Academic year
2024 - 2025

Context-Aware Retention Time Prediction for Enhanced Proteomics Analysis

Wannes Lamberts

Master's thesis

Supervisor
prof. dr. Wout Bittremieux, University of Antwerp

Cosupervisor
Ceder Dens, University of Antwerp

University of Antwerp
Faculty of Science

Acknowledgements

I thank my promoter, Prof. Dr. Wout Bittremieux, for providing the opportunity to do this thesis, offering valuable feedback, and including me in weekly lab meetings, which kept me motivated and on track.

Special thanks to Ceder Dens for his frequent guidance and meetings, which helped me grasp the biological background I initially lacked and improved the quality of my thesis through his feedback and suggestions. I also thank Charlotte Adam for providing final helpful feedback.

For the data collection pipeline, I acknowledge Wim Cuypers for his clear Nextflow explanations and useful references, which accelerated the learning process. Halil Ceylan is also thanked for assisting with data retrieval from the MassiveKB website.

Finally, I would like to acknowledge the CalcUA team for granting me access to their infrastructure and for providing comprehensive documentation that greatly facilitated its use.

Disclaimer

This thesis was written with the assistance of artificial intelligence tools for specific editorial and technical purposes. Throughout the writing process, AI was employed to assist with copy editing and text improvement through prompts such as ”Can you rewrite x” for improving clarity and readability, ”Can you restructure x” for enhancing organization and flow, and ”Can you look for grammar errors in x” for grammatical corrections. Additionally, AI tools were utilized to assist with coding tasks.

All content, research methodology, experimental design, data analysis, conclusions, and work presented in this thesis remain original work. AI tools were used as editorial and technical assistants to improve the presentation and implementation of our research and ideas. The use of AI did not influence the research findings and scientific conclusions of the work presented. I take full responsibility for the accuracy and integrity of all content in this thesis.

Abstract

Proteomics, the large-scale study of proteins, is essential for understanding biological processes. An important technique within this field is liquid chromatography-mass spectrometry (LC-MS), which enables the identification and quantification of peptides in complex biological samples. A crucial element in LC-MS analyses is the retention time (RT) of peptides, which serves as an additional dimension to improve the accuracy of peptide identifications. However, traditional RT prediction models treat peptides as isolated entities and do not account for the influence of sample context, such as interactions between peptides and matrix effects, which can influence chromatographic behavior.

In this thesis, we investigated whether it is feasible and meaningful to add sample context to models for predicting retention time. We conducted a feasibility study in two steps. First, we analyzed the extensive MassiveKB dataset to test the hypothesis that sample context actually influences peptide retention time during liquid chromatography. This analysis showed that retention time variation for the same peptide was greater between different MS runs than within a single MS run. Since different MS runs typically analyze different samples, this result supports the hypothesis and shows the potential value of context-aware models.

Subsequently, we developed a context-aware retention time prediction model based on machine learning techniques, aiming to improve prediction accuracy compared to traditional sequence-based approaches. The model uses amino acid sequences and integrates information about sample context by analyzing all peptides in an MS run. To isolate the impact of sample context as effectively as possible, we used the base model from [Dens et al., 2024], which has a similar structure but only uses sequences without context information. The results show a clear improvement: the context-aware model achieved a 12.3% improvement compared to the base model.

These findings show that incorporating sample context is not only feasible but could also improve the accuracy of retention time predictions. The model has potential for applications in peptide-spectrum match (PSM) rescoring, which can strengthen the reliability of peptide identifications. This study opens new possibilities for more robust and informative proteomics analyses.

Nederlandstalige samenvatting

Proteomics, de grootschalige studie van eiwitten, is essentieel voor het begrijpen van biologische processen. Een belangrijke techniek binnen dit veld is liquid chromatography–mass spectrometry (LC-MS), die de identificatie en kwantificering van peptiden in complexe biologische monsters mogelijk maakt. Een cruciaal element in LC-MS-analyses is de retentietijd (RT) van peptiden, die dient als een extra dimensie om de nauwkeurigheid van peptide-identificaties te vergroten. Traditionele RT-voorspellingsmodellen behandelen peptiden echter als geïsoleerde entiteiten en houden geen rekening met de invloed van de monstercontext, zoals interacties tussen peptiden en matrixeffecten, die het chromatografische gedrag kunnen beïnvloeden.

In deze thesis werd onderzocht of het haalbaar en zinvol is om samplecontext toe te voegen aan modellen voor het voorspellen van retentietijd. Hiervoor voerden we een haalbaarheidsstudie uit in twee stappen. Eerst analyseerden we de uitgebreide MassiveKB-dataset om de hypothese te toetsen dat de context van het monster daadwerkelijk invloed heeft op de retentietijd van peptiden tijdens liquid chromatography. Uit deze analyse bleek dat de variatie in retentietijd voor eenzelfde peptide groter was tussen verschillende MS-runs dan binnen één enkele MS-run. Aangezien verschillende MS-runs doorgaans verschillende monsters analyseren, ondersteunt dit resultaat de hypothese en benadrukt het de meerwaarde van contextbewuste modellen.

Vervolgens ontwikkelden we een contextbewust retentietijd-voorspellingsmodel, gebaseerd op machine learning-technieken, met als doel de voorspellingsnauwkeurigheid te verbeteren ten opzichte van traditionele sequentie benaderingen. Het model gebruikt de aminozuursequenties en integreert informatie over de monstercontext door alle peptiden in een MS-run te analyseren. Om de impact van de monstercontext zo goed mogelijk te isoleren, werd het basismodel van [Dens et al., 2024]. gebruikt, dat een vergelijkbare structuur heeft maar enkel de sequenties gebruikt zonder contextinformatie. De resultaten tonen een duidelijke verbetering: het contextbewuste model behaalde een verbetering van 12,3% in de mediane absolute fout ten opzichte van het basismodel.

Deze bevindingen tonen aan dat het incorporeren van monstercontext niet alleen haalbaar is, maar ook de nauwkeurigheid van retentietijdvoorspellingen verbetert. Het model heeft potentieel voor toepassingen in peptide-spectrum match (PSM) rescoring, wat de betrouwbaarheid van peptide-identificaties in MS-workflows kan versterken. Deze studie opent nieuwe mogelijkheden voor robuustere en informatievere proteomics-analyses.

Contents

List of Figures

List of Tables

List of Acronyms

RT Retention Time

iRT Indexed Retention Time

PSM Peptide Spectrum Match

MAD Median Absolute Deviation

MAE Mean Absolute Error

MedAE Median Absolute Error

MSE Mean Squared Error

LC Liquid Chromatography

MS Mass Spectrometry

KDE Kernel Density Estimation

LC-MS Liquid Chromatography - Mass Spectrometry

FDR False Discovery Rate

MAE Mean Absolute Deviation

PEP Posterior Error Probability

MLP Multilayer Perceptron

TPE Tree-structured Parzen Estimator

1. Background

This thesis operates within the field of bioinformatics, an interdisciplinary domain that requires foundational knowledge of both computational and biological sciences to fully appreciate the research contributions. This chapter provides essential background information necessary to understand the work presented in subsequent chapters. We begin with a focused overview of molecular biology fundamentals covering proteins, peptides, and amino acids. This is followed by an introduction to mass spectrometry-based proteomics, the analytical technique that generated the data that we used in this study.

1.1. Background molecular biology

1.1.1. Proteins

Proteins are large, complex molecules that perform essential functions in all living organisms. They serve numerous functions including catalyzing biochemical reactions as enzymes, providing structural support, regulating physiological processes as hormones, and defending against pathogens as antibodies [Boundless, 2023b]. Proteins are composed of long chains of amino acids that fold into specific three-dimensional structures, with this structure determining their function [Boundless, 2023a][Alberts et al., 2024, pp. 6, 115-116].

In proteomics research, proteins are typically too large and complex to analyze directly by mass spectrometry. Therefore, they must first be broken down into smaller and more manageable fragments called peptides through enzymatic digestion, most commonly using the enzyme trypsin [Shuken, 2023].

1.1.2. Peptides

Peptides are shorter chains of amino acids that typically contain fewer than 50 amino acid residues [Cell Guidance Systems,]. In mass spectrometry-based proteomics, peptides serve as analytical targets that represent the proteins from which they originated [Shuken, 2023]

Peptides can be viewed as informative fragments of proteins. When proteins are enzymatically digested, they produce characteristic peptide patterns that can be used to identify the original proteins. As these peptide chains are analyzed, their amino acid sequences provide the key information needed for protein identification. The boundary between what constitutes a peptide versus a protein is somewhat arbitrary but is generally based on size and structural complexity.

1.1.3. Amino Acids

Amino acids are the fundamental building blocks of proteins and peptides [Boundless, 2023a][Alberts et al., 2024, p. 6]. There are 20 standard amino acids found in the human body, each with unique properties that contribute to protein structure and function. Every amino acid shares a common core structure consisting of

a central carbon atom (alpha carbon), an amino group (-NH2), a carboxyl group (-COOH), a hydrogen atom, and a variable side chain (R group) [Boundless, 2023c]. This structure can be seen in Figure 1.1.

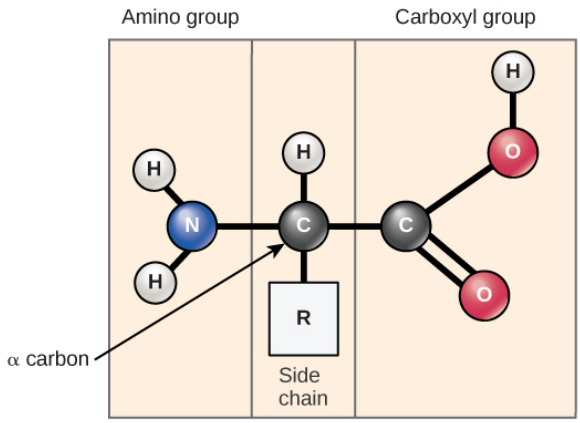


Figure 1.1.: Structural framework of amino acids showing the common core structure with variable side chain[Boundless, 2023c]

This common structural framework allows amino acids to link through peptide bonds, forming chains that ultimately create proteins and peptides. It is the side chain that distinguishes one amino acid from another. These side chains vary greatly in their chemical properties, which determines how amino acids interact with each other and their surrounding environment, ultimately influencing the chromatographic behavior crucial to this thesis.

For convenience in scientific communication, each amino acid can be represented by a three-letter abbreviation (e.g., Ala for alanine, Cys for cysteine) or a single-letter code (e.g., A for alanine, C for cysteine). These single-letter codes allow protein and peptide sequences to be expressed as strings of letters, enabling efficient communication and analysis of the complex sequences that make up proteins.

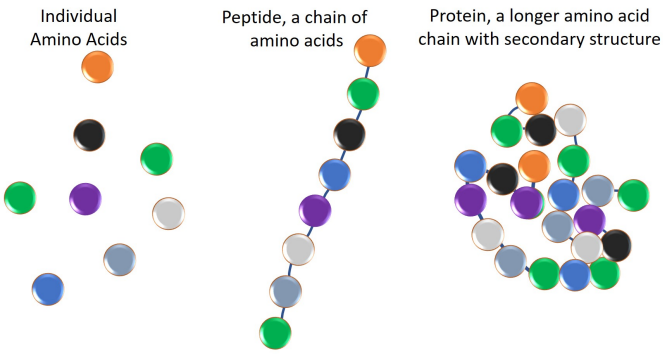


Figure 1.2.: Simple representation of the relation between Amino Acids, peptides, and proteins [Cell Guidance Systems,]

1.1.4. Synthesis of proteins

DNA serves as the genetic blueprint in all living organisms, storing information as a double-stranded helix composed of four nucleotides (A, T, C, and G). Through the processes of transcription and translation, this genetic information is directing the synthesis of proteins. During transcription, DNA is used

to create messenger RNA (mRNA), which then guides the assembly of amino acids into specific protein sequences during translation [Alberts et al., 2024, p. 5].

This flow from DNA to RNA to proteins is often referred to as the central dogma of molecular biology, describing how genetic information is expressed in the form of functional proteins. Figure 1.3 provides a simple view of this process.

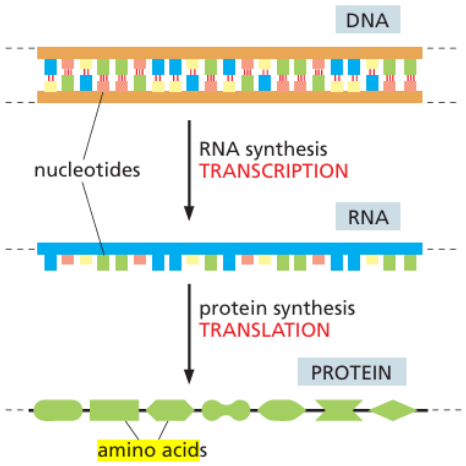


Figure 1.3.: From DNA to protein: the flow of genetic information through transcription and translation [Alberts et al., 2024, p. 5].

1.2. Mass Spectrometry-based proteomics

Mass spectrometry-based proteomics is a powerful analytical approach that enables the identification and quantification of proteins in complex biological samples. The workflow begins with protein extraction and enzymatic digestion to generate peptides, followed by separation using liquid chromatography, mass spectrometric analysis, and computational database searching for peptide identification. Figure 1.4 illustrates this comprehensive workflow, showing how biological samples are processed through each stage to ultimately identify and quantify proteins.

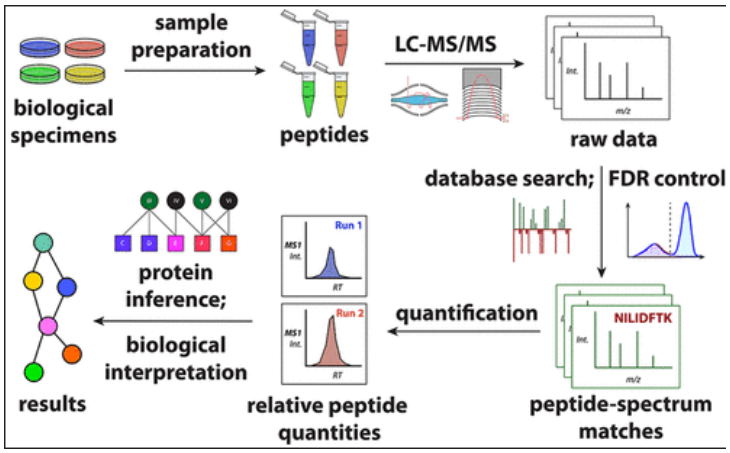


Figure 1.4.: Overview of standard LC-MS workflow and components leading to protein identification [Shuken, 2023].

1.2.1. Liquid chromatography and retention time

Liquid Chromatography (LC) serves as the separation technique that precedes mass spectrometric analysis [Kailasam, 2021]. The process involves introducing the peptide mixture into a column containing a stationary phase. A mobile phase carries the sample through the column, where peptides interact with the stationary phase on the basis of properties such as hydrophobicity. These interactions cause peptides to separate and exit the column at different times, known as Retention Time (RT).

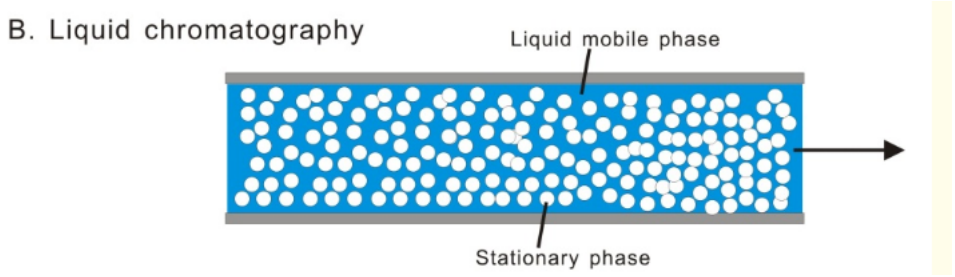


Figure 1.5.: Illustration of chromatographic column displaying mobile and stationary phases in the column.

RT represents the duration it takes for a specific peptide to travel through the chromatographic column. In LC-MS workflows, retention time serves as an essential piece of experimental data, complementing mass-to-charge (m/z) measurements to characterize peptides in complex biological samples [Kailasam, 2021]. This additional dimension can be particularly valuable for distinguishing between spectra that appear similar but originate from different peptides.

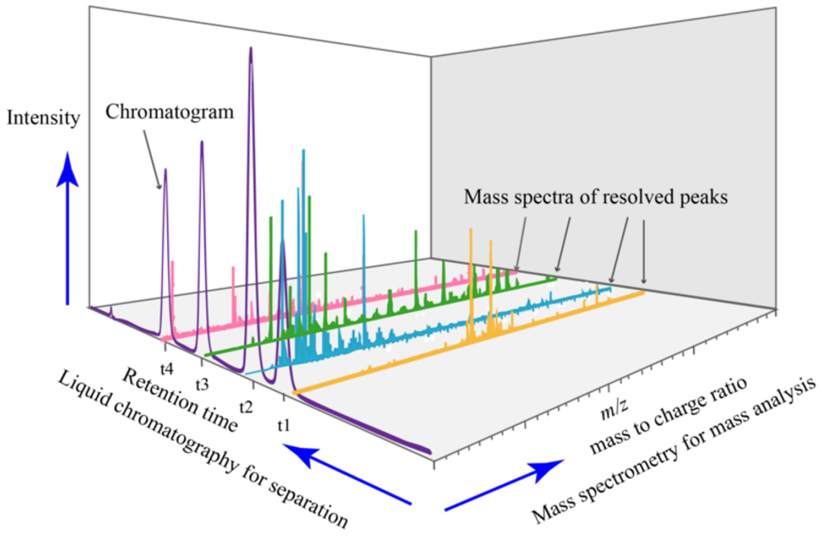


Figure 1.6.: Figure demonstrating how retention time adds an orthogonal dimension to m/z analysis [Kailasam, 2021].

LC reduces sample complexity by spreading peptides across a time axis, preventing the mass spectrometer from being overwhelmed by simultaneous detection. Effective LC produces narrow elution peaks, enhancing signal intensity and aiding in the detection of low-abundance proteins. An example of a chromatogram can be seen in figure 1.7.

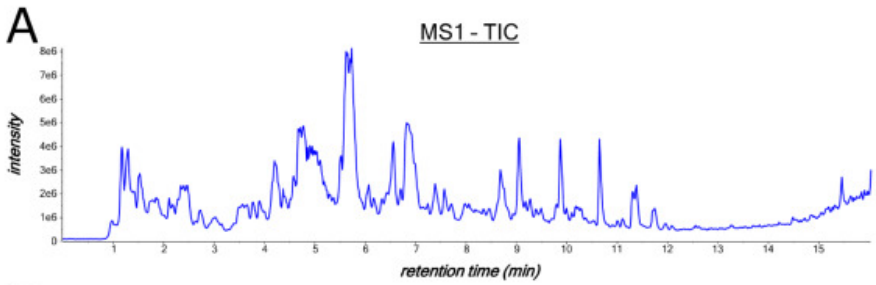


Figure 1.7.: Chromatogram displaying narrow elution peaks [Klont and Hopfgartner, 2021].

1.2.2. Peptide Interactions in Chromatography have impact on retention time

In complex biological samples analyzed by LC-MS, hundreds to thousands of peptides are simultaneously present during the chromatographic separation process. This creates a dynamic environment where peptides interact with each other during LC, potentially influencing their RT as they travel through the column.[ML et al., 2023, Mant et al., 2007, Åsberg et al., 2017] Understanding these interactions is crucial because most current retention time prediction models treat each peptide as an isolated entity, ignoring the potential influence of co-eluting compounds. However, several mechanisms suggest that the behavior of peptides in chromatography is inherently context-dependent.

1. **Co-eluting peptides competing for binding sites:** When multiple peptides travel through the chromatography column simultaneously, they compete for limited binding sites in the stationary phase (the material inside the column). This competition can alter the strength with which each peptide binds, potentially causing some peptides to elute (exit the column) earlier than predicted based on their properties alone. For example, in LC, peptides compete for hydrophobic binding sites, so a more hydrophobic peptide competes more successfully compared to a less hydrophobic peptide [Mant et al., 2007], this competition intensifies in complex samples with many peptides.

2. **Peptide-peptide interactions during separation:** During separation, peptides can interact directly through mechanisms such as ion-pairing (where oppositely charged peptides form bonds), hydrogen bonding, or hydrophobic interactions. [Åsberg et al., 2017] describes similar interactions between peptides and the mobile phase, but these principles also apply to peptide-peptide interactions. These interactions can alter the effective size, charge, or hydrophobicity of the peptides, thereby affecting their RT. For example, a negatively charged peptide pairing with a positively charged one may undergo interactions leading to chromatographic behavior distinct from that of the individual peptides.

3. **Matrix effects from sample composition:** The overall sample matrix (all components present in the sample) can significantly impact how peptides behave during chromatography [ML et al., 2023]. High-abundance components can alter the properties of the mobile phase, previously eluted peptides can modify the column surface, and complex biological systems can create microenvironments with conditions different from those of the bulk mobile phase. These effects become particularly relevant in real-world proteomic samples, where thousands of peptides are analyzed simultaneously.

These interaction mechanisms demonstrate that peptide behavior in LC is inherently context-dependent, suggesting that accurate RT prediction may require consideration of the complete sample composition rather than treating peptides as isolated entities.

1.2.3. Mass spectrometry

Following chromatographic separation, the peptides enter the mass spectrometer, where they undergo ionization and analysis in two phases. First, MS1 spectra are collected, recording the mass-to-charge (m/z) ratios of intact peptides to map the peptide population. Specific peptide ions from each MS1 spectrum are then selected for fragmentation, producing multiple MS2 spectra that capture the m/z ratios of peptide fragments [Shuken, 2023].

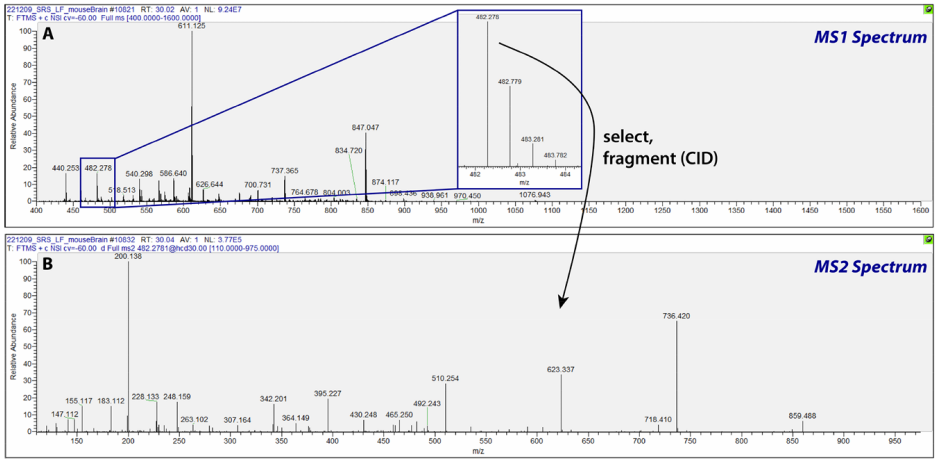


Figure 1.8.: MS1 spectrum paired with its corresponding MS2 fragmentation spectrum [Shuken, 2023].

A single MS1 scan can yield several MS2 spectra according to the number of targeted precursor ions, providing detailed fragmentation information necessary for the identification of peptides. The fragmentation patterns captured in the MS2 spectra are crucial for the identification of peptides.

1.2.4. Database search and peptide identification

The final step involves matching experimental MS2 spectra with theoretical spectra from known protein databases to identify peptides. By comparing the observed MS2 fragment ions with the predicted fragmentation patterns, peptides can be accurately identified, allowing reliable protein identification in the sample.

This process can be enhanced by incorporating additional peptide properties, such as RT. Tools such as MaxQuant [Tyanova et al., 2016] demonstrate how RT information can be integrated with spectral data to improve peptide identification accuracy through correlation analysis and advanced scoring algorithms.

1.3. Computer science background

This section provides essential background on the computational techniques used in the RT prediction models developed in this thesis. We cover neural networks as the foundational machine learning architecture and transformer architecture as a breakthrough in sequence modeling that forms the basis of our approach.

1.3.1. Neural networks

A neural network is a computational model inspired by the way biological neurons in the brain process information. It consists of interconnected layers of artificial neurons that receive input data, apply mathematical transformations through weighted connections, and pass signals forward to produce an output. Each connection has an adjustable weight that determines the strength of the signal, and these weights are learned through data training. The network learns to recognize patterns by adjusting these weights to minimize errors between its predictions and the correct answers.

1.3.2. Transformer Architecture

The transformer is a neural network architecture introduced by [Vaswani et al., 2017] that revolutionized how machines process sequential data like text or, in our case, amino acid sequences. The main benefit is that transformers can examine all parts of a sequence simultaneously through a mechanism called ”attention.”

The key is the attention mechanism, which allows the model to focus on different parts of the input sequence when making predictions. For example, when a sequence is processed, the transformer can simultaneously consider how different amino acids relate to each other, regardless of their position in the sequence.

What makes transformers especially powerful for this thesis is their ability to learn long-range dependencies. The separation of amino acids by many positions can still influence each other’s behavior and the effect on the RT on the peptide as a whole.

1.3.3. Bert

The BERT model [Devlin et al., 2019], is a transformer-based architecture derived from the encoder component of the transformer framework. Unlike traditional unidirectional language models, BERT processes input sequences bidirectionally, capturing the context of the preceding and following tokens simultaneously. This bidirectional approach enables BERT to generate rich, context-aware representations, achieving state-of-the-art performance across numerous natural language processing (NLP) tasks.

In the context of this thesis, BERT is adapted to process amino acid sequences for retention time prediction. Consider the sequence ’...-C-E-D-E-R-...’. A unidirectional model would analyze the central amino acid D using either the preceding context (C, E) or the following context (E, R), but not both. However, BERT examines both, enabling a nuanced understanding of the role of D based on its full context. This capability is critical for accurate RT predictions, as peptide behavior depends on complex interactions among amino acids.

2. Research objective

2.1. Central Hypothesis

Although the mechanisms of peptide interactions during chromatography are well-established (Section 1.2.2), the quantitative impact of these interactions on the prediction of retention time has not been fully explored. We hypothesize that the magnitude of peptide-peptide interactions within a sample affects individual peptide RT during LC, and that the incorporation of sample context into predictive models will improve the accuracy retention time prediction compared to models that treat peptides as isolated entities.

2.2. Reasearch aims

This feasibility study aims to evaluate the feasibility of a context-aware RT prediction model, Specifically, we will:

1. **Quantify the impact of sample context on peptide retention times** by analyzing how the RT of the same peptide behaves differently between samples, confirming the relevance of the problem.

2. **Develop a context-aware prediction model** that incorporates the complete sample composition alongside the target peptide sequence, testing the potential to account for peptide interactions and matrix effects.

3. **Compare context-aware predictions with traditional approaches** to assess whether including sample context improves accuracy, which shows us the effectiveness of adding context.

By addressing these objectives, this feasibility study will contribute to our understanding of the scale peptide interactions influence RT during LC in complex samples and potentially enhance the accuracy of RT prediction in LC-MS workflows. Improved RT prediction, in turn, can lead to more reliable peptide identification and quantification in proteomics studies.

3. Related Work

The prediction of peptide RT in LC-MS has been a cornerstone of proteomics research, providing an orthogonal dimension to enhance peptide identification and quantification. Recent advances in computational methods, particularly deep learning, have significantly improved the precision and applicability of RT prediction.

3.1. DeepLC

In the domain of peptide RT prediction for LC-MS proteomics, DeepLC, introduced by [Bouwmeester et al., 2021], represents a significant advance, particularly for the handling of modified peptides. DeepLC employs a deep convolutional neural network to predict retention times, leveraging a novel peptide encoding strategy based on atomic composition (e.g., counts of carbon, hydrogen, oxygen) rather than traditional one-hot encoding of amino acids or modifications. This approach enables DeepLC to generalize to modified peptides, including those not encountered during training, addressing a critical limitation of prior models such as DeepRT and Prosit, which struggle with unseen modifications due to their reliance on modification-specific training data.

DeepLC achieves performance comparable to that of state-of-the-art models for unmodified peptides, with Pearson correlations exceeding 0.98 in diverse data sets (e.g. SWATH Library, HeLa HF) and relative mean absolute errors below 1.5% for most LC setups. Its ability to predict RT for modified peptides is particularly relevant for open search workflows, where various post-translational modifications are common. By encoding modifications at the atomic level, DeepLC accurately predicts RT for unseen modifications, as demonstrated in the ProteomeTools PTM dataset. Additionally, DeepLC can flag potentially incorrect Peptide Spectrum Match (PSM) in open searches by identifying large RT prediction errors for unexpected modifications, enhancing identification reliability.

However, deep LC does not account for the context in which a peptide lives for the prediction of the RT.

3.2. Chronologer

[Wilburn et al., 2023] introduced Chronologer, an open-source software tool designed for rapid and accurate RT prediction of peptides, including those with diverse post-translational modifications (PTMs). Chronologer leverages a residual convolutional neural network (RCNN) trained on a massive, harmonized database (Chronologer-DB) comprising over 2.2 million peptides from eleven community datasets, covering ten PTM types. This extensive dataset enables Chronologer to outperform other deep learning tools, such as Prosit by achieving a 36-65% reduction in prediction error in various experimental conditions. The authors emphasize the importance of data harmonization, achieved through a novel in silico alignment strategy using KDE to map diverse datasets into a common RT space. This approach mitigates the challenges posed by experimental variations in LC systems, columns, and gradients, which are critical for robust RT prediction. Because of the good mapping of RT this chronologer-DB dataset will be used in this thesis for RT calibration between different runs.

Chronologer's ability to predict RT for peptides with rare PTMs, such as O-GlcNAc, using as few as 10-100 training examples, highlights its efficiency in transfer learning. The tool's architecture incorporates a Laplace-based loss function with False Discovery Rate (FDR) masking, which accounts for inherent errors in proteomics data, further enhancing the reliability of predictions. Although Chronologer represents a significant advancement, its predictions are sequence-based, focusing on amino acid sequences, without explicitly considering the broader sample context of the sample in which the peptide lives.

3.3. Machine learning strategies

[Dens et al., 2024] provided a comprehensive survey of machine learning strategies in proteomics, emphasizing the need for large-scale, high-quality datasets to train robust models. Their analysis revealed that RT prediction accuracy improves with larger training datasets, supporting the need for expansive datasets to capture the complexity of peptide behavior under diverse experimental conditions.

To address data scarcity, [Dens et al., 2024]. explored algorithmic strategies such as self-supervised pretraining and multitask learning. Their evaluation of transformer encoders pre-trained on large protein datasets demonstrated improved RT prediction performance compared to models trained from scratch. The use of multitask learning, where models simultaneously predict multiple peptide properties, further mitigates data limitations by leveraging shared features across tasks.

These advanced machine learning approaches provide a methodological foundation for our research objective. By incorporating contextual features from the entire sample, we can extend these methods to develop a more comprehensive model that takes into account peptide interactions during LC.

4. Dataset

A key aspect of this thesis was securing an appropriate dataset to test the hypothesis that sample a peptide lives in influences its RT during LC. To confirm this, we will compare RT variations for the same peptide between multiple measurements within a single sample to the variation between different samples. Greater variation between different samples would support the hypothesis. To achieve this, the dataset must fulfill two essential criteria:

1. Sufficient peptides must appear multiple times within a single MS run.
2. Sufficient peptides must appear across heterogeneous MS runs.

If the second criterion is met and a trained model is developed, we can further assess how much the sample context affects RT predictions.

Throughout this and subesquent chapters, we use specific terminology: ”sample” refers to the collection of peptides analyzed together in a single MS run, while ”run” denotes the individual MS run performed on a sample. Each run generates a set of PSMs with associated retention times. Importantly, different runs typically analyze different samples, so when comparing peptide RT between runs, we are effectively comparing across different samples. Additionally, the MassiveKB dataset, used later in this thesis, is composed of several distinct datasets, which we will refer to as ”experiments”.

4.1. Data Processing and Analysis Methods

4.1.1. Retention time calibration

Retention times between different MS runs are often shifted due to variations in experimental conditions, including differences in chromatographic columns, mobile phase compositions, flow rates, temperature conditions, and instrument-specific factors. These systematic shifts can cause the same peptide to elute at different RT between runs, even when analyzed under identical conditions.

For RT prediction models, it is essential that retention times are on a comparable scale across all runs to enable effective learning and accurate predictions. Therefore, RT must be calibrated when creating datasets that combine data from multiple experimental conditions and instruments.

In this thesis, retention times were calibrated to a iRT scale using the Chronologer dataset [Wilburn et al., 2023] as a reference . For each LC-MS run, a linear regression model was fitted to map the retention times of detected calibration peptides to their corresponding iRT values. Then, this model was applied to predict the iRT values for all the peptides in the run. An example can be seen in figure 4.1, where the x-axis shows the original RT and the y-axis shows the calibrated iRT.

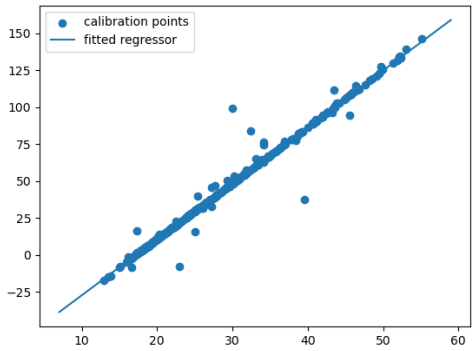


Figure 4.1.: Example of retention time calibration through linear regression using Chronologer for calibration points.

4.1.2. Median absolute deviation

For the analysis of datasets, MAD is used. The formal definition of MAD is presented in Equation 4.1.

$$\text{MAD} = \text{median}(|X_i - \text{median}(X)|) \tag{4.1}$$

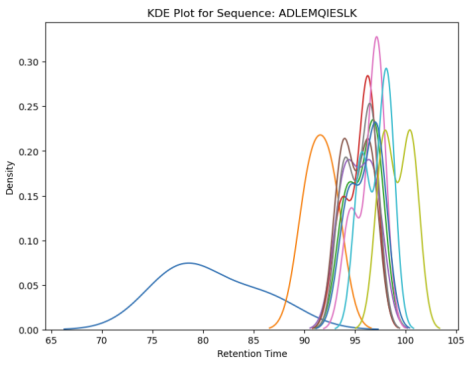
where X is the dataset and X_i represents each individual data point within the dataset.

Initially, MAE was employed in this thesis. However, we transitioned to using MAD because we found that our datasets had a considerable number of outliers. These outliers adversely affect MAE calculations, resulting in misleading analytical results that did not accurately represent the true nature of the data.

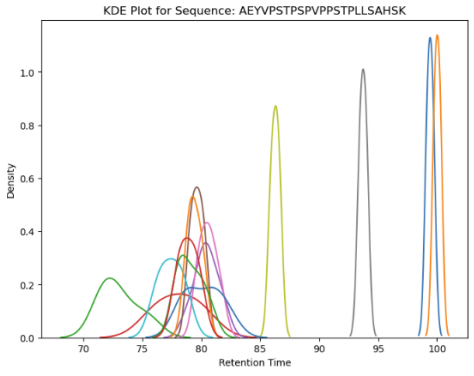
4.1.3. KDE plots

In the analysis, we will use KDE plots [Waskom, 2021] to visualize the distribution of RT for specific peptides between different runs. If we observe variations in these distributions, it could indicate that sample structure influences peptide RT.

In figure 4.2 you can observe an example of where the distributions are similar and where they differ significantly.



(a) Example of similar distributions



(b) Example of different distributions

Figure 4.2.: Example of KDE plots for peptide retention time distributions

4.1.4. Wasserstein distance

While the KDE plots are good for visualizing the distributions for specific peptides, it is impossible to do this for all peptides manually. However, there is a distance that is called the Wasserstein distance [Leo et al., 2023] that we use as a measure to measure the distance between the RT distributions in different runs. The Wasserstein distance, also known as the earth mover distance, provides a measure of the dissimilarity between two probability distributions by quantifying the minimal amount of 'work' needed to transform one distribution into the other. This metric is particularly well suited for comparing the overall shape and alignment of RT distributions.

$$W_1(\mu,\nu) = \inf_{\gamma\in\Gamma(\mu,\nu)} \int_{\mathbb{R}\times\mathbb{R}} |x-y|\, d\gamma(x,y) \tag{4.2}$$

1. Where μ, ν are retention time distributions for a peptide in two different samples
2. Where $\Gamma(\mu,\nu)$ is the set of all joint distributions

For each peptide, we calculate all pairwise Wasserstein distances and compute their mean. These mean distances across all peptides can be visualized in a KDE plot to show the distribution of mean Wasserstein distances. Although this plot alone does not provide direct conclusions, it enables comparison between datasets.

4.2. ProteomeTools

The ProteomeTools project [Zolg et al., 2017] aims to derive molecular and digital tools from the human proteome to facilitate biomedical and life science research. This initiative has generated and performed MS analysis of more than 350,000 synthetic tryptic peptides that represent nearly all human gene proteins. The resource is projected to expand to 1.4 million peptides in a two-year period, with all data made publicly available through ProteomicsDB [Schmidt et al., 2018]. Individual tryptic peptides were synthesized using solid phase synthesis methodology, a technique where peptides are built step-by-step while attached to an insoluble support, allowing for efficient sequential addition of amino acids. These peptides were then combined into pools of approximately 1,000 peptides each. These pools were subsequently analyzed on an Orbitrap Fusion mass spectrometer. For each peptide pool, an inclusion list was created to specifically target peptides for fragmentation in additional MS experiments. Five distinct fragmentation methods were used: HCD, CID, ETD, EThCD, and ETciD, with readout performed using either ion trap or Orbitrap technologies. Furthermore, HCD spectra were recorded at six different collision energy levels to provide comprehensive fragmentation profiles.

All MS runs underwent individual analysis using MaxQuant version 1.5.3.30, ensuring consistent processing parameters in the dataset. MaxQuant is a widely used computational proteomics software platform specifically designed for analyzing large MS datasets [Tyanova et al., 2016] . The software performs peptide identification by matching experimental MS spectra against theoretical fragmentation patterns from peptide sequence databases.

4.2.1. Data structure

The dataset is available here under the project files, specifically within the ZIP files, which are the only required components. Each ZIP file is named in a way that encodes key information about the experimental run, such as the type of mass spectrometer used, the fragmentation strategy, and other relevant metadata. Of particular importance to this thesis is the pool number, which is also included in the filename.

Within each ZIP archive, there is a file referred to as filename that lists all the peptides identified from a single pool, along with associated data.

4.2.2. Preprocessing

Before analyzing the dataset, we performed two pre-processing steps to ensure data quality and consistency.

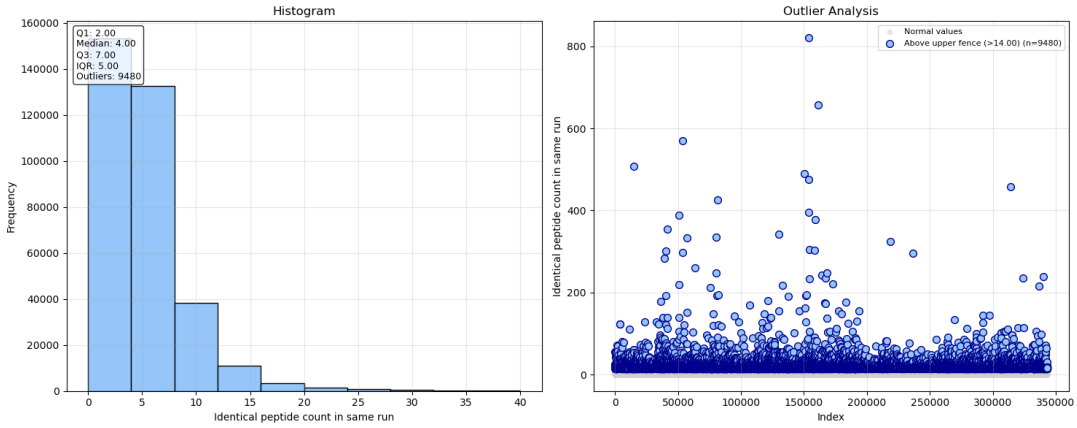
First, we applied high-confidence filtering to retain only peptides identified with high reliability. Each PSM in the Proteometools dataset includes a Score, which reflects how closely the observed spectrum matches a theoretical peptide (higher scores indicate better matches) and a Posterior Error Probability (PEP), which estimates the likelihood of incorrect peptide identification (lower values signify greater confidence). For this study, we kept only peptide identifications with a score above 90 and a PEP below 0.01 to ensure high-quality data.

Next, we performed retention time calibration to standardize RT across all runs, obtaining iRT values as described in Section 4.1.1. This step was essential for making RT comparable across the varied experimental conditions in the ProteomeTools dataset, allowing consistent and reliable analysis.

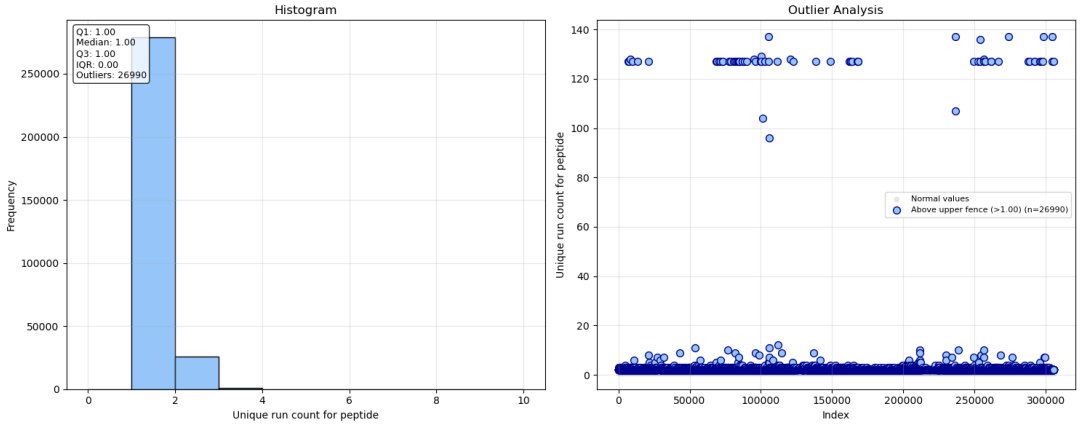
4.2.3. Analysis

After preprocessing, the dataset contained 1.7 million PSM's, comprising 137 unique pools and over 300,000 unique peptides. Each pool is considered one MS run, as the same sample composition was used for all MS runs within a pool, as explained in Section 4.2.

Peptide distribution analysis



(a) Histogram displaying peptide frequency distribution within a single run, with scatterplot highlighting outlier observations.



(b) Histogram displaying frequency of unique runs per peptide, with scatterplot highlighting outlier observations.

Figure 4.3.: Distribution analysis of peptides across the proteometools dataset.

To evaluate the suitability of the dataset for studying sample-dependent RT effects, we analyzed peptide distribution patterns both within individual runs and across different runs. Figure 4.3a shows that many peptides appear multiple times within individual MS runs, with some extreme outliers showing a very high frequency.

However, Figure 4.3b reveals a critical limitation: the analysis shows that nearly all peptides are unique to a single run, with only a few outliers present in almost all runs. These outliers consist mainly of calibration peptides intentionally added to samples and common contaminants. The lack of cross-run peptide overlap significantly limits the dataset's utility for studying sample-dependent RT effects,

Retention time analysis

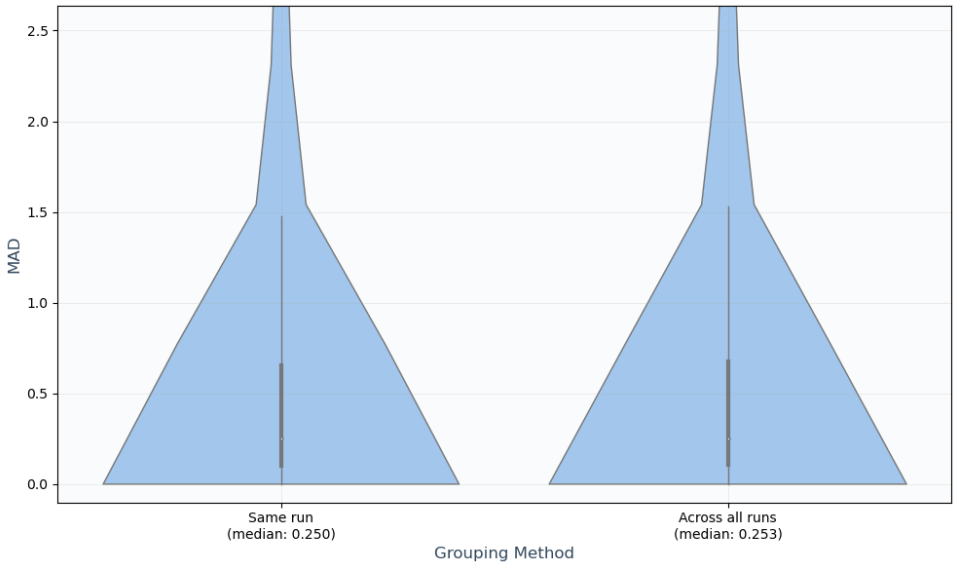


Figure 4.4.: Violin plots comparing MAD iRT distributions within the same run versus across all runs, showing nearly identical distributions.

We investigated the MAD of iRT for identical peptides within the same run and in different runs. However, this comparison was limited because most peptides were unique to a single run, as noted earlier. Figure 4.4 displays the distribution of iRT for the same peptide in all runs and within the same run, revealing nearly the same distributions. This similarity was anticipated as a result of the predominance of peptides unique to individual runs.

To further explore RT behavior across runs, we generated KDE plots for peptides present in multiple runs, illustrating the iRT distributions for specific peptides in each run. These plots, shown in Figure 4.4, indicate highly similar distributions, suggesting minimal variation in iRT across runs in this dataset.

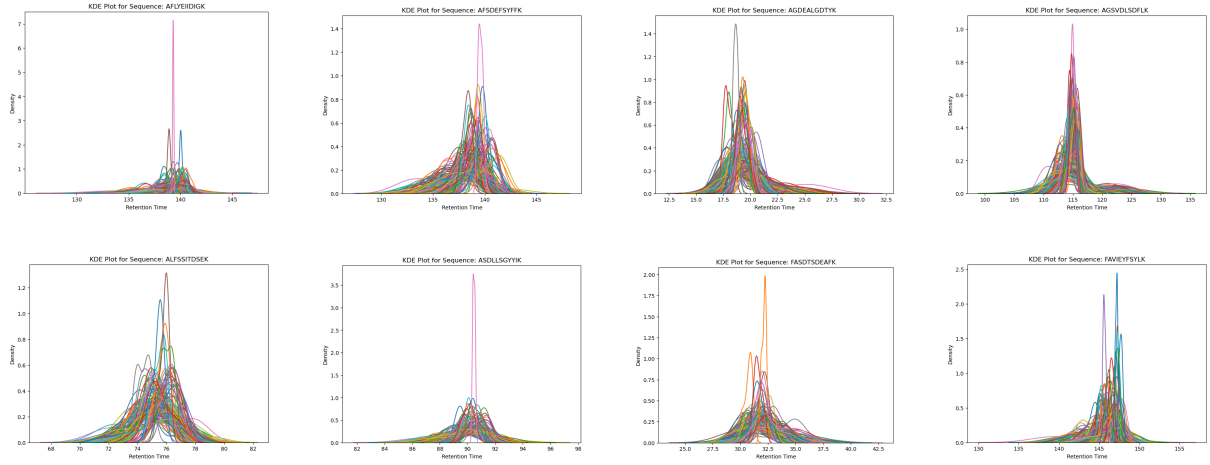


Figure 4.5.: KDE plots of peptides present in multiple runs showing similar iRT distributions across runs.

4.3. MassiveKB

As highlighted in the previous section, the initial Proteometools dataset showed fundamental limitations. To address this, we adopted the MassiveKB project data set, a large-scale initiative focused on reanalyzing public proteomics data to build a robust spectral library. This project, detailed in [Wang et al., 2018], provides a community-scale knowledge base that significantly improves the depth and reliability of proteomics data for this study.

The MassiveKB project starts with the community contributing data to the MassIVE repository. This repository contains 31.4 TB of human HCD data with 658 million MS spectra, comprising 227 datasets covering 27,404 MS runs. This represents a massive collection of proteomics data from 120 research groups around the world, ensuring a diverse representation of biological samples and experimental conditions. In figure 4.6 is an overview of the community working together to build the massive-KB-knowledge base.

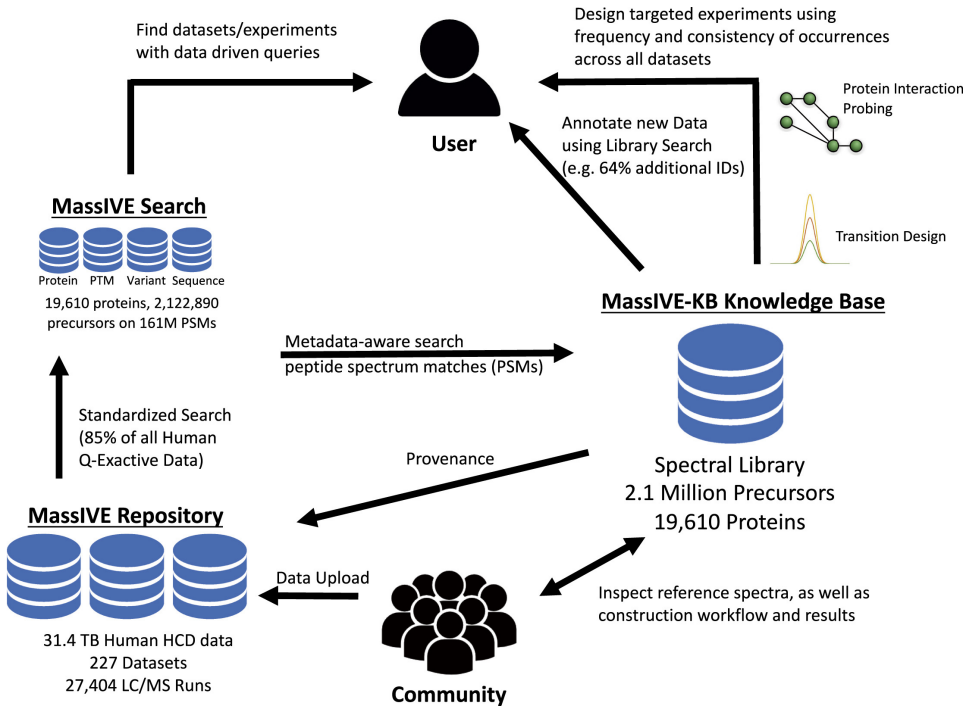


Figure 4.6.: Visualisation of community working together to build massive-KB-knowledge base [Wang et al., 2018]

Of the 658 million spectra, 191 million PSM's were confidently identified at a 1% FDR using the MSGF+ search engine [Kim and Pevzner, 2014]. To handle precursors with multiple spectra, the team selected a single representative spectrum for each precursor based on its similarity to others, resulting in 30.16 million PSM's. A final MSGF+ analysis removed ambiguous spectra with multiple annotations, ensuring the dataset's reliability. This process yielded a high-quality spectral library containing 2.12 million precursors from 19,610 proteins.

To ensure sufficient and diverse data for this thesis, we will utilize a subset of the 191 million PSM's from the MassiveKB dataset, which conveniently includes RT information ideal for our analysis. Due to the large volume of data, collecting it required a specialized pipeline, which is detailed in the Appendix B.

4.3.1. Data structure

The MassiveKB dataset is accessible through a public repository, which provides the 30 million representative PSM's along with their associated metadata. For researchers requiring the complete dataset, all 191 million PSM's can be retrieved by accessing the original search tasks referenced in each PSM's metadata.

Each search task contains an mzTab file that stores comprehensive information about its PSM's. However, a significant limitation is that the RT is not included directly in these mzTab files. Instead, RT information must be extracted from raw mass spectrometry files (in mzML or mzXML format) that are referenced in the metadata.

These raw MS files contain the complete scan data for each experiment and include the RT values that we require. To link a specific PSM to its retention time, we developed a mapping system using a composite key consisting of the filename and scan number. This mapping allows us to systematically retrieve retention times for all PSM's in the dataset, enabling the analysis required for this thesis. A diagram showing the structure can be seen in figure 4.7.

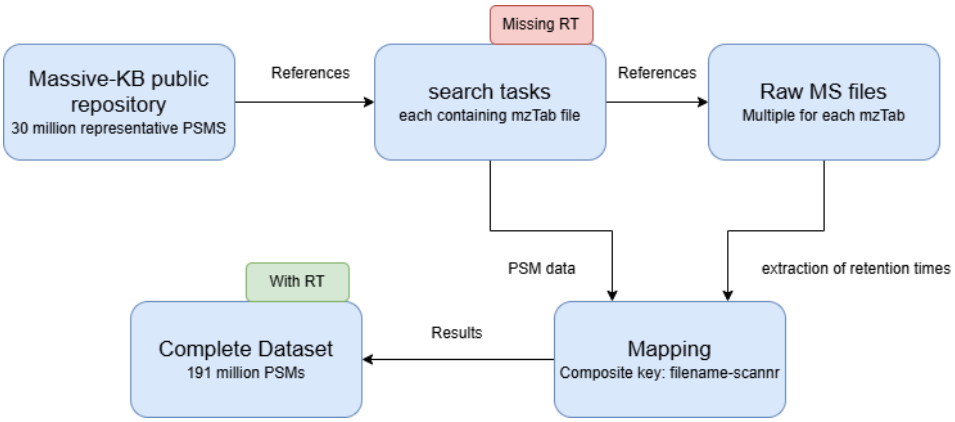


Figure 4.7.: Massive-KB structural diagram showing the process for retention time data collection.

4.3.2. Preprocessing

Before analyzing the dataset, we performed two preprocessing steps to ensure data quality and consistency.

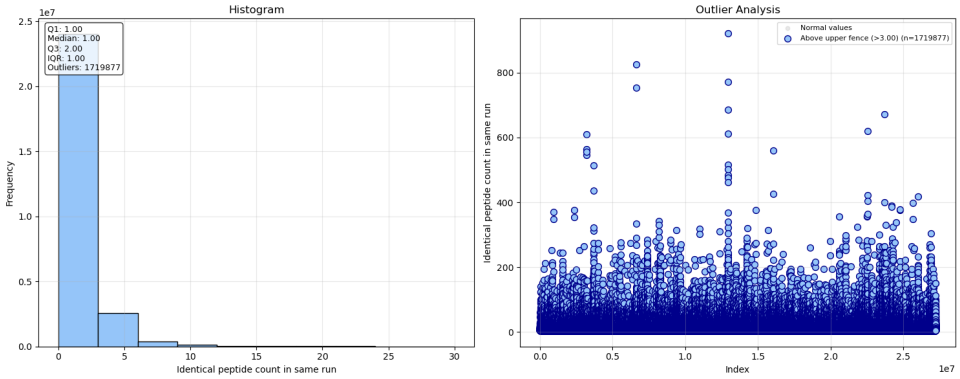
First, we removed duplicates identified during data collection from MassiveKB. We found instances where multiple sequences in the same run mapped to identical scan numbers in the raw file, corresponding to the exact same retention times. To eliminate this redundancy, we retained only the first occurrence of each duplicate.

Next, we performed RT calibration to standardize RT across all runs, obtaining iRT values as detailed in Section 4.1.1. This calibration step was particularly important for the MassiveKB dataset, as this represent a collection from a considerable number of different sources, making RT standardization essential for reliable analysis.

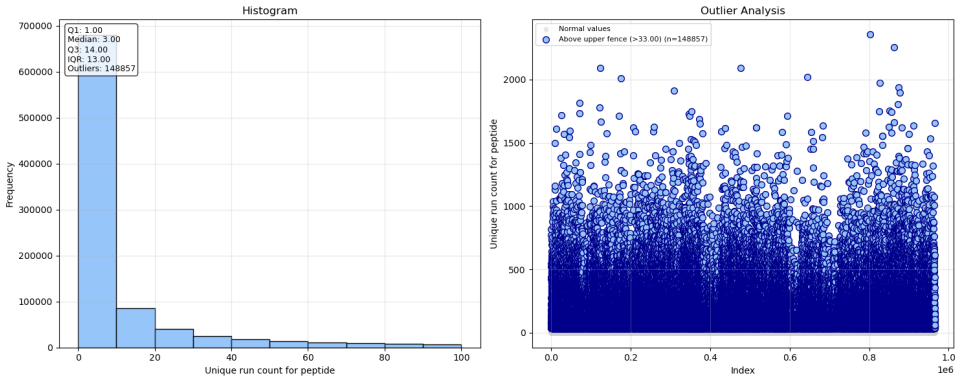
4.3.3. Analysis

Due to the enormous volume of massiveKB data, we began with a limited dataset of 80 tasks. As model performance improved with additional training, we expanded our collection to 165 tasks. Our final collected dataset encompasses 45 million PSM's from 62 distinct datasets which we will refer to as experiments, representing 2,892 independent MS runs. In this section, we analyze how well this data supports our thesis objectives.

Peptide distribution analysis



(a) Histogram displaying peptide frequency distribution within a single run, with scatterplot highlight- ing outlier observations.



(b) Histogram displaying frequency of unique runs per peptide, with scatterplot highlighting outlier observations.

Figure 4.8.: Distribution analysis of peptides in the MassiveKB dataset.

We performed the same analysis as performed for the ProteomeTools dataset, examining peptide distribution patterns both within individual runs and across different runs. Figure 4.8a shows that while many peptides appear only once within the same MS run, a substantial 8 million times a peptide was found at least twice within the same run.

In contrast to the ProteomeTools data, Figure 4.8b reveals that the MassiveKB dataset contains a considerable number of peptides that exist across runs. Around 600,000 peptides appear in at least two different runs, with some rare cases occurring in up to 2,000 heterogeneous runs.

Retention time analysis

We conducted a RT analysis to assess the suitability of the MassiveKB dataset for this thesis by examining the MAD of RT for identical peptides under three different conditions: within the same run, across runs within the same experiment, and across all runs in the entire dataset. Figure 4.9 illustrates

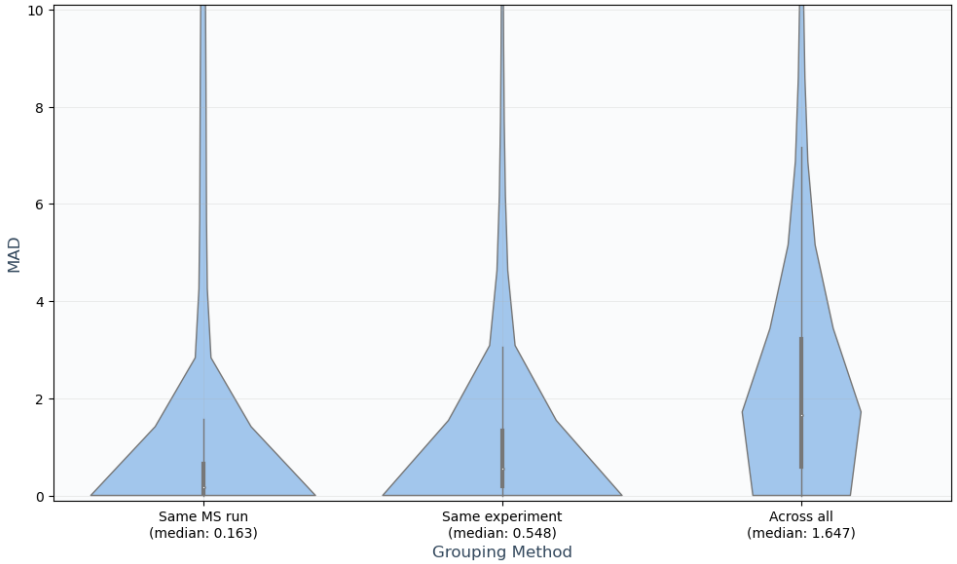


Figure 4.9.: Violin plot showing the MAD iRT for the same peptide within the same MS run, across runs within the same experiment and across all runs in the massiveKB dataset.

these three distributions, revealing a clear pattern. The MAD of retention times was lowest for identical peptides within the same MS run, which is expected since these peptides originate from the same sample. Slightly higher MAD values were observed for identical peptides within the same experiment, consistent with a modest variation among similar samples in an experiment. Finally, the highest MAD values were found when comparing identical peptides in all runs in the dataset, reflecting the greater variability between the different conditions.

The KDE plots for certain peptides in the MassiveKB dataset revealed distributions that appear less similar than those observed in the ProteomeTools dataset. However, since it is difficult to draw definitive conclusions from these visualization plots alone, we rely on the Wasserstein distance analysis (described in Section 4.1.4) presented in the next section to support our findings.

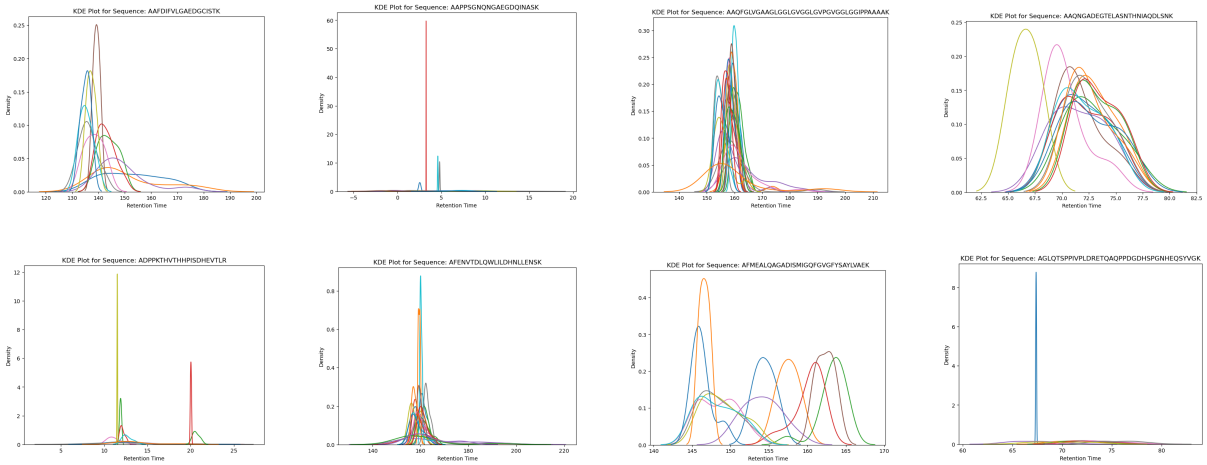


Figure 4.10.: KDE plots showing less similar iRT distribution for peptides across multiple runs.

4.4. Dataset Evaluation Summary

The ProteomeTools dataset, despite its high quality, was determined to be unsuitable for this thesis because of a critical limitation: insufficient peptide presence across heterogeneous MS runs. A primary requirement of this research was the ability to track peptides under diverse sample conditions, but most peptides in ProteomeTools were confined to individual runs, preventing effective assessment of RT variability across different samples.

MassiveKB emerged as a superior alternative, offering a substantial collection of PSMs. Distribution analysis confirmed that numerous peptides appear across multiple runs in MassiveKB, satisfying the heterogeneity requirements essential for this research. Furthermore, RT analysis revealed meaningful sample-dependent variation in peptide behavior.

While KDE plots suggested somewhat less similarity in RT distributions compared to ProteomeTools (though these represent only selected examples rather than the complete dataset), the Wasserstein distance analysis presented in Figure 4.11 provided definitive evidence supporting MassiveKB's superiority. For peptides appearing across multiple runs, the Wasserstein distance analysis demonstrates higher RT variance in MassiveKB compared to ProteomeTools, further validating MassiveKB's suitability for studying peptide behavior across heterogeneous conditions. This increased variance confirms MassiveKB's alignment with the thesis objectives, which specifically require the examination of peptide behavior under diverse experimental conditions.

The scale, diversity, and demonstrable retention time characteristics across experimental runs establish MassiveKB as the optimal resource for this research, effectively addressing the limitations identified in the ProteomeTools dataset.

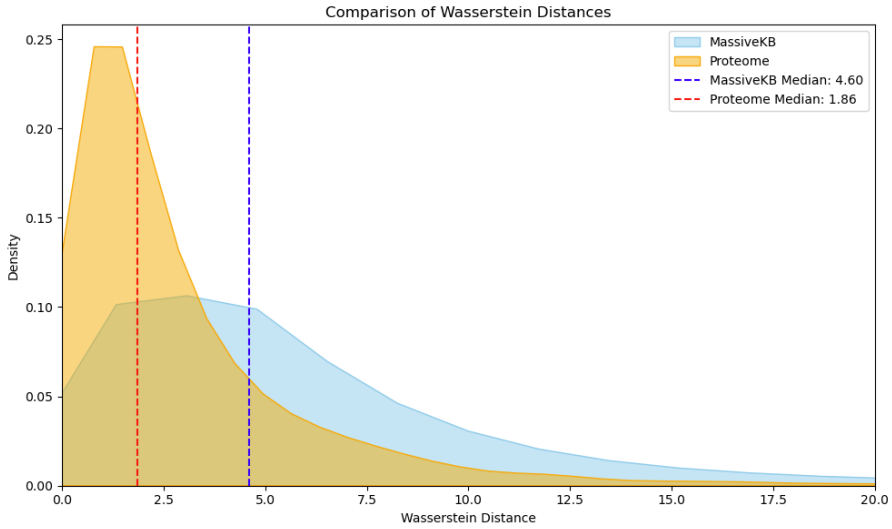


Figure 4.11.: KDE plot showing distribution of mean Wasserstein distances revealing higher values for MassiveKB compared ProteomeTools.

5. Methodology

5.1. Data collection and processing

5.1.1. Dataset Selection Rationale

Based on the comprehensive evaluation of the dataset presented in Chapter 4, we selected the MassiveKB dataset for model training and evaluation. As demonstrated in the dataset analysis, MassiveKB provided superior characteristics for studying sample-dependent retention time effects.

- **Scale:** 191 million PSMs from 227 distinct datasets across 27,404 independent runs
- **Heterogeneity:** Sufficient peptide overlap across different runs.
- **Retention time variability:** Evidence of sample-dependent effects with meaningful variation between runs.

The ProteomeTools dataset, while high quality, was excluded due to insufficient peptide presence across heterogeneous MS runs, which limited its utility for studying context-dependent RT effects.

5.1.2. MassiveKB Pipeline

The MassiveKB dataset required a sophisticated pipeline due to its larger volume and complex data structure. We developed a Nextflow-based workflow pipeline with flexibility in mind, enabling its use beyond the scope of this thesis. The pipeline incorporates several key features: user-friendly deployment requiring only Docker/Apptainer and Nextflow, flexible command-line configuration and cross-platform support for both local and HPC execution.

The pipeline was executed on the Calcua High-Performance Computing cluster [Mant et al., 2007]. Our collected dataset comprises 45 million PSMs derived from 62 datasets encompassing 2,892 individual MS runs. For a detailed description of the pipeline we strongly suggest to take a look at Appendix B, as it constitutes a significant portion of the work in this thesis.

5.1.3. Preprocessing

While we already did some preprocessing for the analysis of MassiveKB which we explained in section 4.3.2 consisting of duplicate removal and RT calibration, we did one additional step to improve model training and evaluation.

We addressed cases where identical peptide sequences appeared multiple times within the same run. To maintain data consistency while preserving representative information, we removed these duplicates retaining only the observation with the median iRT value among all instances. This approach effectively minimized the impact of potential outliers while ensuring that each unique peptide-run combination was

represented only once, thus reducing noise in our data. Important to note is that this wasn't done in our analysis stage because we needed these identical peptide-run instances for comparison.

5.1.4. Data split

To ensure robust model training and evaluation, we implemented a careful data splitting strategy that prioritized preventing data leakage while maintaining realistic evaluation conditions. We assigned two complete experiments from the MassiveKB collection for held-out evaluation: MSV000080865 for validation and MSV000080276 for testing, with all remaining experiments designated for training. We chose to split at the experiment level rather than randomly because runs within the same experiment tend to share similar characteristics and experimental conditions.

Preventing peptide overlap was critical to avoid biased evaluation results. Without proper separation, models could appear to perform well on test data simply because they had encountered identical peptides during training, leading to overly optimistic performance estimates that would not generalize to real-world applications where researchers analyze novel peptide sequences.

Our splitting procedure followed the following steps:

1. **Identification of overlapping peptides:** We systematically identified all peptides that appeared in multiple experiments in the dataset.

2. **Balanced overlap distribution:** These shared peptides were evenly divided (50/50) between the validation and test sets to ensure that both sets of evaluation contained comparable complexity.

3. **Complete removal from training:** All peptides present in the validation or test sets were completely removed from the training data, ensuring zero overlap between the training, validation, and test sets.

This approach was essential for realistic performance evaluation, as it simulates the real-world scenario where models must predict RT for previously unseen peptides. Without this separation, we risked developing models that memorized specific peptide-retention time relationships.

An important constraint was to limit our remaining data to one experiment each for validation and testing. Although using multiple experiments for each evaluation set would have been ideal, our overlap removal procedure meant that including additional experiments in the test and validation sets would have required the removal of substantially more peptides from the training data, potentially compromising model learning. This trade-off balanced evaluation rigor with sufficient training data availability.

5.1.5. Final Dataset Composition

After collecting the data, applying all preprocessing steps and executing the data split, we get the dataset composition shown in Table 5.1.

Dataset	Unique Peptides	PSMs	Experiments	Unique Runs
Train Set	784,016	7,786,779	60	2,710
Validation Set	89,039	249,607	1 (MSV000080865)	82
Test Set	79,949	247,344	1 (MSV000080276)	100
Total	**953,004**	**8,283,730**	**62**	**2,892**

Table 5.1.: Table showing final dataset composition for model training and result evaluation.

5.2. Model architectures

5.2.1. Architecture Overview and Design Rationale

This thesis compares two models for peptide RT prediction: a base model that uses only peptide sequences and a context-aware model that incorporates the composition of the sample. Both models are based on the transformer architecture by [Vaswani et al., 2017], specifically the BERT architecture [Devlin et al., 2019], adapted to processing amino acid sequences.

To ensure we can measure the impact of adding sample context, both models utilize identical architectures for sequence processing. They share the same sequence processing pipeline architecture and identical pre-trained weights from [Dens et al., 2024]. This design ensures that any performance differences can be attributed to the incorporation of sample context rather than fundamental architectural variations, potentially showing that incorporating context is feasible.

The key innovation lies in the dual encoder approach of the context-aware model, which processes both individual peptide sequences and complete sample compositions to generate context-aware predictions.

5.2.2. Sequence Processing Pipeline

In this section we will go over the components of the shared sequence processing pipeline, the complete configuration parameters can be found in table 5.2.

Parameter	Value
encoder blocks	9
Attention Heads	12
Hidden Size	180
feed forward Size	720
Dropout	0.1
Activation Function	gelu
Sequence Length	50 (padded)

Table 5.2.: BERT sequence processing pipeline configuration (Adopted from [Dens et al., 2024])

The shared processing pipeline works as follows; input sequences are first padded to a uniform length of 50 amino acids and tokenized using a predefined vocabulary. The embedding layer transforms these tokenized inputs into 180-dimensional vectors through two complementary components: token ID embeddings that provide unique vectors for each amino acid and token position embeddings that encode sequential information. As illustrated in Figure 5.1, these two embedding types are summed together and then layer normalized to create the initial input representations that enter the encoder stack.

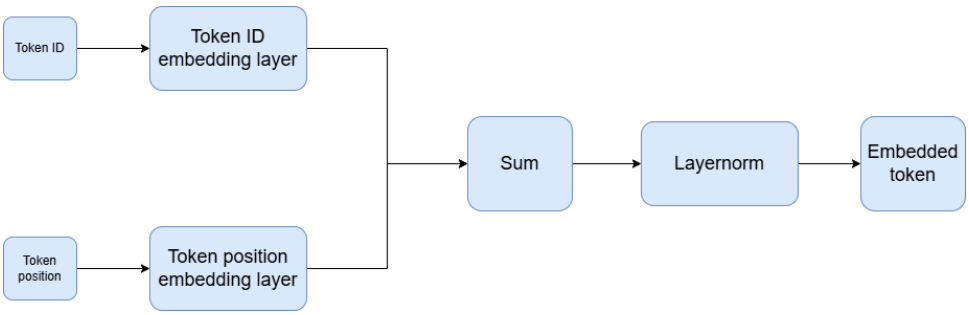


Figure 5.1.: Embedding block architecture converting token IDs and positional information into token representations.

The encoder processes these embeddings through multiple stacked BERT encoder blocks, each incorporating multi-head self-attention to capture contextual relationships among amino acids, feedforward networks for token-wise processing, and layer normalization with residual connections for training stability. This encoding stage produces enriched peptide representations that capture the complex interactions necessary for RT prediction.

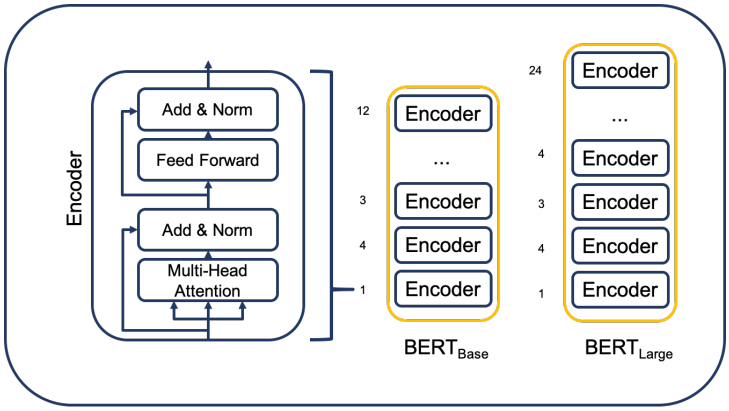


Figure 5.2.: BERT encoder architecture transforming embeddings into enriched peptide representations.

5.2.3. Base Model architecture

The base model architecture serves as our comparison baseline, treating each peptide as an isolated entity. After processing through the shared sequence processing pipeline described above, the sequence representation flows to a Multilayer Perceptron (MLP) that transforms the pooled sequence representation into RT predictions, mapping the 180-dimensional output to a scalar RT value.

This architecture successfully captures sequence-specific retention time patterns, but ignores potential influences from the sample context, such as peptide interactions and matrix effects. An overview of the base model architecture from sequence to prediction can be seen in figure 5.3.

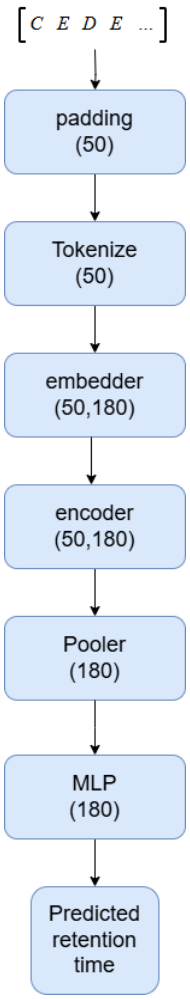


Figure 5.3.: Complete Base model architecture by [Dens et al., 2024]. Showing the flow from peptide sequence to retention time prediction.

5.2.4. context aware model architecture

The context-aware model extends the base architecture through a dual encoder approach that incorporates the information of the sample environment. The model first processes the complete sample, consisting of all peptides present in a given run, through a frozen version of the sequence processing pipeline to generate contextualized representations of the sample. These representations are then aggregated through mean pooling across all dimensions to create a comprehensive context token that encodes the overall sample environment.

Simultaneously, the target peptide sequence is processed through a separate, trainable sequence processing pipeline to produce sequence-specific representations. The peptide's representation is then concatenated with the sample context token before feeding into the MLP output layer for RT prediction. An overview of our context-aware architecture can be seen in figure 5.4

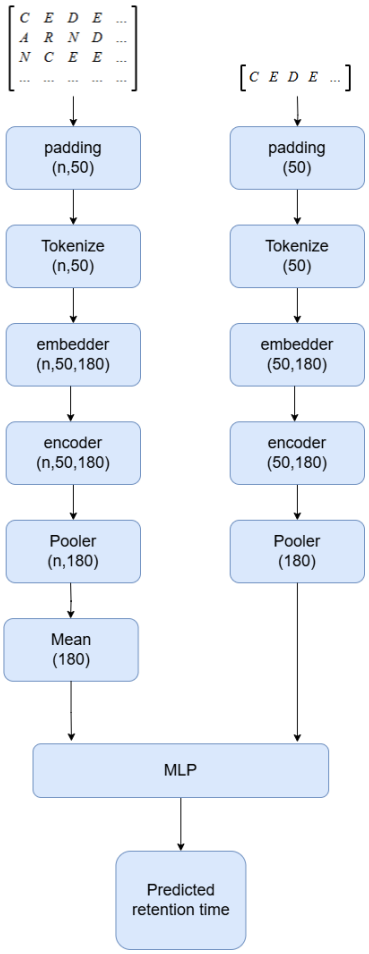


Figure 5.4.: Complete context-aware model architecture: left processes sample context tokens, right handles target peptide sequences, with concatenation for retention time prediction.

Computational Optimisation

To enhance computational efficiency, the context-aware model optimizes the calculation of the sample context. Since the components used to generate the sample context have frozen weights, the context for a given sample remains constant during training and prediction. Instead of recalculating the sample context for each peptide, the model computes it once per sample and stores it in a lookup table. For each peptide's RT prediction, the corresponding sample context is retrieved from this table and concatenated with the peptide's enriched token representationsr. This approach significantly reduces redundant computations, improving the model's efficiency.

5.3. Training Methodology

The training process for the BERT-based models (base and context-aware) consists of two sequential phases: pretraining and fine tuning. These steps enable models to learn contextual relationships in peptide sequences and predict RT accurately. This type of training is know as semi-supervised learning.

5.3.1. pretraining

In the first phase, known as pretraining, a mask head is added on top of the encoder. During this phase, a few amino acids in the peptide are hidden using a masking technique. By predicting these hidden tokens, the model learns to understand the context and relationships within the peptide sequence, enabling it to capture meaningful patterns and dependencies. An example of this process, but for sentence prediction, can be seen in figure 5.5

The main advantage here is that there is no need for labeled data, since the only input required is the peptide itself. This type of training is known as unsupervised learning. In this thesis, we used the pre-trained weights from [Dens et al., 2024].

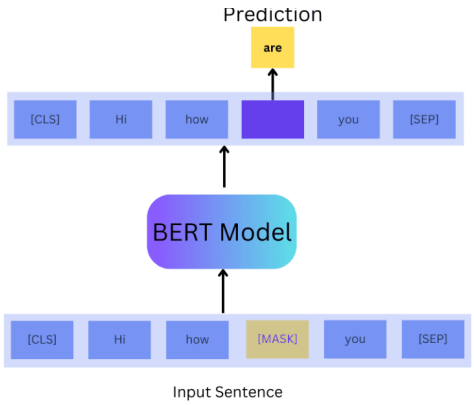


Figure 5.5.: Pretraining example demonstrating BERT's masked token prediction process.

5.3.2. Standardization

Before starting the fine-tuning process, we standardized the calibrated iRT values using the standard Scaler transformation. This process transforms the data to have zero mean and unit variance, mathematically defined as:

$$\text{standardized iRT} = \frac{\text{iRT} - \mu}{\sigma} \tag{5.1}$$

where μ is the mean and σ is the standard deviation of the calibrated retention time values in the training set. This standardization should help maintain stable gradient flow during backpropagation, accelerate convergence, and prevent the large numerical range of RT values from causing training instability, as demonstrated by [Ahsan et al., 2021]. Importantly, we only use training set statistics to avoid data leakage to validation and test sets.

5.3.3. Fine tuning

In the fine-tuning phase, the mask head is removed, and an MLP output layer is added to the encoder. The model is then trained on the standardized iRT to enable the MLP output layer to accurately predict RT. During training, we optimize the MAE loss function:

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{5.2}$$

where y_i and $\hat{y}_i$ represent the true and predicted RT respectively. We selected MAE for its robustness to outliers compared to Mean Squared Error (MSE) and because it is differentiable unlike the Median Absolute Error (MedAE).

5.4. Parameter tuning

The context-aware model architecture introduces several hyperparameters that impact the performance of the model. Unlike the base model, which had already been extensively optimized by [Dens et al., 2024], our context-aware approach required parameter tuning to achieve optimal performance. This section outlines our approach to hyperparameter optimization, addressing the computational challenges inherent in training large-scale models on extensive datasets. Important to note is that we will measure how good the parameter configuration is on the validation set, this is done so that we do not overfit on the test set, which is kept for final model evaluation.

5.4.1. Search space definition

We defined a focused search space comprising six key hyperparameters, deliberately excluding the parameters of the sequence processing pipeline components, which we maintained from [Dens et al., 2024]. We focused our tuning efforts on the MLP component and the training parameters.

Five hyperparameters each had three selected values, while the hidden size MLP parameter had four options, resulting in $3^5 4 = 972$ possible combinations. We used base model configuration as a starting point, with variants designed to explore alternative settings. Our complete search space can be seen in table 6.2. This approach ensured that our search space included both conservative choices (similar to the proven base model) and more exploratory alternatives that might better suit our context-aware architecture.

Hyperparameter	**Base Model**	**Variant 1**	**Variant 2**	**Variant 3**
Learning Rate	0.00033	0.0001	0.001	-
Optimizer	AdamW	Adam	SGD	-
Scheduler	Warmup Decay Cosine	Warmup	None	-
Dropout	0.1	0	0.3	-
Hidden Size	128	256-128	512-256-128	128-64-32
Activation	ReLU	Tanh	Sigmoid	-

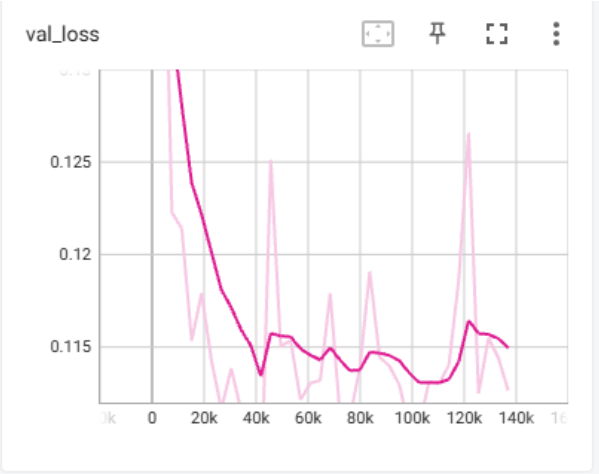
Table 5.3.: Hyperparameter Search Space for Context-Aware Model

5.4.2. Computational Efficiency Strategies

Training Time Optimization

Given the substantial computational requirements of our context-aware model and dataset scale, each full training run required approximately 10 hours, making an exhaustive search infeasible. To address this challenge, we executed an early stopping strategy based on validation loss convergence analysis.

Initial experiments using base model parameters revealed that validation loss typically converged around 10 epochs (approximately 40,000 steps), with minimal improvement thereafter. Figure 5.6 illustrates this



(a) Validation loss convergence during training (1,024 examples per step)

(b) Training progression showing 10 epochs corresponds to approximately 40,000 steps

Figure 5.6.: Training dynamics analysis. (a) Validation loss convergence over training steps. (b) Epoch-to-step relationship.

convergence pattern, showing the relationship between the progression of validation loss and the training epochs. Based on these findings, we limited parameter tuning experiments to 10 epochs, reducing the computation time from 10 hours to approximately 3 hours per configuration.

Search Strategy

To obtain a good first estimate of the hyperparameter impact and help guide further optimization, we initially performed a random search for the first 10 configurations. This exploratory phase provided a diverse sampling of the hyperparameter space, offering insight into the sensitivity of the model to different parameter choices.

To avoid the high computational cost of grid search, we subsequently implemented Tree-structured Parzen Estimator (TPE) optimization to efficiently navigate the hyperparameter space. TPE represents a sophisticated Bayesian optimization approach that intelligently samples promising regions based on historical performance data [Bergstra et al., 2011]. For the implementation of TPE we used Optuna's parameter tuning framework [Akiba et al., 2019].

The TPE algorithm works by first defining a performance threshold y^* (typically set as a percentile of observed results, we chose the top 10%). It then builds two separate probability density models:

- $\ell(x)$: density of parameter configurations that produced good results ($y < y^*$)
- $g(x)$: density of parameter configurations that produced poor results ($y \geq y^*$)

Using these densities, TPE applies Bayes' theorem to estimate the probability that a new configuration x will produce good performance:

$$p(\text{good performance}|x) \propto \frac{\ell(x)}{\ell(x) + g(x)} \tag{5.3}$$

However, TPE uses the ratio $\frac{\ell(x)}{g(x)}$, configurations with high $\ell(x)$ (similar to successful trials) and low $g(x)$

(dissimilar to failed trials) are prioritized for evaluation. By iteratively updating these models with each new result and selecting the most promising configurations, TPE can efficiently navigate the parameter space and find near-optimal hyperparameters while requiring only a fraction of the evaluations needed by grid search.

The combination of early stopping and TPE optimization enabled parameter tuning within reasonable computational limits, ensuring that our context-aware model reached good performance without excessive resource consumption.

5.4.3. Hyperparameter Importance Analysis

Optuna provides a hyperparameter importance analysis by decomposing the validation loss variance into contributions from individual hyperparameters and their interactions. This method quantifies how much each parameter contributes to the overall variance in the objective function, providing importance scores that sum to one. This analysis helps identify which hyperparameters have the most significant impact on model performance and guides future optimization efforts.

5.5. Reproducibility Methodology

To ensure the reproducibility of our results, we implemented two key strategies. First, given the stochastic nature of many processes in this study, we standardized the use of a fixed seed value of 42 for random number generation across all experiments, unless otherwise specified. Second, to minimize variations due to system differences and facilitate the use of our implementation, we utilized Docker and Apptainer containers. These containers also guarantee the use of the correct versions of all dependencies.

6. Results and discussion

This chapter evaluates two models for predicting peptide RT in LC-MS proteomics: a base model and a context-aware model. The base model predicts RT using only the amino acid sequence of a peptide, treating it as an isolated entity. The context-aware model extends this by also considering the sample context, incorporating information about all peptides in an MS run to account for peptide interactions and matrix effects.

All sections in this chapter use the same preprocessed subset of the MassiveKB dataset. From the complete MassiveKB repository containing 191 million PSMs, we collected a subset of 45 million PSMs across 2,892 MS runs from 62 experiments. Following preprocessing steps including duplicate removal, RT calibration and thoughtful data splitting, we got the following dataset composition:

Dataset	Unique Peptides	PSMs	Experiments	Unique Runs
Train Set	784,016	7,786,779	60	2,710
Validation Set	89,039	249,607	1 (MSV000080865)	82
Test Set	79,949	247,344	1 (MSV000080276)	100
Total	**953,004**	**8,283,730**	**62**	**2,892**

Table 6.1.: Table showing final dataset composition for model training and result evaluation.

The split was performed at the experiment level to prevent data leakage, with complete removal of peptide overlap between training, validation, and test sets to ensure realistic performance evaluation on unseen peptides. This splitting strategy ensures that both the validation and test results represent genuine generalization to novel experimental conditions and peptide sequences.

6.1. Parameter tuning results

In this section, we present the results of our parameter tuning experiments, highlighting how tuning impacts model performance and identifying the most influential parameters. As mentioned in Section 5.4, we utilized Optuna [Akiba et al., 2019], which provides both the optimization framework and the analysis tools.

6.1.1. Inspecting performance of our search approach

Figure 6.1 shows the progression of the validation MAE loss between trials, each representing a different parameter configuration. The first 10 trials used random parameter selection for initial exploration, exhibiting a higher and more variable validation MAE as expected. Trial 3 achieved the lowest validation MAE across all experiments, establishing our optimal parameter configuration.

Following trial 10, we transitioned to TPE optimization. The results demonstrate TPE's effectiveness:

validation MAE consistently decreased compared to the random phase, indicating successful identification and exploitation of promising hyperparameter regions.

Manual inspection revealed that TPE was sampling variations around trial 3's successful configuration, confirming that TPE correctly identified this as a promising region. However, trial 3 remained the best configuration within this explored region. suggesting that TPE had converged on a local optimum. This validates our hybrid approach of random exploration followed by TPE-guided exploitation. These results demonstrate the effectiveness of combining initial random sampling with subsequent TPE optimization.

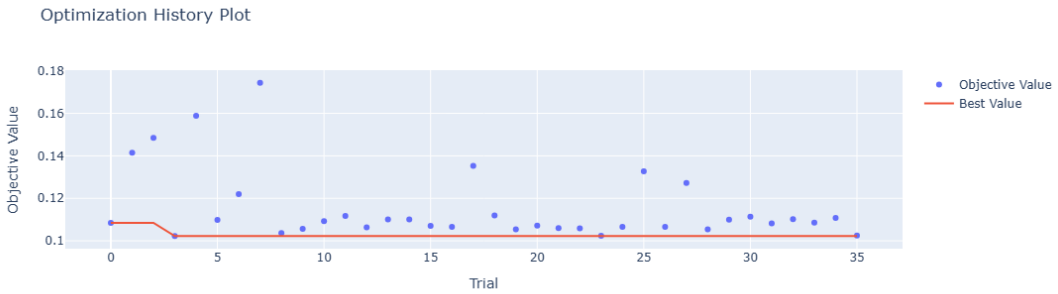


Figure 6.1.: Validation MAE progression across 35 hyperparameter optimization trials.

6.1.2. Parameter Effect analysis

The hyperparameter importance analysis (Figure 6.3) reveals that the optimizer choice has the most substantial impact on model performance, with an importance score of 0.86. This suggests that the selection of the optimization algorithm is critical to achieving good convergence. The learning rate and the scheduler follow as secondary factors with importance scores of 0.06 and 0.05 respectively, indicating their moderate influence on the training dynamics.

Interestingly, the MLP architecture parameters (dropout, activation function, and hidden layer sizes) show minimal importance (all < 0.01), suggesting that the model's performance is relatively robust to these architectural choices within the tested ranges. This robustness can be attributed to the sequence processing pipeline that provides highly informative representations that any reasonable MLP configuration can effectively map to RT predictions.

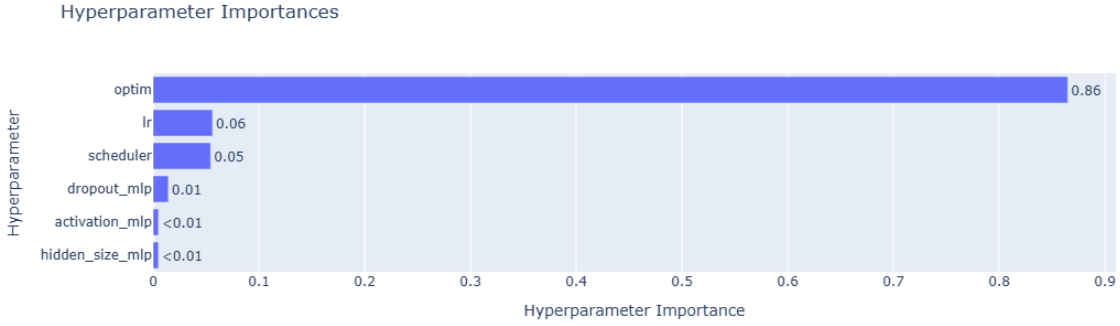


Figure 6.2.: Hyperparameter importance analysis showing the relative impact of each parameter on model performance.

Figure 6.3 presents hyperparameter plots that delve deeper into the analysis by showing the validation MAE for various configurations of each parameter. From this figure, we observe that the SGD optimizer performs noticeably worse compared to Adam and AdamW. Regarding the learning rate, performance improves as the learning rate increases. Similarly, using a more complex learning rate scheduler appears to degrade performance compared to using no scheduler at all. However, these effects may be influenced by the fact that our hyperparameter tuning was limited to only 10 training epochs. Lower learning rates and more sophisticated schedulers often require longer training times to show their full potential.

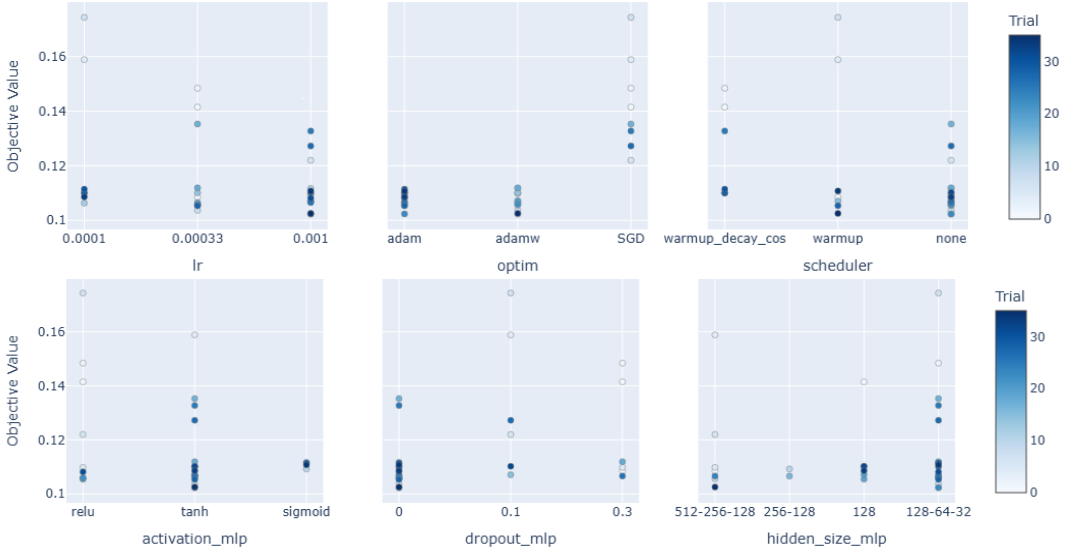


Figure 6.3.: Hyperparameter effect analysis showing validation MAE for different parameter configurations.

6.1.3. Optimal Parameter Configuration and Validation

Trial 3 emerged as the optimal configuration during initial hyperparameter tuning. The complete parameter specification for this configuration is presented in Table 6.2

Learning Rate	Optimizer	Scheduler	Dropout	Hidden Size	Activation
0.001	AdamW	None	0	128-64-32	Tanh

Table 6.2.: Trial 3 parameter configuration adopted as final configuration for the context-aware model.

To validate trial 3 under extended training conditions, we tested an alternative configuration with a reduced learning rate of 0.00033 and a cosine warm-up scheduler. The validation results demonstrated Trial 3's superiority, achieving a validation MAE of 0.1017 compared to 0.1074 for the alternative configuration.

Given the marginal expected improvements from the further exploration of hyperparameters and the computational cost of full training runs, we conclude the search process at this point. The configuration specified in Table 6.2 was therefore adopted as our final context aware model architecture.

6.2. Model Evaluation and Interpretation

This section evaluates the core hypothesis of this thesis: that incorporating the sample context is feasible to improve the prediction accuracy of peptide RT compared to traditional sequence-only approaches. The motivation stems from the understanding that peptide LC behavior is influenced not only by intrinsic sequence properties but also by sample-dependent factors such as peptide interactions and matrix effects during LC.

To test this hypothesis, we developed and compared two models that are built incorporating the same BERT sequence processing pipeline architecture to ensure that any performance differences stem from incorporating sample context rather than architectural improvements. The base model processes individual peptide sequences in isolation, representing the current approach that treats each peptide as an independent entity. Our context-aware model extends this by using the BERT sequence processing pipeline twice: one to process all peptides in the sample and create a comprehensive context token, and one to process the target peptide. The context token and peptide representation are then concatenated prior to the final prediction.

By accounting for the sample environment, this approach addresses the gap in existing RT prediction methods, which often ignore sample-dependent effects influencing chromatographic behavior in complex biological samples. Our evaluation focuses on whether the context-aware model achieves better prediction accuracy, successfully learns to utilize sample context information, and captures the observed sample-dependent RT variability.

6.2.1. Validation and test set results

To assess how well the base and context-aware models performed, we calculated their MedAE on both validation and test sets. We chose MedAE as our main evaluation metric because it offers a stable indicator of the typical prediction accuracy and is less affected by extreme values compared to the MAE. MedAE is defined as:

$$MedAE = \text{median}(|y_i - \hat{y}_i|) \tag{6.1}$$

where y_i and $\hat{y}_i$ represent the true and predicted RT respectively.

Since neural network performance can vary due to stochastic factors such as random weight initialization and data shuffling during training, we trained and evaluated both models across three different random seeds (1, 117, 42) to ensure that our results represent robust, generalizable performance rather than artifacts of a single training run.

The MedAE results for both models are presented in table 6.3. The context-aware model consistently outperformed the base model, achieving a mean validation MedAE of 2.77 ± 0.06 compared to 3.160 ± 0.088 for the base model, reflecting an improvement of 12. 3%. In the test set, the context-aware model yielded a mean MedAE of 7.01 ± 0.16 , compared to 7.99 ± 0.04 for the base model, an improvement of 12.3%. These results show that the incorporation of the sample context improves the accuracy of RT prediction, supporting the hypothesis that peptide interactions and matrix effects influence chromatographic behavior.

The low standard deviations in MedAE across different seeds indicate stable model performance, suggesting that the observed improvements are robust and not due to random variations. This consistent reduction in MedAE for the context-aware model highlights the importance of sample context in RT prediction, offering a measurable improvement over the sequence-only approach.

Model	Validation MedAE	Test MedAE
Base Model (Seed 1)	3.146	8.001
Base Model (Seed 117)	3.080	7.946
Base Model (Seed 42)	3.254	8.032
Base Model (Mean ± Std)	**3.160 ± 0.088**	**7.993 ± 0.044**
Context Aware Model (Seed 1)	2.703	7.133
Context Aware Model (Seed 117)	2.821	7.061
Context Aware Model (Seed 42)	2.787	6.821
Context Aware Model (Mean ± Std)	**2.770 ± 0.061**	**7.005 ± 0.163**

Table 6.3.: Results of the Models on Validation and Test Set Across Multiple Seeds.

6.2.2. Analysis of test set performance

Both models exhibited higher test MedAE compared to validation MedAE, with the test set showing approximately double the values (e.g., 3.160 vs. 7.993 for the context aware model). This performance gap resulted in further investigation into the underlying data characteristics.

To explore this, we analyzed RT variability for identical peptides across different runs using the MAD The test set exhibited significantly higher variability, with a median MAD of 3.243 iRT units, compared to 0.977 iRT units in the validation set, indicating roughly three times greater variability (Figure 6.4). This suggests that the test set includes runs with more diverse experimental conditions or complex sample matrices, which poses a greater challenge for accurate prediction.

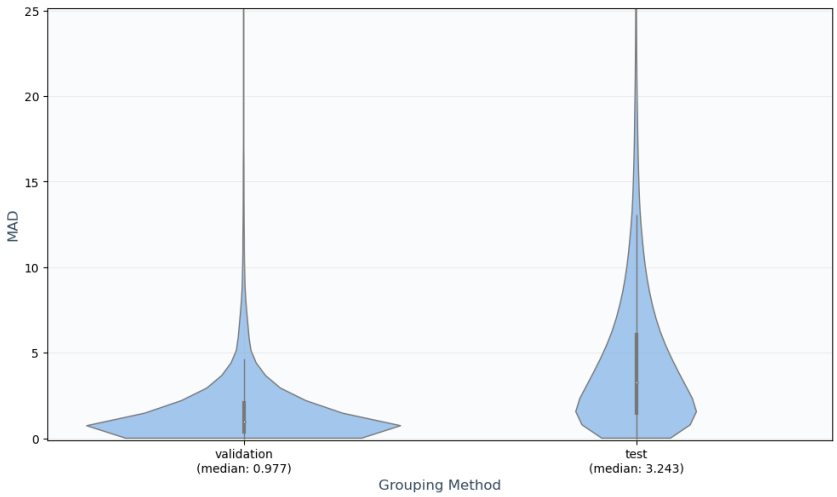


Figure 6.4.: MAD retention time distribution for identical peptides across runs, comparing test and validation sets with higher test set median.

Further analysis examined RT variability across all experiments in the dataset using KDE (Figure 6.5). The test set experiment ID (MSV000080276) showed the highest median variability among all experiments. This high variability in the test set serves as a stress test for the models, highlighting the context-aware model's ability to generalize to challenging, unseen conditions.

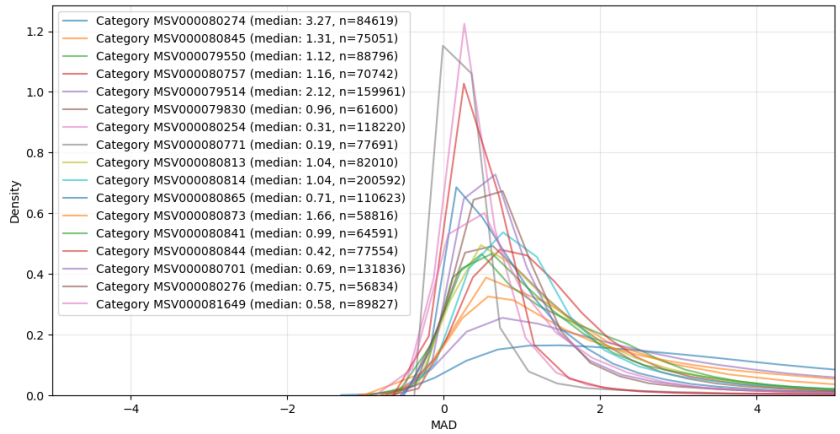


Figure 6.5.: KDE plot displaying MAD retention time distribution across runs for MassiveKB experiment IDs

6.2.3. Directional improvement

To gain deeper insights into the performance differences between our model and the base model, we performed a directional improvement analysis. For each peptide prediction, we calculated the directional improvement using the following formula:

$$\text{improvement} = |\text{base\_prediction} - \text{truth}| - |\text{pool\_pred} - \text{truth}| \tag{6.2}$$

Where positive values indicate the context-aware model performed better , negative values indicate the base model performed better and zero indicates identical performance.

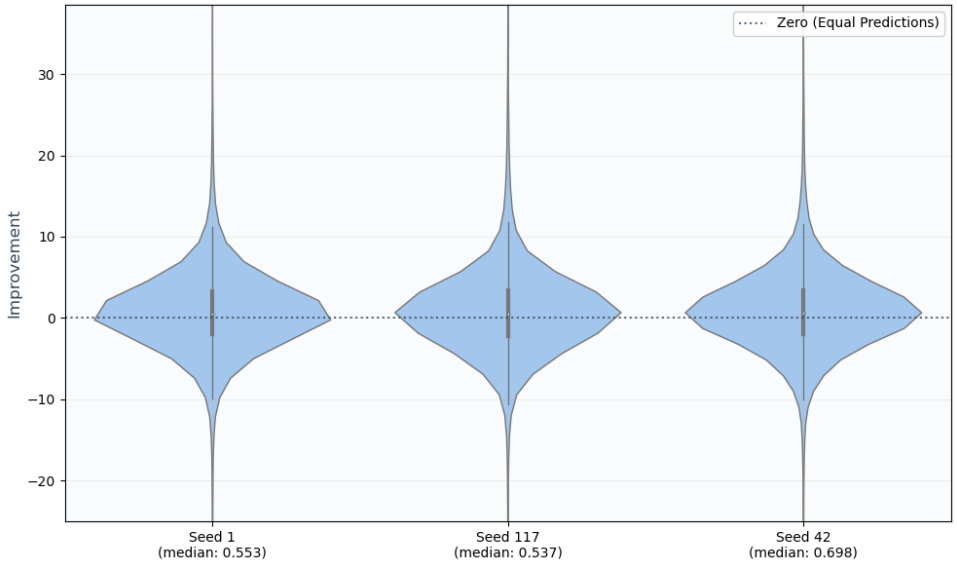


Figure 6.6.: Violin plot demonstrating directional improvement across three seeds with positive bias (medians: 0.533, 0.537, 0.698)

figure 6.6 presents the results of the directional improvement analysis illustrating the distribution of directional improvement values across the test set. The plot shows a small positive bias with positive medians for all seeds. indicating that the context-aware model typically reduces prediction errors compared to the base model. Notably, the distributions have a long tail of positive improvements, suggesting

that when the context-aware model performs better, it can achieve substantially lower errors. However, there are cases where the base model performs better, indicating that sample context is not always beneficial and may introduce noise in certain predictions.

To further quantify the directional improvement, we categorized the directional improvement results into three binary outcomes: "Improved" (context-aware model performs better), "Equal" (similar performance within 0.1 iRT units), and "Worse" (base model performs better). The 0.1 iRT unit threshold accounts for minor fluctuations that may not be practically significant in LC-MS workflows. Table 6.4 summarizes these results across the three seeds.

Metric	Improved	Equal	Worse	Total
Seed 1	135,640 (54.8%)	5,719 (2.3%)	105,985 (42.9%)	247,344 (100%)
Seed 117	134,715 (54.5%)	5,485 (2.2%)	107,144 (43.3%)	247,344 (100%)
Seed 42	139,200 (56.3%)	5,713 (2.3%)	102,431 (41.4%)	247,344 (100%)
Mean ± Std (Raw)	**136,518 ± 2,368**	**5,639 ± 133**	**105,187 ± 2,456**	**-**
Mean ± Std (%)	**55.2% ± 1.0%**	**2.3% ± 0.1%**	**42.5% ± 1.0%**	**-**

Table 6.4.: Binary Directional Improvement Across Three Seeds.

The results demonstrate that the context-aware model provides better predictions for 55.2% of the peptides, with fewer cases where the base model performs better. To evaluate the statistical significance of these results, we applied McNemar's test [McNemar, 1947], which is specifically designed to compare paired binary outcomes and tests whether the number of "Improved" versus "Worse" predictions differs significantly. McNemar's test excludes the "Equal" cases from the analysis. The test statistic is calculated as:

$$\chi^2 = \frac{(|\text{Improved} - \text{Worse}| - 1)^2}{\text{Improved} + \text{Worse}} = \frac{(|136{,}518 - 105{,}187| - 1)^2}{136{,}518 + 105{,}187} = 4{,}061.02 \tag{6.3}$$

Using the mean values across seeds , this chi-square statistic with 1 degree of freedom yields a p-value that approaches zero, confirming that the context-aware model's superior performance is highly statistically significant and not attributable to random chance.

6.2.4. Quantifying Sample Context Utilization

To evaluate how effectively the context-aware model incorporates sample context, we analyzed the MAD of predicted RT for peptides appearing in multiple runs within the test set. This analysis allows us to measure whether the model successfully captures the sample-dependent variation observed in experimental data.

For comparison, we calculated the same MAD values for the ground truth retention times of these identical peptides across runs, establishing the baseline level of sample-dependent variation that the model should ideally capture. Figure 6.7 presents this comparison between the predicted and observed RT variability.

The analysis revealed that, while the model successfully captures sample-dependent effects, it does so conservatively. The model's predictions showed a median variability of 0.589 iRT units compared to the ground truth's median of 3.243 iRT units, indicating that the model accounts for approximately 18% of the sample-dependent variation observed in the test set. Note that the base model would show a MAD of zero in this analysis, as it does not incorporate sample context and would predict identical retention times for the same peptide across all runs.

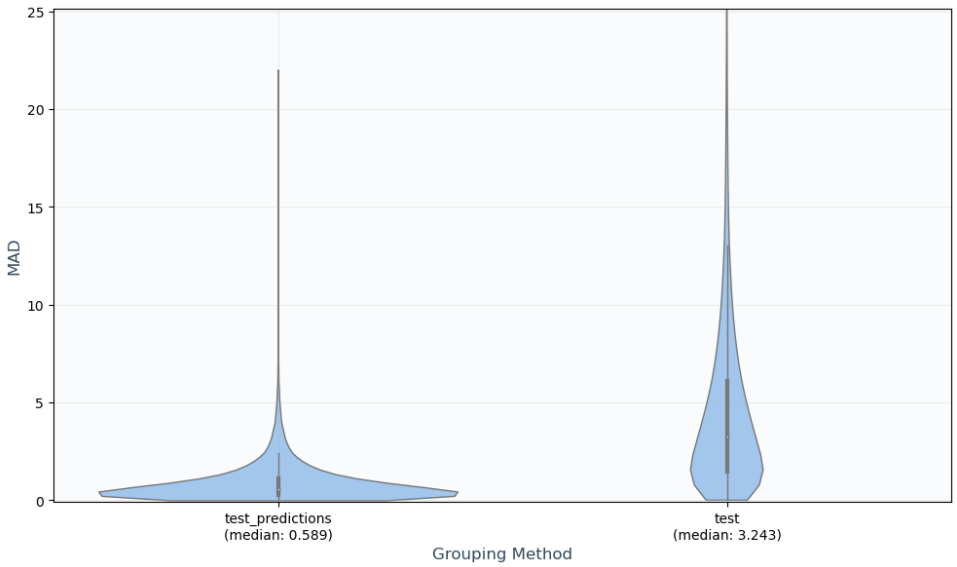


Figure 6.7.: MAD distribution violin plot comparison of predicted versus true retention times for identical peptides across test set (left: model predictions, right: actual values)

This conservative approach probably originates from several factors. First, the model's regularization prevents overfitting to sample-specific variations, prioritizing generalizable patterns over sample-specific noise. Second, the current method of representing the context of the sample through the mean pooling of peptide embeddings may not fully capture the complexity of sample-dependent factors. Finally, the model must balance sequence-based predictions with context information, possibly favoring more reliable sequence features.

6.2.5. Did the model learn sample context?

To determine whether the sample-aware model learned to utilize sample context, we analyzed how model predictions varied for identical peptides across different runs. Figure 6.8 shows the distribution of the prediction variability for the same peptides in different contexts for the validation and test sets.

The context-aware model demonstrated clear context-dependent behavior, with predictions showing greater variability in the test set (median: 0.589) compared to the validation set (median: 0.289). This pattern aligns closely with the variability observed in the ground truth data presented earlier in figure 6.4, suggesting that the model successfully captures sample-specific effects. The similarity in distributional shapes between the model's predictions and ground truth data further indicates that the model is not merely memorizing peptide sequences but instead leveraging sample context.

It is important to note that our model was parameter tuned using the validation set, which may introduce slight bias to the validation results. However, as demonstrated in the parameter tuning section, hyperparameter optimization yielded only modest performance improvements. Critically, the test set remained completely unseen throughout the entire development process, ensuring that test performance represents genuine generalization of sample context learning to unseen experimental conditions.

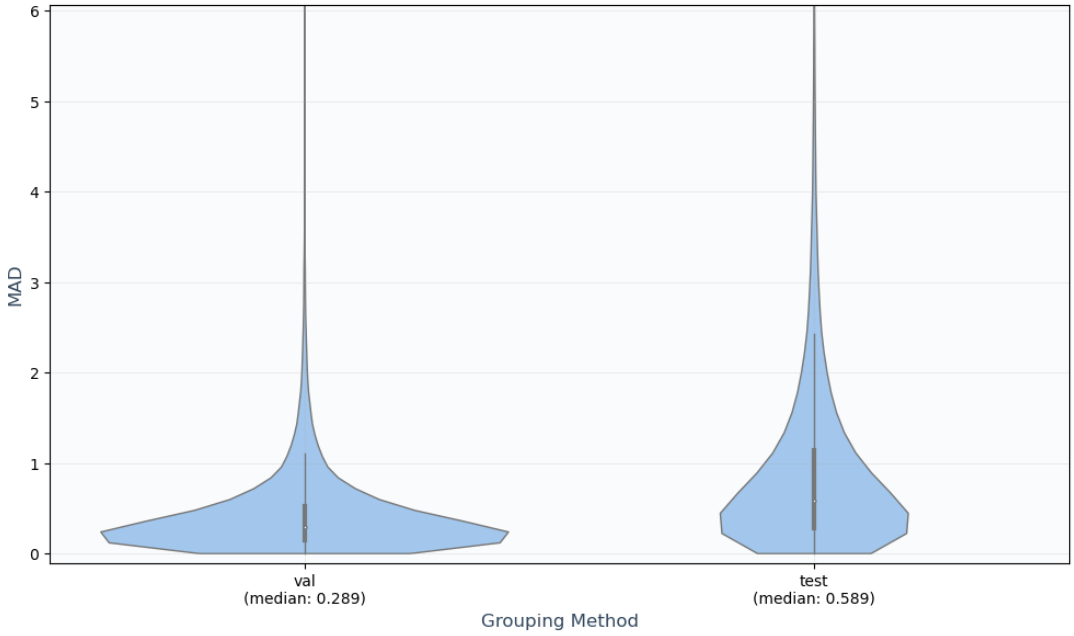


Figure 6.8.: Violin plots showing distribution of predicted retention time MAD for identical peptides across runs, comparing validation and test sets.

6.2.6. Training Data Scale Impact Analysis

To understand how training data volume affects model performance and determine whether additional data collection would yield meaningful improvements, we performed a scaling analysis. We trained multiple instances of our context-aware model using progressively increasing amounts of training data, ranging from 271 MS runs to 2,710 MS runs. For each training subset, we measured the median absolute error (MedAE) on the test set to assess performance scaling characteristics.

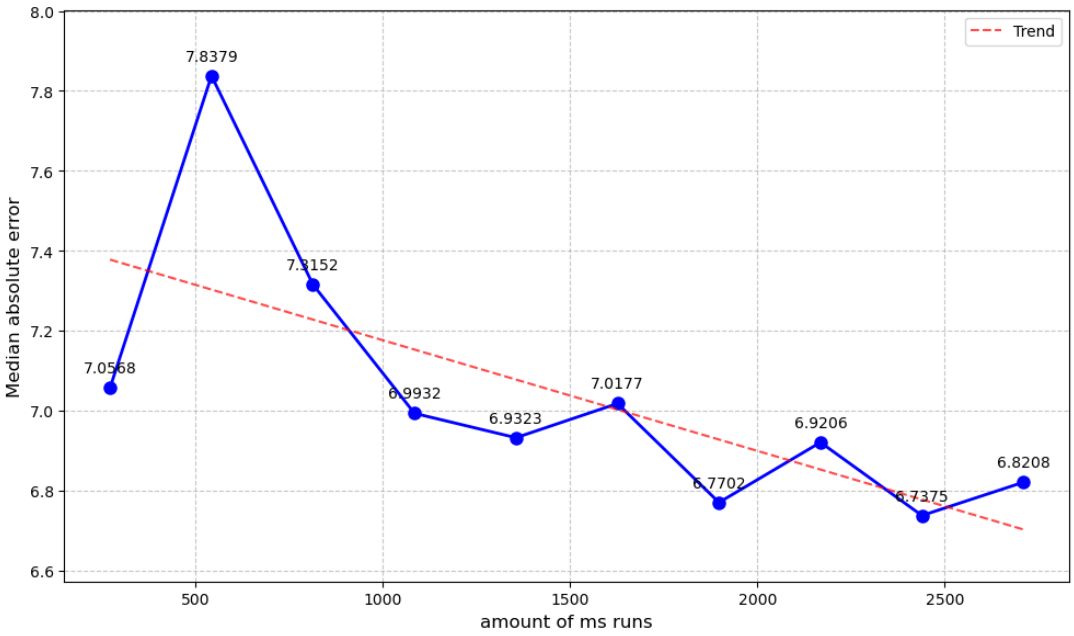


Figure 6.9.: Relationship between number of runs used during trainingand test set performance measured by MedAE.

The scaling analysis revealed a relationship between training data volume and model performance, as illustrated in Figure 6.9. Initially, from 271-813 runs, the results were counterintuitive: the model achieved its best performance at 271 runs with a MedAE of approximately 7.06 iRT units. Performance then degraded at 542 runs (7.84 iRT units) and 813 runs (7.32 iRT units), suggesting training instability during this phase.

This initial performance degradation likely reflects the challenge of learning from increasingly heterogeneous data sources. As additional MS runs are introduced, the model may experience training instability while trying to learn conflicting patterns from different data sources. The early peak at 271 runs could also represent a fortunate alignment between a relatively homogeneous subset of training data and the specific characteristics of our test set.

However, beyond 542 runs, the model exhibits the expected improvement pattern, with performance steadily increasing as more training data becomes available. This transition point appears to represent a threshold where the model can effectively generalize across data heterogeneity. From this stabilization point onward, additional training data consistently improve performance, following the initially expected scaling behavior. This shows that our context-aware model benefits from increased training data volume and exposure to different contexts.

7. Application

The context-aware RT prediction model developed in this thesis has the potential to improve peptide identification workflows, particularly in PSM rescoring. This section demonstrates how our model could improve rescoring algorithms and how our model could be incorporated.

7.1. PSM Rescoring Enhancement

In MS-based proteomics, peptide identification relies on matching experimental spectra with theoretical spectra from protein databases. These PSMs, receive confidence scores based on spectral similarity. However, when searching against large databases, particularly common in immunopeptidomics, the increased search space leads to more potential matches, increasing the likelihood that false identifications occur by random chance. Traditional approaches address this challenge through score thresholds and FDR control using decoy sequences. While effective, this basic approach leaves room for improvement,

PSM rescoring has emerged as a powerful method for improving peptide identification accuracy. In this approach, machine learning models such as Percolator [Käll et al., 2007] or Mokapot [Fondrie and Noble, 2021] incorporate multiple features beyond the initial spectral score, including predicted retention times, charge states, and other peptide properties, to refine match confidence. The rescoring process follows an iterative framework where the model first trains on initial features and PSM scores from both target and decoy matches, then uses learned patterns to re-rank PSMs. Based on these updated rankings, new positive and negative examples are inferred and used to retrain the model. This cycle repeats until convergence to stable, high-confidence identifications. We based our understanding of rescoring on the paper by [Adams et al., 2024] where they explain how rescoring works in depth. They also provide a clear overview of this process in figure 7.1.

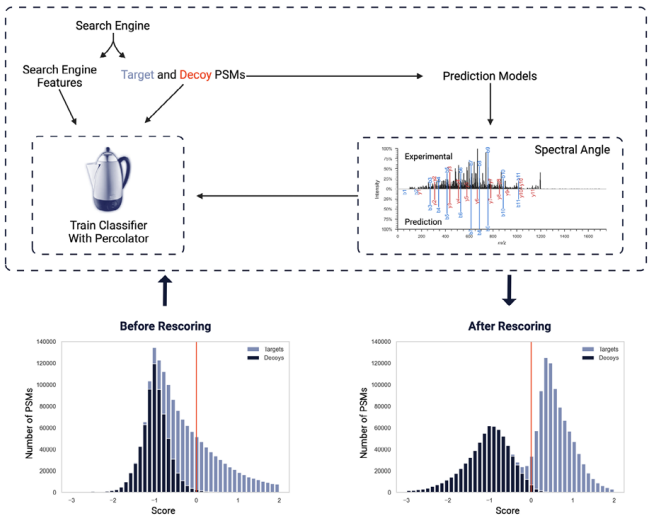


Figure 7.1.: Overview of PSM rescoring process [Adams et al., 2024].

Our context-aware model could improve this process by contributing more accurate RT predictions as input features. Traditional RT predictors treat each peptide in isolation, potentially missing important sample-dependent effects that influence the chromatographic behavior. More accurate RT predictions provide better discrimination between correct and incorrect PSMs, as large deviations between predicted and observed PSM often indicate false matches. By incorporating sample context, our model accounts for matrix effects and peptide interactions that influence RT in complex biological samples, leading to more reliable predictions than sequence-only approaches. These higher quality RT features can potentially improve the overall performance of the rescoring models, leading to better PSMs.

7.2. Proposed integration strategy

The integration of context-aware prediction into PSM rescoring presents an interesting implementation challenge. Our model requires knowledge of the sample composition to generate accurate predictions, yet the rescoring process aims to determine which peptides are actually present. We propose a solution that takes advantage of the iterative nature of rescoring workflow, transforming this challenge into an opportunity for mutual improvement.

Our implementation strategy begins with the highest-scoring PSMs from the initial database search as a preliminary sample context. While imperfect, this provides a reasonable starting approximation of the sample composition. Using this estimated context, we generate RT predictions for all candidate PSMs, providing more accurate RT features than traditional sequence-only models. These improved RT predictions, along with other peptide features, feed into the rescoring model to generate updated PSM confidence scores.

As the rescoring model improves PSM rankings through its iterative process, the set of high-confidence PSMs used for context estimation becomes more accurate, leading to progressively better RT predictions. This creates a beneficial feedback loop where better context leads to more accurate RT predictions, which enable more accurate PSM scoring and better context estimation in subsequent iterations. An overview of our proposed integration can be seen in figure 7.2

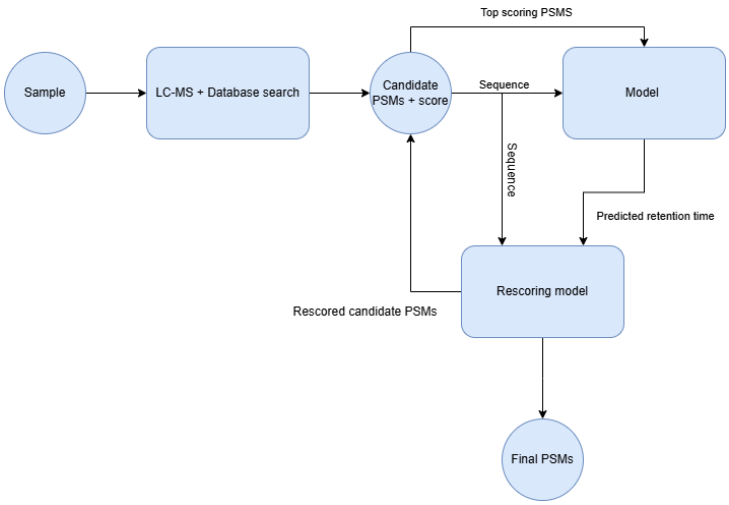


Figure 7.2.: Overview of proposed iterative integration strategy.

Through this implementation strategy, our context-aware RT prediction model can be successfully integrated into real-world proteomics workflows, providing enhanced peptide identification accuracy while working within the established framework of PSM re-scoring methodologies.

8. Conclusion

Our investigation of the MassiveKB dataset showed evidence supporting the hypothesis that sample context influences peptide RT. The dataset analysis revealed meaningful sample-dependent variation in peptide behavior, with RT variability demonstrably higher across different MS runs compared to within the same MS run(runs generally use different samples). This confirmed the relevance of the problem and provided the basis for developing context-sensitive prediction models. In response, our feasibility study showed that adding sample context as a feature is both feasible and beneficial, as the context-aware model outperformed the baseline sequence-only model.

These findings open new avenues for improving the accuracy and reliability of MS-based peptide identification, potentially contributing to more robust and informative proteomics analyses with applications in PSM rescoring workflows.

8.1. Future work

This feasibility study has demonstrated the potential of context-aware retention time RT prediction to enhance peptide identification in proteomics. However, several avenues for future research could further improve the model's performance, applicability, and impact on computational proteomics. The following directions outline potential extensions to build upon this work:

8.1.1. Improved Sample Context Representation

The current sample-aware model uses the mean of peptide embeddings to represent the sample context, which may not fully capture the complexity of sample-dependent factors such as peptide interactions and matrix effects. Future work could explore more sophisticated methods for context representation, such as attention mechanisms [Vaswani et al., 2017] or graph neural networks [Scarselli et al., 2009]. These approaches could dynamically weigh the contributions of individual peptides within a sample and model complex inter-peptide relationships more effectively.

Attention mechanisms could allow the model to focus on the most relevant peptides for predicting a target peptide's retention time, while graph neural networks could explicitly model peptide-peptide interactions as edges in a graph. Developing such representations could lead to more accurate RT predictions by better modeling the chromatographic environment and capturing subtle but important sample-dependent effects.

8.1.2. Integration Validation in Rescoring Workflows

While this thesis proposes a framework for integrating context-aware models into rescoring workflows, validation of this integration represents a next step. Future work could implement the proposed inte-

gration strategy and evaluate its impact on peptide identification rates, FDR, and overall proteomics workflow performance.

8.1.3. MassiveKB Dataset Creation

This study used a subset from the MassiveKB repository, but the complete dataset contains significantly more data, offering opportunities for further exploration. Future work could involve collecting and processing the entire MassiveKB repository using the developed Nextflow pipeline to create a comprehensive, publicly accessible dataset for the proteomics community.

This expanded dataset could enable more robust training and evaluation of context-aware models, capturing a wider range of experimental conditions and sample complexities. The resulting resource would not only benefit RT prediction research but could serve as a valuable dataset for various computational proteomics applications.

Bibliography

[Adams et al., 2024] Adams, C., Laukens, K., Bittremieux, W., and Boonen, K. (2024). Machine learning-based peptide-spectrum match rescoring opens up the immunopeptidome. *PROTEOMICS*, 24(8). xiv, 45

[Ahsan et al., 2021] Ahsan, M. M., Mahmud, M. A. P., Saha, P. K., Gupta, K. D., and Siddique, Z. (2021). Effect of data scaling methods on machine learning algorithms and model performance. *Technologies*, 9(3). 31

[Akiba et al., 2019] Akiba, T., Sano, S., Yanase, T., Ohta, T., and Koyama, M. (2019). Optuna: A next-generation hyperparameter optimization framework. 33, 35

[Alberts et al., 2024] Alberts, B., Johnson, A., Lewis, J., Morgan, D., Raff, M., Roberts, K., and Walter, P. (2024). *Molecular Biology of the Cell*. W. W. Norton & Company, New York, 7 edition. xiii, 1, 3

[Bergstra et al., 2011] Bergstra, J., Bardenet, R., Kégl, B., and Bengio, Y. (2011). Algorithms for hyper-parameter optimization. 33

[Boundless, 2023a] Boundless (2023a). Proteins - Protein Structure. In *General Biology*, chapter 3.9. LibreTexts. 1

[Boundless, 2023b] Boundless (2023b). Proteins - Types and Functions of Proteins. In *General Biology*, chapter 3.7. LibreTexts. 1

[Boundless, 2023c] Boundless (2023c). Proteins -Amino Acids. In *General Biology*, chapter 3.8. LibreTexts. xiii, 2

[Bouwmeester et al., 2021] Bouwmeester, R., Gabriels, R., Hulstaert, N., Martens, L., and Degroeve, S. (2021). DeepLC can predict retention times for peptides that carry as-yet unseen modifications. *Nature Methods*, 18(11):1363–1369. 11

[Cell Guidance Systems,] Cell Guidance Systems. The difference between peptides and proteins. xiii, 1, 2

[Dens et al., 2024] Dens, C., Adams, C., Laukens, K., and Bittremieux, W. (2024). Machine learning strategies to tackle data challenges in mass spectrometry-based proteomics. *Journal of the American Society for Mass Spectrometry*, 35(9):2143–2155. PMID: 39074335. v, vii, xiii, xv, 12, 27, 29, 31, 32

[Devlin et al., 2019] Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). Bert: Pre-training of deep bidirectional transformers for language understanding. 7, 27

[Fondrie and Noble, 2021] Fondrie, W. E. and Noble, W. S. (2021). mokapot: Fast and Flexible Semisupervised Learning for Peptide Detection. *Journal of Proteome Research*, 20(4):1966–1971. 45

[Kailasam, 2021] Kailasam, S. (2021). LC-MS – What Is LC-MS, LC-MS Analysis and LC-MS/MS. *Technology Networks*. xiii, 4

[Kim and Pevzner, 2014] Kim, S. and Pevzner, P. A. (2014). MS-GF+ makes progress towards a universal database search tool for proteomics. *Nature Communications*, 5(1):5277. 19

[Klont and Hopfgartner, 2021] Klont, F. and Hopfgartner, G. (2021). Mass spectrometry based approaches and strategies in bioanalysis for qualitative and quantitative analysis of pharmaceutically relevant molecules. *Drug Discovery Today: Technologies*, 40:64–68. xiii, 5

[Käll et al., 2007] Käll, L., Canterbury, J. D., Weston, J., Noble, W. S., and MacCoss, M. J. (2007). Semi-supervised learning for peptide identification from shotgun proteomics datasets. *Nature Methods*, 4(11):923–925. Epub 2007 Oct 21. 45

[Leo et al., 2023] Leo, J., Ge, E., and Li, S. (2023). Wasserstein Distance in Deep Learning. SSRN Electronic Journal. Available at SSRN. 15

[Mant et al., 2007] Mant, C. T., Chen, Y., Yan, Z., Popa, T. V., Kovacs, J. M., Mills, J. B., Tripet, B. P., and Hodges, R. S. (2007). HPLC analysis and purification of peptides. *Methods in Molecular Biology*, 386:3–55. 5, 25

[McNemar, 1947] McNemar, Q. (1947). Note on the sampling error of the difference between correlated proportions or percentages. *Psychometrika*, 12(2):153–157. 41

[ML et al., 2023] ML, W., AA., O., RV., E., NH., G., and E, G. (2023). Matrix effects demystified: Strategies for resolving challenges in analytical separations of complex samples. *Journal of Separation Science*, 46(23):e2300571. 5

[Scarselli et al., 2009] Scarselli, F., Gori, M., Tsoi, A. C., Hagenbuchner, M., and Monfardini, G. (2009). The graph neural network model. *IEEE Transactions on Neural Networks*, 20(1):61–80. 47

[Schmidt et al., 2018] Schmidt, T., Samaras, P., Frejno, M., Gessulat, S., Barnert, M., Kienegger, H., Krcmar, H., Schlegl, J., Ehrlich, H.-C., Aiche, S., Kuster, B., and Wilhelm, M. (2018). ProteomicsDB in 2018: a user-friendly protein database. *Nucleic Acids Research*, 46(D1):D1271–D1281. 16

[Shuken, 2023] Shuken, S. R. (2023). An introduction to mass spectrometry-based proteomics. *Journal of Proteome Research*, 22(7):2151–2171. PMID: 37260118. xiii, 1, 3, 6

[Tyanova et al., 2016] Tyanova, S., Temu, T., and Cox, J. (2016). The maxquant computational platform for mass spectrometry-based shotgun proteomics. *Nature Protocols*, 11(12):2301–2319. 6, 16

[Vaswani et al., 2017] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017). Attention is all you need. 7, 27, 47

[VSC,] VSC. Flemish Supercomputer Center (VSC). HPC Infrastructure. 57

[Wang et al., 2018] Wang, M., Wang, J., Carver, J., Pullman, B. S., Cha, S. W., and Bandeira, N. (2018). Assembling the Community-Scale Discoverable Human Proteome. *Cell Systems*, 7(4):412–421.e5. xiii, 19

[Waskom, 2021] Waskom, M. L. (2021). seaborn: statistical data visualization. *Journal of Open Source Software*, 6(60):3021. 14

[Wilburn et al., 2023] Wilburn, D. B., Shannon, A. E., Spicer, V., Richards, A. L., Yeung, D., Swaney, D. L., Krokhin, O. V., and Searle, B. C. (2023). Deep learning from harmonized peptide libraries enables retention time prediction of diverse post translational modifications. *bioRxiv*. 11, 13

[Zolg et al., 2017] Zolg, D. P., Wilhelm, M., Schnatbaum, K., Zerweck, J., Knaute, T., Delanghe, B., Bailey, D. J., Gessulat, S., Ehrlich, H.-C., Weininger, M., Yu, P., Schlegl, J., Kramer, K., Schmidt, T., Kusebauch, U., Deutsch, E. W., Aebersold, R., Moritz, R. L., Wenschuh, H., Moehring, T., Aiche, S., Huhmer, A., Reimer, U., and Kuster, B. (2017). Building ProteomeTools based on a complete synthetic human proteome. *Nature Methods*, 14. 16

[Åsberg et al., 2017] Åsberg, D., Langborg Weinmann, A., Leek, T., Lewis, R. J., Klarqvist, M., Leśko, M., Kaczmarski, K., Samuelsson, J., and Fornstedt, T. (2017). The importance of ion-pairing in peptide purification by reversed-phase liquid chromatography. *Journal of Chromatography A*, 1496:80–91. 5

A. Implementation details

We organized our work into two public code repositories:

- **MassiveKB Pipeline Repository** This repository contains the Nextflow pipeline used for dataset collection. It is designed standalone tool which hopefully can be used for other purposes as this thesis. `https://github.com/WannesLamberts/Massivekb-Pipeline`
- **Main Thesis Repository** This repository includes all implementations specific to this thesis, such as data analysis notebooks, model implementations and parameter tuning. `https://github.com/WannesLamberts/MTL-peptide-property-prediction-pool`

The MassiveKB repository includes detailed instructions for use, along with a script (`slurm_scripts/full_thesis_workflow.slurm`) detailing the exact commands executed on the supercomputer.

The Main Thesis Repository provides an `apptainer.def` file to build an image with all required dependencies. The exact scripts we ran for experiments are located in the `slurm_scripts` directory.

B. MassiveKB Pipeline

For the massiveKB dataset, a more sophisticated pipeline was required due to the significantly larger volume and more complex structure of the data, as detailed in section 4.3.1. The pipeline was designed with flexibility in mind, enabling its use beyond the scope of this thesis. The project can be found here.

B.1. Nextflow Framework and Pipeline Architecture

The pipeline was developed using Nextflow, a workflow management system designed to create scalable and reproducible bioinformatics pipelines. Nextflow employs a domain-specific language, built by extending Groovy, which simplifies pipeline development. Its dataflow programming model enables processes to connect via inputs and outputs, automatically forming an execution graph that orchestrates workflow tasks. Another strong point is that nextflow has a lot of documentation and tutorials, which speeded up the learning process a lot.

The pipeline follows a distributed memory multiprocessing architecture to significantly enhance computational efficiency. Multiprocessing leverages multiple CPU cores to execute processes in parallel, in contrast to sequential processing which utilizes only a single core.

B.2. Pipeline Design Philosophy and Features

The pipeline incorporates numerous features; this section briefly outlines the most notable ones.

B.2.1. User-Friendly Experience

The pipeline requires only Docker or Apptainer and Nextflow to operate. All dependencies and container setup are handled automatically, making the system accessible to users with varying technical backgrounds. For ease of use, detailed documentation with examples is provided in the README of the github repo.

B.2.2. Flexible Configuration System

Rather than requiring code modifications, the pipeline implements numerous parameters accessible via the command line, including things as CPU allocation per process, maximum concurrent processes, and various execution settings.

B.2.3. Cross-Platform Support

There is support for both local execution (suitable for small data collections) and High-Performance Computing (HPC) systems (recommended for large-scale data processing as used in this thesis). Users can easily switch between these environments via preconfigured settings specified through command-line arguments, ensuring the pipeline can adapt to different computational needs.

B.2.4. Real-Time feedback

Given the significant runtime required for large datasets, the pipeline includes optional Seqera integration that can be enabled via command-line. This provides detailed visualizations and comprehensive information of all processes during execution–tracking CPU usage, memory consumption, and other key metrics. allowing users to track progress and identify potential issues.

B.2.5. Error logging

Experience revealed that MassiveKB isn't always stable, with some data potentially inaccessible or in unexpected formats. The pipeline addresses this challenge through error logging with specific exit codes that indicate failure points. Importantly, process failures are isolated, if one process crashes the pipeline continues processing other tasks without interruption. Error logs are output as TSV files which can be used to identify and rerun failed tasks.

B.3. Workflow System and Components

B.3.1. data collections workflows

The main workflow requires no input and will collect all PSMs with their RT. While this workflow is easy to use, it is resource intensive and it takes a long time. Also, not everyone needs the full dataset. For this reason the subset data workflow is made where you simply give an input file with all task id's data is wanted from.

B.3.2. Simple data collection workflows

Since downloading the raw MS files represents the main pipeline bottleneck, this workflow generates PSMs exclusively from mzTAB files. While significantly faster, it offers a trade-off: retention times are not included in the PSM data.

B.3.3. preprocessing workflows

There is also a RT calibration workflow implemented and a parquet workflow that collects all the outputs into a single big parquet file that can be used as a dataset for this thesis.

B.4. Data Collection Process Implementation

We executed the pipeline described in the previous section on the calcua High-Performance Computing (HPC) cluster provided by [VSC,].

Table B.1 presents the allocation of resources for each pipeline phase, including memory requirements, concurrent processes, and CPU utilization per task. Note that the initial data collection was limited to 10 concurrent processes, as MassiveKB cannot reliably handle higher volumes of simultaneous downloads without increasing task failure rates.

Table B.2 summarizes the task completion statistics and processing duration for each pipeline phase. During initial data collection, 35 tasks failed: 33 due to missing mzTAB files in MassiveKB and 2 due to unexpected mzTAB file formatting issues. The data collection phase consumed the most time overall, primarily due to the download requirements for numerous large raw mass spectrometry files.

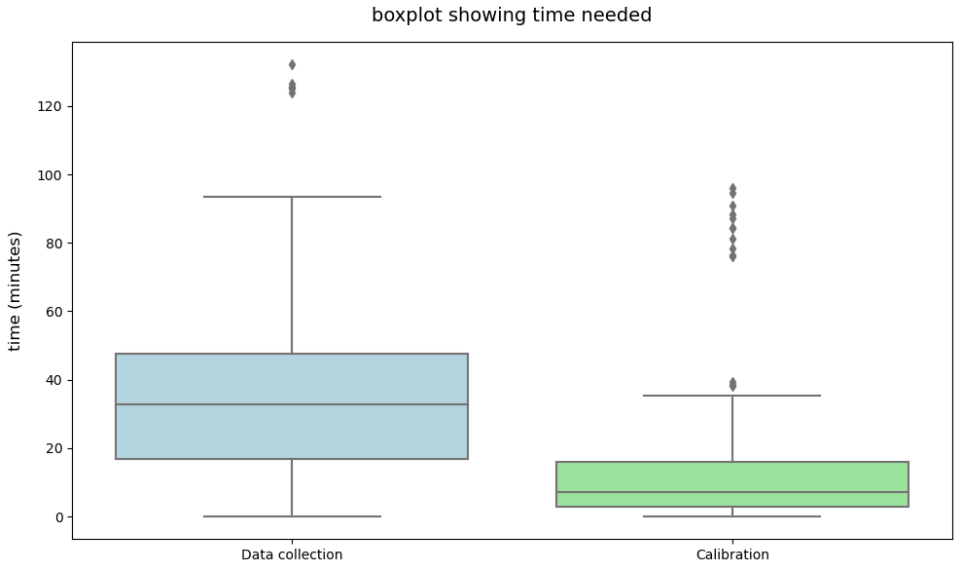
Phase	Memory per Task	Concurrent Processes	cpu's per task
Initial Data Collection	4 GB	10	1
Retention Time Calibration	4 GB	50	1
Parquet Combination	32 GB	1	1

Table B.1.: Table showing the resource allocation for each phase.

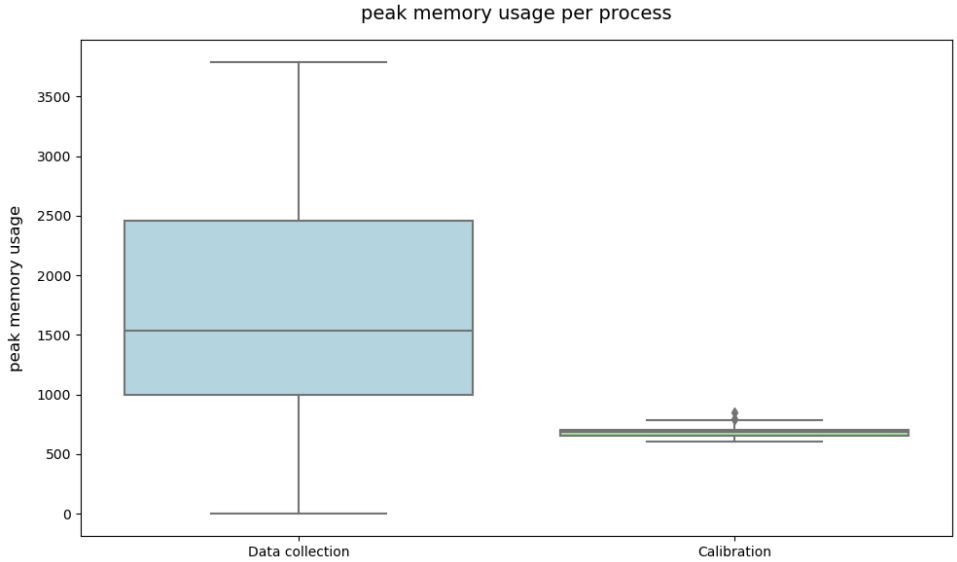
Phase	Successful	Failed	Total	Processing Time
Initial Data Collection	166	35	201	11 hours
Retention Time Calibration	166	0	166	1.5 hours
Parquet Combination	1	0	1	28 minutes

Table B.2.: Table showing task completion status with processing time by phase.

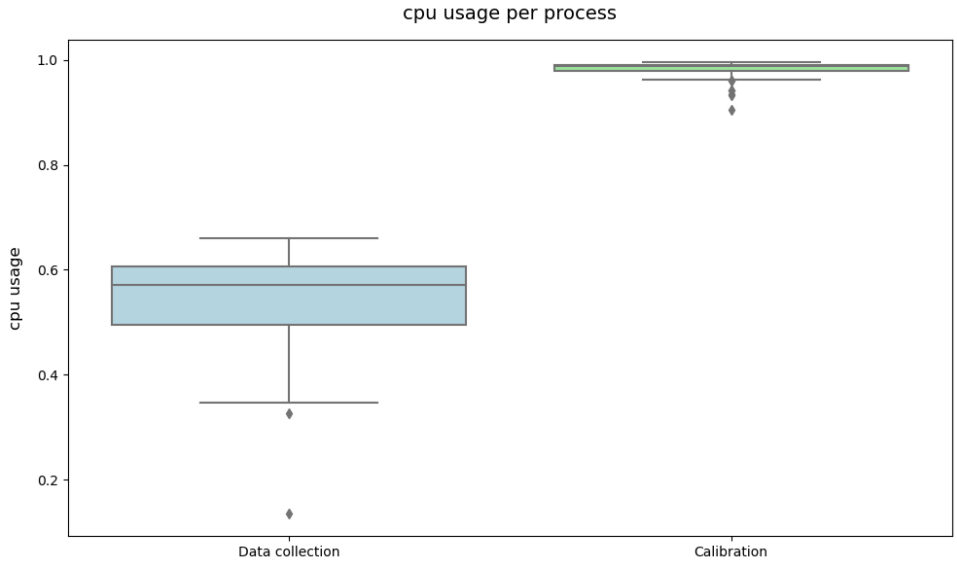
We visualized performance statistics for the successful data collection and calibration processes in Figures B.1 and B.2. The visualization reveals significant variability in process duration across tasks. Data collection processes required substantially more memory due to the large size of raw MS files, while calibration processes demonstrated higher CPU intensity, with all processes approaching 100% CPU utilization.



(a) Boxplot comparing time distributions (minutes) for data collection and calibration processes.

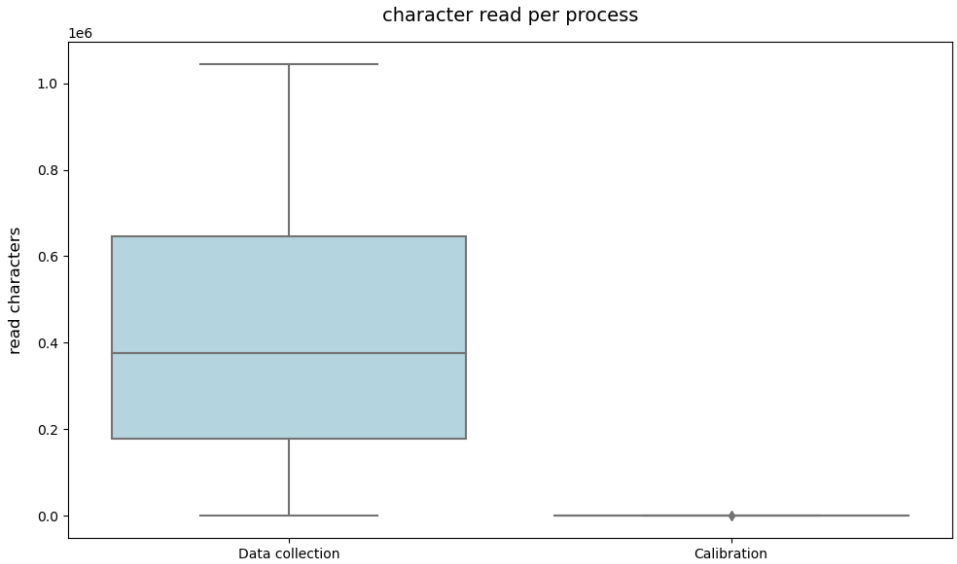


(b) Boxplot comparing memory usage distributions (megabyte) for data collection and calibration processes.

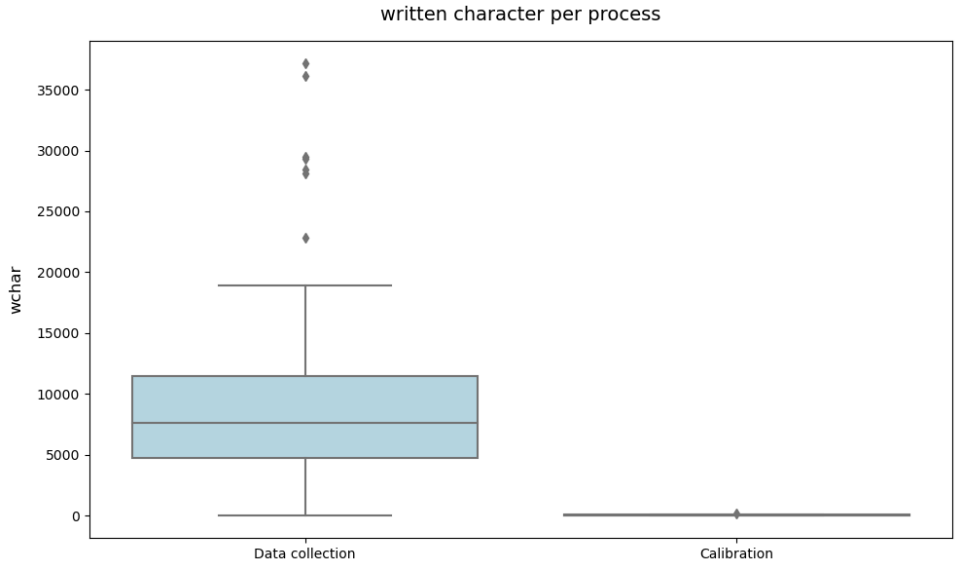


(c) Boxplot comparing cpu usage distributions (percent) for data collection and calibration processes.

Figure B.1.: Pipeline performance statistics



(a) Boxplot comparing characters read for data collection and calibration processes.



(b) Boxplot comparing characters written for data collection and calibration processes.

Figure B.2.: Pipeline I/O statistics