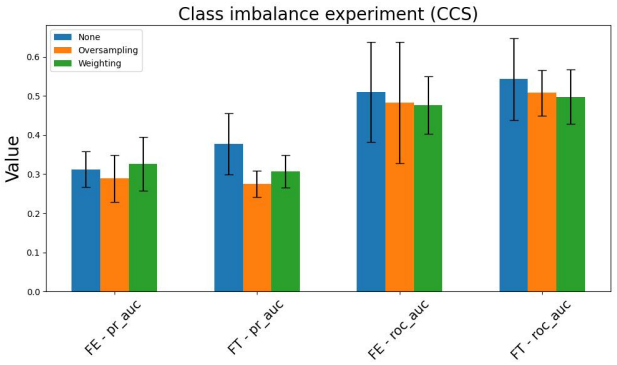
There is two last models, RT + MLP and ESM + MLP stand out due to their more complex behavior. In both cases, the optimal class imbalance handling technique varies depending on the transfer learning strategy applied. For the ESM + MLP model (Figure 4.6e), feature extraction leads to the best overall performance when no class imbalance technique is applied. On the other hand, when fine tuning is used, oversampling yields better results for PR AUC, while weighting is more effective for ROC AUC. Despite these variations, the smallest standard deviation in terms of PR AUC is when using fine tuning and oversampling. A similar trend is observed for the RT + MLP model (Figure 4.6d), feature extraction performs best in combination with oversampling, while fine tuning works better with weighting. Although the mean scores are very close in both cases, the lower standard deviation associated with feature extraction and oversampling suggests that this combination is more stable and therefore preferable for this model.



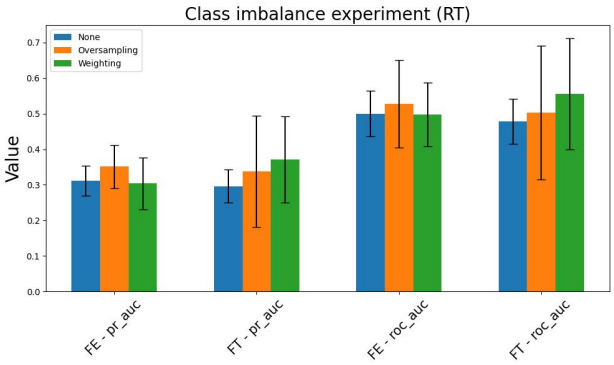
(a) MLP



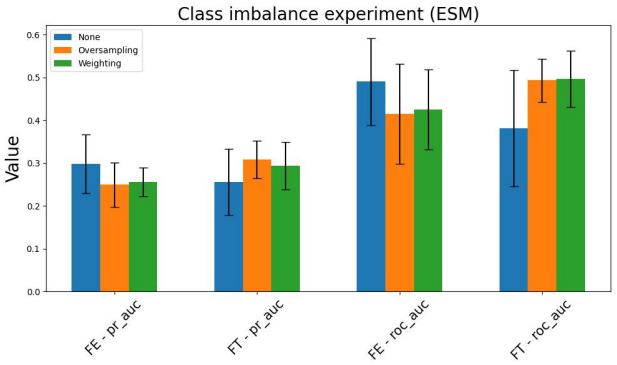
(b) BiLSTM



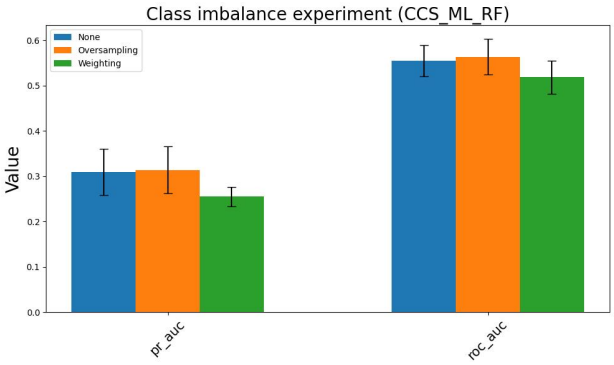
(c) CCS + MLP



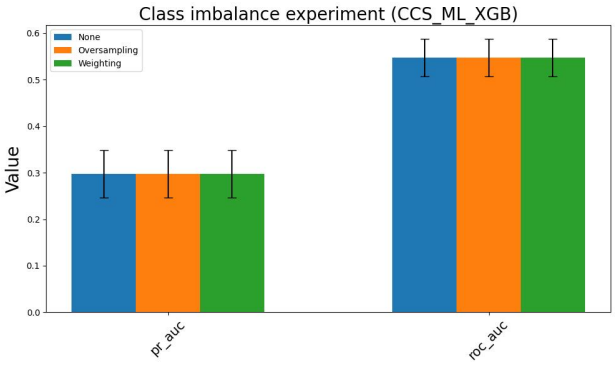
(d) RT + MLP



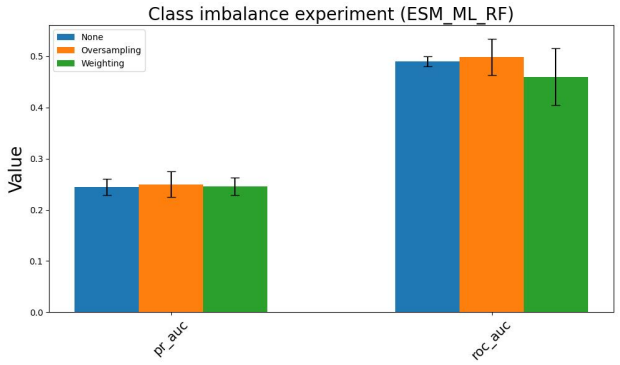
(e) ESM + MLP



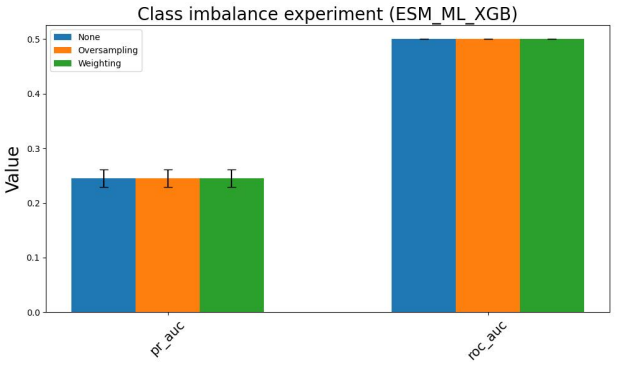
(f) CCS + RF



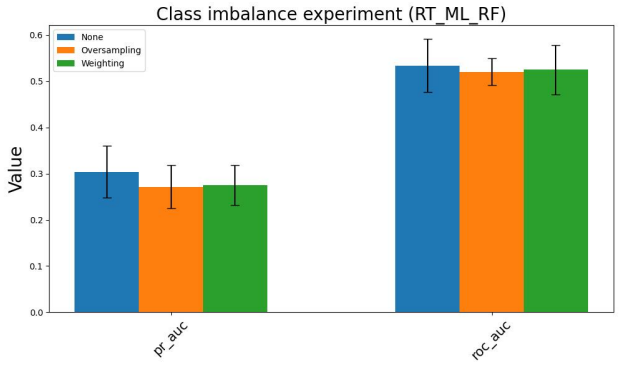
(g) CCS + XGB



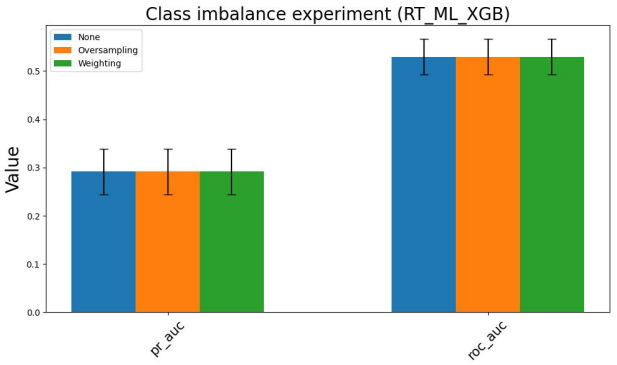
(h) ESM + RF



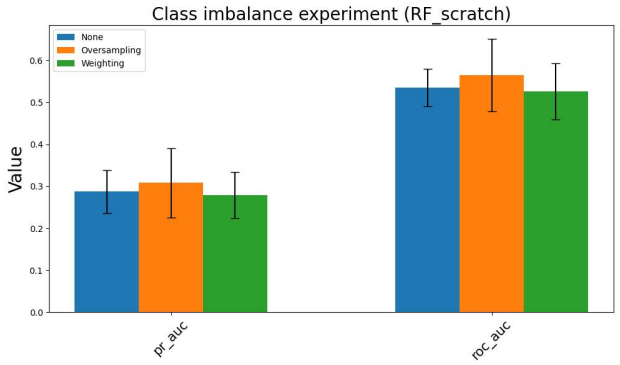
(i) ESM + XGB



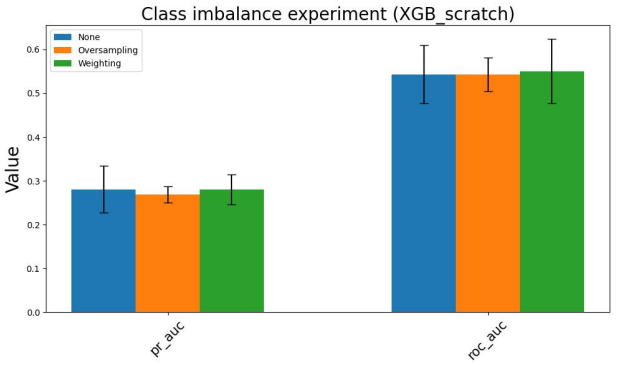
(j) RT + RF



(k) RT + XGB



(l) RF



(m) XGB

Figure 4.6: Class imbalance experiment across multiple models and metrics. In the case of the MLP model, ES refers to Early Stopping, while NES indicates that no early stopping was applied. For transfer learning models (RT + MLP, CCS + MLP, and ESM + MLP), FT stands for Fine Tuning and FE for Feature Extraction. Among the 13 configurations tested, three show identical results for both PR AUC and ROC AUC, indicating that these models fail to learn meaningful patterns; all correspond to models using a transfer learning encoder followed by an XGB classifier. Among the remaining models, some exhibit a clear preference: two perform best without any class imbalance correction (CCS + MLP and RT + RF), two with cost-sensitive learning (MLP and XGB), and four with oversampling (BiLSTM, CCS + RF, ESM + RF, and RF). However, for ESM + MLP and RT + MLP, the optimal strategy depends on the transfer learning configuration. For ESM + MLP, feature extraction achieves the highest performance when no class imbalance strategy is applied. In contrast, fine tuning performs better with oversampling for PR AUC, and with weighting for ROC AUC. Nevertheless, the overall best results are obtained with fine tuning combined with oversampling as it is the smallest standard deviation in terms of PR AUC. For RT + MLP, a similar trend is observed: feature extraction performs best in combination with oversampling, while fine tuning works better with weighting. Although the mean scores are very close in both cases, the lower standard deviation associated with feature extraction and oversampling suggests that this combination is more stable and therefore preferable for this model.

4.2.4 Inter model comparison

Table 4.2 presents a comparison of the best models, showing the best configuration for each of the models tested. For example, among all the MLP models (across different variant configurations), the best results are achieved when using early stopping combined with the weighting technique. This is consistent with the conclusions drawn earlier in the section discussing the effects of different variants.

When comparing the performance of the different models across the various metrics, it is clear that the two models built from scratch (MLP and BiLSTM) outperform all others. They are the only ones to achieve the highest PR AUC and ROC AUC scores, reaching around 0.42 and 0.6 respectively. Among them, the MLP performs better than the BiLSTM, with a higher mean and lower standard deviation for both PR AUC and ROC AUC. Regarding transfer learning, the results are particularly disappointing for the ESM encoder, with a ROC AUC below 0.5 on average, indicating performance worse than a dummy classifier. Other transfer learning models perform slightly better but still achieve low values. Models based on classical machine learning algorithms, such as Random Forest (RF) or XGBoost (XGB), also show poor results, with PR AUC and ROC AUC values far behind those achieved by MLP and BiLSTM, regardless of the encoding technique used, from simple one-hot encoding (scratch) to transfer learning encoders (ESM, RT, or CCS).

Overall, the results are not fully convincing, as even the best models fail to achieve high PR AUC and ROC AUC scores. However, the MLP and BiLSTM models seem to capture the most meaningful patterns, achieving the best results. To continue this work, we will apply self-training on the MLP and BiLSTM models, as they showed the strongest performance. Finally, it is important to recall that we are working with a very limited amount of data, which likely contributes to the disappointing results in this supervised learning task.

Table 4.2: Inter-model comparison: this table presents the best configuration identified for each model across different variant settings. The first four columns describe the model configuration: the first indicates the model name; the second (ES) specifies whether early stopping was applied ('Yes' for enabled, 'No' for disabled, and '/' for not evaluated); the third (TL) refers to the transfer learning strategy used ('FT' for Fine Tuning, 'FE' for Feature Extraction, and '/' if transfer learning was not used); and the fourth (CI) denotes the class imbalance handling strategy ('N' for none, 'W' for weighting, and 'O' for oversampling). The comparison shows that MLP and BiLSTM models, both trained from scratch, consistently outperform other approaches, achieving the highest PR AUC and ROC AUC scores. In contrast, transfer learning models using ESM, CCS, or RT encoders perform poorly, and traditional machine learning models such as Random Forest and XGBoost also fail to deliver strong results regardless of the encoder used. Overall, while the general performance remains modest, MLP and BiLSTM stand out as the most effective models in this study.

Model	ES	TL	CI	Accuracy	Precision	Recall	F1	PR AUC	ROC AUC
MLP	Yes	/	W	0.67 ± 0.09	0.44 ± 0.30	0.32 ± 0.18	0.30 ± 0.12	**0.43 ± 0.10**	**0.61 ± 0.12**
BiLSTM	/	/	O	0.65 ± 0.12	0.32 ± 0.15	0.31 ± 0.23	0.30 ± 0.16	0.42 ± 0.15	0.60 ± 0.19
ESM	/	FT	O	0.61 ± 0.09	0.18 ± 0.10	0.26 ± 0.20	0.21 ± 0.13	0.31 ± 0.04	0.49 ± 0.05
CCS	/	FT	N	0.72 ± 0.02	0.28 ± 0.15	0.23 ± 0.18	0.25 ± 0.16	0.38 ± 0.08	0.54 ± 0.10
RT	/	FT	W	0.70 ± 0.07	0.20 ± 0.20	0.08 ± 0.06	0.11 ± 0.09	0.37 ± 0.12	0.56 ± 0.16
scratch_RF	/	/	O	0.65 ± 0.14	0.38 ± 0.20	0.41 ± 0.16	0.37 ± 0.13	0.31 ± 0.08	0.56 ± 0.09
scratch_XGB	/	/	W	0.64 ± 0.08	0.31 ± 0.11	0.38 ± 0.16	0.33 ± 0.12	0.28 ± 0.03	0.55 ± 0.07
RT_RF	/	/	N	0.74 ± 0.07	0.57 ± 0.42	0.13 ± 0.08	0.20 ± 0.12	0.30 ± 0.06	0.53 ± 0.06
RT_XGB	/	/	W	0.74 ± 0.07	0.55 ± 0.40	0.13 ± 0.08	0.18 ± 0.09	0.29 ± 0.05	0.53 ± 0.04
ESM_RF	/	/	O	0.72 ± 0.06	0.11 ± 0.16	0.07 ± 0.10	0.09 ± 0.12	0.25 ± 0.03	0.50 ± 0.04
ESM_XGB	/	/	W	0.75 ± 0.02	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.25 ± 0.02	0.50 ± 0.00
CCS_RF	/	/	O	0.77 ± 0.02	0.57 ± 0.33	0.15 ± 0.09	0.24 ± 0.13	0.31 ± 0.05	0.56 ± 0.04
CCS_XGB	/	/	W	0.76 ± 0.02	0.53 ± 0.32	0.13 ± 0.08	0.20 ± 0.12	0.30 ± 0.05	0.55 ± 0.04

Chapter 5

Self-training

Self-training is a semi-supervised learning method that uses unlabeled data as input to generate pseudo-labels, thus increasing the number of examples available for training the model. Initially, the model is trained on a small set of labeled data before being used to predict labels for a larger set of unlabeled data. The predictions that meet a certain criterion are then selected as pseudo-labels and incorporated into the training set, allowing the model to be retrained on an expanded dataset. This iterative process continue until a stopping condition is met.

In self-training, the quality of pseudo-labels plays a crucial role in the overall success of the learning process. Assigning incorrect or uncertain labels can lead the model to learn misleading patterns and degrade its performance over time. Therefore, it is essential to use reliable strategies to select and apply pseudo-labels effectively. In this chapter, we begin by presenting different pseudo-labeling strategies, including threshold-based, proportion-based, and optimal threshold-based methods (*Pseudo labeling techniques*). We then explore other important aspects of the self-training process, such as whether to assign soft or hard targets (*Soft labels and hard labels*), when to stop the self-training loop (*Stopping criteria*), and how to design the training objective when combining labeled and pseudo-labeled data (*Double loss*). Each of these components helps ensure that pseudo-labeling contributes positively to model learning, rather than introducing noise or bias.

5.1 Methodology

5.1.1 Pseudo labeling techniques

5.1.1.1 Threshold-based method

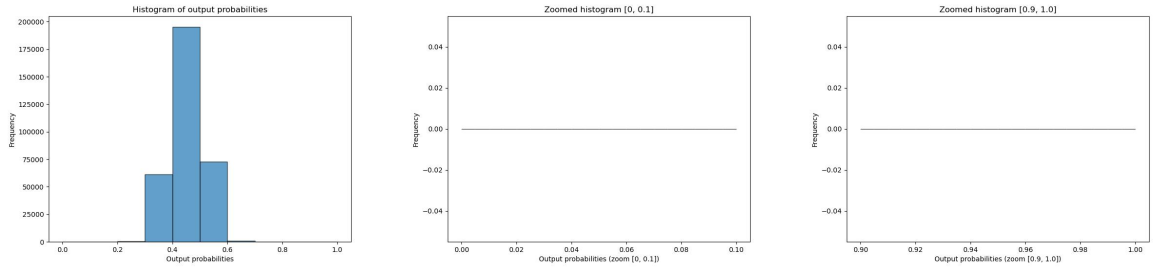
In practice, pseudo-labeling strategies can follow different approaches to determine which unlabeled examples are reliable enough to be added to the training set. One common strategy is the threshold-based method [15], where pseudo-labels are assigned only if the model's confidence exceeds a predefined threshold. By setting high absolute confidence thresholds, this approach ensures that only predictions with strong certainty are incorporated, thereby reducing the risk of introducing erroneous labels. However, fixed thresholds may not always be ideal since the optimal confidence level can vary as the model improves through successive iterations. For instance, Dong-Hyun Lee [35] demonstrates a straightforward application of this idea. In that work, the pseudo-labels are generated by simply selecting the class with the maximum predicted probability to train deep neural networks on both labeled and unlabeled data. Although the paper employ this strategy to a multi-class problem (handwritten digit recognition), the principle can be applied to binary classification as well.

By looking at the output probability distributions on the unlabeled samples for the MLP (Figure 5.1) and BiLSTM (Figure 5.2), I decided to test four different threshold values: 0.7, 0.9, 0.95, and 0.99. The

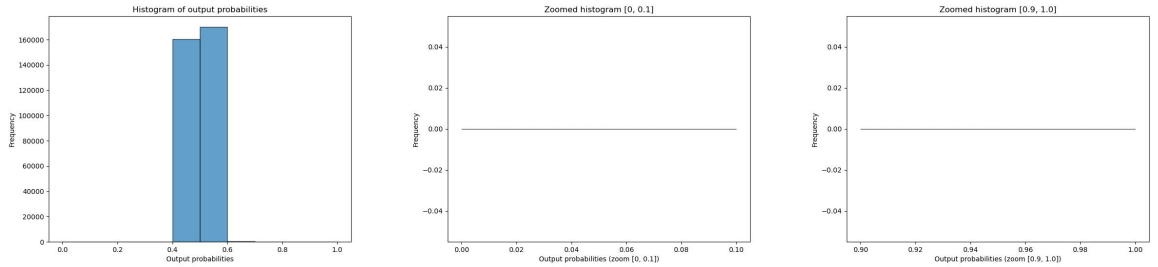
reason is that the probability distributions vary a lot from one model to another and even between folds. Sometimes, when zooming in on a smaller range of probabilities, the distribution appears almost uniform, which suggests that we should try higher threshold values. However, in other cases, the model is not confident at all, and we don't see any probabilities in that zoomed range. This is not a major issue, as it simply means that no pseudo-labels are added when the model is unsure, which helps avoid adding noisy or incorrect labels.

For the MLP model, the last two folds (Figure 5.1j to Figure 5.1o) are particularly interesting for testing different threshold values. In these cases, the model shows some confidence, so applying thresholds can help select reliable pseudo-labels. However, for the first three folds (Figure 5.1a to Figure 5.1i), no pseudo-labels are added because the model does not output any probabilities above the zoomin thresholds (0.9, 0.95, 0.99). This is actually a good thing, if any pseudo-labels had been added, they would have likely been uncertain and potentially misleading.

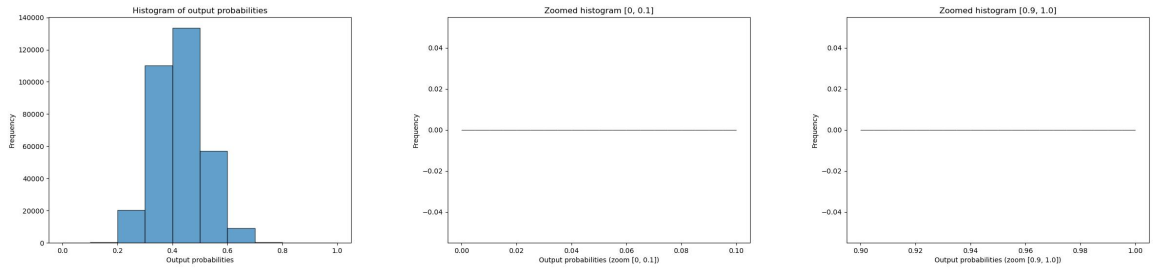
For the BiLSTM model, only one fold (Figure 5.2c) shows some predictions in the 0.9–1.0 range, with a few visible bars in the histogram. In the zoomed-in view of the 0.0–0.1 range, the model seems quite confident for the first two folds (Figure 5.2a to Figure 5.2f), with many predictions close to 0. So, when using thresholds like 0.9, 0.95, or 0.99, we will end up with almost the same number of pseudo-labels. This is because the distribution is nearly the same at the right (resp. left) of the threshold value (resp. 1-threshold).



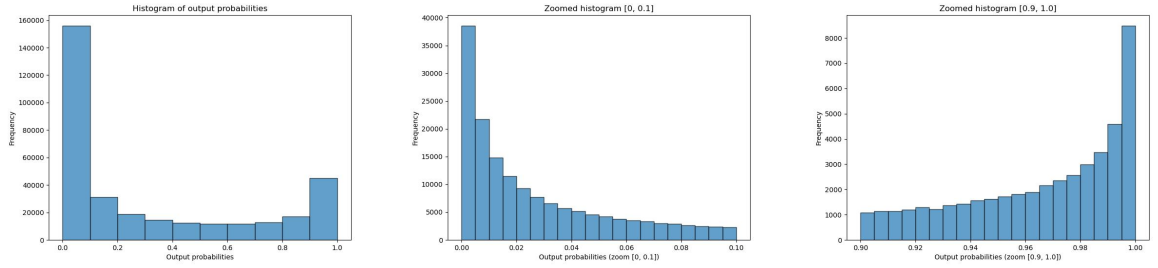
(a) Fold 1 - Full (b) Fold 1 - Zoom [0.0 ; 0.1] (c) Fold 1 - Zoom [0.9 ; 1.0]



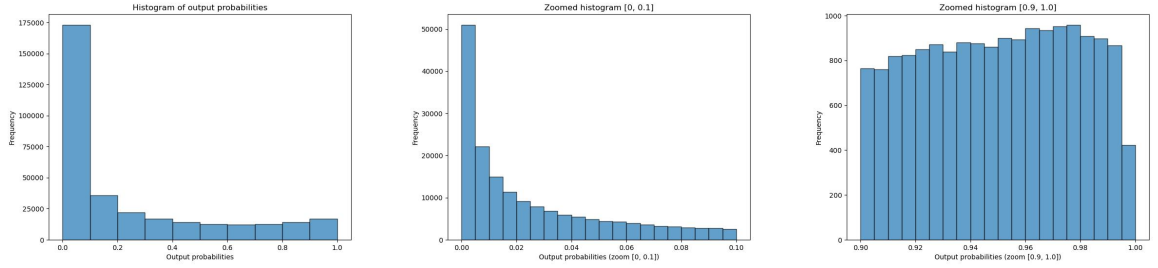
(d) Fold 2 - Full (e) Fold 2 - Zoom [0.0 ; 0.1] (f) Fold 2 - Zoom [0.9 ; 1.0]



(g) Fold 3 - Full (h) Fold 3 - Zoom [0.0 ; 0.1] (i) Fold 3 - Zoom [0.9 ; 1.0]

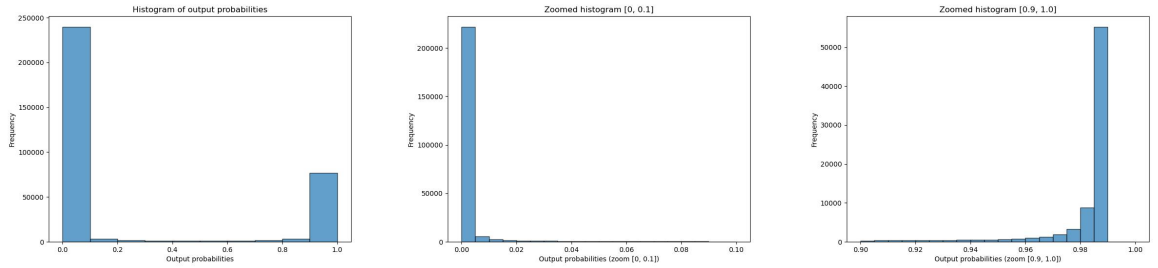


(j) Fold 4 - Full (k) Fold 4 - Zoom [0.0 ; 0.1] (l) Fold 4 - Zoom [0.9 ; 1.0]

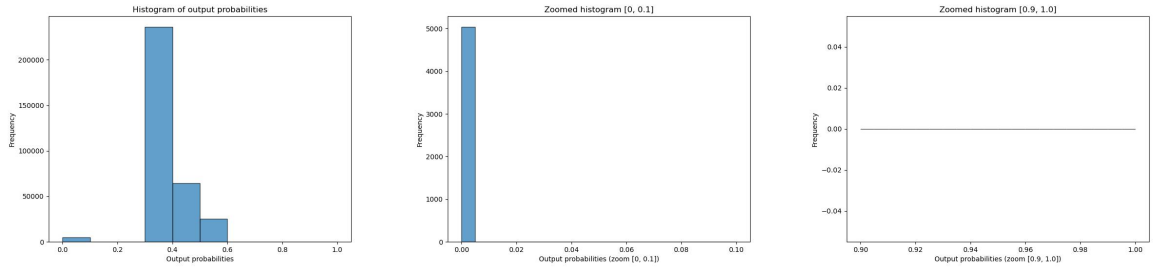


(m) Fold 5 - Full (n) Fold 5 - Zoom [0.0 ; 0.1] (o) Fold 5 - Zoom [0.9 ; 1.0]

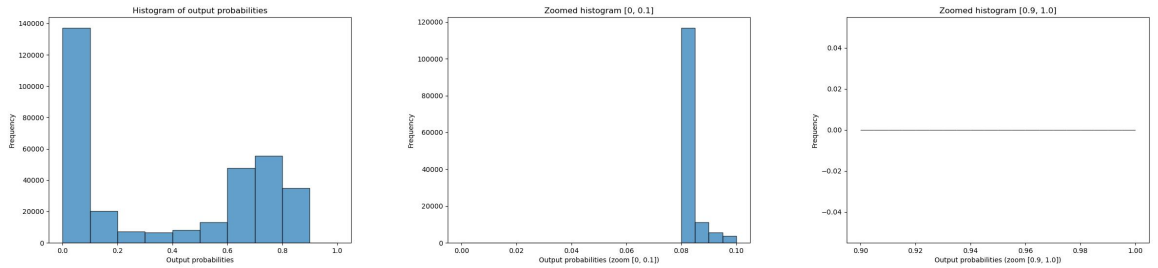
Figure 5.1: MLP output probabilities on the unlabeled samples across folds. Each row shows the full view and two zoomed-in views ([0.0 ; 0.1] and [0.9 ; 1.0]) for a specific fold.



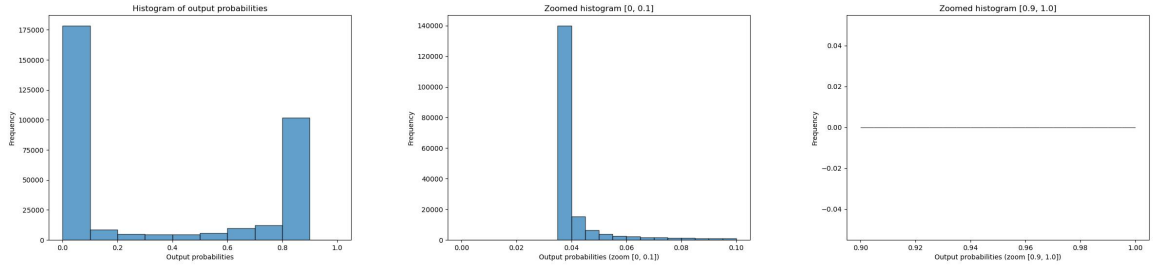
(a) Fold 1 - Full (b) Fold 1 - Zoom [0.0 ; 0.1] (c) Fold 1 - Zoom [0.9 ; 1.0]



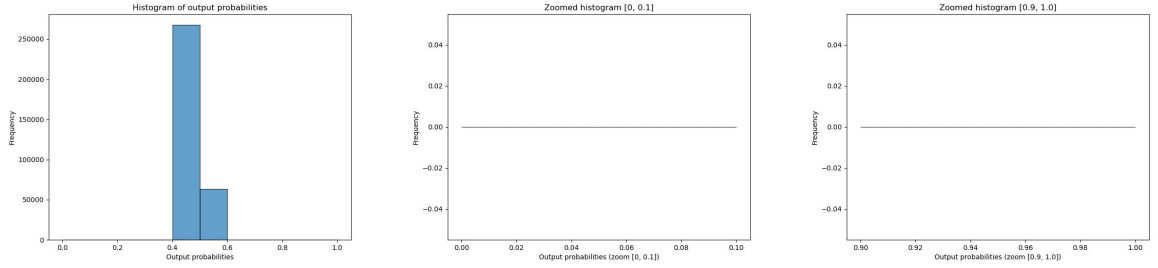
(d) Fold 2 - Full (e) Fold 2 - Zoom [0.0 ; 0.1] (f) Fold 2 - Zoom [0.9 ; 1.0]



(g) Fold 3 - Full (h) Fold 3 - Zoom [0.0 ; 0.1] (i) Fold 3 - Zoom [0.9 ; 1.0]



(j) Fold 4 - Full (k) Fold 4 - Zoom [0.0 ; 0.1] (l) Fold 4 - Zoom [0.9 ; 1.0]



(m) Fold 5 - Full (n) Fold 5 - Zoom [0.0 ; 0.1] (o) Fold 5 - Zoom [0.9 ; 1.0]

Figure 5.2: BiLSTM output probabilities on the unlabeled samples across folds. Each row shows the full view and two zoomed-in views ([0.0 ; 0.1] and [0.9 ; 1.0]) for a specific fold.

5.1.1.2 Proportion-based method

An alternative is the proportion-based method [15], which does not rely on an absolute confidence threshold. Instead, it selects a predetermined proportion of the most confident unlabeled examples for pseudo-labeling. In this approach, one might start by pseudo-labeling only a small percentage of the most confidently predicted samples, and then gradually increase this proportion as the model becomes more reliable. This dynamic adjustment allows the pseudo-labeling process to adapt to the model's evolving performance, helping to balance the trade-off between incorporating a sufficient amount of new data and minimizing the introduction of noise. For instance, Zou *et al.* [36] demonstrate a self-paced learning strategy in which pseudo-labeling begins with only 20% of the most confident predictions and is incrementally increased by 5% per round, thereby dynamically adapting the pseudo-label selection to the model's evolving accuracy. This approach improved classification performance and also achieved superior results on several different datasets in the context of image segmentation. Once again, this example is done on a multi-class problem (image segmentation) but the principle can also be applied to a binary classification problem. In this work, we will explore proportion-based pseudo-labeling strategies by selecting the top 0.1%, 1%, and 5% most confident predictions. Additionally, we will investigate a dynamic evolution strategy, where the proportion of selected pseudo-labels progressively increases from 1% up to 50%, following the sequence 0.01, 0.05, 0.1, 0.2, 0.3, 0.4, 0.5. This gradual inclusion of pseudo-labeled samples is intended to mirror the self-paced learning paradigm, allowing the pseudo-labeling process to adapt to the improving reliability of the model over time.

5.1.1.3 Optimal thresholding

The last pseudo-labeling technique is inspired from Radhakrishnan *et al.* [37]. In their work, the authors propose a method for selecting pseudo-labels based on entropy thresholding, which aims to better account for the uncertainty present in the model's predictions. Rather than simply using the maximum softmax score of the predicted class to determine whether a pseudo-label should be trusted, this method evaluates the entropy of the entire softmax output vector, which provides a more global and reliable measure of prediction confidence.

To apply this approach, for each prediction on the validation set, we compute its normalized entropy, which is obtained by dividing the entropy of the prediction by the logarithm of the number of classes (i.e., $\log(N_c)$ for N_c classes). This normalization ensures that entropy values lie within the range $[0, 1]$, regardless of the number of classes. Next, a grid search is performed over a wide range of possible entropy thresholds (typically 500 values between 0 and 1). For each threshold, predictions with entropy below the threshold are selected as confident pseudo-labels, while those above the threshold are discarded. We then evaluate the true positive rate (TPR) and false positive rate (FPR) of these selections on the validation set, where ground-truth labels are available. This allows us to assess the effectiveness of each entropy threshold in identifying reliable pseudo-labels. To select the optimal entropy threshold, we perform Receiver Operating Characteristic (ROC) analysis, the best threshold is selected as the one corresponding to the point on the ROC curve that has the smallest Euclidean distance to the top-left corner of the plot, which represents ideal classification performance.

However, it is important to note that the utility of entropy thresholding depends on the number of classes. In binary classification, where the softmax output is a two-dimensional vector (e.g., $[p, 1-p]$), the entropy is a simple, symmetric function that reaches its maximum at $p = 0.5$ and minimum at $p = 0$ or $p = 1$. In such cases, the entropy and the softmax score are strongly correlated, and thresholding on entropy is effectively equivalent to thresholding on the softmax score. Hence, in this work, since we are dealing with a binary classification problem, we propose to apply a similar strategy directly on the softmax score. Specifically, instead of thresholding the entropy, two distinct methods will be evaluated. In the first method, we will perform a grid search over possible softmax score thresholds and select the optimal

threshold based on ROC analysis, by choosing the threshold that minimizes the Euclidean distance to the ideal point (TPR = 1, FPR = 0) on the ROC curve. In the second method, we will conduct a similar grid search but based on Precision-Recall (PR) analysis, selecting the threshold that minimizes the Euclidean distance to the ideal point (precision = 1, recall = 1) on the PR curve.

5.1.2 Soft labels and hard labels

In binary classification tasks, once pseudo-labels are selected, we still need to decide how to use the model's predictions as targets during training. There are two main approaches: hard pseudo-labeling and soft pseudo-labeling [18, 37].

Hard pseudo-labeling transforms the probability into a strict binary label. If the predicted probability is greater than 0.5, the target becomes 1; otherwise, it becomes 0. This method forces the model to learn hard decisions rather than probabilities. Even if the original model was only slightly confident, the model will receive a full 0 or 1 label. This can simplify the learning process but may introduce more noise, especially if the model's predictions are not very reliable.

Soft pseudo-labeling means using the raw probability output by the model (after the sigmoid activation) as the training target. For example, if the model predicts a probability of 0.85 for the positive class, this value is directly used as the target. During training, the model minimizes the difference between its own output and this soft target, usually using binary cross-entropy. This method allows the model to learn not only the predicted class but also the model's level of confidence in that prediction, providing a smoother and potentially more robust training signal.

In our work, the method for selecting which pseudo-labels to accept (based on confidence thresholds or proportions) is handled separately. Here, hard and soft pseudo-labeling only refer to how we transform the model's output into a training target once a pseudo-label has been accepted.

5.1.3 Stopping criteria

In self-training, it is important to decide when to stop the process. This helps avoid wasting time and prevents the model from learning bad or useless information. There are several simple ways to decide when to stop.

The most basic rule is to stop when there are no more unlabeled data left. In this case, everything has been labeled, and the process is complete.

Another rule is based on how many new data points are added at each step. If the number is very small compared to the current training set, less than 1% for example, it may not be worth continuing. For instance, if only 2 new peptides are added while the training set already has 300 000 examples, this will not make a big difference.

Another common method is to use a proportion-based strategy. Here, we always add a fixed percentage of the most confident predictions, for example, the top 1% for each class. In this case, we can decide in advance how many steps to run, and stop when we reach that number.

Finally, we can also stop based on performance. After each step, we check how well the model performs, using a validation set, and compare the results with the previous step. If the PR AUC gets worse, it may mean the new data is hurting the model. In that case, we stop to avoid making things worse.

In this work, we compare the previously described methods with the performance-based approach to evaluate the impact of each stopping criterion.

5.1.4 Double loss

In many self-training approaches, pseudo-labels are treated the same as true labels. They are added directly to the training set, and the loss is calculated on all the data without making any difference between real labels and pseudo-labels. However, this can be a problem because pseudo-labels are not always correct, especially in the early stages of training.

To reduce this risk, a better approach is to separate the loss into two parts [35]: one part for the labeled data and another part for the pseudo-labeled data. The total loss is then a combination of both, where the second part is weighted by a factor α to control its influence:

$$\mathcal{L}_{\text{total}} = \frac{1}{n}\sum_{i=1}^{n}\mathcal{L}(y_i^l, f_\theta(x_i^l)) + \alpha \cdot \frac{1}{m}\sum_{j=1}^{m}\mathcal{L}(\hat{y}_j^u, f_\theta(x_j^u)) \tag{5.1}$$

Where:

- x_i^l and y_i^l are the i-th labeled input and its true label,
- x_j^u and $\hat{y}_j^u$ are the j-th unlabeled input and its pseudo-label,
- $f_\theta(\cdot)$ is the model's prediction,
- $\mathcal{L}$ is the loss function (e.g., binary cross-entropy),
- α is a weight that controls how much we trust the pseudo-labels.

In some works, α increases over time, so that pseudo-labels get more importance as the model improves. But in our case, since there is a few self-training steps, we test fixed values of α from 0.1 to 1.0, increasing by 0.1 each time. This helps us understand how the weight of pseudo-labels affects the final results. We will also perform a grid search over the 10 values within each fold to select the optimal α value.

5.2 Results

All the detailed numerical results can be found in Appendix B.2.

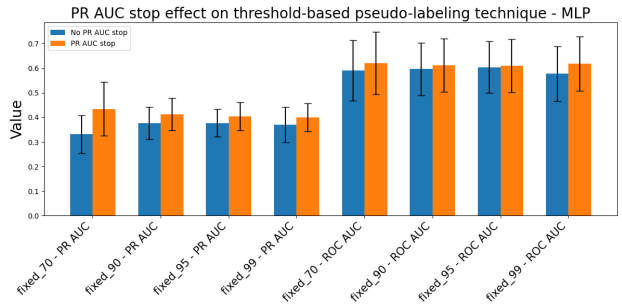
5.2.1 PR AUC as stopping criterion effect

As a reminder, I applied a basic stopping criterion for the self-training loop at first. The process stop under one of two conditions: either when the entire unlabeled dataset had been pseudo-labeled, or when the number of newly pseudo-labeled samples was too small relative to the size of the current training set. In the case of proportion-based pseudo-labeling, where a fixed percentage of pseudo-labeled data is added to the training set at each iteration, it is a bit different, I chose to limit the process to a maximum of five iterations. After conducting experiments using these criteria, I introduced a more dynamic stopping condition based on performance monitoring. Specifically, I monitored the evolution of the PR AUC score on the validation set and stopped the process when the addition of new pseudo-labeled data began to degrade the model's performance. The following plots illustrate the impact of this new stopping criterion in comparison to the basic approach.

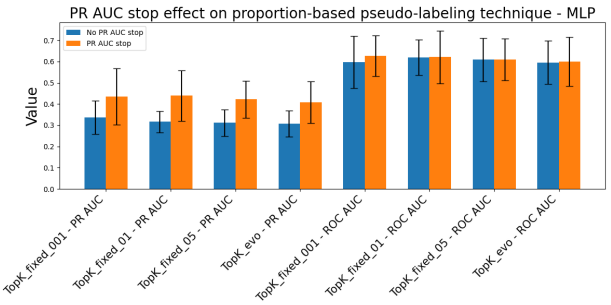
For the MLP model (first three plots of Figure 5.3), applying the PR AUC stopping criterion is clearly beneficial across all pseudo-labeling techniques and for both PR AUC and ROC AUC metrics. The magnitude of improvement varies depending on the pseudo-labeling strategy. For the optimal thresholding technique, the gain is more moderate, with an increase of up to 0.05 in the mean performance. In contrast, for the other two techniques, the improvement reaches up to 0.1, which is notable.

For the BiLSTM model (last three plots of Figure 5.3), the conclusions are less straightforward than for the MLP. As shown in Figure 5.3d, the first two threshold-based values appear to perform better without applying the PR AUC stopping criterion. However, for all other pseudo-labeling techniques, applying the PR AUC stopping criterion consistently leads to better performance.

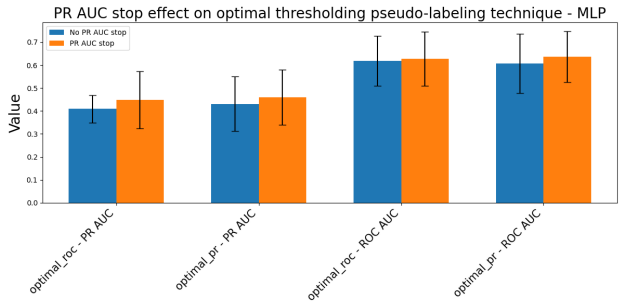
I decided to keep the PR AUC stopping criterion, as it improves the overall performance of the model in nearly all cases for both MLP and BiLSTM.



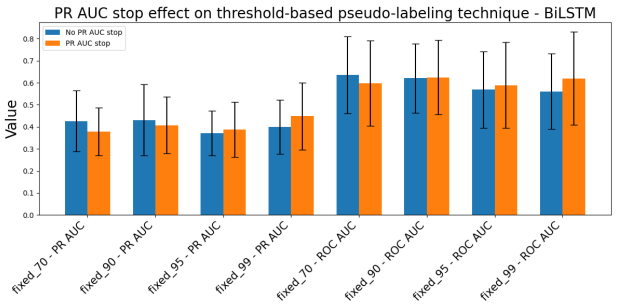
(a) MLP - Threshold-based



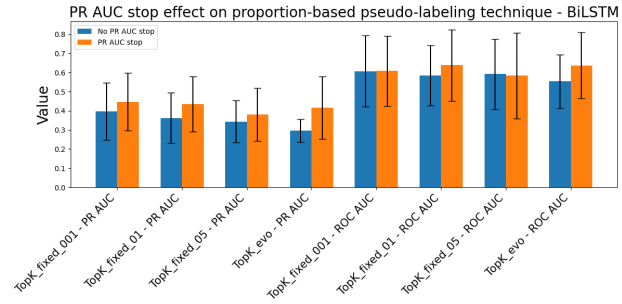
(b) MLP - Proportion-based



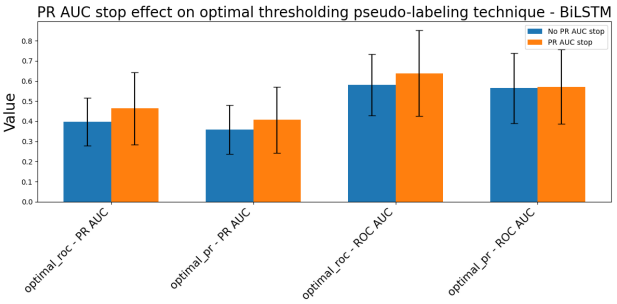
(c) MLP - Optimal thresholding



(d) BiLSTM - Threshold-based



(e) BiLSTM - Proportion-based



(f) BiLSTM - Optimal thresholding

Figure 5.3: PR AUC stop effect on MLP and BiLSTM grouped by pseudo-labeling techniques with hard labels. The first three plots correspond to MLP results, while the last three correspond to BiLSTM results. For MLP, applying the PR AUC stopping criterion is clearly beneficial across all pseudo-labeling techniques and for both PR AUC and ROC AUC metrics. For BiLSTM, the conclusions are less straightforward. The first two threshold-based values perform better without applying the stopping criterion, while for all other techniques, applying the PR AUC stopping criterion consistently improves performance. Overall, applying the performance-based stopping criterion leads to better results than not using it.

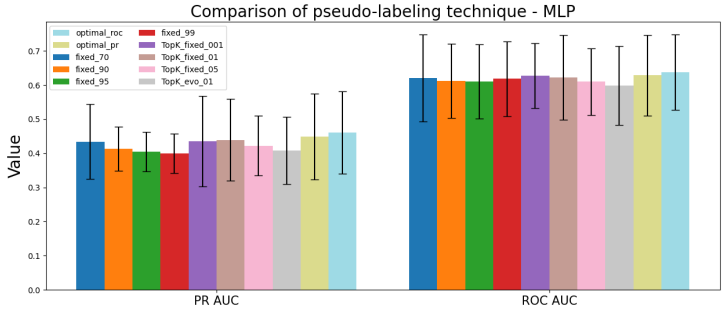
5.2.2 Comparison of pseudo labeling techniques

As a reminder, we explored ten different pseudo-labeling configurations, including three main approaches: threshold-based (referred to as fixed), proportion-based (referred to as TopK), and optimal thresholding (referred to as optimal), each with multiple hyper-parameters tested.

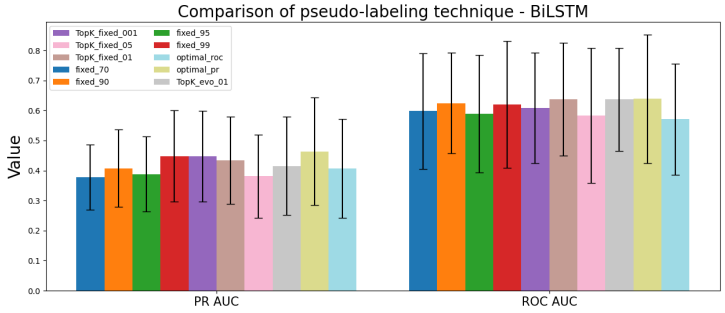
For the MLP model (see Figure 5.4a), it is surprising that within the threshold-based approach, the lowest threshold value (0.7) yields better PR AUC and ROC AUC results. This is counterintuitive, as one would expect that increasing the threshold would lead to better performance. However, this was not observed in our case. For the proportion-based approach, the best results were obtained when using the 0.1 value. In the optimal thresholding approach, the highest performance was achieved by performing a grid search of the thresholds on the ROC curve. The performance values are very close to each other, par-

ticularly among the best configurations. It appears that the optimal ROC configuration performs slightly better than the other top-performing configurations.

For the BiLSTM model (see Figure 5.4b), the results are more intuitive. In the threshold-based approach, the best performance is achieved with the most restrictive threshold. Similarly, for the proportion-based approach, the most restrictive setting yields the best results. Finally, in the optimal thresholding approach, the best performance is obtained by performing a grid search on the PR curve. As previously discussed, PR AUC is considered a more reliable performance metric than ROC AUC in the presence of class imbalance. When comparing the different approaches, it can be seen that optimal PR outperforms the others, even though the values are really close among the top-performing methods.



(a) MLP



(b) BiLSTM

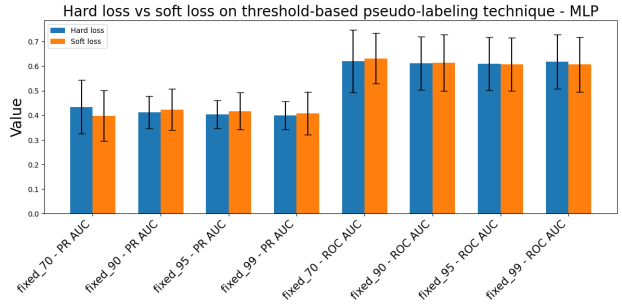
Figure 5.4: Comparison of the different pseudo-labeling techniques using hard labels and with the PR AUC stopping criterion enabled. For MLP, the threshold-based approach with the lowest threshold (70) surprisingly yielded the best results, while for the proportion-based approach, the best performance was obtained with a proportion of 1%. Optimal thresholding performed best when using a grid search on the ROC curve, with very close performance values overall. For BiLSTM, results were more intuitive, with the most restrictive thresholds leading to the best outcomes for both threshold-based and proportion-based approaches. Optimal thresholding achieved the highest performance when applying a grid search on the PR curve. Optimal PR slightly outperformed the other top configurations despite the close performance values overall.

5.2.3 Hard vs soft labels

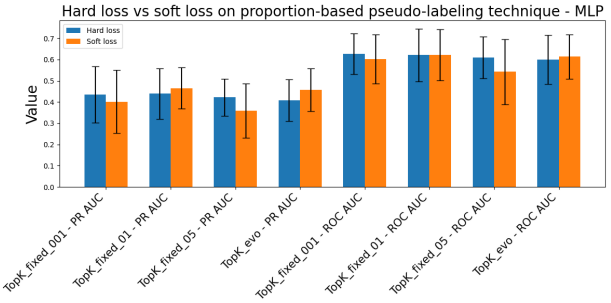
As a reminder, when incorporating pseudo-labels, there are two approaches: converting those probabilities into a discrete class (in the binary case: 1 or 0), referred to as hard labels, or keeping the probabilities output by the model, referred to as soft labels. In this way, the data from true labels differ from pseudo-labels, as the target values are different. Additionally, the soft label approach can help the model learn that some pseudo-labels are less certain, since the probabilities reflect lower confidence. In contrast, with

the hard label approach, a pseudo-label with low probability might be converted into a definitive 1 (or 0), giving the model a much more confident target despite the underlying uncertainty.

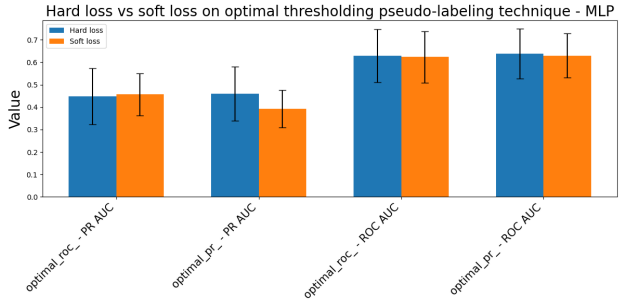
The effect is not straightforward because the results vary depending on the model and the pseudo-labeling configuration. For the MLP, in the threshold-based approach (Figure 5.5a), soft labels perform better in 3 out of 4 configurations for both PR AUC and ROC AUC. In the proportion-based approach (Figure 5.5b), soft labels outperform in 2 cases, while hard labels perform better in the other 2. In the optimal thresholding approach (Figure 5.5c), the results are also inconclusive, with soft labels being better in one case and hard labels in the other. For the BiLSTM, the situation is similar. In the threshold-based approach (Figure 5.5d), soft labels perform better in 3 out of 4 configurations. However, in the proportion-based configuration (Figure 5.5e), hard labels perform better in 3 out of 4 configurations. In the optimal thresholding approach (Figure 5.5f), hard labels show better performance. In conclusion, it is not possible to definitively determine the superiority of soft or hard labels, as performance depends on both the pseudo-labeling configuration and the model. Therefore, to continue the self-training experiments, I decided to evaluate both soft and hard label approaches while also incorporating the double loss strategy as described in the methodology. This will allow us to compare the impact of double loss across both models and both soft and hard label strategies.



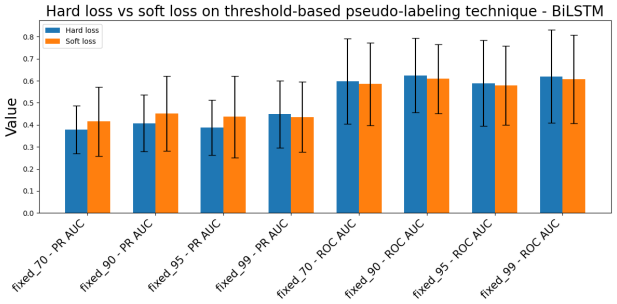
(a) MLP - Threshold-based



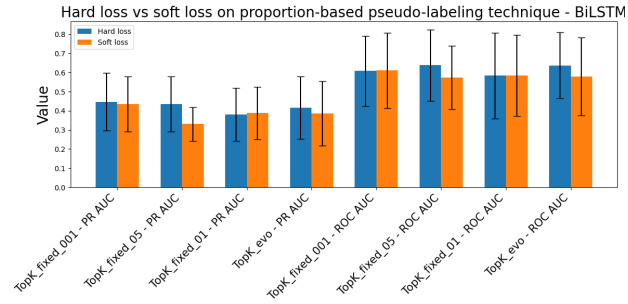
(b) MLP - Proportion-based



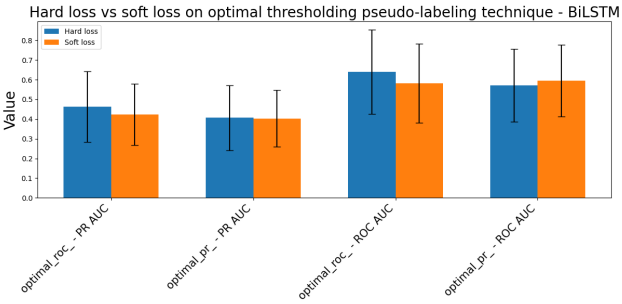
(c) MLP - Optimal thresholding



(d) BiLSTM - Threshold-based



(e) BiLSTM - Proportion-based



(f) BiLSTM - Optimal thresholding

Figure 5.5: Hard loss vs Soft loss on MLP and BiLSTM grouped by pseudo-labeling techniques. The first three plots correspond to MLP results, while the last three correspond to BiLSTM results. The effect is not straightforward, as performance varies by model and pseudo-labeling configuration. For the MLP, soft labels perform better in most threshold-based configurations (3 out of 4) and in half of the proportion-based configurations, while results are mixed for optimal thresholding. For the BiLSTM, soft labels dominate in threshold-based settings (3 out of 4), whereas hard labels outperform in most proportion-based configurations (3 out of 4) and in the optimal thresholding approach. No clear superiority of soft or hard labels can be made, as results vary depending on the model and pseudo-labeling configuration.

5.2.4 Impact of double loss

As a reminder, the impact of the loss related with pseudo-labels can be mitigated by introducing a regularization weight, referred to as α. We tested ten different values of α, ranging from 0.1 to 1.0 in increments of 0.1. This approach treats true labels and pseudo-labels as different. A lower α places greater penalty on misclassifying true labels compared to pseudo-labels, which is the behavior we wanted to evaluate.

By analyzing Figure 5.6, it can be observed that, within the proportion-based approach (Figure 5.6e to Figure 5.6h), MLP models generally outperform BiLSTM models. Moreover, the use of hard labels consistently provides better results than the soft label variant across most α values. For the threshold-based approach (Figure 5.6a to Figure 5.6d), the results vary depending on the configuration: the best model

depends on the chosen threshold (BiLSTM with soft labels for a threshold of 70%, MLP with hard labels for thresholds of 90% and 95%, and BiLSTM with hard labels for a threshold of 99%). In the optimal PR setting (Figure 5.6i and Figure 5.6j), MLP outperforms BiLSTM for most α values. In contrast, in the optimal ROC setting, MLP clearly emerges as the best option.

It can also be noted that the combination of double loss with hard labels appears to be more favorable than with soft labels. In most cases, and for a given model, the hard label configuration outperforms the soft label one. The only exceptions occur in the threshold-based configuration (70, 90, and 95) for BiLSTM, and in the optimal PR, TopK 05, and fixed 90 configurations for MLP, each pseudo-labeling approach having at least one configuration where soft labels outperform hard labels. This is particularly interesting, as analyzing the effect of hard vs soft labels alone made it difficult to determine which approach was more beneficial.

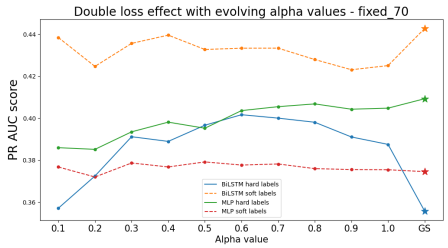
We also evaluated the performance of selecting the α value using a grid search. This method consists of choosing the α that achieves the highest PR AUC score on the validation set. Its performance is indicated by a star in the graphs and is placed at the last position on the x-axis, labeled GS (for GridSearch). In most situations, the grid search strategy yields a PR AUC score that is comparable or higher than the best performing value of α.

We also investigated the α values yielding the highest PR AUC scores for each model and configuration. By counting how often each α produced the best score, we obtained the following distribution:

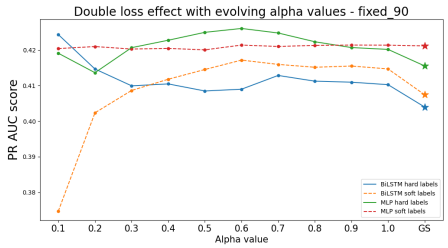
α	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
count	6	9	4	4	4	5	0	2	1	5

Table 5.1: Distribution of best-performing α values across all runs.

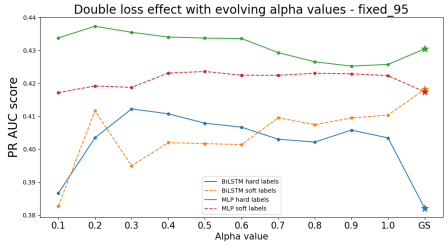
It is clear that the majority of models favor an α value of 0.5 or lower, with 82.5% of the best results falling within this range. The most frequently selected α value is 0.2, which corresponds to an important regularization. This observation validates the underlying intuition behind the use of the double loss approach: penalizing the error on true labels more heavily appears to yield better performance when α is small, as it increases the regularization effect.



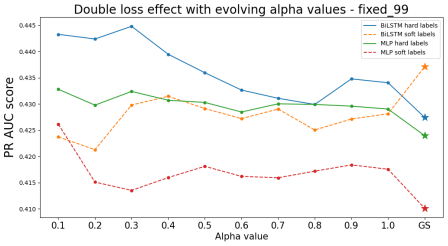
(a) Threshold-based - Fixed 70



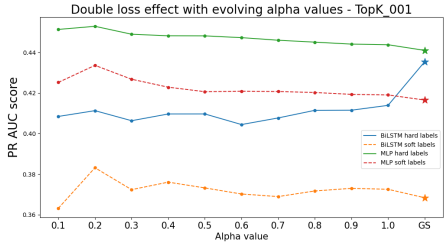
(b) Threshold-based - Fixed 90



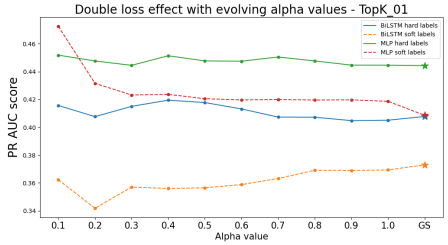
(c) Threshold-based - Fixed 95



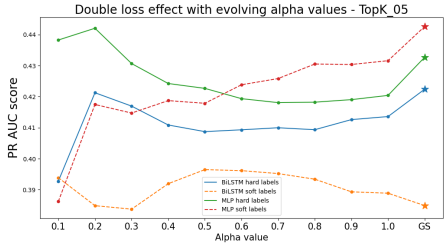
(d) Threshold-based - Fixed 99



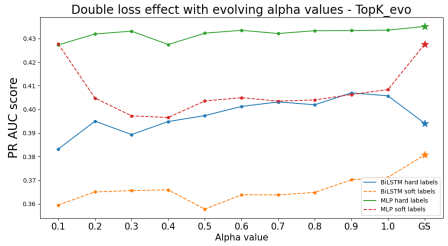
(e) Proportion-based - TopK 001



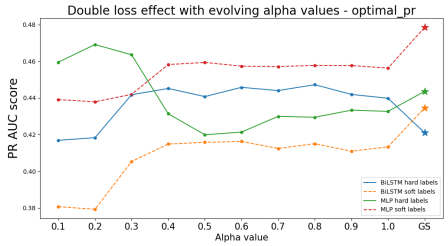
(f) Proportion-based - TopK 01



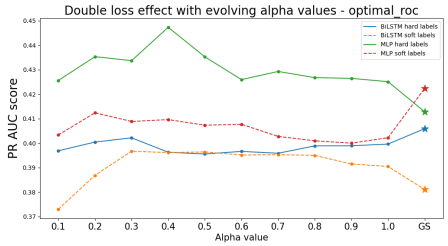
(g) Proportion-based - TopK 05



(h) Proportion-based - TopK evo



(i) Optimal thresholding - Optimal PR



(j) Optimal thresholding - Optimal ROC

Figure 5.6: Mean PR AUC scores obtained when training with a double loss, across pseudo-labeling strategies and configurations, with different α values. For each setting, the result obtained via grid search is highlighted with a star and displayed as the final point on the x-axis (labeled GS). In the proportion-based setting, MLP models generally outperform BiLSTM. The threshold-based approach shows varying results depending on the chosen threshold (BiLSTM with soft labels for a threshold of 70%, MLP with hard labels for thresholds of 90% and 95%, and BiLSTM with hard labels for a threshold of 99%). In the optimal PR setting, MLP models often perform better than BiLSTM, while in the optimal ROC setting, MLP consistently achieves the best results. Hard labels show better performance than soft labels across most α values and configurations. Moreover, applying grid search can occasionally improve performance, only 14 out of 40 configurations outperformed the best fixed α. This suggests that grid search is not consistently beneficial and should not be used as the only strategy.

5.2.5 Model's performance comparison

A total of 250 different models were tested during the self-training phase for both the MLP and BiLSTM architectures, resulting in 500 models in total. Table 5.2 shows that only 21.6% of the MLP models improved upon the baseline performance (i.e., a MLP trained using only the true labels) in terms of PR AUC. For the BiLSTM models, the proportion is slightly higher, at 23.6%. These results suggest that self-training often degrades the model's ability to generalize patterns compared to the baseline. However, since some models do achieve better performance than the baseline, this implies that a subset of models successfully learn meaningful patterns, which we will further investigates.

Table 5.2: Number of self-trained models that outperform or underperform the baseline PR AUC value.

Model	Better	Worse	%
MLP	54	196	21.6%
BiLSTM	59	191	23.6%

Table 5.3 presents the five best MLP models and the five best BiLSTM models selected from all configurations, including pseudo-labeling techniques, hard or soft label settings, and the presence or absence of the PR AUC stopping criterion. The first two rows correspond to the best MLP and BiLSTM models obtained from the supervised learning task, providing a baseline to assess whether self-training leads to improved model performance.

It can be observed that, compared to the baseline, the five best models achieved improved performance in both PR AUC and ROC AUC metrics. This is encouraging, as it suggests that the application of self-training was beneficial. For both model types, the best configuration yielded an improvement of 0.05 in PR AUC over the MLP baseline and of 0.04 in PR AUC over the BiLSTM baseline. Moving on to the comparison between MLP and BiLSTM, MLP consistently remains the best model, which is consistent with its superior performance in the supervised learning part. Additionally, five out of the ten selected models use the optimal thresholding approach as the pseudo-labeling technique, which is consistent when we compared the different pseudo-labeling strategies. In contrast, there is a clear under-representation of models using the double loss strategy, indicating that models trained without double loss generally performed better. This is somewhat surprising, as the intention was to penalize the misclassification of true labels more heavily. However, it appears that, in general, the models performed better without this additional regularization. Finally, regarding the comparison between hard and soft labels, no definitive conclusion can be made: five models used soft labels, and five used hard labels, resulting in only a slight superiority of hard label configurations. We can conclude that the best-performing model is the MLP when using the optimal thresholding strategy on the PR curve for the pseudo-labeling technique, applying soft labels, using the PR AUC as a stopping criterion, and performing a grid search to find the optimal α value for the double loss.

Despite the improvement, it is not sufficient to consider the solution fully satisfactory. Although the metrics have increased, the PR AUC and ROC AUC values remain relatively low, indicating that the models still struggle to identify highly convincing and meaningful patterns for predicting the quantotypic property.

Table 5.3: Comparison of the five best MLP models with the five best BiLSTM models. The first two rows serve as a reminder of the best supervised learning model. The first five columns describe the configuration of each model: the first column indicates the model name; PL specifies the pseudo-labeling technique used; LT refers to the label type, with hard (H) or soft (S) labels; PR indicates whether the PR AUC STOP criterion is enabled; and DL refers to the double loss strategy. For DL, a value of '/' means double loss was not applied, 'GS' indicates that a grid search was performed over 10 α values, and a numeric value represents a fixed α. Self-training led to overall performance improvements over the baseline for both architectures, with MLP consistently outperforming BiLSTM. The optimal thresholding approach was frequently selected. On the other hand, models employing the double loss strategy were under-represented among the top models, suggesting that additional penalization of true label misclassification did not yield consistent benefits. No clear advantage was observed between soft and hard labels. However, despite the improvements, overall PR AUC and ROC AUC values remain relatively low, indicating limited model ability to capture highly predictive patterns for the quantotypic property.

Model	PL	LT	PR	DL	Accuracy	Precision	Recall	F1	PR AUC	ROC AUC
MLP_ES_N_W					0.67 ± 0.09	0.44 ± 0.30	0.32 ± 0.18	0.30 ± 0.12	0.43 ± 0.10	0.61 ± 0.12
BiLSTM_NES_N_O					0.65 ± 0.12	0.32 ± 0.15	0.31 ± 0.23	0.30 ± 0.16	0.42 ± 0.15	0.60 ± 0.19
MLP	Opt - pr	S	Yes	GS	0.43 ± 0.24	0.31 ± 0.12	0.78 ± 0.30	0.40 ± 0.11	0.48 ± 0.09	0.64 ± 0.11
MLP	TopK fixed - 0.01	S	Yes	0.1	0.67 ± 0.14	0.36 ± 0.23	0.21 ± 0.10	0.26 ± 0.14	0.47 ± 0.13	0.64 ± 0.14
MLP	TopK fixed - 0.01	S	Yes	/	0.41 ± 0.20	0.28 ± 0.05	0.72 ± 0.34	0.36 ± 0.05	0.47 ± 0.10	0.62 ± 0.12
MLP	Opt - roc	H	Yes	/	0.64 ± 0.11	0.28 ± 0.11	0.29 ± 0.17	0.28 ± 0.13	0.46 ± 0.12	0.64 ± 0.11
MLP	Opt - pr	S	Yes	/	0.43 ± 0.24	0.31 ± 0.12	0.78 ± 0.30	0.40 ± 0.11	0.46 ± 0.09	0.62 ± 0.11
BiLSTM	Opt - pr	H	Yes	/	0.69 ± 0.14	0.39 ± 0.20	0.26 ± 0.19	0.30 ± 0.19	0.46 ± 0.18	0.64 ± 0.21
BiLSTM	fixed - 0.99	H	Yes	/	0.68 ± 0.14	0.47 ± 0.31	0.34 ± 0.27	0.33 ± 0.20	0.45 ± 0.15	0.62 ± 0.21
BiLSTM	TopK fixed - 0.001	H	Yes	/	0.66 ± 0.14	0.47 ± 0.31	0.40 ± 0.28	0.35 ± 0.20	0.45 ± 0.15	0.61 ± 0.18
BiLSTM	fixed - 0.9	S	Yes	/	0.72 ± 0.09	0.39 ± 0.20	0.32 ± 0.19	0.35 ± 0.19	0.45 ± 0.17	0.61 ± 0.16
BiLSTM	Opt - pr	H	Yes	0.4	0.69 ± 0.14	0.34 ± 0.28	0.27 ± 0.25	0.28 ± 0.23	0.45 ± 0.17	0.61 ± 0.19

Chapter 6

Discussion

This chapter aims to critically examine the limitations and challenges encountered in this study, as well as to propose potential directions for improvement. While the goal was to predict the quantotypic property of peptides using only their amino acid sequence, the results were not as promising as expected. Several factors may have contributed to this outcome, including the limited amount of training data, the chosen labeling method, and the use of raw sequence information without additional biochemical context. By analyzing these aspects, we can better understand the weaknesses of the current approach and identify opportunities for future work to improve model performance and reliability.

6.1 Quantity of data

One possible explanation for the disappointing results is the limited amount of data. In artificial intelligence, data plays a key role in achieving good performance. Traditional machine learning algorithms such as Random Forest or XGBoost usually require less data to perform well, mainly because they are simpler and less data-hungry than deep learning models. Deep learning models, on the other hand, typically require much more data due to their large number of parameters [38].

AlphaPeptDeep models have around 4 million parameters, which means they need a large dataset to properly update the weights during training. In the AlphaPeptDeep paper [31], they mention using approximately 40 million spectra for training the MS2 models and about 500,000 values for the RT and CCS models.

In contrast, our dataset only contains 155 peptides. To follow a proper training protocol, this dataset is split into training, validation, and test sets. The training set includes about 64% of the data, which corresponds to roughly 100 data points. When the task is as complex as predicting a peptide's quantotypic property based solely on its amino acid sequence, such a small training set is likely not sufficient to expect good results. Deep learning models need enough examples to learn meaningful patterns.

In other machine learning domains, data augmentation techniques are often used to artificially increase the size of the training set. However, in the context of peptide sequences, generating artificial data is not straightforward. Randomly modifying or creating new sequences could result in biologically invalid peptides.

6.2 Labeling technique

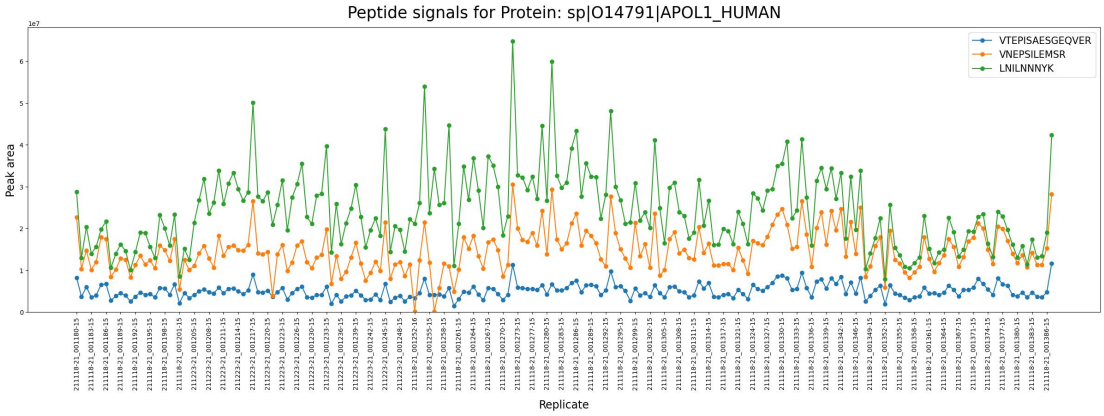
Another possible reason for the disappointing results could be the way the peptides were labeled. Although Worboys *et al.* showed that using the correlation matrix of peptide signals can help identify quantotypic

peptides, there are still limitations to this approach. We have access to another dataset similar to SPARE, called PREMIDID, which comes from a different study. This dataset also contains 230 peptides, but each peptide is measured across 58 samples. The preprocessing steps used for this dataset are the same as those described in Chapter 3.

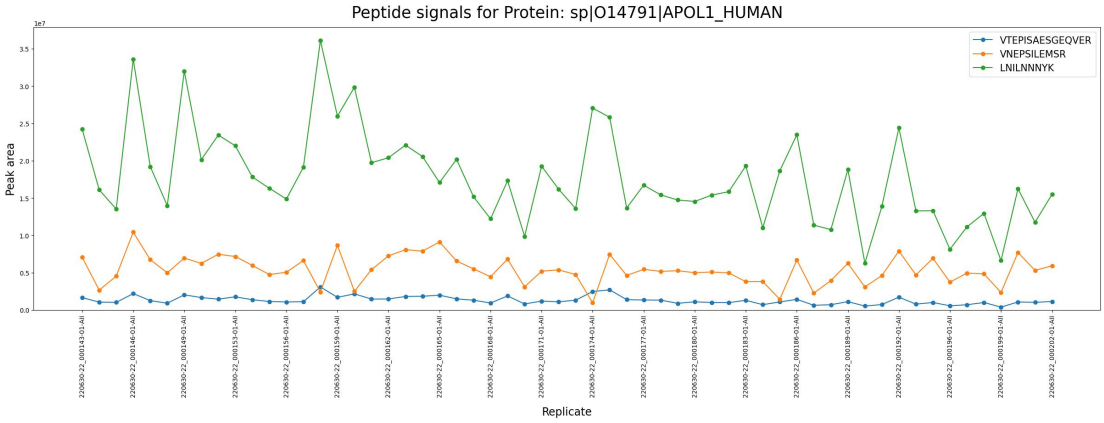
This allows us to compare the peptide signals and the resulting correlation matrices for the same protein across both datasets (SPARE vs PREMIDID) (see Figure 6.1 and Figure 6.2). We can observe two different behaviors when comparing the peptide signals and their corresponding correlation matrices between the SPARE and PREMIDID datasets. These differences are likely due to experimental variability introduced during data acquisition. In proteomics, factors like chemical environment, temperature, humidity, or even slight variations in sample preparation can influence peptide ionization and detection. As a result, the same peptide may show different intensity patterns across datasets, even if the biological context is similar. These variations impact the correlation between peptides and can lead to inconsistent labeling of quantotypic peptides when using correlation-based approaches.

Based on the threshold defined in Chapter 3 (0.9), the correlation matrix from the SPARE dataset would label the peptides VNEPSILEMSR and VTEPISAESGEQVER as quantotypic, and the other peptide as non-quantotypic. In addition, in the SPARE dataset, the strongest correlation is between VNEPSILEMSR and VTEPISAESGEQVER, while in the PREMIDID dataset, the highest correlation is between two different peptides: VTEPISAESGEQVER and LNILNNNYK. This shows that the labeling can change depending on the dataset. Figure 6.3 also highlights that the distributions of correlation values differ between the two datasets. In the SPARE dataset, 155 peptides were retained after labeling, whereas only 98 peptides remained in the PREMIDID dataset. This inconsistency suggests a difference between the datasets, although such variation should not exist as a peptide that is truly quantotypic should be consistently labeled across different datasets. Of all the peptides, 85 are present in both datasets, 13 are unique to PREMIDID, and 70 are unique to SPARE. Among the 85 shared peptides, 59 are labeled as quantotypic in both datasets, 12 are labeled as non-quantotypic in both, and 14 receive conflicting labels (being considered quantotypic in one dataset and non-quantotypic in the other).

This kind of variation creates inconsistencies in the labeling process and raises questions about how reliable and meaningful the current labeling method really is.



(a) SPARE



(b) PREMIDID

Figure 6.1: Peptide signals for the protein APOL1.

Correlation matrix for Protein: sp|O14791|APOL1_HUMAN

	VTEPISAESGEQVER	VNEPSILEMSR	LNILNNNYK
VTEPISAESGEQVER	1.0	0.93148	0.75129
VNEPSILEMSR	0.93148	1.0	0.75323
LNILNNNYK	0.75129	0.75323	1.0

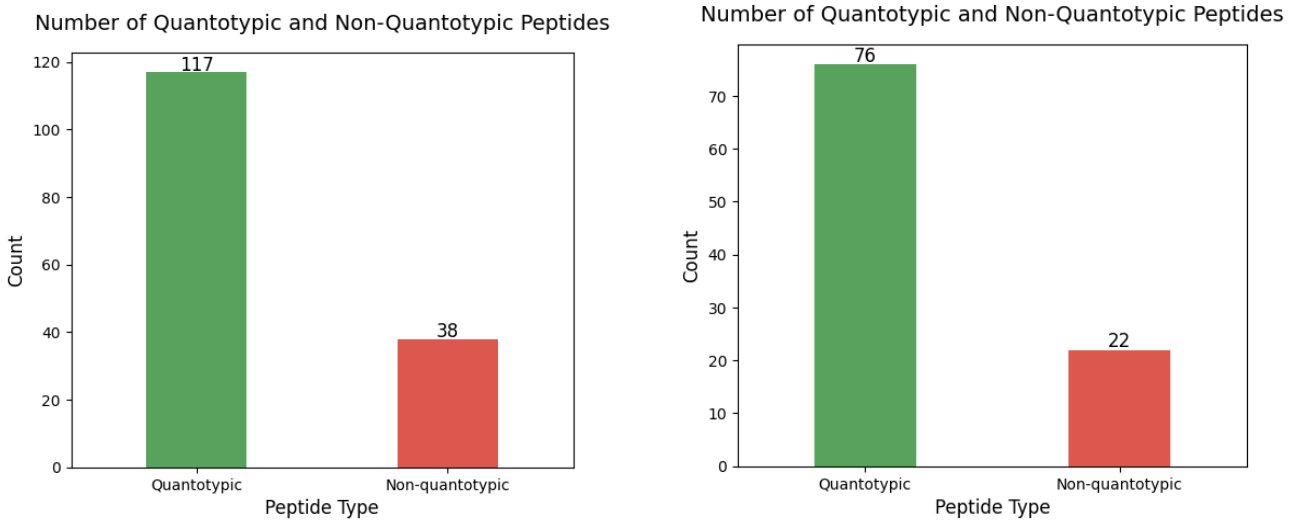
(a) SPARE

Correlation matrix for Protein: sp|O14791|APOL1_HUMAN

	VTEPISAESGEQVER	VNEPSILEMSR	LNILNNNYK
VTEPISAESGEQVER	1.0	0.55077	0.89123
VNEPSILEMSR	0.55077	1.0	0.53681
LNILNNNYK	0.89123	0.53681	1.0

(b) PREMIDID

Figure 6.2: Correlation matrices derived from the peptide signals for the protein APOL1.



(a) SPARE (b) PREMIDID

Figure 6.3: Distribution of quantotypic and non-quantotypic peptides for both dataset applying a 0.9 threshold to process the correlation matrices.

6.3 Only using sequence as input

Another limitation of this work may come from the fact that we chose to use only the raw peptide sequence as input. Although there are likely meaningful patterns hidden in the sequences, the models receive very limited information to support accurate predictions. Amino acids have specific physicochemical properties such as hydrophobicity, charge, size, and polarity which influence peptide behavior in biological and experimental contexts. These properties are available in databases such as AAindex [39], as used in the CONSeQuence paper [11], where the authors extracted physicochemical features from peptide sequences.

In that study, a feature analysis was performed to remove redundancy and retain only the most informative features, which helped reduce noise and improve model performance. A similar strategy could be beneficial in our work, especially since our results show that sequence information alone may not be sufficient for accurate quantotypic peptide classification. Moreover, physicochemical properties are inherently interpretable, which can help researchers understand why certain peptides are predicted as quantotypic, offering potential biological insights.

6.4 Future work

As discussed earlier, using only the peptide sequence as input may not provide enough information for accurate prediction. Therefore, one possible direction for future work would be to enhance the input data by incorporating additional features related to the physicochemical properties of the amino acids. These properties can offer valuable insights into peptide behavior and improve model performance when used alongside sequence-based representations.

Another important aspect to consider is the labeling process. The current method for defining peptides as quantotypic or non-quantotypic relies on correlation thresholds, which may introduce inconsistencies across datasets. Exploring alternative labeling strategies could help build a more reliable ground truth. Improving both the input representation and the quality of the labels would likely lead to more robust and generalizable models.

Lastly, this study was conducted using data from targeted proteomics experiments, where only selected peptides are measured across a limited number of samples. While this approach provides high precision, it also restricts the amount of available data. Moving toward untargeted proteomics could help address this limitation. Untargeted approaches generate much larger datasets by measuring a broad range of peptides without prior selection, which could provide a richer foundation for training AI models requiring large dataset such as deep learning architectures. However, untargeted data typically requires much more cleaning, processing, and filtering before it can be used effectively, as it is generally less precise and reliable than data from targeted proteomics. Nevertheless, the significantly larger volume of data available in untargeted approaches could mitigate these challenges when training AI models requiring large datasets.

Chapter 7

Conclusions

In the field of proteomics, several solutions exist to predict proteotypic properties, but there is still a critical lack of tools capable of predicting the quantotypic property of peptides. Such a tool could reduce the dependence on costly and equipment-intensive a posteriori methods. This work focused on developing an artificial intelligence approach to explore the feasibility of such a prediction task. We deliberately limited the input to peptide sequences alone in order to train the models. Our research was conducted within the framework of targeted proteomics, which, although providing fewer data compared to untargeted proteomics, offers greater reliability. The project was divided into two main parts. The first part involved the use of a labeled dataset to perform supervised learning. The second part explored the use of self-training methods, addressing the limited availability of annotated data.

In the first part of the study, we explored various artificial intelligence models, including machine learning models (Random Forest and XGBoost) and deep learning architectures (MLP and BiLSTM). We investigated both models trained from scratch and transfer learning approaches, leveraging pre-trained models initially developed for different tasks but still based on peptide (AlphaPeptDeep) or protein (ESM) data. The dataset presented a significant class imbalance problem. To address this, we tested cost-sensitive learning and oversampling techniques, particularly RandomOverSampling. We also evaluated two transfer learning strategies: fine-tuning (updating the weights of the pre-trained models) and feature extraction (freezing the pre-trained weights). The results showed that deep learning models trained from scratch outperformed both transfer learning models and traditional machine learning algorithms. The transfer learning results were particularly disappointing, and the machine learning models did not exceed the performance of the deep learning approaches. Overall, the performance of all models remained modest, with low PR AUC and ROC AUC values.

In the second part of the study, we aimed to incorporate additional data by applying self-training methods. We experimented with several pseudo-labeling techniques, including the threshold-based approach, the proportion-based approach, and the optimal thresholding approach. We also implemented a stopping criterion based on PR AUC performance. Both hard labels and soft labels were tested to assess their impact on model performance. Additionally, we explored splitting the loss function by introducing a regularization term on the pseudo-label loss, with the objective of penalizing misclassification of true labels more strongly. Although the best models showed some improvement in metrics such as PR AUC and ROC AUC, the overall performance remained low and unsatisfactory.

Several factors may explain the poor performance of the models. First, the limited amount of data is a major challenge, especially for deep learning, as neural networks require large datasets to effectively learn and identify relevant patterns, particularly in such a complex prediction task. Another critical factor may be the reliability of the dataset. The labels were generated based on signal measurements, and inconsistencies were observed when comparing two similar datasets. Specifically, the same peptides across different

datasets were sometimes assigned conflicting labels, indicating inconsistency in the labeling process.

For future work, increasing the size of the dataset could help improve the models. It may also be useful to explore different ways of labeling peptides to avoid differences between datasets. In this study, only the peptide sequence was used. Future work could combine this with additional information, such as the physicochemical properties of amino acids, by using available databases. Another idea would be to test datasets from untargeted proteomics, as they contain much larger amounts of data. With careful selection of good quality data, it may be possible to build a more reliable dataset.

Despite the modest results, this work represents a first step towards the development of predictive tools for the quantotypic property of peptides and provides valuable insights for future research in this field.

Bibliography

[1] Essential amino acids: Chart, abbreviations and structure. https://www.technologynetworks.com/applied-sciences/articles/essential-amino-acids-chart-abbreviations-and-structure-324357. Accessed: 2025-02-21.

[2] B. Chen, K. A. Brown, Z. Lin et Y. Ge : Top-down proteomics: Ready for prime time? *Analytical Chemistry*, 90(1):110–127, 2018.

[3] Dario Di Silvestre, Francesca Brambilla, Giulio Agnetti et Pierluigi Mauri : *Bottom-Up Proteomics*, pages 155–185. 09 2016.

[4] Vinzenz Lange, P. Picotti, Bruno Domon et R. Aebersold : Selected reaction monitoring for quantitative proteomics: A tutorial. *Mol. Syst. Biol.*, 4, 10 2008.

[5] Cristina Chiva et Eduard Sabido : On peptide selection for targeted protein quantitation. *Journal of Proteome Research*, 16, 01 2017.

[6] Jonathan Worboys, John Sinclair, Yinyin Yuan et Claus Jørgensen : Systematic evaluation of quantotypic peptides for targeted analysis of the human kinome. *Nature methods*, 11, 08 2014.

[7] Vincent A Fusaro, D R Mani, Jill P Mesirov et Steven A Carr : Prediction of high-responding peptides for targeted protein assays by mass spectrometry. *Nature Biotechnology*, 27(2):190–198, 2009.

[8] K Demeure, E Duriez, B Domon et S P Niclou : Peptidemanager: a peptide selection tool for targeted proteomic studies involving mixed samples from different species. *Frontiers in Genetics*, 5:305, 2014.

[9] Brian Searle, Jarrett Egertson, James Bollinger et Andrew Stergachis : Using data independent acquisition (dia) to model high-responding peptides for targeted proteomics experiments. *Molecular cellular proteomics : MCP*, 14, 06 2015.

[10] Andrew B Stergachis, Brendan MacLean, Kristen Lee, John A Stamatoyannopoulos et Michael J MacCoss : Rapid empirical discovery of optimal peptides for targeted proteomics. *Nature Methods*, 8(12):1041–1043, 2011.

[11] Claire Eyers, Craig Lawless, David Wedge, King Lau, Simon Gaskell et Simon Hubbard : Consequence: Prediction of reference peptides for absolute quantitative proteomics using consensus machine learning approaches. *Molecular cellular proteomics : MCP*, 10:M110.003384, 08 2011.

[12] Nicolas Pierre, Vân Huynh-Thu, Dominique Baiwir, Gabriel Mazzucchelli, Maximilien Fléron, Lisette Trzpiot, Gauthier Eppe, Edwin De Pauw, David Laharie, Jack Satsangi, Peter Bossuyt, Lucine Vuitton, Sophie Vieujean, Jean-Frédéric Colombel, Marie-Alice Meuwis et Ezekiel Louis : External validation of serum biomarkers predicting short-term and mid/long-term relapse in patients with crohn's disease stopping infliximab. *Gut*, pages gutjnl–2024, 08 2024.

[13] Skyline software. https://skyline.ms/project/home/begin.view. Accessed: 2025-02-21.

[14] Pierre Geurts et Louis Wehenkel : Introduction to machine learning course at uliège, 2023.

[15] Massih-Reza Amini, Vasilii Feofanov, Loïc Pauletto, Liès Hadjadj, Émilie Devijver et Yury Maximov : Self-training: A survey. *Neurocomputing*, 616:128904, février 2025.

[16] A gentle introduction to self-training and semi-supervised learning. https://towardsdatascience.com/a-gentle-introduction-to-self-training-and-semi-supervised-learning-ceee73178b38/. Accessed: 2025-02-07.

[17] Fuzhen Zhuang, Zhiyuan Qi, Keyu Duan, Dongbo Xi, Yongchun Zhu, Hengshu Zhu, Hui Xiong et Qing He : A comprehensive survey on transfer learning. *Proceedings of the IEEE*, 109:43–76, 2020.

[18] Romain Mormont : *Addressing data scarcity with deep transfer learning and self-training in digital pathology*. Thèse de doctorat, ULiège - Université de Liège [Faculté des Sciences Appliquées], Liège, Belgium, September 2022.

[19] A tutorial to understand decision tree ID3 learning algorithm. https://nulpointerexception.com/2017/12/16/a-tutorial-to-understand-decision-tree-id3-learning-algorithm/. Accessed: 2025-02-17.

[20] Random Forest Algorithm in Machine Learning. https://www.geeksforgeeks.org/random-forest-algorithm-in-machine-learning/. Accessed: 2025-03-18.

[21] Alexey Natekin et Alois Knoll : Gradient boosting machines, a tutorial. *Frontiers in neurorobotics*, 7:21, 12 2013.

[22] Xgboost regression in depth. https://medium.com/@fraidoonomarzai99/xgboost-regression-in-depth-cb2b3f623281. Accessed: 2025-02-17.

[23] Gilles Louppe : Deep learning course at uliège, 2024.

[24] Ashwin Ittoo : Web and text analytics course at uliège, 2024.

[25] Multi-layer perceptrons explained and illustrated. https://medium.com/towards-data-science/multi-layer-perceptrons-8d76972afa2b. Accessed: 2025-02-17.

[26] Rishikesh Gawde : Image caption generation methodologies. 04 2021.

[27] What is LSTM – Long Short Term Memory? https://www.geeksforgeeks.org/deep-learning-introduction-to-long-short-term-memory/. Accessed: 2025-02-22.

[28] Jesse Davis et Mark Goadrich : The relationship between precision-recall and roc curves. volume 06, 06 2006.

[29] Nitesh Chawla, Kevin Bowyer, Lawrence Hall et W. Kegelmeyer : Smote: Synthetic minority oversampling technique. *J. Artif. Intell. Res. (JAIR)*, 16:321–357, 06 2002.

[30] Haibo He, Yang Bai, Edwardo Garcia et Shutao Li : Adasyn: Adaptive synthetic sampling approach for imbalanced learning. pages 1322 – 1328, 07 2008.

[31] Wen-Feng Zeng, Xie-Xuan Zhou, Sander Willems, Constantin Ammar, Maria Wahle, Isabell Bludau, Eugenia Voytik, Maximillian Strauss et Matthias Mann : Alphapeptdeep: a modular deep learning framework to predict peptide properties for proteomics. *Nature Communications*, 13, 11 2022.

[32] AlphaPeptDeep github repository. https://github.com/MannLabs/alphapeptdeep. Accessed: 2025-03-12.

[33] ESM github repository. https://github.com/facebookresearch/esm. Accessed: 2025-03-12.

[34] Zeming Lin, Halil Akin, Roshan Rao, Brian Hie, Zhongkai Zhu, Wenting Lu, Nikita Smetanin, Robert Verkuil, Ori Kabeli, Yaniv Shmueli, Allan dos Santos Costa, Maryam Fazel-Zarandi, Tom Sercu, Salvatore Candido et Alexander Rives : Evolutionary-scale prediction of atomic level protein structure with a language model. *bioRxiv*, 2022.

[35] Dong-Hyun Lee : Pseudo-label : The simple and efficient semi-supervised learning method for deep neural networks. *ICML 2013 Workshop : Challenges in Representation Learning (WREPL)*, 07 2013.

[36] Yang Zou, Zhiding Yu, B. V. K. Vijaya Kumar et Jinsong Wang : Domain adaptation for semantic segmentation via class-balanced self-training, 2018.

[37] Aswathnarayan Radhakrishnan, Jim Davis, Zachary Rabin, Benjamin Lewis, Matthew Scherreik et Roman Ilin : Enhancing self-training methods, 01 2023.

[38] Deep learning vs machine learning. https://cloud.google.com/discover/deep-learning-vs-machine-learning. Accessed: 2025-05-08.

[39] Aaindex: Amino acid index database. https://www.genome.jp/aaindex/. Accessed: 2025-05-08.

Appendices

A Hyperparameter tuning for supervised learning task

Here is a detail overview of the hyperparameter tuning for the supervised learning task.

A.1 Machine learning models

A.1.1 Random forest

To train a Random Forest, several hyperparameters must be specified. Two key hyperparameters to fine-tune are `max_depth` and `max_features`.

The `max_depth` parameter limits how deep the decision trees can grow. In our work with 155 data points, the upper bound for `max_depth` was set to 8. This is because random forests typically perform binary splits, and at a depth of 8 the maximum number of terminal nodes would be $2^8 = 256$ (for a balanced tree) which already exceeds our dataset size. Consequently, while we test multiple values for `max_depth`, none extend beyond 8.

The `max_features` parameter restricts the number of features considered when making a split at each node. Below is a detailed explanation of the options we explore:

- **sqrt**: The algorithm considers a number of features equal to the square root of the total number of available features.
- **log2**: The algorithm considers a number of features equal to the base-2 logarithm of the total number of features.
- **None**: No restriction is placed on the number of features; all available features are considered at each split.

`n_estimators` controls the number of trees in the forest, increasing the number of trees generally improves performance. Therefore, we have fixed the `n_estimators` parameter at 1000.

The specific parameter values tested during our grid search were:

Max depth: {None, 1, 3, 5, 6}

Max features: {sqrt, log2, None}

A.1.2 XGBoost

To train an XGBoost model, several hyperparameters must be tuned. For our XGBoost model, two key parameters that we fine-tune are `max_depth` and `learning_rate`.

The `max_depth` parameter controls the maximum depth of the decision trees used in the boosting process. We tested the same parameter values as for the Random Forest model.

The `learning_rate` parameter, also referred to as *eta*, determines the contribution of each tree during the boosting process. A lower learning rate often leads to slower but more refined learning, which can help prevent overfitting, while a higher learning rate accelerates training but may compromise model accuracy.

The different parameter values tested during our grid search were:

Max depth:{None, 1, 3, 5, 6}

Learning rate:{0.01, 0.1, 0.2, 0.3, 0.4}

We fixed the `n_estimators` parameter at 1000.

A.2 Deep Learning Models

A.2.1 MLP

To train our MLP, we modify several hyperparameters. Specifically, we explore the following values:

- **Learning Rate (LR):** {0.0001, 0.001, 0.01, 0.1}
  This hyperparameter determines the step size during gradient descent updates. Lower learning rates (e.g., 0.0001 or 0.001) allow for finer adjustments, whereas higher values (e.g., 0.01 or 0.1) can speed up convergence but might risk overshooting the optimum.
- **Number of Hidden Layers:** {1, 2, 3}
  This parameter sets the number of hidden layers in the network. Increasing the number of layers can enable the model to learn more complex representations, although it may also increase the risk of overfitting, particularly with limited data.
- **Dropout Rate:** {0.2, 0.3}
  Dropout is a regularization technique that randomly deactivates a portion of neurons during training to prevent overfitting. We test dropout rates of 0.2 and 0.3 to determine the optimal balance between retaining sufficient model capacity and ensuring robust regularization.
- **Hidden and Embedding Sizes:** {(64, 64), (128, 128)}
  Here, each tuple corresponds to the hidden layer size and the embedding dimension, respectively. For simplicity, we set both values to be equal and evaluate two different configurations.

This lead to a total of 48 different configurations to be tested.

A.2.2 BiLSTM

To train our BiLSTM model, we adjust several hyperparameters. Specifically, we explore the following values:

- **Learning Rate (LR):** {0.0001, 0.001, 0.01, 0.1}
- **Number of LSTM Layers:** {1, 2, 3}
- **Dropout Rate:** {0.2, 0.3}
- **Hidden and Embedding Sizes:** {(64, 64), (128, 128)}

- **Number of Epochs:** {25, 50, 100}
  This hyperparameter defines the number of complete passes over the training dataset. We vary the number of epochs to assess the impact on training performance and convergence.

This lead to a total of 144 different configurations to be tested.

A.2.3 AlphaPeptDeep and ESM

For the transfer learning models, we reuse the encoder from each architecture and append a feed-forward network (MLP) as a decoder. Consequently, the hyperparameters tuned are nearly identical to those detailed in Section A.2.1, with the additional tuning of the number of epochs. The following values were tested:

- **Learning Rate (LR):** {0.0001, 0.001, 0.01, 0.1}
- **Number of Hidden Layers:** {1, 2, 3}
- **Dropout Rate:** {0.2, 0.3}
- **Hidden and Embedding Sizes:** {(64, 64), (128, 128)}
- **Number of Epochs:** {25, 50, 100}

This lead to a total of 144 different configurations to be tested.

B Tables

This section includes all tables that may be helpful in understanding this thesis, but are not essential and do not provide significant insights.

B.1 Supervised learning

The following tables show the results for the supervised learning task. The accuracy, precision, recall, F1 are computed using a threshold of 0.5 (to classify probabilities). The first four column correspond to the following:

- **Model**: Indicates the name of the model evaluated.
- **ES (Early Stopping)**: Specifies whether early stopping was applied.
  - `Yes` – Early stopping enabled
  - `No` – Early stopping disabled
  - `/` – Early stopping not evaluated
- **TL (Transfer Learning)**: Describes the transfer learning strategy used.
  - `FT` – Fine Tuning
  - `FE` – Feature Extraction
  - `/` – Transfer learning not used
- **CI (Class Imbalance)**: Indicates the class imbalance handling strategy.
  - `N` – No class imbalance technique applied

- `W` – Weighting strategy
- `O` – Oversampling strategy

Model	ES	TL	CI	Accuracy	Precision	Recall	F1	PR AUC	ROC AUC
MLP	No	/	W	0.73 ± 0.06	0.45 ± 0.15	0.24 ± 0.05	0.30 ± 0.07	0.42 ± 0.04	0.60 ± 0.07
MLP	Yes	/	N	0.72 ± 0.05	0.23 ± 0.23	0.12 ± 0.14	0.16 ± 0.17	0.39 ± 0.11	0.61 ± 0.10
MLP	No	/	O	0.65 ± 0.05	0.24 ± 0.08	0.18 ± 0.06	0.20 ± 0.05	0.34 ± 0.08	0.55 ± 0.13
MLP	Yes	/	W	0.67 ± 0.09	0.44 ± 0.30	0.32 ± 0.18	0.30 ± 0.12	0.43 ± 0.10	0.61 ± 0.12
MLP	Yes	/	O	0.61 ± 0.16	0.21 ± 0.14	0.21 ± 0.14	0.20 ± 0.12	0.36 ± 0.05	0.56 ± 0.11
MLP	No	/	N	0.69 ± 0.07	0.33 ± 0.35	0.13 ± 0.08	0.18 ± 0.13	0.38 ± 0.13	0.60 ± 0.13

Table 7.1: MLP intra model comparison

Model	ES	TL	CI	Accuracy	Precision	Recall	F1	PR AUC	ROC AUC
BiLSTM	/	/	W	0.50 ± 0.24	0.31 ± 0.35	0.45 ± 0.45	0.22 ± 0.14	0.37 ± 0.09	0.54 ± 0.12
BiLSTM	/	/	N	0.65 ± 0.07	0.18 ± 0.16	0.21 ± 0.19	0.20 ± 0.17	0.33 ± 0.07	0.54 ± 0.09
BiLSTM	/	/	O	0.65 ± 0.12	0.32 ± 0.15	0.31 ± 0.23	0.30 ± 0.16	0.42 ± 0.15	0.60 ± 0.19

Table 7.2: BiLSTM intra model comparison

Model	ES	TL	CI	Accuracy	Precision	Recall	F1	PR AUC	ROC AUC
ESM	/	FE	O	0.49 ± 0.24	0.13 ± 0.11	0.45 ± 0.42	0.20 ± 0.17	0.25 ± 0.05	0.41 ± 0.12
ESM	/	FT	N	0.72 ± 0.07	0.05 ± 0.11	0.07 ± 0.15	0.06 ± 0.13	0.26 ± 0.08	0.38 ± 0.14
ESM	/	FE	N	0.74 ± 0.04	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.30 ± 0.07	0.49 ± 0.10
ESM	/	FT	O	0.61 ± 0.09	0.18 ± 0.10	0.26 ± 0.20	0.21 ± 0.13	0.31 ± 0.04	0.49 ± 0.05
ESM	/	FE	W	0.65 ± 0.18	0.05 ± 0.10	0.20 ± 0.40	0.08 ± 0.16	0.26 ± 0.03	0.42 ± 0.09
ESM	/	FT	W	0.57 ± 0.19	0.15 ± 0.12	0.36 ± 0.37	0.20 ± 0.17	0.29 ± 0.06	0.50 ± 0.07

Table 7.3: ESM intra model comparison

Model	ES	TL	CI	Accuracy	Precision	Recall	F1	PR AUC	ROC AUC
CCS	/	FE	N	0.75 ± 0.02	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.31 ± 0.05	0.51 ± 0.13
CCS	/	FE	W	0.48 ± 0.21	0.23 ± 0.12	0.64 ± 0.37	0.32 ± 0.16	0.33 ± 0.07	0.48 ± 0.07
CCS	/	FE	O	0.48 ± 0.15	0.26 ± 0.08	0.58 ± 0.21	0.35 ± 0.10	0.29 ± 0.06	0.48 ± 0.16
CCS	/	FT	N	0.72 ± 0.02	0.28 ± 0.15	0.23 ± 0.18	0.25 ± 0.16	0.38 ± 0.08	0.54 ± 0.10
CCS	/	FT	W	0.70 ± 0.07	0.22 ± 0.24	0.11 ± 0.11	0.14 ± 0.15	0.31 ± 0.04	0.50 ± 0.07
CCS	/	FT	O	0.67 ± 0.04	0.03 ± 0.07	0.03 ± 0.05	0.03 ± 0.06	0.28 ± 0.03	0.51 ± 0.06

Table 7.4: CCS intra model comparison

Model	ES	TL	CI	Accuracy	Precision	Recall	F1	PR AUC	ROC AUC
RT	/	FE	N	0.71 ± 0.05	0.03 ± 0.07	0.03 ± 0.06	0.03 ± 0.06	0.31 ± 0.04	0.50 ± 0.06
RT	/	FE	W	0.50 ± 0.17	0.15 ± 0.10	0.40 ± 0.40	0.21 ± 0.16	0.30 ± 0.07	0.50 ± 0.09
RT	/	FE	O	0.65 ± 0.06	0.30 ± 0.05	0.29 ± 0.11	0.28 ± 0.07	0.35 ± 0.06	0.53 ± 0.12
RT	/	FT	N	0.62 ± 0.05	0.22 ± 0.07	0.21 ± 0.06	0.21 ± 0.06	0.30 ± 0.05	0.48 ± 0.06
RT	/	FT	W	0.70 ± 0.07	0.20 ± 0.20	0.08 ± 0.06	0.11 ± 0.09	0.37 ± 0.12	0.56 ± 0.16
RT	/	FT	O	0.68 ± 0.05	0.24 ± 0.15	0.25 ± 0.19	0.24 ± 0.17	0.34 ± 0.16	0.50 ± 0.19

Table 7.5: RT intra model comparison

Model	ES	TL	CI	Accuracy	Precision	Recall	F1	PR AUC	ROC AUC
scratch_RF	/	/	N	0.75 ± 0.04	0.38 ± 0.39	0.10 ± 0.09	0.16 ± 0.14	0.29 ± 0.05	0.53 ± 0.04
scratch_RF	/	/	W	0.74 ± 0.04	0.22 ± 0.27	0.10 ± 0.15	0.14 ± 0.18	0.28 ± 0.06	0.53 ± 0.07
scratch_RF	/	/	O	0.65 ± 0.14	0.38 ± 0.20	0.41 ± 0.16	0.37 ± 0.13	0.31 ± 0.08	0.56 ± 0.09
scratch_XGB	/	/	N	0.71 ± 0.06	0.29 ± 0.20	0.21 ± 0.16	0.24 ± 0.16	0.28 ± 0.05	0.54 ± 0.07
scratch_XGB	/	/	W	0.64 ± 0.08	0.31 ± 0.11	0.38 ± 0.16	0.33 ± 0.12	0.28 ± 0.03	0.55 ± 0.07
scratch_XGB	/	/	O	0.65 ± 0.06	0.31 ± 0.06	0.34 ± 0.10	0.32 ± 0.05	0.27 ± 0.02	0.54 ± 0.04

Table 7.6: scratch intra model comparison

Model	ES	TL	CI	Accuracy	Precision	Recall	F1	PR AUC	ROC AUC
RT_XGB	/	/	N	0.74 ± 0.07	0.55 ± 0.40	0.13 ± 0.08	0.18 ± 0.09	0.29 ± 0.05	0.53 ± 0.04
RT_RF	/	/	N	0.74 ± 0.07	0.57 ± 0.42	0.13 ± 0.08	0.20 ± 0.12	0.30 ± 0.06	0.53 ± 0.06
RT_RF	/	/	O	0.75 ± 0.02	0.37 ± 0.37	0.07 ± 0.06	0.12 ± 0.10	0.27 ± 0.05	0.52 ± 0.03
RT_XGB	/	/	O	0.74 ± 0.07	0.55 ± 0.40	0.13 ± 0.08	0.18 ± 0.09	0.29 ± 0.05	0.53 ± 0.04
RT_RF	/	/	W	0.71 ± 0.07	0.32 ± 0.37	0.16 ± 0.16	0.17 ± 0.15	0.27 ± 0.04	0.52 ± 0.05
RT_XGB	/	/	W	0.74 ± 0.07	0.55 ± 0.40	0.13 ± 0.08	0.18 ± 0.09	0.29 ± 0.05	0.53 ± 0.04

Table 7.7: RT_ML intra model comparison

Model	Es	TL	CI	Accuracy	Precision	Recall	F1	PR AUC	ROC AUC
ESM_XGB	/	/	W	0.75 ± 0.02	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.25 ± 0.02	0.50 ± 0.00
ESM_XGB	/	/	O	0.75 ± 0.02	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.25 ± 0.02	0.50 ± 0.00
ESM_XGB	/	/	N	0.75 ± 0.02	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.25 ± 0.02	0.50 ± 0.00
ESM_RF	/	/	W	0.65 ± 0.07	0.11 ± 0.13	0.10 ± 0.12	0.10 ± 0.13	0.25 ± 0.02	0.46 ± 0.06
ESM_RF	/	/	O	0.72 ± 0.06	0.11 ± 0.16	0.07 ± 0.10	0.09 ± 0.12	0.25 ± 0.03	0.50 ± 0.04
ESM_RF	/	/	N	0.72 ± 0.06	0.05 ± 0.10	0.05 ± 0.10	0.05 ± 0.10	0.24 ± 0.02	0.49 ± 0.01

Table 7.8: ESM_ML intra model comparison

Model	ES	TL	CI	Accuracy	Precision	Recall	F1	PR AUC	ROC AUC
CCS_XGB	/	/	W	0.76 ± 0.02	0.53 ± 0.32	0.13 ± 0.08	0.20 ± 0.12	0.30 ± 0.05	0.55 ± 0.04
CCS_RF	/	/	O	0.77 ± 0.02	0.57 ± 0.33	0.15 ± 0.09	0.24 ± 0.13	0.31 ± 0.05	0.56 ± 0.04
CCS_RF	/	/	N	0.76 ± 0.04	0.60 ± 0.39	0.15 ± 0.09	0.22 ± 0.12	0.31 ± 0.05	0.56 ± 0.03
CCS_XGB	/	/	N	0.76 ± 0.02	0.53 ± 0.32	0.13 ± 0.08	0.20 ± 0.12	0.30 ± 0.05	0.55 ± 0.04
CCS_RF	/	/	W	0.65 ± 0.05	0.22 ± 0.12	0.27 ± 0.15	0.24 ± 0.13	0.26 ± 0.02	0.52 ± 0.04
CCS_XGB	/	/	O	0.76 ± 0.02	0.53 ± 0.32	0.13 ± 0.08	0.20 ± 0.12	0.30 ± 0.05	0.55 ± 0.04

Table 7.9: CCS_ML intra model comparison

B.2 Self-training

This section presents the tables related to the second part of this work. As a reminder, we applied self-training on both the BiLSTM and MLP models. For each model, three different tables are shown. The first table includes all the results using the threshold-based pseudo-labeling technique (named 'fixed'). The second table shows the results using the proportion-based pseudo-labeling technique (named 'TopK fixed' or 'TopK evo'). The third table presents the results when using the optimal thresholding pseudo-labeling technique (named 'optimal pr' or 'optimal roc'). For the two first approaches, the table is divided in two due to its large size.

The first row of each table corresponds to the best model (either MLP or BiLSTM) from the first part of this work (Supervised Learning). The first 5 columns correspond to the following:

- **Model**: Name of the model evaluated.
- **PL (Pseudo Labeling)**: Indicates the pseudo-labeling technique used during self-training.
- **LT (Label Type)**: Specifies whether hard or soft labels were used.
  - `H` – Hard labels
  - `S` – Soft labels
- **PR (PR AUC STOP)**: Indicates whether the PR AUC stopping criterion was used.
- **DL (Double Loss)**: Refers to the use of a double loss strategy during training.
  - `/` – Double loss not used
  - `GS` – Grid search performed over the 10 α values
  - `<value>` – A fixed value of α was used

B.2.1 MLP

Model	PL	LT	PR	DL	Accuracy	Precision	Recall	F1	PR AUC	ROC AUC
MLP_ES_N_W					0.67 ± 0.09	0.44 ± 0.30	0.32 ± 0.18	0.30 ± 0.12	0.43 ± 0.10	0.61 ± 0.12
MLP	fixed - 0.7	H	No	/	0.59 ± 0.07	0.25 ± 0.11	0.38 ± 0.21	0.29 ± 0.14	0.33 ± 0.08	0.59 ± 0.12
MLP	fixed - 0.9	H	No	/	0.66 ± 0.09	0.42 ± 0.30	0.32 ± 0.18	0.29 ± 0.10	0.38 ± 0.07	0.60 ± 0.11
MLP	fixed - 0.95	H	No	/	0.67 ± 0.09	0.44 ± 0.29	0.32 ± 0.18	0.30 ± 0.11	0.38 ± 0.06	0.60 ± 0.11
MLP	fixed - 0.99	H	No	/	0.67 ± 0.09	0.44 ± 0.30	0.29 ± 0.17	0.29 ± 0.10	0.37 ± 0.07	0.58 ± 0.11
MLP	fixed - 0.7	H	Yes	/	0.64 ± 0.06	0.30 ± 0.08	0.35 ± 0.15	0.31 ± 0.09	0.43 ± 0.11	0.62 ± 0.13
MLP	fixed - 0.9	H	Yes	/	0.66 ± 0.09	0.43 ± 0.30	0.29 ± 0.17	0.28 ± 0.09	0.41 ± 0.07	0.61 ± 0.11
MLP	fixed - 0.95	H	Yes	/	0.66 ± 0.09	0.43 ± 0.30	0.32 ± 0.18	0.30 ± 0.11	0.40 ± 0.06	0.61 ± 0.11
MLP	fixed - 0.99	H	Yes	/	0.66 ± 0.09	0.45 ± 0.28	0.34 ± 0.16	0.32 ± 0.08	0.40 ± 0.06	0.62 ± 0.11
MLP	fixed - 0.99	S	Yes	/	0.66 ± 0.09	0.45 ± 0.29	0.37 ± 0.19	0.33 ± 0.12	0.41 ± 0.09	0.61 ± 0.11
MLP	fixed - 0.95	S	Yes	/	0.66 ± 0.09	0.44 ± 0.29	0.37 ± 0.19	0.33 ± 0.11	0.42 ± 0.08	0.61 ± 0.11
MLP	fixed - 0.9	S	Yes	/	0.67 ± 0.10	0.46 ± 0.29	0.39 ± 0.23	0.35 ± 0.15	0.42 ± 0.08	0.61 ± 0.11
MLP	fixed - 0.7	S	Yes	/	0.54 ± 0.16	0.33 ± 0.12	0.69 ± 0.28	0.42 ± 0.12	0.40 ± 0.10	0.63 ± 0.10
MLP	fixed - 0.7	H	Yes	0.1	0.60 ± 0.16	0.19 ± 0.15	0.24 ± 0.21	0.20 ± 0.16	0.39 ± 0.10	0.59 ± 0.14
MLP	fixed - 0.7	H	Yes	0.2	0.55 ± 0.17	0.22 ± 0.13	0.35 ± 0.28	0.25 ± 0.14	0.39 ± 0.11	0.59 ± 0.13
MLP	fixed - 0.7	H	Yes	0.3	0.55 ± 0.15	0.24 ± 0.12	0.39 ± 0.30	0.27 ± 0.14	0.39 ± 0.11	0.59 ± 0.13
MLP	fixed - 0.7	H	Yes	0.4	0.56 ± 0.14	0.25 ± 0.12	0.40 ± 0.29	0.28 ± 0.14	0.40 ± 0.11	0.60 ± 0.13
MLP	fixed - 0.7	H	Yes	0.5	0.56 ± 0.14	0.25 ± 0.12	0.41 ± 0.28	0.29 ± 0.13	0.40 ± 0.11	0.60 ± 0.13
MLP	fixed - 0.7	H	Yes	0.6	0.56 ± 0.13	0.26 ± 0.11	0.41 ± 0.27	0.29 ± 0.13	0.40 ± 0.12	0.60 ± 0.13
MLP	fixed - 0.7	H	Yes	0.7	0.57 ± 0.13	0.26 ± 0.11	0.40 ± 0.26	0.29 ± 0.12	0.41 ± 0.12	0.60 ± 0.13
MLP	fixed - 0.7	H	Yes	0.8	0.57 ± 0.12	0.26 ± 0.11	0.41 ± 0.25	0.30 ± 0.13	0.41 ± 0.11	0.60 ± 0.13
MLP	fixed - 0.7	H	Yes	0.9	0.57 ± 0.12	0.26 ± 0.11	0.41 ± 0.25	0.30 ± 0.12	0.40 ± 0.11	0.60 ± 0.13
MLP	fixed - 0.7	H	Yes	1.0	0.57 ± 0.12	0.26 ± 0.11	0.41 ± 0.24	0.30 ± 0.12	0.40 ± 0.11	0.60 ± 0.13
MLP	fixed - 0.7	H	Yes	GS	0.57 ± 0.11	0.27 ± 0.09	0.43 ± 0.20	0.31 ± 0.10	0.41 ± 0.08	0.62 ± 0.12
MLP	fixed - 0.9	H	Yes	0.1	0.68 ± 0.09	0.48 ± 0.27	0.32 ± 0.15	0.32 ± 0.07	0.42 ± 0.11	0.61 ± 0.12
MLP	fixed - 0.9	H	Yes	0.2	0.68 ± 0.09	0.48 ± 0.27	0.32 ± 0.15	0.32 ± 0.06	0.41 ± 0.09	0.61 ± 0.11
MLP	fixed - 0.9	H	Yes	0.3	0.68 ± 0.09	0.47 ± 0.27	0.32 ± 0.15	0.32 ± 0.06	0.42 ± 0.09	0.61 ± 0.11
MLP	fixed - 0.9	H	Yes	0.4	0.68 ± 0.09	0.47 ± 0.28	0.32 ± 0.15	0.31 ± 0.06	0.42 ± 0.09	0.61 ± 0.11
MLP	fixed - 0.9	H	Yes	0.5	0.68 ± 0.09	0.47 ± 0.28	0.32 ± 0.15	0.32 ± 0.07	0.42 ± 0.09	0.61 ± 0.11
MLP	fixed - 0.9	H	Yes	0.6	0.68 ± 0.09	0.47 ± 0.28	0.32 ± 0.15	0.31 ± 0.07	0.43 ± 0.09	0.61 ± 0.11
MLP	fixed - 0.9	H	Yes	0.7	0.68 ± 0.09	0.46 ± 0.28	0.31 ± 0.16	0.31 ± 0.07	0.42 ± 0.09	0.61 ± 0.11
MLP	fixed - 0.9	H	Yes	0.8	0.67 ± 0.09	0.46 ± 0.28	0.31 ± 0.16	0.30 ± 0.08	0.42 ± 0.09	0.61 ± 0.11
MLP	fixed - 0.9	H	Yes	0.9	0.67 ± 0.09	0.45 ± 0.28	0.31 ± 0.16	0.30 ± 0.08	0.42 ± 0.09	0.61 ± 0.11
MLP	fixed - 0.9	H	Yes	1.0	0.67 ± 0.09	0.45 ± 0.28	0.31 ± 0.16	0.30 ± 0.08	0.42 ± 0.09	0.61 ± 0.11
MLP	fixed - 0.9	H	Yes	GS	0.66 ± 0.09	0.43 ± 0.29	0.29 ± 0.17	0.28 ± 0.08	0.42 ± 0.10	0.62 ± 0.11
MLP	fixed - 0.95	H	Yes	0.1	0.68 ± 0.09	0.47 ± 0.27	0.32 ± 0.15	0.31 ± 0.06	0.43 ± 0.10	0.61 ± 0.11
MLP	fixed - 0.95	H	Yes	0.2	0.67 ± 0.09	0.46 ± 0.28	0.32 ± 0.15	0.31 ± 0.06	0.44 ± 0.11	0.61 ± 0.11
MLP	fixed - 0.95	H	Yes	0.3	0.67 ± 0.09	0.46 ± 0.28	0.32 ± 0.15	0.31 ± 0.06	0.44 ± 0.10	0.61 ± 0.11
MLP	fixed - 0.95	H	Yes	0.4	0.67 ± 0.09	0.45 ± 0.28	0.32 ± 0.15	0.31 ± 0.06	0.43 ± 0.10	0.61 ± 0.11
MLP	fixed - 0.95	H	Yes	0.5	0.67 ± 0.09	0.46 ± 0.28	0.32 ± 0.15	0.31 ± 0.06	0.43 ± 0.10	0.61 ± 0.11
MLP	fixed - 0.95	H	Yes	0.6	0.67 ± 0.09	0.45 ± 0.28	0.31 ± 0.15	0.30 ± 0.06	0.43 ± 0.10	0.61 ± 0.11
MLP	fixed - 0.95	H	Yes	0.7	0.67 ± 0.09	0.45 ± 0.28	0.31 ± 0.15	0.30 ± 0.07	0.43 ± 0.10	0.61 ± 0.11
MLP	fixed - 0.95	H	Yes	0.8	0.67 ± 0.09	0.45 ± 0.28	0.31 ± 0.15	0.31 ± 0.07	0.43 ± 0.09	0.61 ± 0.11
MLP	fixed - 0.95	H	Yes	0.9	0.67 ± 0.09	0.45 ± 0.28	0.32 ± 0.15	0.31 ± 0.07	0.43 ± 0.09	0.61 ± 0.11
MLP	fixed - 0.95	H	Yes	1.0	0.67 ± 0.09	0.45 ± 0.28	0.31 ± 0.15	0.30 ± 0.07	0.43 ± 0.09	0.61 ± 0.11
MLP	fixed - 0.95	H	Yes	GS	0.66 ± 0.09	0.43 ± 0.29	0.29 ± 0.17	0.28 ± 0.08	0.43 ± 0.09	0.61 ± 0.11
MLP	fixed - 0.99	H	Yes	0.1	0.65 ± 0.09	0.43 ± 0.29	0.32 ± 0.15	0.30 ± 0.07	0.43 ± 0.10	0.61 ± 0.11
MLP	fixed - 0.99	H	Yes	0.2	0.65 ± 0.10	0.43 ± 0.29	0.32 ± 0.15	0.30 ± 0.07	0.43 ± 0.09	0.61 ± 0.11
MLP	fixed - 0.99	H	Yes	0.3	0.66 ± 0.09	0.44 ± 0.29	0.32 ± 0.15	0.30 ± 0.07	0.43 ± 0.09	0.61 ± 0.11
MLP	fixed - 0.99	H	Yes	0.4	0.65 ± 0.09	0.43 ± 0.29	0.31 ± 0.15	0.30 ± 0.06	0.43 ± 0.09	0.61 ± 0.11
MLP	fixed - 0.99	H	Yes	0.5	0.65 ± 0.09	0.43 ± 0.29	0.31 ± 0.15	0.30 ± 0.06	0.43 ± 0.09	0.61 ± 0.11
MLP	fixed - 0.99	H	Yes	0.6	0.65 ± 0.09	0.43 ± 0.29	0.31 ± 0.15	0.30 ± 0.06	0.43 ± 0.09	0.61 ± 0.11
MLP	fixed - 0.99	H	Yes	0.7	0.65 ± 0.09	0.43 ± 0.29	0.31 ± 0.15	0.30 ± 0.06	0.43 ± 0.09	0.61 ± 0.11
MLP	fixed - 0.99	H	Yes	0.8	0.65 ± 0.09	0.43 ± 0.29	0.31 ± 0.15	0.30 ± 0.07	0.43 ± 0.09	0.61 ± 0.11
MLP	fixed - 0.99	H	Yes	0.9	0.66 ± 0.09	0.43 ± 0.29	0.31 ± 0.15	0.30 ± 0.07	0.43 ± 0.09	0.61 ± 0.11
MLP	fixed - 0.99	H	Yes	1.0	0.66 ± 0.09	0.43 ± 0.29	0.32 ± 0.15	0.30 ± 0.08	0.43 ± 0.09	0.61 ± 0.11
MLP	fixed - 0.99	H	Yes	GS	0.66 ± 0.09	0.44 ± 0.29	0.34 ± 0.16	0.32 ± 0.09	0.42 ± 0.09	0.60 ± 0.11

Table 7.10: MLP - threshold-based pseudo-labeling technique - 1/2

Model	PL	LT	PR	DL	Accuracy	Precision	Recall	F1	PR AUC	ROC AUC
MLP_ES_N_W					0.67 ± 0.09	0.44 ± 0.30	0.32 ± 0.18	0.30 ± 0.12	0.43 ± 0.10	0.61 ± 0.12
MLP	fixed - 0.7	S	Yes	0.1	0.58 ± 0.10	0.29 ± 0.08	0.45 ± 0.12	0.35 ± 0.08	0.38 ± 0.11	0.60 ± 0.13
MLP	fixed - 0.7	S	Yes	0.2	0.58 ± 0.12	0.30 ± 0.11	0.47 ± 0.17	0.35 ± 0.13	0.37 ± 0.11	0.59 ± 0.13
MLP	fixed - 0.7	S	Yes	0.3	0.58 ± 0.11	0.30 ± 0.11	0.48 ± 0.18	0.36 ± 0.12	0.38 ± 0.11	0.60 ± 0.13
MLP	fixed - 0.7	S	Yes	0.4	0.59 ± 0.11	0.31 ± 0.11	0.49 ± 0.17	0.36 ± 0.12	0.38 ± 0.11	0.59 ± 0.13
MLP	fixed - 0.7	S	Yes	0.5	0.58 ± 0.11	0.31 ± 0.11	0.50 ± 0.18	0.37 ± 0.12	0.38 ± 0.12	0.60 ± 0.13
MLP	fixed - 0.7	S	Yes	0.6	0.58 ± 0.11	0.30 ± 0.12	0.48 ± 0.18	0.36 ± 0.12	0.38 ± 0.12	0.60 ± 0.13
MLP	fixed - 0.7	S	Yes	0.7	0.58 ± 0.12	0.30 ± 0.11	0.49 ± 0.18	0.36 ± 0.11	0.38 ± 0.12	0.60 ± 0.13
MLP	fixed - 0.7	S	Yes	0.8	0.57 ± 0.12	0.30 ± 0.11	0.48 ± 0.18	0.36 ± 0.11	0.38 ± 0.12	0.59 ± 0.13
MLP	fixed - 0.7	S	Yes	0.9	0.57 ± 0.12	0.30 ± 0.11	0.48 ± 0.17	0.36 ± 0.11	0.38 ± 0.12	0.59 ± 0.13
MLP	fixed - 0.7	S	Yes	1.0	0.57 ± 0.12	0.30 ± 0.12	0.48 ± 0.17	0.36 ± 0.12	0.38 ± 0.12	0.59 ± 0.13
MLP	fixed - 0.7	S	Yes	GS	0.59 ± 0.15	0.31 ± 0.16	0.45 ± 0.19	0.36 ± 0.17	0.37 ± 0.13	0.58 ± 0.15
MLP	fixed - 0.9	S	Yes	0.1	0.66 ± 0.09	0.45 ± 0.28	0.34 ± 0.16	0.32 ± 0.09	0.42 ± 0.08	0.61 ± 0.11
MLP	fixed - 0.9	S	Yes	0.2	0.67 ± 0.09	0.45 ± 0.28	0.34 ± 0.16	0.32 ± 0.09	0.42 ± 0.08	0.61 ± 0.11
MLP	fixed - 0.9	S	Yes	0.3	0.67 ± 0.09	0.46 ± 0.28	0.35 ± 0.17	0.33 ± 0.10	0.42 ± 0.08	0.61 ± 0.11
MLP	fixed - 0.9	S	Yes	0.4	0.67 ± 0.09	0.46 ± 0.28	0.35 ± 0.17	0.33 ± 0.10	0.42 ± 0.08	0.61 ± 0.11
MLP	fixed - 0.9	S	Yes	0.5	0.67 ± 0.09	0.46 ± 0.29	0.35 ± 0.18	0.33 ± 0.10	0.42 ± 0.08	0.61 ± 0.11
MLP	fixed - 0.9	S	Yes	0.6	0.67 ± 0.09	0.45 ± 0.29	0.35 ± 0.17	0.32 ± 0.10	0.42 ± 0.08	0.61 ± 0.11
MLP	fixed - 0.9	S	Yes	0.7	0.66 ± 0.09	0.45 ± 0.29	0.34 ± 0.17	0.32 ± 0.10	0.42 ± 0.08	0.61 ± 0.11
MLP	fixed - 0.9	S	Yes	0.8	0.67 ± 0.09	0.45 ± 0.29	0.35 ± 0.18	0.32 ± 0.10	0.42 ± 0.08	0.61 ± 0.11
MLP	fixed - 0.9	S	Yes	0.9	0.67 ± 0.09	0.45 ± 0.29	0.35 ± 0.18	0.32 ± 0.10	0.42 ± 0.08	0.61 ± 0.11
MLP	fixed - 0.9	S	Yes	1.0	0.66 ± 0.09	0.45 ± 0.29	0.35 ± 0.17	0.32 ± 0.10	0.42 ± 0.08	0.61 ± 0.11
MLP	fixed - 0.9	S	Yes	GS	0.65 ± 0.10	0.43 ± 0.30	0.34 ± 0.16	0.31 ± 0.09	0.42 ± 0.08	0.61 ± 0.11
MLP	fixed - 0.95	S	Yes	0.1	0.66 ± 0.10	0.43 ± 0.31	0.32 ± 0.18	0.30 ± 0.12	0.42 ± 0.08	0.61 ± 0.11
MLP	fixed - 0.95	S	Yes	0.2	0.66 ± 0.10	0.45 ± 0.30	0.34 ± 0.19	0.32 ± 0.12	0.42 ± 0.08	0.61 ± 0.11
MLP	fixed - 0.95	S	Yes	0.3	0.67 ± 0.10	0.45 ± 0.29	0.35 ± 0.19	0.33 ± 0.13	0.42 ± 0.08	0.61 ± 0.11
MLP	fixed - 0.95	S	Yes	0.4	0.67 ± 0.09	0.45 ± 0.29	0.34 ± 0.18	0.32 ± 0.11	0.42 ± 0.09	0.61 ± 0.12
MLP	fixed - 0.95	S	Yes	0.5	0.67 ± 0.09	0.45 ± 0.29	0.34 ± 0.18	0.32 ± 0.11	0.42 ± 0.09	0.61 ± 0.12
MLP	fixed - 0.95	S	Yes	0.6	0.67 ± 0.09	0.45 ± 0.29	0.34 ± 0.17	0.32 ± 0.10	0.42 ± 0.09	0.61 ± 0.11
MLP	fixed - 0.95	S	Yes	0.7	0.67 ± 0.09	0.45 ± 0.29	0.34 ± 0.17	0.32 ± 0.10	0.42 ± 0.09	0.61 ± 0.11
MLP	fixed - 0.95	S	Yes	0.8	0.67 ± 0.09	0.45 ± 0.29	0.33 ± 0.17	0.31 ± 0.09	0.42 ± 0.09	0.61 ± 0.11
MLP	fixed - 0.95	S	Yes	0.9	0.67 ± 0.09	0.45 ± 0.29	0.33 ± 0.17	0.32 ± 0.09	0.42 ± 0.09	0.61 ± 0.11
MLP	fixed - 0.95	S	Yes	1.0	0.67 ± 0.09	0.45 ± 0.29	0.34 ± 0.17	0.32 ± 0.09	0.42 ± 0.09	0.61 ± 0.11
MLP	fixed - 0.95	S	Yes	GS	0.66 ± 0.09	0.44 ± 0.28	0.34 ± 0.16	0.31 ± 0.08	0.42 ± 0.07	0.61 ± 0.11
MLP	fixed - 0.99	S	Yes	0.1	0.68 ± 0.10	0.56 ± 0.36	0.29 ± 0.15	0.31 ± 0.06	0.43 ± 0.09	0.60 ± 0.11
MLP	fixed - 0.99	S	Yes	0.2	0.66 ± 0.11	0.50 ± 0.34	0.33 ± 0.17	0.31 ± 0.09	0.42 ± 0.08	0.60 ± 0.11
MLP	fixed - 0.99	S	Yes	0.3	0.66 ± 0.11	0.47 ± 0.32	0.33 ± 0.17	0.31 ± 0.09	0.41 ± 0.08	0.60 ± 0.11
MLP	fixed - 0.99	S	Yes	0.4	0.66 ± 0.10	0.47 ± 0.32	0.34 ± 0.17	0.32 ± 0.09	0.42 ± 0.08	0.60 ± 0.11
MLP	fixed - 0.99	S	Yes	0.5	0.66 ± 0.10	0.46 ± 0.31	0.33 ± 0.17	0.31 ± 0.09	0.42 ± 0.08	0.60 ± 0.11
MLP	fixed - 0.99	S	Yes	0.6	0.66 ± 0.10	0.46 ± 0.31	0.33 ± 0.16	0.31 ± 0.09	0.42 ± 0.08	0.60 ± 0.11
MLP	fixed - 0.99	S	Yes	0.7	0.66 ± 0.10	0.46 ± 0.31	0.32 ± 0.16	0.31 ± 0.08	0.42 ± 0.08	0.60 ± 0.11
MLP	fixed - 0.99	S	Yes	0.8	0.66 ± 0.10	0.45 ± 0.30	0.32 ± 0.16	0.31 ± 0.08	0.42 ± 0.08	0.60 ± 0.11
MLP	fixed - 0.99	S	Yes	0.9	0.66 ± 0.10	0.45 ± 0.30	0.33 ± 0.16	0.31 ± 0.08	0.42 ± 0.08	0.60 ± 0.11
MLP	fixed - 0.99	S	Yes	1.0	0.66 ± 0.10	0.45 ± 0.30	0.33 ± 0.17	0.31 ± 0.09	0.42 ± 0.08	0.60 ± 0.11
MLP	fixed - 0.99	S	Yes	GS	0.66 ± 0.09	0.44 ± 0.30	0.34 ± 0.21	0.31 ± 0.14	0.41 ± 0.07	0.60 ± 0.11

Table 7.11: MLP - threshold-based pseudo-labeling technique - 2/2

Model	PL	LT	PR	DL	Accuracy	Precision	Recall	F1	PR AUC	ROC AUC
MLP_ES_N_W					0.67 ± 0.09	0.44 ± 0.30	0.32 ± 0.18	0.30 ± 0.12	0.43 ± 0.10	0.61 ± 0.12
MLP	TopK fixed - 0.001	H	No	/	0.61 ± 0.08	0.29 ± 0.08	0.38 ± 0.12	0.32 ± 0.07	0.34 ± 0.08	0.60 ± 0.12
MLP	TopK fixed - 0.01	H	No	/	0.56 ± 0.08	0.32 ± 0.06	0.71 ± 0.12	0.44 ± 0.07	0.32 ± 0.05	0.62 ± 0.08
MLP	TopK fixed - 0.05	H	No	/	0.53 ± 0.08	0.31 ± 0.07	0.73 ± 0.19	0.43 ± 0.10	0.31 ± 0.06	0.61 ± 0.10
MLP	TopK evo	H	No	/	0.59 ± 0.09	0.33 ± 0.08	0.60 ± 0.15	0.42 ± 0.11	0.31 ± 0.06	0.60 ± 0.10
MLP	TopK fixed - 0.001	H	Yes	/	0.67 ± 0.10	0.27 ± 0.17	0.23 ± 0.12	0.24 ± 0.14	0.44 ± 0.13	0.63 ± 0.10
MLP	TopK fixed - 0.01	H	Yes	/	0.65 ± 0.08	0.31 ± 0.11	0.31 ± 0.05	0.31 ± 0.08	0.44 ± 0.12	0.62 ± 0.12
MLP	TopK fixed - 0.05	H	Yes	/	0.61 ± 0.07	0.31 ± 0.10	0.50 ± 0.16	0.39 ± 0.12	0.42 ± 0.09	0.61 ± 0.10
MLP	TopK evo	H	Yes	/	0.61 ± 0.07	0.25 ± 0.09	0.31 ± 0.12	0.28 ± 0.10	0.40 ± 0.10	0.60 ± 0.12
MLP	TopK fixed - 0.001	S	Yes	/	0.51 ± 0.19	0.31 ± 0.10	0.56 ± 0.26	0.36 ± 0.05	0.40 ± 0.15	0.60 ± 0.11
MLP	TopK fixed - 0.01	S	Yes	/	0.41 ± 0.20	0.28 ± 0.05	0.72 ± 0.34	0.36 ± 0.05	0.47 ± 0.10	0.62 ± 0.12
MLP	TopK fixed - 0.05	S	Yes	/	0.41 ± 0.21	0.28 ± 0.07	0.78 ± 0.30	0.38 ± 0.08	0.36 ± 0.13	0.54 ± 0.15
MLP	TopK evo	S	Yes	/	0.42 ± 0.23	0.30 ± 0.10	0.725 ± 0.34	0.37 ± 0.06	0.46 ± 0.10	0.61 ± 0.10
MLP	TopK fixed - 0.05	H	Yes	0.1	0.62 ± 0.09	0.31 ± 0.16	0.35 ± 0.15	0.31 ± 0.11	0.44 ± 0.12	0.59 ± 0.13
MLP	TopK fixed - 0.05	H	Yes	0.2	0.64 ± 0.08	0.31 ± 0.14	0.35 ± 0.18	0.31 ± 0.11	0.44 ± 0.11	0.60 ± 0.13
MLP	TopK fixed - 0.05	H	Yes	0.3	0.63 ± 0.09	0.30 ± 0.13	0.34 ± 0.15	0.31 ± 0.11	0.43 ± 0.10	0.60 ± 0.13
MLP	TopK fixed - 0.05	H	Yes	0.4	0.62 ± 0.10	0.33 ± 0.20	0.34 ± 0.14	0.31 ± 0.10	0.42 ± 0.10	0.60 ± 0.12
MLP	TopK fixed - 0.05	H	Yes	0.5	0.62 ± 0.10	0.33 ± 0.18	0.36 ± 0.14	0.32 ± 0.10	0.42 ± 0.10	0.60 ± 0.12
MLP	TopK fixed - 0.05	H	Yes	0.6	0.62 ± 0.11	0.33 ± 0.18	0.36 ± 0.13	0.32 ± 0.10	0.42 ± 0.10	0.60 ± 0.13
MLP	TopK fixed - 0.05	H	Yes	0.7	0.62 ± 0.10	0.32 ± 0.17	0.36 ± 0.13	0.32 ± 0.10	0.42 ± 0.10	0.60 ± 0.13
MLP	TopK fixed - 0.05	H	Yes	0.8	0.62 ± 0.10	0.32 ± 0.16	0.37 ± 0.13	0.32 ± 0.10	0.42 ± 0.10	0.60 ± 0.13
MLP	TopK fixed - 0.05	H	Yes	0.9	0.62 ± 0.10	0.32 ± 0.16	0.36 ± 0.12	0.32 ± 0.10	0.42 ± 0.10	0.60 ± 0.12
MLP	TopK fixed - 0.05	H	Yes	1.0	0.62 ± 0.10	0.32 ± 0.15	0.36 ± 0.12	0.32 ± 0.09	0.42 ± 0.10	0.60 ± 0.12
MLP	TopK fixed - 0.05	H	Yes	GS	0.61 ± 0.09	0.28 ± 0.10	0.34 ± 0.07	0.31 ± 0.09	0.43 ± 0.11	0.61 ± 0.12
MLP	TopK fixed - 0.01	H	Yes	0.1	0.74 ± 0.07	0.49 ± 0.20	0.26 ± 0.08	0.34 ± 0.12	0.45 ± 0.12	0.62 ± 0.14
MLP	TopK fixed - 0.01	H	Yes	0.2	0.72 ± 0.10	0.50 ± 0.32	0.25 ± 0.09	0.32 ± 0.14	0.45 ± 0.13	0.62 ± 0.14
MLP	TopK fixed - 0.01	H	Yes	0.3	0.69 ± 0.10	0.43 ± 0.29	0.26 ± 0.10	0.31 ± 0.14	0.44 ± 0.13	0.61 ± 0.14
MLP	TopK fixed - 0.01	H	Yes	0.4	0.70 ± 0.11	0.46 ± 0.32	0.25 ± 0.10	0.30 ± 0.14	0.45 ± 0.13	0.62 ± 0.14
MLP	TopK fixed - 0.01	H	Yes	0.5	0.69 ± 0.10	0.44 ± 0.31	0.26 ± 0.11	0.30 ± 0.14	0.45 ± 0.13	0.62 ± 0.14
MLP	TopK fixed - 0.01	H	Yes	0.6	0.69 ± 0.10	0.42 ± 0.29	0.25 ± 0.10	0.30 ± 0.14	0.45 ± 0.13	0.62 ± 0.14
MLP	TopK fixed - 0.01	H	Yes	0.7	0.69 ± 0.10	0.42 ± 0.29	0.27 ± 0.14	0.30 ± 0.14	0.45 ± 0.13	0.62 ± 0.14
MLP	TopK fixed - 0.01	H	Yes	0.8	0.69 ± 0.11	0.43 ± 0.30	0.27 ± 0.14	0.31 ± 0.14	0.45 ± 0.13	0.62 ± 0.14
MLP	TopK fixed - 0.01	H	Yes	0.9	0.68 ± 0.11	0.41 ± 0.29	0.28 ± 0.13	0.31 ± 0.14	0.44 ± 0.13	0.62 ± 0.14
MLP	TopK fixed - 0.01	H	Yes	1.0	0.68 ± 0.11	0.41 ± 0.29	0.28 ± 0.13	0.31 ± 0.14	0.44 ± 0.13	0.62 ± 0.14
MLP	TopK fixed - 0.01	H	Yes	GS	0.67 ± 0.10	0.40 ± 0.31	0.29 ± 0.11	0.32 ± 0.14	0.44 ± 0.12	0.62 ± 0.14
MLP	TopK fixed - 0.001	H	Yes	0.1	0.72 ± 0.07	0.39 ± 0.34	0.18 ± 0.13	0.24 ± 0.17	0.45 ± 0.14	0.62 ± 0.14
MLP	TopK fixed - 0.001	H	Yes	0.2	0.72 ± 0.06	0.34 ± 0.28	0.20 ± 0.15	0.24 ± 0.17	0.45 ± 0.14	0.62 ± 0.14
MLP	TopK fixed - 0.001	H	Yes	0.3	0.71 ± 0.06	0.31 ± 0.26	0.20 ± 0.15	0.24 ± 0.17	0.45 ± 0.14	0.62 ± 0.14
MLP	TopK fixed - 0.001	H	Yes	0.4	0.70 ± 0.06	0.31 ± 0.25	0.20 ± 0.15	0.23 ± 0.17	0.45 ± 0.14	0.62 ± 0.14
MLP	TopK fixed - 0.001	H	Yes	0.5	0.70 ± 0.06	0.30 ± 0.24	0.20 ± 0.15	0.23 ± 0.17	0.45 ± 0.14	0.62 ± 0.14
MLP	TopK fixed - 0.001	H	Yes	0.6	0.70 ± 0.06	0.30 ± 0.23	0.20 ± 0.15	0.23 ± 0.17	0.45 ± 0.14	0.62 ± 0.14
MLP	TopK fixed - 0.001	H	Yes	0.7	0.69 ± 0.06	0.30 ± 0.22	0.21 ± 0.14	0.24 ± 0.16	0.45 ± 0.14	0.62 ± 0.14
MLP	TopK fixed - 0.001	H	Yes	0.8	0.69 ± 0.06	0.29 ± 0.21	0.21 ± 0.14	0.24 ± 0.16	0.44 ± 0.14	0.62 ± 0.14
MLP	TopK fixed - 0.001	H	Yes	0.9	0.69 ± 0.06	0.29 ± 0.21	0.21 ± 0.14	0.24 ± 0.16	0.44 ± 0.14	0.62 ± 0.14
MLP	TopK fixed - 0.001	H	Yes	1.0	0.69 ± 0.06	0.30 ± 0.20	0.21 ± 0.14	0.24 ± 0.15	0.44 ± 0.14	0.62 ± 0.14
MLP	TopK fixed - 0.001	H	Yes	GS	0.68 ± 0.07	0.31 ± 0.16	0.21 ± 0.10	0.25 ± 0.12	0.44 ± 0.14	0.62 ± 0.14
MLP	TopK evo	H	Yes	0.1	0.68 ± 0.08	0.33 ± 0.35	0.16 ± 0.10	0.20 ± 0.14	0.43 ± 0.12	0.60 ± 0.14
MLP	TopK evo	H	Yes	0.2	0.69 ± 0.08	0.35 ± 0.35	0.18 ± 0.13	0.22 ± 0.15	0.43 ± 0.12	0.60 ± 0.14
MLP	TopK evo	H	Yes	0.3	0.69 ± 0.11	0.40 ± 0.35	0.21 ± 0.12	0.26 ± 0.16	0.43 ± 0.12	0.61 ± 0.14
MLP	TopK evo	H	Yes	0.4	0.67 ± 0.10	0.36 ± 0.31	0.21 ± 0.11	0.25 ± 0.14	0.43 ± 0.12	0.61 ± 0.14
MLP	TopK evo	H	Yes	0.5	0.68 ± 0.10	0.36 ± 0.32	0.21 ± 0.12	0.25 ± 0.15	0.43 ± 0.12	0.61 ± 0.14
MLP	TopK evo	H	Yes	0.6	0.68 ± 0.10	0.37 ± 0.32	0.22 ± 0.12	0.25 ± 0.15	0.43 ± 0.12	0.61 ± 0.14
MLP	TopK evo	H	Yes	0.7	0.67 ± 0.10	0.36 ± 0.30	0.22 ± 0.12	0.26 ± 0.15	0.43 ± 0.12	0.61 ± 0.14
MLP	TopK evo	H	Yes	0.8	0.67 ± 0.10	0.36 ± 0.30	0.23 ± 0.12	0.26 ± 0.15	0.43 ± 0.12	0.61 ± 0.14
MLP	TopK evo	H	Yes	0.9	0.67 ± 0.10	0.36 ± 0.30	0.24 ± 0.12	0.27 ± 0.15	0.43 ± 0.12	0.61 ± 0.14
MLP	TopK evo	H	Yes	1.0	0.67 ± 0.10	0.36 ± 0.29	0.24 ± 0.11	0.27 ± 0.14	0.43 ± 0.12	0.61 ± 0.14
MLP	TopK evo	H	Yes	GS	0.68 ± 0.10	0.39 ± 0.22	0.26 ± 0.08	0.30 ± 0.12	0.44 ± 0.12	0.63 ± 0.13

Table 7.12: MLP - proportion-based pseudo-labeling technique - 1/2

Model	PL	LT	PR	DL	Accuracy	Precision	Recall	F1	PR AUC	ROC AUC
MLP_ES_N_W					0.67 ± 0.09	0.44 ± 0.30	0.32 ± 0.18	0.30 ± 0.12	0.43 ± 0.10	0.61 ± 0.12
MLP	TopK fixed - 0.05	S	Yes	0.1	0.56 ± 0.13	0.24 ± 0.10	0.36 ± 0.24	0.27 ± 0.14	0.39 ± 0.11	0.58 ± 0.13
MLP	TopK fixed - 0.05	S	Yes	0.2	0.57 ± 0.14	0.29 ± 0.14	0.42 ± 0.25	0.31 ± 0.14	0.42 ± 0.12	0.59 ± 0.13
MLP	TopK fixed - 0.05	S	Yes	0.3	0.55 ± 0.14	0.28 ± 0.13	0.45 ± 0.26	0.31 ± 0.13	0.41 ± 0.11	0.59 ± 0.13
MLP	TopK fixed - 0.05	S	Yes	0.4	0.55 ± 0.14	0.28 ± 0.12	0.46 ± 0.25	0.32 ± 0.12	0.42 ± 0.10	0.59 ± 0.13
MLP	TopK fixed - 0.05	S	Yes	0.5	0.54 ± 0.14	0.28 ± 0.11	0.47 ± 0.27	0.32 ± 0.11	0.42 ± 0.10	0.60 ± 0.12
MLP	TopK fixed - 0.05	S	Yes	0.6	0.53 ± 0.14	0.28 ± 0.11	0.48 ± 0.27	0.32 ± 0.11	0.42 ± 0.10	0.60 ± 0.13
MLP	TopK fixed - 0.05	S	Yes	0.7	0.53 ± 0.15	0.28 ± 0.12	0.48 ± 0.27	0.32 ± 0.12	0.43 ± 0.10	0.60 ± 0.13
MLP	TopK fixed - 0.05	S	Yes	0.8	0.53 ± 0.15	0.28 ± 0.11	0.49 ± 0.27	0.32 ± 0.12	0.43 ± 0.10	0.60 ± 0.13
MLP	TopK fixed - 0.05	S	Yes	0.9	0.52 ± 0.15	0.28 ± 0.11	0.50 ± 0.27	0.33 ± 0.11	0.43 ± 0.10	0.60 ± 0.13
MLP	TopK fixed - 0.05	S	Yes	1.0	0.52 ± 0.16	0.28 ± 0.11	0.52 ± 0.28	0.33 ± 0.11	0.43 ± 0.10	0.60 ± 0.13
MLP	TopK fixed - 0.05	S	Yes	GS	0.49 ± 0.15	0.30 ± 0.05	0.70 ± 0.32	0.38 ± 0.05	0.44 ± 0.11	0.61 ± 0.15
MLP	TopK fixed - 0.01	S	Yes	0.1	0.67 ± 0.14	0.36 ± 0.23	0.21 ± 0.10	0.26 ± 0.14	0.47 ± 0.13	0.64 ± 0.14
MLP	TopK fixed - 0.01	S	Yes	0.2	0.66 ± 0.13	0.36 ± 0.22	0.28 ± 0.15	0.30 ± 0.15	0.43 ± 0.14	0.62 ± 0.14
MLP	TopK fixed - 0.01	S	Yes	0.3	0.64 ± 0.14	0.35 ± 0.22	0.31 ± 0.17	0.30 ± 0.14	0.42 ± 0.14	0.61 ± 0.13
MLP	TopK fixed - 0.01	S	Yes	0.4	0.64 ± 0.14	0.35 ± 0.21	0.33 ± 0.16	0.31 ± 0.13	0.42 ± 0.14	0.61 ± 0.13
MLP	TopK fixed - 0.01	S	Yes	0.5	0.63 ± 0.13	0.34 ± 0.20	0.33 ± 0.17	0.31 ± 0.14	0.42 ± 0.14	0.60 ± 0.13
MLP	TopK fixed - 0.01	S	Yes	0.6	0.62 ± 0.14	0.34 ± 0.20	0.34 ± 0.18	0.31 ± 0.14	0.42 ± 0.14	0.60 ± 0.13
MLP	TopK fixed - 0.01	S	Yes	0.7	0.62 ± 0.14	0.35 ± 0.22	0.35 ± 0.18	0.32 ± 0.14	0.42 ± 0.14	0.60 ± 0.14
MLP	TopK fixed - 0.01	S	Yes	0.8	0.62 ± 0.14	0.34 ± 0.22	0.35 ± 0.18	0.31 ± 0.14	0.42 ± 0.14	0.60 ± 0.13
MLP	TopK fixed - 0.01	S	Yes	0.9	0.61 ± 0.15	0.34 ± 0.21	0.36 ± 0.18	0.31 ± 0.13	0.42 ± 0.15	0.60 ± 0.14
MLP	TopK fixed - 0.01	S	Yes	1.0	0.61 ± 0.14	0.33 ± 0.21	0.36 ± 0.19	0.31 ± 0.14	0.42 ± 0.15	0.60 ± 0.13
MLP	TopK fixed - 0.01	S	Yes	GS	0.59 ± 0.12	0.28 ± 0.17	0.35 ± 0.22	0.29 ± 0.14	0.41 ± 0.15	0.58 ± 0.13
MLP	TopK fixed - 0.001	S	Yes	0.1	0.64 ± 0.10	0.31 ± 0.15	0.29 ± 0.11	0.29 ± 0.12	0.43 ± 0.13	0.63 ± 0.14
MLP	TopK fixed - 0.001	S	Yes	0.2	0.64 ± 0.10	0.30 ± 0.14	0.31 ± 0.15	0.29 ± 0.13	0.43 ± 0.13	0.62 ± 0.14
MLP	TopK fixed - 0.001	S	Yes	0.3	0.63 ± 0.10	0.30 ± 0.14	0.33 ± 0.20	0.30 ± 0.14	0.43 ± 0.14	0.62 ± 0.14
MLP	TopK fixed - 0.001	S	Yes	0.4	0.62 ± 0.10	0.30 ± 0.14	0.34 ± 0.21	0.30 ± 0.13	0.42 ± 0.14	0.62 ± 0.13
MLP	TopK fixed - 0.001	S	Yes	0.5	0.62 ± 0.10	0.29 ± 0.14	0.35 ± 0.24	0.30 ± 0.14	0.42 ± 0.14	0.62 ± 0.13
MLP	TopK fixed - 0.001	S	Yes	0.6	0.62 ± 0.11	0.29 ± 0.14	0.36 ± 0.25	0.30 ± 0.14	0.42 ± 0.14	0.62 ± 0.13
MLP	TopK fixed - 0.001	S	Yes	0.7	0.61 ± 0.11	0.29 ± 0.14	0.36 ± 0.26	0.30 ± 0.14	0.42 ± 0.14	0.62 ± 0.13
MLP	TopK fixed - 0.001	S	Yes	0.8	0.61 ± 0.11	0.29 ± 0.14	0.37 ± 0.27	0.30 ± 0.14	0.42 ± 0.14	0.62 ± 0.13
MLP	TopK fixed - 0.001	S	Yes	0.9	0.61 ± 0.11	0.29 ± 0.14	0.38 ± 0.28	0.30 ± 0.14	0.42 ± 0.14	0.62 ± 0.13
MLP	TopK fixed - 0.001	S	Yes	1.0	0.61 ± 0.11	0.29 ± 0.14	0.38 ± 0.28	0.30 ± 0.14	0.42 ± 0.14	0.62 ± 0.13
MLP	TopK fixed - 0.001	S	Yes	GS	0.59 ± 0.13	0.30 ± 0.15	0.44 ± 0.33	0.32 ± 0.16	0.42 ± 0.14	0.62 ± 0.12
MLP	TopK evo	S	Yes	0.1	0.64 ± 0.12	0.43 ± 0.29	0.38 ± 0.12	0.34 ± 0.06	0.43 ± 0.09	0.62 ± 0.12
MLP	TopK evo	S	Yes	0.2	0.61 ± 0.13	0.35 ± 0.24	0.32 ± 0.12	0.30 ± 0.09	0.40 ± 0.11	0.59 ± 0.12
MLP	TopK evo	S	Yes	0.3	0.61 ± 0.12	0.33 ± 0.21	0.33 ± 0.15	0.30 ± 0.10	0.40 ± 0.10	0.58 ± 0.12
MLP	TopK evo	S	Yes	0.4	0.60 ± 0.12	0.32 ± 0.20	0.33 ± 0.16	0.29 ± 0.11	0.40 ± 0.11	0.59 ± 0.12
MLP	TopK evo	S	Yes	0.5	0.61 ± 0.12	0.31 ± 0.19	0.34 ± 0.17	0.30 ± 0.11	0.40 ± 0.12	0.59 ± 0.12
MLP	TopK evo	S	Yes	0.6	0.60 ± 0.12	0.31 ± 0.18	0.35 ± 0.17	0.30 ± 0.11	0.40 ± 0.12	0.59 ± 0.12
MLP	TopK evo	S	Yes	0.7	0.59 ± 0.12	0.30 ± 0.17	0.36 ± 0.19	0.30 ± 0.11	0.40 ± 0.12	0.59 ± 0.12
MLP	TopK evo	S	Yes	0.8	0.59 ± 0.12	0.30 ± 0.17	0.35 ± 0.19	0.29 ± 0.12	0.40 ± 0.12	0.59 ± 0.12
MLP	TopK evo	S	Yes	0.9	0.59 ± 0.12	0.29 ± 0.17	0.35 ± 0.19	0.29 ± 0.12	0.41 ± 0.13	0.59 ± 0.12
MLP	TopK evo	S	Yes	1.0	0.59 ± 0.13	0.29 ± 0.17	0.36 ± 0.19	0.29 ± 0.12	0.41 ± 0.13	0.59 ± 0.12
MLP	TopK evo	S	Yes	GS	0.55 ± 0.17	0.31 ± 0.16	0.40 ± 0.17	0.31 ± 0.09	0.43 ± 0.14	0.61 ± 0.12

Table 7.13: MLP - proportion-based pseudo-labeling technique - 2/2

Model	PL	LT	PR	DL	Accuracy	Precision	Recall	F1	PR AUC	ROC AUC
MLP_ES_N_W					0.67 ± 0.09	0.44 ± 0.30	0.32 ± 0.18	0.30 ± 0.12	0.43 ± 0.10	0.61 ± 0.12
MLP	Opt - pr	H	No	/	0.68 ± 0.07	0.35 ± 0.13	0.32 ± 0.18	0.31 ± 0.13	0.41 ± 0.06	0.62 ± 0.11
MLP	Opt - roc	H	No	/	0.61 ± 0.10	0.27 ± 0.10	0.33 ± 0.16	0.28 ± 0.10	0.43 ± 0.12	0.61 ± 0.13
MLP	Opt - pr	H	Yes	/	0.68 ± 0.06	0.36 ± 0.12	0.32 ± 0.18	0.31 ± 0.13	0.45 ± 0.13	0.63 ± 0.12
MLP	Opt - roc	H	Yes	/	0.64 ± 0.11	0.28 ± 0.11	0.29 ± 0.17	0.28 ± 0.13	0.46 ± 0.12	0.64 ± 0.11
MLP	Opt - pr	S	Yes	/	0.43 ± 0.24	0.31 ± 0.12	0.78 ± 0.30	0.40 ± 0.11	0.46 ± 0.09	0.62 ± 0.11
MLP	Opt - roc	S	Yes	/	0.56 ± 0.18	0.29 ± 0.11	0.46 ± 0.30	0.32 ± 0.11	0.39 ± 0.08	0.63 ± 0.10
MLP	Opt - pr	H	Yes	0.1	0.67 ± 0.10	0.43 ± 0.30	0.29 ± 0.11	0.31 ± 0.10	0.47 ± 0.10	0.62 ± 0.12
MLP	Opt - pr	H	Yes	0.2	0.68 ± 0.09	0.44 ± 0.29	0.31 ± 0.13	0.32 ± 0.11	0.46 ± 0.12	0.62 ± 0.12
MLP	Opt - pr	H	Yes	0.3	0.68 ± 0.09	0.41 ± 0.25	0.30 ± 0.14	0.31 ± 0.10	0.46 ± 0.11	0.62 ± 0.12
MLP	Opt - pr	H	Yes	0.4	0.68 ± 0.09	0.41 ± 0.26	0.30 ± 0.15	0.30 ± 0.10	0.46 ± 0.11	0.62 ± 0.12
MLP	Opt - pr	H	Yes	0.5	0.68 ± 0.08	0.42 ± 0.27	0.30 ± 0.16	0.30 ± 0.10	0.46 ± 0.11	0.62 ± 0.12
MLP	Opt - pr	H	Yes	0.6	0.67 ± 0.08	0.40 ± 0.25	0.30 ± 0.15	0.30 ± 0.10	0.46 ± 0.11	0.62 ± 0.11
MLP	Opt - pr	H	Yes	0.7	0.67 ± 0.08	0.39 ± 0.24	0.30 ± 0.15	0.30 ± 0.10	0.46 ± 0.11	0.62 ± 0.12
MLP	Opt - pr	H	Yes	0.8	0.68 ± 0.08	0.40 ± 0.24	0.30 ± 0.16	0.30 ± 0.10	0.46 ± 0.11	0.62 ± 0.12
MLP	Opt - pr	H	Yes	0.9	0.68 ± 0.08	0.40 ± 0.25	0.29 ± 0.15	0.29 ± 0.10	0.46 ± 0.10	0.62 ± 0.12
MLP	Opt - pr	H	Yes	1.0	0.67 ± 0.08	0.40 ± 0.24	0.29 ± 0.15	0.29 ± 0.10	0.46 ± 0.10	0.62 ± 0.12
MLP	Opt - pr	H	Yes	GS	0.66 ± 0.07	0.33 ± 0.11	0.29 ± 0.17	0.28 ± 0.09	0.44 ± 0.11	0.62 ± 0.11
MLP	Opt - roc	H	Yes	0.1	0.59 ± 0.15	0.23 ± 0.14	0.40 ± 0.31	0.26 ± 0.16	0.44 ± 0.09	0.60 ± 0.09
MLP	Opt - roc	H	Yes	0.2	0.57 ± 0.19	0.24 ± 0.17	0.41 ± 0.34	0.26 ± 0.16	0.43 ± 0.10	0.61 ± 0.10
MLP	Opt - roc	H	Yes	0.3	0.59 ± 0.17	0.24 ± 0.16	0.38 ± 0.31	0.26 ± 0.16	0.45 ± 0.12	0.62 ± 0.10
MLP	Opt - roc	H	Yes	0.4	0.58 ± 0.18	0.24 ± 0.17	0.38 ± 0.32	0.26 ± 0.16	0.44 ± 0.12	0.62 ± 0.10
MLP	Opt - roc	H	Yes	0.5	0.58 ± 0.19	0.24 ± 0.18	0.38 ± 0.33	0.25 ± 0.16	0.43 ± 0.13	0.61 ± 0.10
MLP	Opt - roc	H	Yes	0.6	0.59 ± 0.18	0.25 ± 0.18	0.37 ± 0.31	0.26 ± 0.16	0.43 ± 0.12	0.61 ± 0.10
MLP	Opt - roc	H	Yes	0.7	0.59 ± 0.17	0.24 ± 0.18	0.35 ± 0.30	0.25 ± 0.16	0.43 ± 0.12	0.61 ± 0.10
MLP	Opt - roc	H	Yes	0.8	0.59 ± 0.18	0.25 ± 0.17	0.36 ± 0.30	0.26 ± 0.15	0.43 ± 0.12	0.61 ± 0.10
MLP	Opt - roc	H	Yes	0.9	0.58 ± 0.18	0.25 ± 0.17	0.37 ± 0.31	0.25 ± 0.15	0.43 ± 0.12	0.61 ± 0.10
MLP	Opt - roc	H	Yes	1.0	0.58 ± 0.18	0.24 ± 0.17	0.37 ± 0.31	0.25 ± 0.15	0.43 ± 0.12	0.61 ± 0.10
MLP	Opt - roc	H	Yes	GS	0.54 ± 0.19	0.22 ± 0.14	0.40 ± 0.35	0.25 ± 0.15	0.41 ± 0.12	0.60 ± 0.11
MLP	Opt - pr	S	Yes	0.1	0.50 ± 0.16	0.31 ± 0.10	0.69 ± 0.24	0.40 ± 0.09	0.43 ± 0.09	0.61 ± 0.11
MLP	Opt - pr	S	Yes	0.2	0.47 ± 0.19	0.31 ± 0.11	0.71 ± 0.25	0.40 ± 0.10	0.42 ± 0.11	0.60 ± 0.11
MLP	Opt - pr	S	Yes	0.3	0.46 ± 0.20	0.30 ± 0.11	0.71 ± 0.27	0.39 ± 0.10	0.42 ± 0.11	0.61 ± 0.11
MLP	Opt - pr	S	Yes	0.4	0.45 ± 0.20	0.30 ± 0.10	0.72 ± 0.29	0.38 ± 0.09	0.43 ± 0.11	0.61 ± 0.12
MLP	Opt - pr	S	Yes	0.5	0.45 ± 0.21	0.30 ± 0.11	0.72 ± 0.29	0.39 ± 0.09	0.43 ± 0.11	0.61 ± 0.12
MLP	Opt - pr	S	Yes	0.6	0.45 ± 0.21	0.30 ± 0.11	0.73 ± 0.29	0.39 ± 0.10	0.43 ± 0.11	0.61 ± 0.12
MLP	Opt - pr	S	Yes	0.7	0.45 ± 0.21	0.30 ± 0.11	0.74 ± 0.29	0.39 ± 0.10	0.43 ± 0.11	0.61 ± 0.12
MLP	Opt - pr	S	Yes	0.8	0.44 ± 0.22	0.30 ± 0.11	0.74 ± 0.29	0.39 ± 0.10	0.44 ± 0.11	0.61 ± 0.12
MLP	Opt - pr	S	Yes	0.9	0.44 ± 0.22	0.30 ± 0.12	0.74 ± 0.29	0.39 ± 0.10	0.44 ± 0.12	0.61 ± 0.13
MLP	Opt - pr	S	Yes	1.0	0.44 ± 0.22	0.31 ± 0.12	0.75 ± 0.29	0.39 ± 0.10	0.44 ± 0.12	0.62 ± 0.12
MLP	Opt - pr	S	Yes	GS	0.43 ± 0.24	0.31 ± 0.12	0.78 ± 0.30	0.40 ± 0.11	0.48 ± 0.09	0.64 ± 0.11
MLP	Opt - roc	S	Yes	0.1	0.64 ± 0.07	0.32 ± 0.15	0.35 ± 0.15	0.31 ± 0.11	0.40 ± 0.12	0.59 ± 0.11
MLP	Opt - roc	S	Yes	0.2	0.65 ± 0.07	0.33 ± 0.14	0.42 ± 0.26	0.34 ± 0.13	0.41 ± 0.13	0.61 ± 0.11
MLP	Opt - roc	S	Yes	0.3	0.64 ± 0.08	0.35 ± 0.21	0.41 ± 0.24	0.34 ± 0.13	0.41 ± 0.13	0.60 ± 0.11
MLP	Opt - roc	S	Yes	0.4	0.63 ± 0.09	0.34 ± 0.20	0.41 ± 0.23	0.33 ± 0.13	0.41 ± 0.13	0.60 ± 0.11
MLP	Opt - roc	S	Yes	0.5	0.63 ± 0.09	0.33 ± 0.19	0.40 ± 0.23	0.33 ± 0.12	0.41 ± 0.13	0.61 ± 0.11
MLP	Opt - roc	S	Yes	0.6	0.63 ± 0.09	0.34 ± 0.21	0.39 ± 0.22	0.32 ± 0.12	0.41 ± 0.13	0.61 ± 0.11
MLP	Opt - roc	S	Yes	0.7	0.63 ± 0.09	0.33 ± 0.20	0.38 ± 0.22	0.32 ± 0.12	0.40 ± 0.13	0.61 ± 0.11
MLP	Opt - roc	S	Yes	0.8	0.62 ± 0.09	0.33 ± 0.19	0.38 ± 0.22	0.31 ± 0.12	0.40 ± 0.12	0.61 ± 0.10
MLP	Opt - roc	S	Yes	0.9	0.63 ± 0.09	0.32 ± 0.19	0.37 ± 0.22	0.31 ± 0.12	0.40 ± 0.12	0.61 ± 0.10
MLP	Opt - roc	S	Yes	1.0	0.63 ± 0.08	0.32 ± 0.18	0.37 ± 0.22	0.31 ± 0.12	0.40 ± 0.12	0.61 ± 0.10
MLP	Opt - roc	S	Yes	GS	0.65 ± 0.05	0.32 ± 0.11	0.35 ± 0.20	0.31 ± 0.11	0.42 ± 0.09	0.63 ± 0.10

Table 7.14: MLP - optimal thresholding pseudo-labeling technique

B.2.2 BiLSTM

Model	PL	LT	PR	DL	Accuracy	Precision	Recall	F1	PR AUC	ROC AUC
BiLSTM_NES_N_O					0.65 ± 0.12	0.32 ± 0.15	0.31 ± 0.23	0.30 ± 0.16	0.42 ± 0.15	0.60 ± 0.19
BiLSTM	fixed - 0.7	H	No	/	0.65 ± 0.13	0.32 ± 0.17	0.37 ± 0.25	0.34 ± 0.20	0.43 ± 0.14	0.64 ± 0.17
BiLSTM	fixed - 0.9	H	No	/	0.73 ± 0.08	0.48 ± 0.31	0.27 ± 0.16	0.33 ± 0.17	0.43 ± 0.16	0.62 ± 0.16
BiLSTM	fixed - 0.95	H	No	/	0.67 ± 0.13	0.43 ± 0.31	0.22 ± 0.12	0.25 ± 0.12	0.37 ± 0.10	0.57 ± 0.17
BiLSTM	fixed - 0.99	H	No	/	0.66 ± 0.13	0.25 ± 0.19	0.31 ± 0.29	0.27 ± 0.23	0.40 ± 0.12	0.56 ± 0.17
BiLSTM	fixed - 0.7	H	Yes	/	0.66 ± 0.13	0.33 ± 0.16	0.40 ± 0.28	0.35 ± 0.21	0.38 ± 0.11	0.60 ± 0.19
BiLSTM	fixed - 0.9	H	Yes	/	0.70 ± 0.07	0.36 ± 0.16	0.35 ± 0.27	0.33 ± 0.18	0.41 ± 0.13	0.62 ± 0.17
BiLSTM	fixed - 0.95	H	Yes	/	0.67 ± 0.14	0.35 ± 0.19	0.32 ± 0.19	0.33 ± 0.19	0.39 ± 0.13	0.59 ± 0.19
BiLSTM	fixed - 0.99	H	Yes	/	0.68 ± 0.14	0.47 ± 0.31	0.34 ± 0.27	0.33 ± 0.20	0.45 ± 0.15	0.62 ± 0.21
BiLSTM	fixed - 0.99	S	Yes	/	0.66 ± 0.13	0.33 ± 0.16	0.34 ± 0.27	0.32 ± 0.20	0.44 ± 0.16	0.61 ± 0.20
BiLSTM	fixed - 0.95	S	Yes	/	0.67 ± 0.14	0.35 ± 0.19	0.32 ± 0.19	0.33 ± 0.19	0.44 ± 0.19	0.58 ± 0.18
BiLSTM	fixed - 0.9	S	Yes	/	0.72 ± 0.09	0.39 ± 0.20	0.32 ± 0.19	0.35 ± 0.19	0.45 ± 0.17	0.61 ± 0.16
BiLSTM	fixed - 0.7	S	Yes	/	0.65 ± 0.13	0.33 ± 0.17	0.34 ± 0.23	0.32 ± 0.18	0.42 ± 0.16	0.59 ± 0.19
BiLSTM	fixed - 0.7	H	Yes	0.1	0.64 ± 0.09	0.29 ± 0.12	0.34 ± 0.23	0.30 ± 0.16	0.36 ± 0.10	0.57 ± 0.18
BiLSTM	fixed - 0.7	H	Yes	0.2	0.65 ± 0.10	0.30 ± 0.13	0.34 ± 0.23	0.31 ± 0.17	0.37 ± 0.11	0.56 ± 0.18
BiLSTM	fixed - 0.7	H	Yes	0.3	0.65 ± 0.11	0.31 ± 0.14	0.36 ± 0.25	0.32 ± 0.18	0.39 ± 0.14	0.57 ± 0.19
BiLSTM	fixed - 0.7	H	Yes	0.4	0.65 ± 0.11	0.30 ± 0.14	0.37 ± 0.26	0.32 ± 0.18	0.39 ± 0.13	0.58 ± 0.19
BiLSTM	fixed - 0.7	H	Yes	0.5	0.65 ± 0.11	0.31 ± 0.15	0.37 ± 0.25	0.32 ± 0.18	0.40 ± 0.14	0.58 ± 0.19
BiLSTM	fixed - 0.7	H	Yes	0.6	0.65 ± 0.12	0.31 ± 0.15	0.37 ± 0.26	0.33 ± 0.19	0.40 ± 0.14	0.59 ± 0.19
BiLSTM	fixed - 0.7	H	Yes	0.7	0.65 ± 0.11	0.31 ± 0.15	0.38 ± 0.26	0.33 ± 0.19	0.40 ± 0.14	0.58 ± 0.19
BiLSTM	fixed - 0.7	H	Yes	0.8	0.65 ± 0.11	0.31 ± 0.15	0.38 ± 0.26	0.33 ± 0.19	0.40 ± 0.14	0.58 ± 0.19
BiLSTM	fixed - 0.7	H	Yes	0.9	0.65 ± 0.11	0.31 ± 0.15	0.38 ± 0.26	0.33 ± 0.19	0.39 ± 0.14	0.58 ± 0.19
BiLSTM	fixed - 0.7	H	Yes	1.0	0.65 ± 0.11	0.31 ± 0.14	0.38 ± 0.26	0.33 ± 0.19	0.39 ± 0.13	0.58 ± 0.19
BiLSTM	fixed - 0.7	H	Yes	GS	0.65 ± 0.09	0.30 ± 0.13	0.37 ± 0.25	0.32 ± 0.17	0.36 ± 0.10	0.57 ± 0.19
BiLSTM	fixed - 0.9	H	Yes	0.1	0.69 ± 0.09	0.38 ± 0.19	0.33 ± 0.22	0.33 ± 0.16	0.42 ± 0.12	0.60 ± 0.17
BiLSTM	fixed - 0.9	H	Yes	0.2	0.69 ± 0.10	0.38 ± 0.19	0.34 ± 0.23	0.34 ± 0.18	0.41 ± 0.12	0.60 ± 0.17
BiLSTM	fixed - 0.9	H	Yes	0.3	0.69 ± 0.09	0.37 ± 0.17	0.32 ± 0.23	0.32 ± 0.17	0.41 ± 0.11	0.60 ± 0.17
BiLSTM	fixed - 0.9	H	Yes	0.4	0.70 ± 0.09	0.37 ± 0.18	0.31 ± 0.22	0.32 ± 0.17	0.41 ± 0.11	0.60 ± 0.16
BiLSTM	fixed - 0.9	H	Yes	0.5	0.69 ± 0.09	0.37 ± 0.18	0.31 ± 0.22	0.32 ± 0.17	0.41 ± 0.12	0.60 ± 0.16
BiLSTM	fixed - 0.9	H	Yes	0.6	0.70 ± 0.09	0.37 ± 0.18	0.31 ± 0.21	0.32 ± 0.17	0.41 ± 0.11	0.61 ± 0.16
BiLSTM	fixed - 0.9	H	Yes	0.7	0.70 ± 0.09	0.38 ± 0.19	0.31 ± 0.21	0.33 ± 0.18	0.41 ± 0.12	0.61 ± 0.16
BiLSTM	fixed - 0.9	H	Yes	0.8	0.70 ± 0.09	0.37 ± 0.19	0.32 ± 0.22	0.33 ± 0.18	0.41 ± 0.12	0.61 ± 0.16
BiLSTM	fixed - 0.9	H	Yes	0.9	0.70 ± 0.09	0.38 ± 0.19	0.32 ± 0.21	0.33 ± 0.18	0.41 ± 0.12	0.61 ± 0.17
BiLSTM	fixed - 0.9	H	Yes	1.0	0.70 ± 0.09	0.37 ± 0.19	0.32 ± 0.21	0.33 ± 0.18	0.41 ± 0.12	0.61 ± 0.16
BiLSTM	fixed - 0.9	H	Yes	GS	0.71 ± 0.06	0.36 ± 0.15	0.35 ± 0.23	0.35 ± 0.18	0.40 ± 0.13	0.61 ± 0.14
BiLSTM	fixed - 0.95	H	Yes	0.1	0.66 ± 0.13	0.33 ± 0.16	0.27 ± 0.16	0.29 ± 0.15	0.39 ± 0.11	0.61 ± 0.21
BiLSTM	fixed - 0.95	H	Yes	0.2	0.67 ± 0.13	0.35 ± 0.17	0.30 ± 0.19	0.31 ± 0.16	0.40 ± 0.13	0.60 ± 0.20
BiLSTM	fixed - 0.95	H	Yes	0.3	0.66 ± 0.12	0.32 ± 0.16	0.30 ± 0.19	0.30 ± 0.15	0.41 ± 0.14	0.59 ± 0.19
BiLSTM	fixed - 0.95	H	Yes	0.4	0.66 ± 0.12	0.33 ± 0.16	0.31 ± 0.20	0.31 ± 0.17	0.41 ± 0.14	0.59 ± 0.19
BiLSTM	fixed - 0.95	H	Yes	0.5	0.66 ± 0.12	0.33 ± 0.16	0.32 ± 0.22	0.31 ± 0.17	0.41 ± 0.14	0.59 ± 0.19
BiLSTM	fixed - 0.95	H	Yes	0.6	0.66 ± 0.12	0.33 ± 0.16	0.32 ± 0.22	0.31 ± 0.17	0.41 ± 0.13	0.59 ± 0.19
BiLSTM	fixed - 0.95	H	Yes	0.7	0.65 ± 0.12	0.32 ± 0.16	0.32 ± 0.21	0.31 ± 0.17	0.40 ± 0.13	0.59 ± 0.19
BiLSTM	fixed - 0.95	H	Yes	0.8	0.66 ± 0.12	0.33 ± 0.16	0.32 ± 0.21	0.31 ± 0.17	0.40 ± 0.13	0.59 ± 0.19
BiLSTM	fixed - 0.95	H	Yes	0.9	0.66 ± 0.12	0.33 ± 0.16	0.32 ± 0.21	0.31 ± 0.17	0.41 ± 0.13	0.59 ± 0.19
BiLSTM	fixed - 0.95	H	Yes	1.0	0.66 ± 0.13	0.33 ± 0.16	0.33 ± 0.21	0.32 ± 0.17	0.40 ± 0.13	0.59 ± 0.19
BiLSTM	fixed - 0.95	H	Yes	GS	0.68 ± 0.14	0.37 ± 0.18	0.35 ± 0.19	0.35 ± 0.19	0.38 ± 0.13	0.59 ± 0.20
BiLSTM	fixed - 0.99	H	Yes	0.1	0.66 ± 0.13	0.25 ± 0.19	0.31 ± 0.29	0.27 ± 0.23	0.44 ± 0.15	0.60 ± 0.20
BiLSTM	fixed - 0.99	H	Yes	0.2	0.66 ± 0.13	0.25 ± 0.19	0.33 ± 0.31	0.27 ± 0.23	0.44 ± 0.15	0.61 ± 0.20
BiLSTM	fixed - 0.99	H	Yes	0.3	0.67 ± 0.14	0.33 ± 0.26	0.35 ± 0.32	0.30 ± 0.24	0.44 ± 0.15	0.61 ± 0.20
BiLSTM	fixed - 0.99	H	Yes	0.4	0.66 ± 0.13	0.30 ± 0.24	0.35 ± 0.32	0.29 ± 0.23	0.44 ± 0.15	0.61 ± 0.20
BiLSTM	fixed - 0.99	H	Yes	0.5	0.66 ± 0.13	0.28 ± 0.23	0.34 ± 0.31	0.28 ± 0.23	0.44 ± 0.15	0.61 ± 0.20
BiLSTM	fixed - 0.99	H	Yes	0.6	0.65 ± 0.13	0.27 ± 0.22	0.33 ± 0.31	0.27 ± 0.22	0.43 ± 0.14	0.61 ± 0.20
BiLSTM	fixed - 0.99	H	Yes	0.7	0.66 ± 0.13	0.30 ± 0.24	0.34 ± 0.31	0.28 ± 0.22	0.43 ± 0.14	0.61 ± 0.20
BiLSTM	fixed - 0.99	H	Yes	0.8	0.66 ± 0.13	0.32 ± 0.26	0.34 ± 0.31	0.29 ± 0.22	0.43 ± 0.14	0.61 ± 0.20
BiLSTM	fixed - 0.99	H	Yes	0.9	0.66 ± 0.13	0.31 ± 0.25	0.34 ± 0.31	0.28 ± 0.22	0.43 ± 0.15	0.61 ± 0.20
BiLSTM	fixed - 0.99	H	Yes	1.0	0.66 ± 0.13	0.30 ± 0.25	0.34 ± 0.30	0.28 ± 0.22	0.43 ± 0.15	0.61 ± 0.20
BiLSTM	fixed - 0.99	H	Yes	GS	0.66 ± 0.13	0.25 ± 0.19	0.31 ± 0.29	0.27 ± 0.23	0.43 ± 0.15	0.61 ± 0.20

Table 7.15: BiLSTM - threshold-based pseudo-labeling technique - 1/2

Model	PL	LT	PR	DL	Accuracy	Precision	Recall	F1	PR AUC	ROC AUC
BiLSTM_NES_N_O					0.65 ± 0.12	0.32 ± 0.15	0.31 ± 0.23	0.30 ± 0.16	0.42 ± 0.15	0.60 ± 0.19
BiLSTM	fixed - 0.7	S	Yes	0.1	0.67 ± 0.06	0.33 ± 0.11	0.42 ± 0.26	0.36 ± 0.17	0.44 ± 0.14	0.62 ± 0.17
BiLSTM	fixed - 0.7	S	Yes	0.2	0.67 ± 0.11	0.34 ± 0.16	0.41 ± 0.27	0.36 ± 0.20	0.42 ± 0.13	0.61 ± 0.17
BiLSTM	fixed - 0.7	S	Yes	0.3	0.67 ± 0.11	0.34 ± 0.17	0.42 ± 0.28	0.36 ± 0.22	0.44 ± 0.16	0.61 ± 0.18
BiLSTM	fixed - 0.7	S	Yes	0.4	0.67 ± 0.12	0.34 ± 0.19	0.39 ± 0.28	0.35 ± 0.22	0.44 ± 0.17	0.61 ± 0.19
BiLSTM	fixed - 0.7	S	Yes	0.5	0.66 ± 0.12	0.33 ± 0.18	0.38 ± 0.27	0.34 ± 0.21	0.43 ± 0.16	0.61 ± 0.19
BiLSTM	fixed - 0.7	S	Yes	0.6	0.66 ± 0.11	0.33 ± 0.17	0.39 ± 0.27	0.34 ± 0.21	0.43 ± 0.16	0.61 ± 0.19
BiLSTM	fixed - 0.7	S	Yes	0.7	0.66 ± 0.12	0.33 ± 0.18	0.39 ± 0.27	0.34 ± 0.21	0.43 ± 0.16	0.61 ± 0.19
BiLSTM	fixed - 0.7	S	Yes	0.8	0.66 ± 0.12	0.33 ± 0.17	0.39 ± 0.27	0.34 ± 0.21	0.43 ± 0.15	0.60 ± 0.19
BiLSTM	fixed - 0.7	S	Yes	0.9	0.66 ± 0.12	0.33 ± 0.17	0.38 ± 0.27	0.34 ± 0.20	0.42 ± 0.15	0.60 ± 0.19
BiLSTM	fixed - 0.7	S	Yes	1.0	0.66 ± 0.12	0.33 ± 0.17	0.38 ± 0.27	0.34 ± 0.20	0.43 ± 0.16	0.60 ± 0.19
BiLSTM	fixed - 0.7	S	Yes	GS	0.66 ± 0.13	0.33 ± 0.17	0.37 ± 0.25	0.34 ± 0.20	0.44 ± 0.17	0.60 ± 0.21
BiLSTM	fixed - 0.9	S	Yes	0.1	0.66 ± 0.08	0.31 ± 0.13	0.27 ± 0.16	0.27 ± 0.12	0.37 ± 0.10	0.58 ± 0.15
BiLSTM	fixed - 0.9	S	Yes	0.2	0.69 ± 0.07	0.34 ± 0.14	0.30 ± 0.18	0.31 ± 0.15	0.40 ± 0.13	0.59 ± 0.15
BiLSTM	fixed - 0.9	S	Yes	0.3	0.70 ± 0.09	0.36 ± 0.18	0.30 ± 0.18	0.32 ± 0.17	0.41 ± 0.14	0.59 ± 0.15
BiLSTM	fixed - 0.9	S	Yes	0.4	0.70 ± 0.08	0.36 ± 0.17	0.30 ± 0.18	0.32 ± 0.17	0.41 ± 0.14	0.60 ± 0.15
BiLSTM	fixed - 0.9	S	Yes	0.5	0.69 ± 0.08	0.35 ± 0.16	0.30 ± 0.19	0.32 ± 0.16	0.41 ± 0.14	0.60 ± 0.16
BiLSTM	fixed - 0.9	S	Yes	0.6	0.69 ± 0.08	0.35 ± 0.16	0.31 ± 0.19	0.32 ± 0.16	0.42 ± 0.14	0.60 ± 0.16
BiLSTM	fixed - 0.9	S	Yes	0.7	0.69 ± 0.08	0.35 ± 0.16	0.30 ± 0.19	0.32 ± 0.16	0.42 ± 0.14	0.60 ± 0.16
BiLSTM	fixed - 0.9	S	Yes	0.8	0.69 ± 0.08	0.35 ± 0.16	0.30 ± 0.19	0.31 ± 0.16	0.42 ± 0.14	0.60 ± 0.16
BiLSTM	fixed - 0.9	S	Yes	0.9	0.69 ± 0.08	0.35 ± 0.16	0.31 ± 0.20	0.32 ± 0.17	0.42 ± 0.14	0.60 ± 0.16
BiLSTM	fixed - 0.9	S	Yes	1.0	0.69 ± 0.08	0.35 ± 0.16	0.31 ± 0.19	0.32 ± 0.17	0.41 ± 0.14	0.61 ± 0.16
BiLSTM	fixed - 0.9	S	Yes	GS	0.70 ± 0.09	0.37 ± 0.16	0.29 ± 0.15	0.32 ± 0.16	0.41 ± 0.14	0.62 ± 0.18
BiLSTM	fixed - 0.95	S	Yes	0.1	0.66 ± 0.13	0.33 ± 0.16	0.33 ± 0.22	0.32 ± 0.18	0.38 ± 0.11	0.59 ± 0.19
BiLSTM	fixed - 0.95	S	Yes	0.2	0.67 ± 0.13	0.35 ± 0.17	0.32 ± 0.20	0.33 ± 0.18	0.41 ± 0.14	0.60 ± 0.20
BiLSTM	fixed - 0.95	S	Yes	0.3	0.67 ± 0.13	0.34 ± 0.17	0.31 ± 0.19	0.31 ± 0.17	0.39 ± 0.13	0.59 ± 0.19
BiLSTM	fixed - 0.95	S	Yes	0.4	0.66 ± 0.13	0.33 ± 0.16	0.32 ± 0.20	0.31 ± 0.17	0.40 ± 0.13	0.59 ± 0.19
BiLSTM	fixed - 0.95	S	Yes	0.5	0.66 ± 0.13	0.33 ± 0.16	0.31 ± 0.21	0.31 ± 0.17	0.40 ± 0.13	0.59 ± 0.19
BiLSTM	fixed - 0.95	S	Yes	0.6	0.66 ± 0.13	0.33 ± 0.16	0.32 ± 0.21	0.31 ± 0.17	0.40 ± 0.14	0.59 ± 0.19
BiLSTM	fixed - 0.95	S	Yes	0.7	0.66 ± 0.13	0.33 ± 0.16	0.33 ± 0.22	0.32 ± 0.18	0.41 ± 0.15	0.59 ± 0.19
BiLSTM	fixed - 0.95	S	Yes	0.8	0.66 ± 0.13	0.34 ± 0.17	0.33 ± 0.22	0.32 ± 0.18	0.41 ± 0.15	0.59 ± 0.19
BiLSTM	fixed - 0.95	S	Yes	0.9	0.67 ± 0.13	0.34 ± 0.17	0.32 ± 0.22	0.32 ± 0.18	0.41 ± 0.15	0.59 ± 0.19
BiLSTM	fixed - 0.95	S	Yes	1.0	0.67 ± 0.13	0.34 ± 0.18	0.33 ± 0.22	0.32 ± 0.19	0.41 ± 0.14	0.59 ± 0.19
BiLSTM	fixed - 0.95	S	Yes	GS	0.69 ± 0.15	0.39 ± 0.20	0.35 ± 0.23	0.36 ± 0.22	0.42 ± 0.13	0.60 ± 0.19
BiLSTM	fixed - 0.99	S	Yes	0.1	0.68 ± 0.15	0.49 ± 0.32	0.34 ± 0.27	0.34 ± 0.21	0.42 ± 0.13	0.60 ± 0.19
BiLSTM	fixed - 0.99	S	Yes	0.2	0.68 ± 0.14	0.47 ± 0.32	0.34 ± 0.27	0.33 ± 0.20	0.42 ± 0.13	0.60 ± 0.20
BiLSTM	fixed - 0.99	S	Yes	0.3	0.67 ± 0.14	0.40 ± 0.30	0.33 ± 0.28	0.31 ± 0.21	0.43 ± 0.14	0.61 ± 0.20
BiLSTM	fixed - 0.99	S	Yes	0.4	0.67 ± 0.14	0.39 ± 0.27	0.34 ± 0.28	0.31 ± 0.21	0.43 ± 0.14	0.61 ± 0.20
BiLSTM	fixed - 0.99	S	Yes	0.5	0.67 ± 0.14	0.38 ± 0.26	0.34 ± 0.28	0.31 ± 0.21	0.43 ± 0.14	0.60 ± 0.20
BiLSTM	fixed - 0.99	S	Yes	0.6	0.67 ± 0.14	0.38 ± 0.24	0.35 ± 0.29	0.32 ± 0.21	0.43 ± 0.14	0.60 ± 0.20
BiLSTM	fixed - 0.99	S	Yes	0.7	0.67 ± 0.13	0.36 ± 0.24	0.35 ± 0.29	0.31 ± 0.21	0.43 ± 0.14	0.61 ± 0.20
BiLSTM	fixed - 0.99	S	Yes	0.8	0.67 ± 0.13	0.36 ± 0.23	0.35 ± 0.29	0.31 ± 0.21	0.43 ± 0.14	0.61 ± 0.20
BiLSTM	fixed - 0.99	S	Yes	0.9	0.67 ± 0.13	0.37 ± 0.24	0.35 ± 0.29	0.31 ± 0.21	0.43 ± 0.14	0.61 ± 0.20
BiLSTM	fixed - 0.99	S	Yes	1.0	0.67 ± 0.13	0.37 ± 0.25	0.35 ± 0.28	0.31 ± 0.20	0.43 ± 0.14	0.61 ± 0.20
BiLSTM	fixed - 0.99	S	Yes	GS	0.67 ± 0.14	0.45 ± 0.31	0.34 ± 0.27	0.32 ± 0.18	0.44 ± 0.14	0.60 ± 0.20

Table 7.16: BiLSTM - threshold-based pseudo-labeling technique - 2/2

Model	PL	LT	PR	DL	Accuracy	Precision	Recall	F1	PR AUC	ROC AUC
BiLSTM_NES_N_O					0.65 ± 0.12	0.32 ± 0.15	0.31 ± 0.23	0.30 ± 0.16	0.42 ± 0.15	0.60 ± 0.19
BiLSTM	TopK fixed - 0.001	H	No	/	0.63 ± 0.10	0.27 ± 0.16	0.27 ± 0.20	0.26 ± 0.16	0.40 ± 0.15	0.61 ± 0.19
BiLSTM	TopK fixed - 0.01	H	No	/	0.55 ± 0.11	0.26 ± 0.14	0.45 ± 0.26	0.33 ± 0.16	0.36 ± 0.13	0.58 ± 0.16
BiLSTM	TopK fixed - 0.05	H	No	/	0.54 ± 0.08	0.26 ± 0.11	0.54 ± 0.29	0.34 ± 0.16	0.34 ± 0.11	0.59 ± 0.18
BiLSTM	TopK evo	H	No	/	0.54 ± 0.09	0.27 ± 0.11	0.56 ± 0.25	0.36 ± 0.14	0.30 ± 0.06	0.55 ± 0.14
BiLSTM	TopK fixed - 0.001	H	Yes	/	0.66 ± 0.14	0.47 ± 0.31	0.40 ± 0.28	0.35 ± 0.20	0.45 ± 0.15	0.61 ± 0.18
BiLSTM	TopK fixed - 0.05	H	Yes	/	0.68 ± 0.10	0.35 ± 0.20	0.38 ± 0.31	0.33 ± 0.23	0.38 ± 0.14	0.58 ± 0.23
BiLSTM	TopK fixed - 0.01	H	Yes	/	0.69 ± 0.11	0.27 ± 0.24	0.30 ± 0.27	0.28 ± 0.26	0.43 ± 0.15	0.64 ± 0.19
BiLSTM	TopK evo	H	Yes	/	0.66 ± 0.09	0.24 ± 0.20	0.35 ± 0.30	0.28 ± 0.23	0.42 ± 0.16	0.64 ± 0.17
BiLSTM	TopK fixed - 0.001	S	Yes	/	0.71 ± 0.09	0.27 ± 0.26	0.26 ± 0.29	0.26 ± 0.26	0.44 ± 0.14	0.61 ± 0.20
BiLSTM	TopK fixed - 0.01	S	Yes	/	0.66 ± 0.12	0.33 ± 0.15	0.40 ± 0.28	0.35 ± 0.20	0.33 ± 0.09	0.57 ± 0.16
BiLSTM	TopK fixed - 0.05	S	Yes	/	0.68 ± 0.13	0.33 ± 0.23	0.38 ± 0.31	0.35 ± 0.26	0.39 ± 0.14	0.59 ± 0.21
BiLSTM	TopK evo	S	Yes	/	0.66 ± 0.09	0.27 ± 0.19	0.35 ± 0.32	0.30 ± 0.24	0.39 ± 0.17	0.58 ± 0.20
BiLSTM	TopK fixed - 0.05	H	Yes	0.1	0.66 ± 0.15	0.34 ± 0.20	0.30 ± 0.22	0.31 ± 0.21	0.39 ± 0.15	0.57 ± 0.17
BiLSTM	TopK fixed - 0.05	H	Yes	0.2	0.70 ± 0.11	0.40 ± 0.16	0.38 ± 0.23	0.37 ± 0.19	0.42 ± 0.13	0.60 ± 0.14
BiLSTM	TopK fixed - 0.05	H	Yes	0.3	0.69 ± 0.12	0.43 ± 0.23	0.38 ± 0.24	0.37 ± 0.20	0.42 ± 0.14	0.59 ± 0.17
BiLSTM	TopK fixed - 0.05	H	Yes	0.4	0.65 ± 0.15	0.38 ± 0.22	0.39 ± 0.26	0.35 ± 0.19	0.41 ± 0.13	0.59 ± 0.16
BiLSTM	TopK fixed - 0.05	H	Yes	0.5	0.66 ± 0.14	0.36 ± 0.22	0.39 ± 0.27	0.34 ± 0.20	0.41 ± 0.13	0.59 ± 0.17
BiLSTM	TopK fixed - 0.05	H	Yes	0.6	0.66 ± 0.14	0.37 ± 0.22	0.39 ± 0.28	0.35 ± 0.21	0.41 ± 0.14	0.59 ± 0.18
BiLSTM	TopK fixed - 0.05	H	Yes	0.7	0.66 ± 0.14	0.37 ± 0.21	0.40 ± 0.28	0.35 ± 0.21	0.41 ± 0.14	0.59 ± 0.18
BiLSTM	TopK fixed - 0.05	H	Yes	0.8	0.66 ± 0.14	0.36 ± 0.21	0.40 ± 0.28	0.35 ± 0.22	0.41 ± 0.14	0.59 ± 0.19
BiLSTM	TopK fixed - 0.05	H	Yes	0.9	0.66 ± 0.14	0.37 ± 0.23	0.40 ± 0.28	0.35 ± 0.22	0.41 ± 0.15	0.59 ± 0.19
BiLSTM	TopK fixed - 0.05	H	Yes	1.0	0.66 ± 0.14	0.37 ± 0.22	0.40 ± 0.28	0.35 ± 0.22	0.41 ± 0.15	0.59 ± 0.20
BiLSTM	TopK fixed - 0.05	H	Yes	GS	0.66 ± 0.14	0.33 ± 0.19	0.37 ± 0.26	0.35 ± 0.22	0.42 ± 0.15	0.58 ± 0.23
BiLSTM	TopK fixed - 0.01	H	Yes	0.1	0.70 ± 0.09	0.31 ± 0.24	0.32 ± 0.29	0.31 ± 0.26	0.42 ± 0.17	0.56 ± 0.22
BiLSTM	TopK fixed - 0.01	H	Yes	0.2	0.70 ± 0.09	0.31 ± 0.24	0.32 ± 0.29	0.31 ± 0.26	0.41 ± 0.16	0.57 ± 0.22
BiLSTM	TopK fixed - 0.01	H	Yes	0.3	0.72 ± 0.08	0.36 ± 0.22	0.34 ± 0.27	0.33 ± 0.24	0.42 ± 0.16	0.59 ± 0.22
BiLSTM	TopK fixed - 0.01	H	Yes	0.4	0.72 ± 0.08	0.35 ± 0.22	0.34 ± 0.29	0.33 ± 0.25	0.42 ± 0.16	0.60 ± 0.21
BiLSTM	TopK fixed - 0.01	H	Yes	0.5	0.72 ± 0.07	0.34 ± 0.22	0.33 ± 0.29	0.32 ± 0.25	0.42 ± 0.15	0.60 ± 0.20
BiLSTM	TopK fixed - 0.01	H	Yes	0.6	0.72 ± 0.08	0.35 ± 0.25	0.33 ± 0.29	0.31 ± 0.25	0.41 ± 0.15	0.59 ± 0.20
BiLSTM	TopK fixed - 0.01	H	Yes	0.7	0.72 ± 0.07	0.36 ± 0.27	0.31 ± 0.28	0.30 ± 0.24	0.41 ± 0.15	0.59 ± 0.20
BiLSTM	TopK fixed - 0.01	H	Yes	0.8	0.71 ± 0.08	0.36 ± 0.28	0.30 ± 0.28	0.29 ± 0.24	0.41 ± 0.14	0.59 ± 0.20
BiLSTM	TopK fixed - 0.01	H	Yes	0.9	0.71 ± 0.08	0.34 ± 0.28	0.29 ± 0.27	0.28 ± 0.23	0.40 ± 0.14	0.59 ± 0.20
BiLSTM	TopK fixed - 0.01	H	Yes	1.0	0.71 ± 0.08	0.33 ± 0.27	0.28 ± 0.27	0.28 ± 0.23	0.41 ± 0.14	0.59 ± 0.20
BiLSTM	TopK fixed - 0.01	H	Yes	GS	0.69 ± 0.08	0.22 ± 0.20	0.24 ± 0.28	0.22 ± 0.23	0.41 ± 0.15	0.58 ± 0.22
BiLSTM	TopK fixed - 0.001	H	Yes	0.1	0.69 ± 0.08	0.25 ± 0.22	0.30 ± 0.27	0.27 ± 0.24	0.41 ± 0.14	0.59 ± 0.17
BiLSTM	TopK fixed - 0.001	H	Yes	0.2	0.70 ± 0.09	0.36 ± 0.31	0.29 ± 0.26	0.29 ± 0.24	0.41 ± 0.16	0.59 ± 0.18
BiLSTM	TopK fixed - 0.001	H	Yes	0.3	0.71 ± 0.09	0.40 ± 0.32	0.33 ± 0.28	0.32 ± 0.24	0.41 ± 0.15	0.59 ± 0.18
BiLSTM	TopK fixed - 0.001	H	Yes	0.4	0.70 ± 0.09	0.41 ± 0.32	0.33 ± 0.28	0.32 ± 0.23	0.41 ± 0.15	0.59 ± 0.18
BiLSTM	TopK fixed - 0.001	H	Yes	0.5	0.71 ± 0.09	0.43 ± 0.33	0.33 ± 0.28	0.32 ± 0.23	0.41 ± 0.15	0.59 ± 0.18
BiLSTM	TopK fixed - 0.001	H	Yes	0.6	0.70 ± 0.09	0.43 ± 0.33	0.33 ± 0.27	0.32 ± 0.22	0.40 ± 0.14	0.59 ± 0.18
BiLSTM	TopK fixed - 0.001	H	Yes	0.7	0.70 ± 0.09	0.43 ± 0.33	0.33 ± 0.27	0.32 ± 0.23	0.41 ± 0.14	0.59 ± 0.18
BiLSTM	TopK fixed - 0.001	H	Yes	0.8	0.70 ± 0.09	0.44 ± 0.33	0.33 ± 0.27	0.32 ± 0.23	0.41 ± 0.14	0.59 ± 0.18
BiLSTM	TopK fixed - 0.001	H	Yes	0.9	0.70 ± 0.09	0.43 ± 0.32	0.34 ± 0.28	0.32 ± 0.23	0.41 ± 0.14	0.59 ± 0.18
BiLSTM	TopK fixed - 0.001	H	Yes	1.0	0.70 ± 0.09	0.43 ± 0.32	0.34 ± 0.28	0.32 ± 0.23	0.41 ± 0.14	0.60 ± 0.18
BiLSTM	TopK fixed - 0.001	H	Yes	GS	0.70 ± 0.10	0.46 ± 0.33	0.35 ± 0.28	0.33 ± 0.22	0.44 ± 0.15	0.60 ± 0.18
BiLSTM	TopK evo	H	Yes	0.1	0.70 ± 0.07	0.33 ± 0.21	0.46 ± 0.34	0.38 ± 0.25	0.38 ± 0.12	0.58 ± 0.20
BiLSTM	TopK evo	H	Yes	0.2	0.68 ± 0.09	0.30 ± 0.20	0.38 ± 0.32	0.33 ± 0.25	0.40 ± 0.15	0.56 ± 0.22
BiLSTM	TopK evo	H	Yes	0.3	0.68 ± 0.09	0.31 ± 0.21	0.39 ± 0.32	0.33 ± 0.24	0.39 ± 0.14	0.56 ± 0.21
BiLSTM	TopK evo	H	Yes	0.4	0.69 ± 0.09	0.35 ± 0.25	0.38 ± 0.31	0.33 ± 0.23	0.39 ± 0.14	0.58 ± 0.20
BiLSTM	TopK evo	H	Yes	0.5	0.69 ± 0.09	0.35 ± 0.24	0.38 ± 0.32	0.33 ± 0.24	0.40 ± 0.13	0.58 ± 0.21
BiLSTM	TopK evo	H	Yes	0.6	0.69 ± 0.10	0.35 ± 0.23	0.40 ± 0.33	0.34 ± 0.24	0.40 ± 0.14	0.58 ± 0.21
BiLSTM	TopK evo	H	Yes	0.7	0.69 ± 0.10	0.37 ± 0.25	0.39 ± 0.32	0.34 ± 0.24	0.40 ± 0.14	0.58 ± 0.21
BiLSTM	TopK evo	H	Yes	0.8	0.69 ± 0.10	0.36 ± 0.26	0.38 ± 0.32	0.33 ± 0.25	0.40 ± 0.14	0.59 ± 0.21
BiLSTM	TopK evo	H	Yes	0.9	0.69 ± 0.10	0.35 ± 0.26	0.37 ± 0.33	0.33 ± 0.26	0.41 ± 0.15	0.59 ± 0.22
BiLSTM	TopK evo	H	Yes	1.0	0.69 ± 0.10	0.33 ± 0.26	0.37 ± 0.33	0.32 ± 0.25	0.41 ± 0.15	0.59 ± 0.21
BiLSTM	TopK evo	H	Yes	GS	0.65 ± 0.08	0.21 ± 0.19	0.35 ± 0.33	0.26 ± 0.24	0.39 ± 0.14	0.57 ± 0.19

Table 7.17: BiLSTM - proportion-based pseudo-labeling technique - 1/2

Model	PL	LT	PR	DL	Accuracy	Precision	Recall	F1	PR AUC	ROC AUC
BiLSTM_NES_N_O					0.65 ± 0.12	0.32 ± 0.15	0.31 ± 0.23	0.30 ± 0.16	0.42 ± 0.15	0.60 ± 0.19
BiLSTM	TopK fixed - 0.05	S	Yes	0.1	0.67 ± 0.14	0.30 ± 0.23	0.22 ± 0.18	0.23 ± 0.18	0.39 ± 0.15	0.60 ± 0.17
BiLSTM	TopK fixed - 0.05	S	Yes	0.2	0.66 ± 0.14	0.33 ± 0.22	0.33 ± 0.22	0.31 ± 0.21	0.38 ± 0.15	0.59 ± 0.18
BiLSTM	TopK fixed - 0.05	S	Yes	0.3	0.66 ± 0.14	0.35 ± 0.21	0.38 ± 0.27	0.34 ± 0.22	0.38 ± 0.15	0.59 ± 0.18
BiLSTM	TopK fixed - 0.05	S	Yes	0.4	0.66 ± 0.14	0.35 ± 0.20	0.38 ± 0.27	0.34 ± 0.22	0.39 ± 0.15	0.60 ± 0.18
BiLSTM	TopK fixed - 0.05	S	Yes	0.5	0.67 ± 0.14	0.35 ± 0.21	0.39 ± 0.29	0.35 ± 0.24	0.40 ± 0.16	0.60 ± 0.19
BiLSTM	TopK fixed - 0.05	S	Yes	0.6	0.68 ± 0.14	0.38 ± 0.24	0.37 ± 0.28	0.34 ± 0.23	0.40 ± 0.15	0.60 ± 0.20
BiLSTM	TopK fixed - 0.05	S	Yes	0.7	0.67 ± 0.14	0.37 ± 0.23	0.38 ± 0.28	0.34 ± 0.23	0.40 ± 0.15	0.60 ± 0.19
BiLSTM	TopK fixed - 0.05	S	Yes	0.8	0.67 ± 0.13	0.37 ± 0.23	0.38 ± 0.28	0.34 ± 0.23	0.39 ± 0.16	0.59 ± 0.20
BiLSTM	TopK fixed - 0.05	S	Yes	0.9	0.67 ± 0.14	0.36 ± 0.23	0.38 ± 0.28	0.34 ± 0.23	0.39 ± 0.15	0.59 ± 0.20
BiLSTM	TopK fixed - 0.05	S	Yes	1.0	0.67 ± 0.14	0.35 ± 0.23	0.38 ± 0.29	0.34 ± 0.24	0.39 ± 0.15	0.59 ± 0.20
BiLSTM	TopK fixed - 0.05	S	Yes	GS	0.69 ± 0.11	0.32 ± 0.22	0.40 ± 0.34	0.35 ± 0.27	0.38 ± 0.15	0.59 ± 0.20
BiLSTM	TopK fixed - 0.01	S	Yes	0.1	0.68 ± 0.04	0.23 ± 0.14	0.29 ± 0.25	0.25 ± 0.18	0.36 ± 0.07	0.57 ± 0.16
BiLSTM	TopK fixed - 0.01	S	Yes	0.2	0.68 ± 0.08	0.29 ± 0.18	0.31 ± 0.24	0.28 ± 0.19	0.34 ± 0.08	0.56 ± 0.16
BiLSTM	TopK fixed - 0.01	S	Yes	0.3	0.68 ± 0.09	0.30 ± 0.20	0.29 ± 0.25	0.28 ± 0.21	0.36 ± 0.13	0.57 ± 0.19
BiLSTM	TopK fixed - 0.01	S	Yes	0.4	0.68 ± 0.08	0.31 ± 0.19	0.31 ± 0.26	0.29 ± 0.21	0.36 ± 0.12	0.56 ± 0.19
BiLSTM	TopK fixed - 0.01	S	Yes	0.5	0.68 ± 0.09	0.31 ± 0.19	0.33 ± 0.27	0.30 ± 0.22	0.36 ± 0.12	0.57 ± 0.19
BiLSTM	TopK fixed - 0.01	S	Yes	0.6	0.68 ± 0.09	0.31 ± 0.20	0.34 ± 0.28	0.31 ± 0.23	0.36 ± 0.12	0.57 ± 0.19
BiLSTM	TopK fixed - 0.01	S	Yes	0.7	0.68 ± 0.09	0.32 ± 0.20	0.35 ± 0.28	0.32 ± 0.23	0.36 ± 0.13	0.58 ± 0.19
BiLSTM	TopK fixed - 0.01	S	Yes	0.8	0.69 ± 0.09	0.32 ± 0.20	0.35 ± 0.29	0.32 ± 0.23	0.37 ± 0.14	0.58 ± 0.20
BiLSTM	TopK fixed - 0.01	S	Yes	0.9	0.69 ± 0.09	0.31 ± 0.20	0.35 ± 0.28	0.32 ± 0.23	0.37 ± 0.14	0.58 ± 0.20
BiLSTM	TopK fixed - 0.01	S	Yes	1.0	0.68 ± 0.09	0.31 ± 0.20	0.35 ± 0.29	0.32 ± 0.23	0.37 ± 0.14	0.58 ± 0.20
BiLSTM	TopK fixed - 0.01	S	Yes	GS	0.65 ± 0.10	0.29 ± 0.19	0.38 ± 0.31	0.31 ± 0.24	0.37 ± 0.16	0.55 ± 0.20
BiLSTM	TopK fixed - 0.001	S	Yes	0.1	0.68 ± 0.09	0.25 ± 0.21	0.32 ± 0.29	0.28 ± 0.24	0.36 ± 0.10	0.58 ± 0.17
BiLSTM	TopK fixed - 0.001	S	Yes	0.2	0.68 ± 0.08	0.23 ± 0.21	0.31 ± 0.30	0.26 ± 0.24	0.38 ± 0.12	0.59 ± 0.17
BiLSTM	TopK fixed - 0.001	S	Yes	0.3	0.68 ± 0.08	0.23 ± 0.21	0.29 ± 0.27	0.25 ± 0.23	0.37 ± 0.12	0.59 ± 0.17
BiLSTM	TopK fixed - 0.001	S	Yes	0.4	0.68 ± 0.08	0.24 ± 0.21	0.29 ± 0.28	0.26 ± 0.24	0.38 ± 0.12	0.59 ± 0.17
BiLSTM	TopK fixed - 0.001	S	Yes	0.5	0.69 ± 0.08	0.24 ± 0.22	0.29 ± 0.28	0.26 ± 0.24	0.37 ± 0.12	0.59 ± 0.17
BiLSTM	TopK fixed - 0.001	S	Yes	0.6	0.68 ± 0.08	0.24 ± 0.22	0.29 ± 0.28	0.26 ± 0.24	0.37 ± 0.11	0.58 ± 0.17
BiLSTM	TopK fixed - 0.001	S	Yes	0.7	0.69 ± 0.08	0.24 ± 0.22	0.29 ± 0.28	0.26 ± 0.24	0.37 ± 0.11	0.58 ± 0.17
BiLSTM	TopK fixed - 0.001	S	Yes	0.8	0.69 ± 0.08	0.24 ± 0.22	0.29 ± 0.28	0.26 ± 0.24	0.37 ± 0.11	0.59 ± 0.17
BiLSTM	TopK fixed - 0.001	S	Yes	0.9	0.69 ± 0.08	0.25 ± 0.22	0.29 ± 0.28	0.26 ± 0.24	0.37 ± 0.12	0.59 ± 0.17
BiLSTM	TopK fixed - 0.001	S	Yes	1.0	0.69 ± 0.09	0.24 ± 0.22	0.29 ± 0.28	0.26 ± 0.24	0.37 ± 0.11	0.59 ± 0.18
BiLSTM	TopK fixed - 0.001	S	Yes	GS	0.68 ± 0.09	0.23 ± 0.22	0.26 ± 0.29	0.24 ± 0.25	0.37 ± 0.10	0.59 ± 0.18
BiLSTM	TopK evo	S	Yes	0.1	0.68 ± 0.09	0.30 ± 0.23	0.35 ± 0.28	0.32 ± 0.25	0.36 ± 0.12	0.58 ± 0.18
BiLSTM	TopK evo	S	Yes	0.2	0.69 ± 0.10	0.34 ± 0.20	0.34 ± 0.27	0.32 ± 0.22	0.37 ± 0.13	0.57 ± 0.16
BiLSTM	TopK evo	S	Yes	0.3	0.69 ± 0.09	0.33 ± 0.19	0.35 ± 0.27	0.32 ± 0.22	0.37 ± 0.12	0.57 ± 0.18
BiLSTM	TopK evo	S	Yes	0.4	0.68 ± 0.10	0.30 ± 0.20	0.31 ± 0.27	0.29 ± 0.22	0.37 ± 0.13	0.57 ± 0.19
BiLSTM	TopK evo	S	Yes	0.5	0.68 ± 0.10	0.30 ± 0.19	0.31 ± 0.27	0.29 ± 0.21	0.36 ± 0.12	0.56 ± 0.19
BiLSTM	TopK evo	S	Yes	0.6	0.68 ± 0.10	0.31 ± 0.20	0.33 ± 0.28	0.30 ± 0.22	0.36 ± 0.13	0.57 ± 0.19
BiLSTM	TopK evo	S	Yes	0.7	0.68 ± 0.10	0.31 ± 0.20	0.33 ± 0.28	0.30 ± 0.23	0.36 ± 0.13	0.57 ± 0.20
BiLSTM	TopK evo	S	Yes	0.8	0.68 ± 0.10	0.31 ± 0.20	0.32 ± 0.28	0.30 ± 0.23	0.36 ± 0.13	0.57 ± 0.20
BiLSTM	TopK evo	S	Yes	0.9	0.68 ± 0.10	0.31 ± 0.20	0.33 ± 0.28	0.30 ± 0.23	0.37 ± 0.14	0.58 ± 0.20
BiLSTM	TopK evo	S	Yes	1.0	0.68 ± 0.10	0.32 ± 0.20	0.33 ± 0.28	0.31 ± 0.22	0.37 ± 0.14	0.58 ± 0.20
BiLSTM	TopK evo	S	Yes	GS	0.66 ± 0.08	0.34 ± 0.14	0.38 ± 0.23	0.33 ± 0.16	0.38 ± 0.15	0.58 ± 0.20

Table 7.18: BiLSTM - proportion-based pseudo-labeling technique - 2/2

Model	PL	LT	PR	DL	Accuracy	Precision	Recall	F1	PR AUC	ROC AUC
BiLSTM_NES_N_O					0.65 ± 0.12	0.32 ± 0.15	0.31 ± 0.23	0.30 ± 0.16	0.42 ± 0.15	0.60 ± 0.19
BiLSTM	Opt - pr	H	No	/	0.69 ± 0.14	0.29 ± 0.24	0.23 ± 0.22	0.25 ± 0.22	0.40 ± 0.12	0.58 ± 0.15
BiLSTM	Opt - roc	H	No	/	0.64 ± 0.11	0.29 ± 0.12	0.34 ± 0.23	0.30 ± 0.16	0.36 ± 0.12	0.56 ± 0.17
BiLSTM	Opt - pr	H	Yes	/	0.69 ± 0.14	0.39 ± 0.20	0.26 ± 0.19	0.30 ± 0.19	0.46 ± 0.18	0.64 ± 0.21
BiLSTM	Opt - roc	H	Yes	/	0.66 ± 0.13	0.35 ± 0.16	0.37 ± 0.25	0.34 ± 0.19	0.41 ± 0.16	0.57 ± 0.19
BiLSTM	Opt - pr	S	Yes	/	0.71 ± 0.15	0.53 ± 0.31	0.29 ± 0.20	0.33 ± 0.20	0.42 ± 0.16	0.58 ± 0.20
BiLSTM	Opt - roc	S	Yes	/	0.65 ± 0.11	0.32 ± 0.12	0.37 ± 0.25	0.32 ± 0.17	0.40 ± 0.14	0.59 ± 0.18
BiLSTM	Opt - pr	H	Yes	0.1	0.70 ± 0.12	0.28 ± 0.26	0.26 ± 0.29	0.26 ± 0.26	0.42 ± 0.14	0.60 ± 0.16
BiLSTM	Opt - pr	H	Yes	0.2	0.69 ± 0.13	0.28 ± 0.24	0.25 ± 0.25	0.26 ± 0.24	0.42 ± 0.15	0.59 ± 0.18
BiLSTM	Opt - pr	H	Yes	0.3	0.69 ± 0.14	0.29 ± 0.25	0.26 ± 0.26	0.27 ± 0.24	0.44 ± 0.16	0.61 ± 0.19
BiLSTM	Opt - pr	H	Yes	0.4	0.69 ± 0.14	0.34 ± 0.28	0.27 ± 0.25	0.28 ± 0.23	0.45 ± 0.17	0.61 ± 0.19
BiLSTM	Opt - pr	H	Yes	0.5	0.70 ± 0.14	0.34 ± 0.28	0.28 ± 0.26	0.28 ± 0.24	0.44 ± 0.17	0.60 ± 0.19
BiLSTM	Opt - pr	H	Yes	0.6	0.70 ± 0.14	0.35 ± 0.27	0.28 ± 0.26	0.29 ± 0.24	0.45 ± 0.17	0.60 ± 0.19
BiLSTM	Opt - pr	H	Yes	0.7	0.70 ± 0.14	0.37 ± 0.28	0.28 ± 0.25	0.30 ± 0.23	0.44 ± 0.16	0.61 ± 0.18
BiLSTM	Opt - pr	H	Yes	0.8	0.69 ± 0.14	0.36 ± 0.27	0.27 ± 0.24	0.28 ± 0.23	0.45 ± 0.16	0.61 ± 0.18
BiLSTM	Opt - pr	H	Yes	0.9	0.69 ± 0.14	0.37 ± 0.28	0.27 ± 0.24	0.29 ± 0.22	0.44 ± 0.16	0.61 ± 0.18
BiLSTM	Opt - pr	H	Yes	1.0	0.69 ± 0.14	0.36 ± 0.28	0.27 ± 0.24	0.28 ± 0.22	0.44 ± 0.16	0.61 ± 0.18
BiLSTM	Opt - pr	H	Yes	GS	0.68 ± 0.14	0.29 ± 0.24	0.23 ± 0.22	0.25 ± 0.22	0.42 ± 0.13	0.61 ± 0.15
BiLSTM	Opt - roc	H	Yes	0.1	0.65 ± 0.13	0.33 ± 0.18	0.34 ± 0.23	0.32 ± 0.18	0.40 ± 0.12	0.59 ± 0.18
BiLSTM	Opt - roc	H	Yes	0.2	0.65 ± 0.12	0.32 ± 0.17	0.34 ± 0.23	0.32 ± 0.18	0.40 ± 0.13	0.59 ± 0.19
BiLSTM	Opt - roc	H	Yes	0.3	0.65 ± 0.13	0.32 ± 0.17	0.35 ± 0.24	0.32 ± 0.18	0.40 ± 0.13	0.59 ± 0.19
BiLSTM	Opt - roc	H	Yes	0.4	0.65 ± 0.13	0.32 ± 0.16	0.36 ± 0.24	0.32 ± 0.18	0.40 ± 0.13	0.58 ± 0.18
BiLSTM	Opt - roc	H	Yes	0.5	0.65 ± 0.12	0.32 ± 0.16	0.35 ± 0.24	0.32 ± 0.18	0.40 ± 0.13	0.58 ± 0.18
BiLSTM	Opt - roc	H	Yes	0.6	0.65 ± 0.12	0.32 ± 0.16	0.36 ± 0.24	0.32 ± 0.18	0.40 ± 0.13	0.58 ± 0.18
BiLSTM	Opt - roc	H	Yes	0.7	0.65 ± 0.12	0.32 ± 0.16	0.36 ± 0.24	0.32 ± 0.18	0.40 ± 0.13	0.58 ± 0.18
BiLSTM	Opt - roc	H	Yes	0.8	0.65 ± 0.12	0.33 ± 0.16	0.36 ± 0.24	0.33 ± 0.18	0.40 ± 0.13	0.58 ± 0.18
BiLSTM	Opt - roc	H	Yes	0.9	0.65 ± 0.12	0.33 ± 0.16	0.36 ± 0.24	0.33 ± 0.18	0.40 ± 0.13	0.58 ± 0.18
BiLSTM	Opt - roc	H	Yes	1.0	0.65 ± 0.12	0.33 ± 0.16	0.36 ± 0.24	0.33 ± 0.18	0.40 ± 0.13	0.58 ± 0.18
BiLSTM	Opt - roc	H	Yes	GS	0.65 ± 0.12	0.33 ± 0.14	0.4 ± 0.23	0.35 ± 0.17	0.41 ± 0.13	0.57 ± 0.18
BiLSTM	Opt - pr	S	Yes	0.1	0.63 ± 0.15	0.22 ± 0.20	0.20 ± 0.22	0.19 ± 0.17	0.38 ± 0.15	0.54 ± 0.21
BiLSTM	Opt - pr	S	Yes	0.2	0.65 ± 0.15	0.27 ± 0.23	0.23 ± 0.20	0.23 ± 0.18	0.38 ± 0.14	0.55 ± 0.20
BiLSTM	Opt - pr	S	Yes	0.3	0.66 ± 0.15	0.31 ± 0.23	0.27 ± 0.21	0.27 ± 0.19	0.41 ± 0.15	0.58 ± 0.21
BiLSTM	Opt - pr	S	Yes	0.4	0.67 ± 0.15	0.37 ± 0.28	0.30 ± 0.23	0.29 ± 0.20	0.41 ± 0.15	0.59 ± 0.21
BiLSTM	Opt - pr	S	Yes	0.5	0.66 ± 0.15	0.38 ± 0.30	0.28 ± 0.23	0.28 ± 0.20	0.42 ± 0.15	0.60 ± 0.21
BiLSTM	Opt - pr	S	Yes	0.6	0.67 ± 0.15	0.39 ± 0.29	0.28 ± 0.23	0.29 ± 0.21	0.42 ± 0.15	0.60 ± 0.21
BiLSTM	Opt - pr	S	Yes	0.7	0.67 ± 0.15	0.39 ± 0.28	0.29 ± 0.22	0.30 ± 0.21	0.41 ± 0.15	0.60 ± 0.21
BiLSTM	Opt - pr	S	Yes	0.8	0.68 ± 0.15	0.40 ± 0.30	0.28 ± 0.22	0.30 ± 0.21	0.42 ± 0.15	0.60 ± 0.21
BiLSTM	Opt - pr	S	Yes	0.9	0.68 ± 0.15	0.40 ± 0.30	0.28 ± 0.22	0.29 ± 0.21	0.41 ± 0.15	0.59 ± 0.21
BiLSTM	Opt - pr	S	Yes	1.0	0.68 ± 0.15	0.41 ± 0.30	0.28 ± 0.22	0.30 ± 0.21	0.41 ± 0.16	0.59 ± 0.21
BiLSTM	Opt - pr	S	Yes	GS	0.71 ± 0.16	0.55 ± 0.33	0.34 ± 0.21	0.38 ± 0.21	0.43 ± 0.18	0.59 ± 0.23
BiLSTM	Opt - roc	S	Yes	0.1	0.66 ± 0.13	0.33 ± 0.16	0.37 ± 0.25	0.34 ± 0.19	0.37 ± 0.13	0.60 ± 0.20
BiLSTM	Opt - roc	S	Yes	0.2	0.66 ± 0.13	0.34 ± 0.17	0.36 ± 0.24	0.33 ± 0.18	0.39 ± 0.14	0.59 ± 0.19
BiLSTM	Opt - roc	S	Yes	0.3	0.65 ± 0.12	0.33 ± 0.16	0.36 ± 0.24	0.33 ± 0.18	0.40 ± 0.15	0.59 ± 0.19
BiLSTM	Opt - roc	S	Yes	0.4	0.65 ± 0.12	0.34 ± 0.16	0.36 ± 0.25	0.33 ± 0.18	0.40 ± 0.15	0.59 ± 0.19
BiLSTM	Opt - roc	S	Yes	0.5	0.65 ± 0.12	0.33 ± 0.16	0.36 ± 0.24	0.33 ± 0.18	0.40 ± 0.15	0.59 ± 0.19
BiLSTM	Opt - roc	S	Yes	0.6	0.65 ± 0.12	0.32 ± 0.15	0.36 ± 0.24	0.32 ± 0.18	0.40 ± 0.15	0.59 ± 0.19
BiLSTM	Opt - roc	S	Yes	0.7	0.65 ± 0.12	0.32 ± 0.15	0.36 ± 0.25	0.32 ± 0.17	0.40 ± 0.15	0.59 ± 0.19
BiLSTM	Opt - roc	S	Yes	0.8	0.65 ± 0.12	0.32 ± 0.15	0.36 ± 0.24	0.32 ± 0.17	0.39 ± 0.15	0.58 ± 0.19
BiLSTM	Opt - roc	S	Yes	0.9	0.65 ± 0.12	0.32 ± 0.15	0.36 ± 0.24	0.32 ± 0.17	0.39 ± 0.15	0.58 ± 0.18
BiLSTM	Opt - roc	S	Yes	1.0	0.65 ± 0.12	0.31 ± 0.15	0.36 ± 0.24	0.32 ± 0.17	0.39 ± 0.15	0.58 ± 0.18
BiLSTM	Opt - roc	S	Yes	GS	0.64 ± 0.11	0.29 ± 0.13	0.34 ± 0.23	0.31 ± 0.17	0.38 ± 0.15	0.56 ± 0.19

Table 7.19: BiLSTM - optimal thresholding pseudo-labeling technique