Application of Machine Learning to the Classification of Spectra-like Data

Yue Sun

Department of Software Engineering

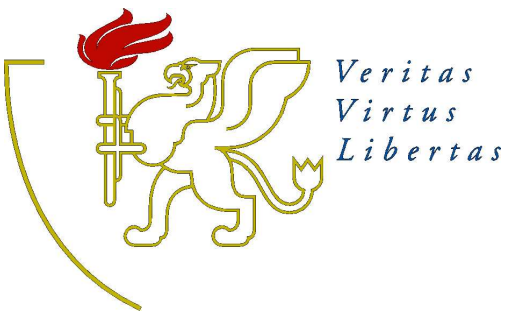
Szeged, 2023.

Supervisors:

Sándor Brockhauser, Péter Hegedűs

A THESIS SUBMITTED FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

OF THE UNIVERSITY OF SZEGED



University of Szeged

PhD School in Computer Science

Preface

Acknowledgements

I would like to thank my supervisors Sándor Brockhauser, Péter Hegedűs for guiding me in my studies and research, without whom I would not have been able to conduct my research. I am very grateful to them for respecting my ideas and giving me the freedom to explore and think. I am also immensely appreciative of their approachability and constant encouragement, which allowed me to enjoy the journey of pursuing my Ph.D. I would like to give special thanks to my supervisor, Sándor Brockhauser, for his inspiration and encouragement during the initial phase of my PhD journey in a foreign country. I joined the European XFEL Data Analysis Group in October 2021 and I would also like to express my sincere thanks to Dr. Luca Gelisio and Dr. Danilo Enoque Ferreira de Lima, whom I regard as my second mentor, for their excellent supervision and support during this period. I would like to thank Christian Plueckthun and Zuzana Konopkova at European XFEL for providing the HED experimental spectral data and valuable discussions and interpretations.

I would also like to thank Dr. Shan Liu from DESY (Hamburg, Germany) for being a mentor and sharing her knowledge and skills with me to help me plan my future academic and life path. I am also very grateful to Dr. Jun Zhu and Dr. Ye Chen for supporting and helping me in my academic career. In addition, I would also like to thank my colleagues in the EUXFEL data analysis group and my fellow EUXFEL PhD students for their continuous help and support over the past years, with which my life has been much more fulfilling and enjoyable. My many thanks also go to my friends, namely Dr. Zhuoni Qian, Dr. Juncheng E, Dr. Jiawei Yan, Dapeng Li, Dr. Zihan Zhu, Dr. Junjie Guo, Dr. Weilun Qin, Dr. Tianyun Long, Dr. Xinchao Huang, Dr. Hao Wang, Dr. Bihan Wang, and Dr. Minxue Tang for their sincere support.

Finally, I would like to thank my boyfriend, Xiayun Pan, and my family for their unwavering support and encouragement throughout my PhD. Their constant support and trust in me has been a source of strength and motivation. I am deeply grateful for their support and love. It should also be mentioned that this research was supported by China Scholarship Council (CSC, No. 201904890020), and the European XFEL.

List of Tables

List of Figures

Contents

1 Introduction

Spectroscopy and diffraction technologies are central to natural sciences and engineering, serving as indispensable tools for investigating matter [144, 96, 85, 88], uncovering new phenomena, and characterizing the properties of various substances and materials [31]. When experiments are performed at modern X-ray facilities, such as synchrotron radiation sources, and X-ray free electron lasers (XFELs), a vast amount of data are potentially collected over short periods of time. For instance, at the European XFEL, X-ray pulses can be generated with only 220ns separation in time and a maximum of 27000 pulses per second. The experimental setups in these facilities are complex, and the high data rates, on the order of 10-15GB per second per detector at the European XFEL [59] necessitate an efficient approach that combines experiments with online data analysis, creating an active feedback system with low feedback latency [121, 8, 9, 24].

The ability to rapidly and accurately extract useful features and provide immediate feedback on the actual state (e.g. time-resolved status of the sample) would be essential to the understanding of the underlying processes governing the experiment and facilitating necessary adjustments. This capability significantly enhances the experiment's efficiency and maximizes the scientific outcomes. On the other hand, when analyzing data already collected –potentially up to hundreds of thousands of data sets –it is crucial to be able to employ some automated or semi-automated methods capable of extracting scientifically interesting features in the data so as to minimize the usage of experts's time and to maximize the scientific output.

In this thesis, we examine 1D diffraction spectra data from high-pressure X-ray diffraction (XRD) experiments and in particular focus on the identification of phase transitions in samples investigated by XRD. High-pressure experiments using either dynamic (shock, ramp) or static compression techniques play a fundamental role in improving our understanding of the behavior of matter under extreme conditions associated with the interior of Earth and exoplanets [86]. In these experiments, changes in pressure will cause changes in the unit cell [86], which are reflected in the spectral peaks, such as differences in peak intensity, position shifts, changes in peak shape (e.g. broadening), and new peaks appearing. One important scientific question in

high-pressure experiments is whether the unit cell undergoes a distortion or a phase transition. Accurate determination of phase transition intervals (the onset and end of the phase transition) is crucial for the calculation of lattice parameters and kinetic investigations [86], which are usually done by experienced experts with the naked eye, dividing the measured spectra into three different categories, i.e., before, during and after the phase transition. However, this process is complex, time-consuming, and of limited accuracy.

Recently, Machine Learning (ML) opens up new avenues for data-driven spectra analysis by offering the possibility of discovering intricate structures and learning representations of high-dimensional data, and implementing an online feedback system that can be used near real-time during data collection. ML encompasses a wide range of computational algorithms and techniques that can train systems from raw data rather than a priori models [20], making it ideal for the automation of repetitive tasks and identification of features and patterns in data sets, thus useful for research facilities that produce large, multidimensional datasets like European XFEL. Several ML applications to data collected at x-ray facilities have been recently published ([141, 44, 1, 143]).

Although Machine learning (ML), especially deep learning (DL) has gained significant traction and demonstrated promising results in various fields, their application in the natural sciences domain is relatively limited compared to areas such as image and video processing and natural language processing (NLP). This is partially attributed to the limitations summarized earlier, which include challenges related to data representativeness, data labeling and annotation, limited public data availability, interpretability and explainability, model generalization, overfitting, and computational resource requirements. These limitations collectively contribute to the cautious adoption of ML and DL techniques in the natural sciences and highlight the need for careful consideration and further research to address these challenges effectively.

It is crucial to address and consider these limitations while applying machine learning and deep learning techniques to 1D diffraction spectra classification to ensure reliable and meaningful results. In this thesis, we employ tools (e.g., data analysis and machine learning) from the field of software engineering to tackle the problem of diffraction spectra classification and in particular the identification of phase transitions in samples investigated by x-ray powder diffraction. Through our research, we aim to demonstrate the immense potential of ML and DL methods in enhancing the scientific efficiency of experiments conducted at large-scale X-ray facilities. By effectively addressing the aforementioned challenges, we can pave the way for improved and reliable results in the classification of diffraction spectra.

Below we provide a concise summary of the thesis, then we summarize the contributions of the author.

1.1 Structure of the Dissertation

In this thesis, we present ML/DL methods tailored for diffraction spectral data classification. The driving force of the approach proposed in this thesis is to design an efficient and automatic classification model for one-dimensional diffraction spectra, based on which we propose several machine learning-based classification models, ranging from traditional ML models with feature engineering, and supervised learning models to self-supervised models. Since spectral classification is the domain-specific problem to

be solved, the key features of the spectral data are considered when designing the ML-based classification models.

The main content and structure of this thesis are as follows:

First, Chapter 1 provides an overview of the work presented in the thesis, describes the motivation and significance of spectral data classification and the application of machine learning methods in this regard, and summarizes the author's contributions. Chapter 2 provides the necessary background for the diffraction spectra used in this thesis and some information about the physical experiment that generates the data. In addition, we review recent works related to traditional ML models and the common state-of-the-art DL techniques (FCNN, CNNs, ResNets, RNNs, and Transformer/Self-attention), including supervised learning methods and self-supervised learning methods applied to spectral time series analysis. The remaining sections of the thesis present and discuss the different kinds of methods we have developed for efficient and automated spectral classification, and also highlight the author's contributions related to the main points of the thesis.

Specifically, Chapter 3 presents a spectra classification model based on the Principle Component Analysis (PCA) and Spectral Clustering. In this model, PCA is applied to reduce the dimensionality, after which the spectral clustering method is applied. To better explain the model, we investigated the contribution of the original variables in spectra to the principle components in PCA and also tested the relationship between the classification results and the number of PCs. In addition, in the absence of ground-truth label information, we propose the classification confidence metric for model evaluation. Despite the high performance we achieved on the example spectral dataset, the method still has some limitations. First, determining the appropriate number or density of clusters to be discovered is a challenge, especially when dealing with complex and high-dimensional spectral data. In addition, clustering algorithms are usually sensitive to the choice of distance metrics, which may lead to inconsistent or unstable outcomes. Lastly, the approach relies on PCA for preprocessing, which can lead to the loss of important discriminatory information. Furthermore, this additional step of feature engineering can impede automation, and introduce extra computational burden and complexity to the interpretation of the model.

Chapter 4 demonstrates a neural network-based statistical model applied to spectra classification. In this ML model, the spectra classification problem is transformed into a 2D space segmentation problem, and machine learning models with a binned-weighting technique are proposed to minimize the misclassification of indistinguishable features in overlapping regions. The solution proposed here can automatically find regions (or bins) with high separability. We also investigated the performance of the model using a different number of bins. The drawback of this method is that it's not an end-to-end solution and is computationally expensive.

In Chapter 5, we first extend the model presented in Chapter 4 to an end-to-end binned FCNN (fully connected neural network) with the automatically capturing weighting factors model. Additionally, the convolutional SCT (spatial-channel-temporal) Attention model is proposed under the setting of 1D time series classification. It is a hybrid model of CNN (convolutional neural network) and self-attention architecture, where self-attention is calculated across spatial, channel, and temporal dimensions, enabling the model to focus on distinguishable features while suppressing noisy and misleading ones. We also designed several other classification model tails for diffraction spectroscopy based on other state-of-the-art ML (machine learning) al-

gorithms, such as FCNN, CNN, Resnet, LSTM, and Transformer. Furthermore, we evaluated and compared the classification performance of these deep learning (DL) models on an example of experimental spectra data from multiple perspectives. In this way, our work provides a standard baseline for 1D spectral time series classification in an end-to-end manner. Finally, we designed a software program based on these end-to-end methods, specifically for spectral classification. The program is implemented in the interactive runtime environment Mybinder and Google Colab, which promotes reproducibility and facilitates the exploration and reuse of ML models.

Chapter 6 introduces a supervised contrastive learning framework combined with data augmentation for 1D spectral representation and classification, the first attempt at its application in the field of diffraction spectra classification. To reveal the underlying and increase the credibility to physicists, We interpret our classification model from multiple perspectives in terms of traditional descriptors of spectral patterns of interest to physicists, such as peak number, peak position, width, and intensity. Furthermore, We evaluate the effectiveness of the network on multiple experimental spectral datasets to demonstrate the high quality and generality of the model.

Although the supervised classification models introduced in the previous chapters demonstrate excellent classification performance by annotating to exploit domain knowledge, the reliance on annotated data poses a challenge for automatic classification algorithms. In Chapter 7, we address this problem by introducing a self-supervised classification framework, namely, the SpecRRMoco-Net, for spectral classification. This method incorporates self-supervised relational reasoning learning [23, 140, 82, 110], and self-supervised contrastive learning [13, 12, 38], along with data augmentation techniques, to enable the extraction of generalizable features from unlabeled data. In addition, the data augmentation techniques presented in this work preserve physically relevant information, and are particularly effective for diffraction spectral time series, allowing efficient definition of pretext tasks. While self-supervised learning requires domain-specific knowledge, the need for human supervision is largely reduced with respect to supervised learning, and the potential for automation is increased. In addition, two other networks, SpecMoco-Net based on self-supervised contrastive learning and SpecRR-Net based on relational inference learning, are proposed and discussed for spectral classification. Experimental results show that SpecRRMoco-Net shows superior performance by benefiting from contrastive learning and relational inference learning. We further demonstrate on multiple experimental spectral datasets that self-supervised machine learning methods can effectively classify spectral data, offering great opportunities to improve the scientific efficiency of experiments at large-scale X-ray facilities.

Finally, Chapter 8 outlines the main content and major contributions of the thesis, and suggests some future directions for further research, such as hyperparameter optimization and extension of our proposed methods to other spectra collected with different spectroscopy and diffraction techniques, and even other common time series data and images.

1.2 Summary of the Results

The main results presented in the thesis are related to the employment of techniques such as data analysis and machine learning in the field of software engineering to develop efficient, automatic, and interpretable ML/DL-based statistical models for 1D

spectral classification. The primary contribution of the thesis revolves around the development of innovative classification models specifically designed for 1D diffraction spectral data, with particular emphasis on identifying phase transitions in samples analyzed through x-ray powder diffraction. The models proposed in this thesis take careful consideration of the unique features of spectral data. To accomplish the objective of efficient, automatic, and interpretable spectral classification, several different types of classification models are systematically proposed in a staged manner. Chapter 3-7 of the thesis delve into the models proposed and the corresponding experiments conducted, providing a comprehensive discussion and analysis of the findings.

The thesis provides a comprehensive and detailed description of these proposed methods for the spectral classification task. Moreover, the validity and efficacy of the models are rigorously tested and verified on various representative experimental diffraction spectra datasets. The thorough evaluation process ensures the reliability and accuracy of the proposed models for 1D spectral classification tasks.

The thesis results are grouped into three major thesis points, effectively demonstrating the author's contribution. The supporting publications related to each thesis point are shown in Table 1.1.

I. **Spectra classification with manual feature engineering.** The contributions of this thesis point are related to applying machine learning methods with feature engineering to spectra classification and will be discussed in Chapter 3 and 4.

   Spectra classification model based on the Principle Component Analysis (PCA) and Spectral Clustering. In this model, PCA is applied to reduce the dimensionality, after which the spectral clustering method is applied. To better explain the model, we investigated the contribution of the original variables in spectra to the principle components in PCA and also tested the relationship between the classification results and the number of PCs. In addition, in the absence of ground-truth label information, we propose the classification confidence metric for model evaluation.

   ML model combined with a binned-weighting technique for spectra classification. As a first step to classify diffraction spectra and detect phase transitions with a neural network-based model, we treat the spectral classification problem as a two-dimensional spatial segmentation problem and propose machine learning models with binned-weighting techniques 4 to minimize misclassification of indistinguishable features in overlapping regions of different spectra.

   In the context of 2D space segmentation, it was observed that data points in overlapping regions of spectra were indistinguishable, rendering the classification of these features meaningless and lacking confidence. To address this issue, a strategy was employed where the spectrum was divided into multiple bins, allowing for the learning of local separability and the derivation of believability weighting factors specific to each bin. By utilizing these weighting factors, a weighted average of individual predictions from all bins was computed to obtain the final classification prediction for each spectral curve. Additionally, the introduction of a new metric called separability, learned from the final accuracy of the trained neural network model in each bin, facilitated the characterization of the trustworthiness of the classification predictions within each bin.

   To validate the initial findings, an experimental spectra dataset was utilized to determine whether the multi-crystal powder system had undergone a phase tran-

sition during a pressure ramping experiment. This validation served to corroborate the efficacy of the proposed approach in discerning phase transitions in real-world scenarios.

The author's contributions. The author evaluated and compared clustering methods and chose the Spectral Clustering method to analyze the spectra data. She studied the principal components (PCs) of the PCA method and analyzed the contribution of the original variables to the PC. In addition, she tested the classification confidence with different cumulative explained variance values in the PCA method. Furthermore, the author performed the empirical validation of the neural-network-based ML model combined with a binned-weighting technique for spectra classification and drew some key conclusions. She implemented, analyzed, evaluated, and presented the results. Finally, she provided the analysis script as Jupyter Notebooks for reproducibility. The publications related to this thesis point are:

- Y. Sun, S. Brockhauser, P. Hegedűs. Machine Learning Applied for Spectra Classification. In *Computational Science and Its Applications-ICCSA 2021: 21st International Conference*, Cagliari, Italy, September 13–16, 2021, Proceeding, Part IX 21, pages 54–68. Springer International Publishing, 2021.
- Y. Sun, S. Brockhauser, P. Hegedűs. Machine Learning Applied for Spectra Classification in X-ray Free Electorn Laser Sciences. *Data Science Journal*, 21, no. 1, 2022.

II. **End-to-End Deep Learning Methods for Spectra Classification.**

The contributions of this thesis point are related to the development of end-to-end deep learning models and an open-source interactive software application for spectra classification, which will be discussed in Chapter 5 and 6.

End-to-End Machine Learning Methods for Spectra Classification. To enhance the robustness of the classification model, we extended the previous approach by incorporating an end-to-end binned Fully Convolutional Neural Network (FCNN) that automatically captures the weighting factors, as presented in Chapter 5. This eliminates the need for hand-designed algorithms and allows the neural network to learn the weighting factors on its own, resulting in a more robust and adaptive model.

More importantly, we realize that time-resolved spectra are more commonly seen as a typical time series. In this context, we propose the convolutional SCT attention model, which can learn better representations by modeling the global dependence of spatial, channel, and temporal dimensions to effectively suppress noisy features with low separation. In addition to this, we have designed and implemented several different classification models based on state-of-the-art deep learning models for spectral classification. These end-to-end solutions bring us closer to achieving efficient and automated classification. We did a comprehensive study comparing these models from several perspectives and concluded that the proposed model in 1D time series classification is superior and that the convolutional SCT attention model is able to best focus on classification-related features and suppress non-suppressible noise features.

Supervised contrastive learning for spectra classification. We further propose a novel classification framework based on supervised contrastive loss for one-dimensional spectral classification of diffraction spectra, inspired by recent studies on supervised contrastive learning, as presented in Chapter 6. Instead of optimizing the model solely with cross-entropy loss, we leverage the power of contrastive learning to enable the model to learn more general representations from the data. The backbone encoder utilized in our framework is based on the convolutional SCT attention model described above.

To ensure the effectiveness and robustness of our proposed framework, we conduct experiments on multiple diverse experimental diffraction spectra datasets, rather than relying on a single example dataset for validation. Additionally, we aim to provide insights into the decision-making process of the deep classification model and bridge the gap between complex models and physicists' understanding. Therefore, we interpret the model in terms of traditional spectral mode descriptors that are of particular interest and relevance to physicists, such as peak position, width, and intensity.

Through the analysis and interpretation of the learned representations, we demonstrate that the network has the capability to automatically extract physically meaningful peak information, which is essential for accurate spectral classification tasks. This interpretation allows physicists to gain valuable insights into the model's behavior and align it with their domain expertise, facilitating a better understanding and utilization of the model in practical applications.

An Interactive Machine Learning Application for Spectra Classification. To enhance data analysis, model reuse, and the advancement of new methodologies, the thesis presents an open-source interactive software program specifically designed for spectral classification. This program incorporates two types of models introduced in the thesis: end-to-end supervised learning methods (Chapter 5) and self-supervised classification methods (Chapter 7, which will be described in next thesis point). By implementing the scripts on the Jupyter Notebook platform and utilizing interactive runtime environments such as Mybinder and Google Colab, the software program facilitates improved reproducibility, as well as effortless evaluation and exploration of data and models.

The author's contributions. First, the author conducted an in-depth analysis of diffraction spectra data and identified the need for new classification models tailored specifically for spectra data. Then, the author introduced novel end-to-end supervised classification models with the objective of improving performance and enhancing automation and efficiency by eliminating the need for manual feature engineering. In order to interpret the classification models, the author conducted a feature importance analysis based on gradient backpropagation, providing insights into the significance of different features. In addition, she proposed a spectral classification model based on supervised contrastive learning, which is the first attempt to apply this method to diffraction spectra. Furthermore, the author interpreted the models from multiple perspectives, employing traditional descriptors of diffraction spectral patterns relevant to physicists, such as peak number, peak position, width, and intensity. Lastly, the author's contribution extends to the development of an open-source interactive software program that facilitates these end-to-end approaches to spectral classification. The publications

related to this thesis point are:

- Y. Sun, S. Brockhauser, P. Hegedűs. Comparing End-to-End Machine Learning Methods for Spectra Classification. In *Applied Sciences*, pages 11520, 2021.
- Y. Sun, C. Plückthun, S. Brockhauser, P. Hegedűs. An Interactive Machine Learning Application for Spectra Classification. In *Computational Science and Its Applications-ICCSA 2023: 23rd International Conference*, 2023 (Accepted, to appear).

III. **Self-Supervised Approaches for Spectra Classification.**

The contributions of this thesis point are related to the development of more automatic spectral classification models based on self-supervised learning, which will be discussed in Chapter 7.

Self-supervised spectra classification models. Although end-to-end supervised classification models, particularly the convolutional SCT attention model, have shown high accuracy in learning data representations and making predictions directly from raw data without the need for feature engineering, the manual intervention required for data labeling still hampers the automation process of spectral classification.

To address this limitation, this thesis introduces self-supervised classification frameworks 7 based on self-supervised relational reasoning (SpecRR-Net), self-supervised contrastive learning (SpecMoco-Net), or a combination of both (SpecRRMoco). The pretext tasks and data augmentations proposed in this work are specifically tailored to the scientific problem at hand, preserving physically meaningful information. Additionally, an inter-temporal relational reasoning module is incorporated to capture temporal dependencies in time-resolved spectral data, which significantly improves the quality of learned representations. These self-supervised models have the ability to learn discriminative features and construct effective representations, thereby reducing the reliance on a large number of labeled samples and advancing the automation of spectral classification. Moreover, the reduced need for labels could lead to significant time savings for scientists. Lastly, the thesis emphasizes the importance of selecting appropriate data augmentations that are specifically designed for the particular study case, ensuring the retention of scientifically meaningful information.

Self-supervised loss function and its validation. The proposed self-supervised loss function in the SpecRRMoco-Net unifies the relational reasoning-based model (SpecRR-Net) and contrastive learning-based model (SpecMoco-Net), and is a linear combination of these two networks. To our best knowledge, this is the first time they have been combined together. We report on an ablation study on the combination coefficient in the loss function, which was performed to understand its impact on learning data representations.

Comparison of these self-supervised methods We compare these three self-supervised classification models from several perspectives, including performance in downstream classification tasks, as well as clustering power analysis, such as visualization of embedded features learned by each model, and evaluation by Silhouette Score and Mutual Information metrics. We concluded that SpecRRMoco-Net

shows superior performance by benefting from contrastive learning and relational inference learning. Furthermore, by comparing with a state-of-the-art self-supervised relational reasoning model without an inter-temporal relational inference module, we also emphasize the importance of this module we introduce.

Ablation studies on the data augmentation. We also conducted an ablation study on data augmentations performed in order to evaluate their impact on the models' performances, emphasizing the need to tailor data augmentation approaches to the specific case study. It is worth noting that the data augmentation involved in the self-supervised learning approaches respects the fundamental invariance of the physical problem. We also observed that the sequence in which data augmentation techniques are applied may have an impact on the results, as different orders yield distinct enhanced data due to the inherent randomness associated with each data enhancement technique. As data augmentation is domain-specific, it must be customized for data sets from different research areas.

The author's contributions. The author's contribution to this thesis point encompasses several key aspects. First, the author proposed novel self-supervised classification models, including the design of pretext tasks and the selection of data augmentation, to achieve improved automation and efficiency in spectral data classification. Second, she conducted an ablation study on the selection of data augmentation techniques, crucial to ensure that scientifically meaningful information is retained. Furthermore, she implemented and extensively validated these models using several experimental spectral datasets for a comprehensive evaluation from multiple perspectives. In addition, the analysis and discussion of the results is also a contribution of hers. The publications related to this thesis point are:

- Y. Sun, S. Brockhauser, P. Hegedűs, C. Plückthun, L. Gelisio, and D. Ferreira. Application of self-supervised approaches to the classification of X-ray diffraction spectra during phase transitions. In *Scientific Reports*, 13, no. 1, pages 9370, 2023.
- Y. Sun, C. Plückthun, S. Brockhauser, P. Hegedűs. An Interactive Machine Learning Application for Spectra Classification. In *Computational Science and Its Applications-ICCSA 2023: 23rd International Conference*, 2023 (Accepted, to appear).

$\mathcal{N}o.$	[106]	[104]	[105]	[107]	[109]
I/1.	•	•			
II/2.			•		•
III/3.				•	•

Table 1.1: Thesis contributions and supporting publications

2
Background

Spectroscopy and X-ray diffraction techniques encode ample information on investigated samples. Advancements in technology, including faster detectors and brighter X-ray sources, have led to an increase in data volume and temporal resolution in X-ray or diffraction experiments. Detecting phase changes in samples during these experiments is crucial, considering the limited and valuable time available in large facilities such as synchrotron, FEL, or neutron sources. Efficient and accurate screening and processing of data, particularly in identifying phase transitions, play a vital role in maximizing the utilization of beam time. This enables researchers to derive meaningful insights from the experimental data within the constraints of the allotted time, leading to more efficient and productive scientific outcomes. However, traditional data processing techniques and algorithms were insufficient to handle the scale and complexity of big data generated in this field.

Fortunately, the development of machine learning technology has provided a solution. Machine learning algorithms have the capability to analyze and comprehend such vast amounts of data by uncovering hidden patterns, trends, and correlations. This has significantly enhanced the processing capabilities of big data, enabling its effective utilization across various fields, including the realm of scientific research. The incorporation of machine learning techniques into the analysis of experimental data in recent years has opened up new possibilities and revolutionized the way researchers approach data processing and interpretation. With machine learning, researchers can now leverage advanced algorithms to extract valuable insights from the immense volume of data generated in X-ray or diffraction experiments, enabling more accurate and efficient detection of phase transitions and facilitating scientific progress in a time-sensitive environment.

Despite the significant advancements in machine learning (ML) techniques for spectra classification, there are still some noteworthy issues and limitations to be addressed, including:

- *Complex spectral patterns*: Spectral data often contain complex patterns and relationships that are challenging to capture using traditional ML algorithms. Therefore, there is a need to explore more advanced ML/DL techniques or develop

specialized models tailored to the specific characteristics of spectral data.

- *Automation and efficiency*: Despite the progress made in automating spectral classification, there is still a need to improve the efficiency and automation of the classification process. It is crucial to develop more efficient models or explore techniques such as self-supervised learning to reduce the reliance on labeled data.

- *Generalizability*: ML models developed in previous works may exhibit limitations in terms of their generalizability to new and unseen spectra. The performance of models trained on specific datasets may not transfer well to different experimental conditions. It is important to develop more robust and generalizable models.

- *Interpretability*: While ML models can achieve high classification accuracy, their decision-making processes are often considered "black-box" and lack interpretability. It is essential to develop methods that not only provide accurate predictions but also offer explanations or insights into how the models arrived at those predictions, especially for scientific applications where interpretability is crucial.

The methods proposed in this thesis (Chapter 3-7) will address the limitations outlined above. In this chapter, we briefly describe the diffraction spectra data and give an overview on the high-pressure X-ray diffraction experiments that generated these data. Furthermore, we present a literature review of spectral time series analysis for related ML applications.

2.1 High Pressure X-ray Diffraction Experiment and Spectra Data

The spectra we examined are 1D diffraction spectra data from high-pressure X-ray diffraction (XRD) experiments. These high-pressure experiments allow examining the evolution of lattice parameters and phase transition during compression and decompression events. All the experiments were performed at PETRA III Extreme Conditions Beamline P02.2 using the LAMBDA (Large Area Medipix-Based Detector Array) GaAs 2M detector system (a multi-megapixel hard X-ray detector to measure the scattering intensity for experiments at synchrotrons), as described in [84]. The measurements were performed at 25.6 keV beam energy (corresponding to the incident beam wavelength of 0.4828 Å) [86]. A simple sketch of the experimental setup is shown in Figure 2.1. The frame rate of the detector enables up to 2000 frames per second, giving the possibility of a 0.5 ms period per image. Samples involved in this thesis were iron (Fe), wüstite (FeO), and $(Mg_{0.2}Fe_{0.8})O$ magnesiowüstite which is a solid solution of the endmembers periclase (MgO) and wüstite (FeO).

In these experiments, diffraction images (Figure 2.2) were continuously collected during predefined trapezoidal ramp profiles. The exposure rates of 1-100 ms enable high resolution of the compression interval and maintain a reasonable quality of the diffraction signal (intensity). Azimuthal integration and baseline subtraction [25] are applied to the 2D raw diffraction images to obtain 1D integrated spectra. Figure 2.2 shows an example of how the spectral peak positions in a spectral curve reflect the intensity distribution in the corresponding raw diffraction image as a function of the scattering angle. A peak in the diffractogram or spectrum corresponds to a diffraction ring [84].

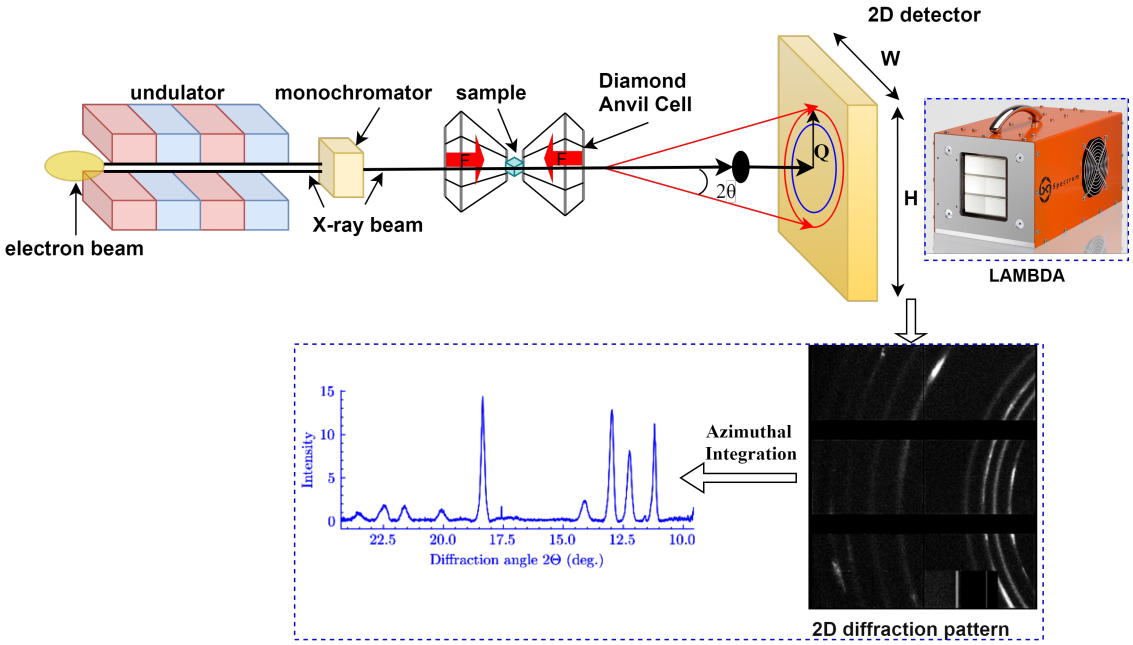


Figure 2.1: Schematic diagram of the High Energy Density (HED) experiment setup.

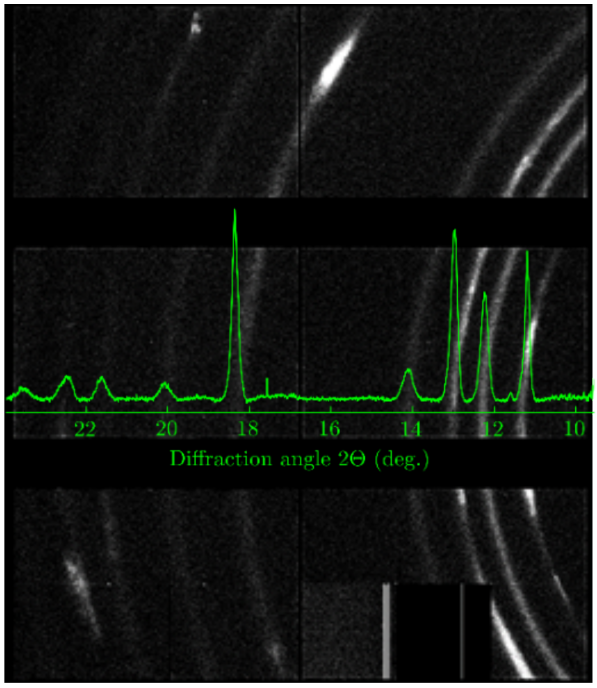


Figure 2.2: This figure was measured at PETRA III beamline P02.2 using a 25.6 keV beam. The 2D raw diffraction image shows the starting condition at 1 GPa and with fcc/B1 $(Mg_{0.2}Fe_{0.8})O$ sample. After azimuthal integration, the corresponding 1D spectrum is shown in green on top of it.

In these high-pressure XRD experiments, changes in pressure will cause changes in the unit cell, which are reflected in the distribution of spectral peaks, such as differences in peak intensity, position, shape (e.g., spread), and number of individuals.

A total of 10 datasets are involved in this thesis. Specifically, one dataset was collected from sample $(Mg_{0.2}Fe_{0.8})O$ magnesiowüstite, which was compressed from 1

to above 100 GPa over a period of $50s$ while maintaining reasonable diffraction image quality in consecutive 100 ms exposures. This dataset consists of 349 spectral profiles, each containing 4020 data points. However, this dataset is not labeled. For sample Fe, a transition from body-centered cubic (BCC) to Hexagonal Close-Packed (HCP) is expected above approximately 13 GPa [95, 86]. In this case, the task set for our machine learning methods is to detect the BCC-HCP phase transition, and in particular to identify patterns corresponding to either the BCC or HCP atomic arrangement which are characterized by different Bragg peaks, or the region corresponding to the transition between the two. For this sample, seven different data sets were collected, each one characterized by the application of pressure at different rates or signal-to-noise ratio (individual data sets are labeled as D1 to D7). Further details for each collected dataset, including the number of scattering curves corresponding to before (BCC), during, and after (HCP) phase transition are shown in Table 2.1. To simplify the presentation of results, they are all summarized as data set "Fe" in later sections. In the case of sample FeO, the target is the identification of the B1 to rB1 crystal structure transformation above approximately 14-15 GPa [27, 86]. Also in this case, two data sets were collected by applying pressures at different rates (D8 and D9, details in Table 2.1). To simplify the presentation, they are summarized as "FeO". Data sets D1 to D9 are composed of a different number of scattering curves, 60 to 460 (see Table 2.1), each containing approximately 4,000 data points (features). These nine datasets are annotated to evaluate the performance of the model presented by the authors in Chapter 6 and 7 in the later part of the thesis. However, these labels are not used to train the model, but only to evaluate it. The number of labels annotated for each class is also reported in Table 2.1. Please notice that only a limited fraction of labels corresponding to the transition between two phases in FeO is available. Representative curves are shown in Fig. 2.3. Pressure, up to peak values of 65 GPa, was applied to the samples using either piezo actuator-driven dynamic diamond anvil cell (dDAC) [22, 49] or the membrane diamond anvil cell (mDAC) [63]. Further details on the experiment are provided in Ref. [86].

Figure 2.3 shows X-ray scattering curves corresponding to an example dataset collected from FeO powder sample. As can be seen from this figure, before the phase transition (the first stage), due to the change in pressure, the peak position shifts, and the peak intensity changes. The second stage, during phase transition, is mainly manifested by the broadening and initial splitting or appearance of new peaks, and the third stage (after phase transition) is mainly manifested by newly emerging peaks. Furthermore, the spectral time series not only has a specific distribution along the diffraction angle dimension but also has a strong dependence on the temporal dimension.

In these experiments, immediate feedback on the actual state (e.g. time-resolved status of the sample) would be essential to quickly judge how to proceed with the experiment. Since spectral changes can indicate the change of the system under investigation and so the progress of the experiment, it is critical to effectively automate and streamline the screening and analysis of spectral data by capturing major spectral changes and extracting useful features.

Machine Learning (ML) excels in discovering patterns and structures in high-dimensional data [62], opening new avenues for data-driven analysis in spectroscopy, offering the possibility of quickly identifying such specific variations and enabling online feedback systems that can be used in near real-time during data acquisition. Therefore, in this thesis, we employ machine learning tools in the field of software engineering to

Dataset	Sample	$T\,(ms)$	$P°(GPa/s)$	N	$B/D/A$
D1	Fe	10	260	60	17/0/43
D2	Fe	100	10	116	29/0/87
D3	Fe	10	160	70	21/0/49
D4	Fe	1	1,030	130	33/0/97
D5	Fe	100	10	130	37/0/93
D6	†Fe	10	262	70	22/0/48
D7	‡Fe	10	287	258	19/0/239
D8	FeO	50	7	222	71/20/131
D9	FeO	50	2.2	460	90/65/305

Table 2.1: Information on the samples used in the X-ray powder diffraction experiments. For every dataset, the sample is either composed of iron (Fe) or wüstite (FeO). The phase transition for all Fe and FeO samples is detected at a pressure of about 15 GPa. The compression rate ($P°$) is also reported, together with the detector exposure time (T). The last two columns show the total number of diffraction patterns collected for a given dataset (N), and the one corresponding to the three possible classes, as labeled by an expert. These are before (B), during (D) and after (A) phase transition, from the BCC to HCP structure for Fe, from B1 to rB1 for FeO. † A pressure calibrant (platinum) was employed. ‡ A pressure medium (neon) was employed.

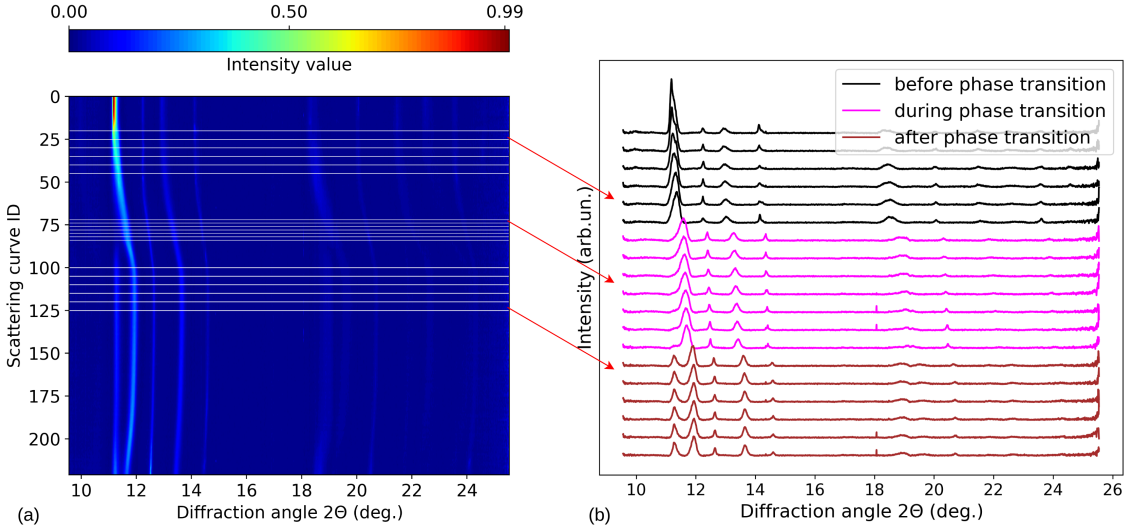


Figure 2.3: X-ray scattering curves corresponding to an example dataset collected from wüstite powder sample. (a) Intensity distribution for different curves collected by applying different pressures drawn as a contour plot. The horizontal lines correspond to representative curves shown in (b). Here, the color black corresponds to the original B1 structure (that is, before the phase transition), magenta to the transition, and brown to the rB1 structure (that is, after the phase transition). In (b) curves are shifted vertically to improve visualization.

solve the diffraction spectra classification problem and to provide immediate information on the experimental phase based on the classification of the measured spectra.

2.2 Related work

In a broader sense, the diffraction spectra data utilized in this thesis can be considered as standard yet typical time series data. As a result, the literature review will primarily focus on the application of machine learning techniques to time series data, with a specific emphasis on those applied to time-resolved spectra. An important point to highlight is that the model proposed for general time series data can be easily expanded to accommodate the specific characteristics of time-resolved spectral data. By incorporating the unique features and properties of spectral data, the machine learning model can effectively capture the temporal dynamics and patterns present in time-resolved spectra for spectral classification and the detection of phase transition.

Various machine learning methods have been successfully applied to time series data, as well as time-resolved spectra data in diffraction and spectroscopy techniques, including traditional machine learning methods and the-state-of-the-art deep learning methods. In this section, we will briefly discuss the successes and limitations of these models.

Traditional machine learning models encompass a range of techniques, including clustering methods and supervised classification methods, applied in various stages of data analysis. Clustering methods such as Spectral Clustering [143, 51], K-Means [37], Agglomerative Clustering [74], DBSCAN [21], are usually employed in the exploratory stage of data analysis. While classification methods such as k-nearest neighbor (KNN) [134], Partial Least Squares Discriminant Analysis (PLS-DA) [11], Extreme Learning Machine (ELM) [138, 137], random forest [72], and support vector machine (SVM) [41] are typically used for supervised classification.

Deep neural networks have received an increasing amount of attention in real-world time series analysis in recent years [53]. A large variety of modeling approaches for univariate and multivariate time series, with deep learning models are recently challenging or replacing the state-of-the-art in a broad range of tasks such as forecasting, regression, and classification [60, 113, 133].

The most commonly established deep learning models in this area are CNN [30, 81, 18], recurrent neural networks (RNN) [60, 35, 64], and attention-based neural networks [119, 125, 4]. The fully connected architecture is the simplest and most basic architecture, and it is the foundation of all deep learning methods. Usually, it is used in conjunction with other architectures. And CNN is based on the shared-weight architecture of the convolution kernels [129], which can easily capture features. However, since the CNN-based models can only learn local neighborhood features, recently, RNN-based models, attention-based models, and hybrid models are increasingly popular for learning from time series data.

These deep learning models are typically trained by supervised learning, which can achieve high performance, but are also built at the cost of extensive manual labeling. Recent advances in the field of self-supervised learning provide a solution to reduce the reliance on data labels [77]. This approach greatly reduces the need for a large number of labels by leveraging data augmentation techniques and pretexting tasks. By learning general representations from unlabeled data, self-supervised learning enables easier classification of data, reduces the reliance on labeled data, and offers the possibility of increased automation. Its training process is more time-consuming due to a large amount of unlabeled data, and since its training is not task-specific, it usually achieves slightly worse results compared to supervised learning. However, it is possible to achieve

comparable performance to supervised learning if the pre-text tasks can be designed efficiently.

2.2.1 Tranditional Machine Learning-Based Approaches

Numerous traditional machine learning methods have been developed for time series data or specifically spectra data classification [138]. These traditional methods are unsupervised learning or nonparametric algorithms that do not contain learnable parameters but are more procedural [14]. Time series data, especially time-resolved spectra data generated in spectroscopy and diffraction experiments, commonly have high dimensionality, and it is very common for the number of input variables (features) to be greater than the number of samples(considering the limited access to new X-ray sources, such as X-ray free electron lasers (XFELs) or synchrotron sources, time-consuming setups and experiments, and the high cost of building and maintaining such large scientific facilities), which is more likely to lead to overfitting. In this case, traditional machine learning algorithms are required to effectively identify underlying patterns or key descriptors in spectra data [14].

Traditional ML classification methods are known for their simplicity and effectiveness in various pattern recognition problems. For instance, principle component analysis (PCA) is usually used as a dimensionality reduction by using an orthogonal transformation to convert original data into a set of uncorrelated variables of higher space; k-means clustering is widely used for exploratory data analysis, applied to data clustering. Among unsupervised clustering methods, spectral clustering is a clustering method that does not make assumptions about the global structure of the data [51]. It can solve very general problems like intertwined spirals and can be implemented efficiently even for large data sets [120]. For example, Jebara et al. [48] combined nonparametric spectral clustering with parametric hidden Markov models for time-series data analysis, and achieved great clustering accuracy. However, one major limitation of unsupervised clustering algorithms is that it can be challenging to determine the appropriate number or density of clusters to be discovered [26, 73, 87]. This often requires fine-tuning certain hyperparameters to obtain accurate results. In addition, it is difficult to exploit label information annotated by experts or self-generated supervision signals from the data.

Traditional supervised ML models are usually applied after the data exploration phase and provide better performance. Ref. [100] presented a novel method called nearest clusters-based PLS-DA (NCPLS-DA) to analyze the classification of spectral data. The proposed method can effectively address the multimodality and nonlinearity issues. The work [138] proposed a framework for spectra data classification using the kernel extreme learning machine (KELM). They used PCA as spectra preprocessing and dimensionality reduction, and then applied the kernel extreme learning machine for classification. One of the most relevant studies to our work [2] presented the PhaseMapper, based on convolutive nonnegative matrix factorization (NMF), which can effectively solve the multi-phase identification problem in the context of x-ray diffraction (XRD). Random forest, which is an ensemble learning method based on multiple decision trees, always results in the convergence of generalization error [99], and can be used for classification and regression. Ref. [111] presents a simple tree-ensemble-based supervised ML model for the determination of crystal system and space group from 1D diffraction patterns. They address the important criticism related to the inter-

pretability of ML models. However, their research is in the proof of concept (POC) stage, where the model is trained on ideal simulated diffraction data rather than actual experimental data. Even though these traditional ML classification methods are widely recognized, there are still some problems. One such limitation is their struggle with high-dimensional data, making it challenging for them to excel in complex classification tasks. For instance, in our case of determining the onset and end of phase transitions, which occur during a phase transition phase, these methods may not perform optimally. Additionally, they often require feature engineering and rely heavily on preprocessing. For example, they usually require the pretreatment of data and rely heavily on preprocessing such as PCA, which makes end-to-end work difficult. Moreover, the dependency on labeled data hinders efficiency and automation processes.

2.2.2 Convolutional-based Approaches

Convolutional neural network (CNN) [30, 81, 18, 125] is one of the most commonly established deep learning models in time series data analysis. It can easily and effectively capture features in multi-scale by different convolution kernels and pooling operations.

Many powerful CNN-based models have been proposed and achieved great success. Among them, the work [129] applied a combination of 1D and 2D CNN to multisource multitemporal satellite imagery for crop classification. It explored the features across spectral and spatial dimensions but did not consider temporal information. Meanwhile, Temporal Convolutional Neural Network (TempCNN) [83] where convolutions are applied in the temporal dimension, is also well explored. In Ref. [83], an exhaustive study of one-dimensional TempCNNs for the classification of Satellite Image Time Series was conducted, their experimental results show that TempCNNs are superior to RNNs in terms of SITS classification. The Conv1D-based classification model designed in Ref. [139] further proved the effectiveness of TempCNNs in processing the temporal dimension of time series classification. Residual Networks (ResNets) [40] are another common backbone architecture in computer vision and are also adapted to time series classification [90, 125, 65], the residual connections or shortcut [40] in the model numerically add higher-level features to the forward propagated representation [90] and helps to propagate the gradient back through the network, so very deep models can be trained [40]. Based on ResNet, Multiscale architectures [65] can process time series at multiple scales and achieve feature fusion by concentrating each stream. Fully convolutional network (FCN) [39] which can yield a fixed-length feature for classification has also been well exploited and has been shown to achieve the state-of-the-art performance for end-to-end time series classification [53]. In addition, dilated convolutions are also widely used to improve the feature resolution [132, 6].

In scientific research, the enormous potential of CNN-based methods has also been demonstrated in spectroscopy and diffraction technologies, including bacterial classification from Raman spectra [42], classification of edible oils using low field nuclear magnetic resonance [43], and spectra classification in the field of Neutron and X-Ray Scattering [14, 79]. For example, Oviedo et al. [79] represented their XRD pattern as time series and applied a CNN-based model to achieve the rapid prediction and classification of crystallographic dimensionality and space group.

Although great progress has been achieved by CNN-based networks, there are still some limitations, namely the inability of leveraging the relationship between features in a global view. To utilize both spatial and temporal features from time series, combi-

nations of convolutional operation and recurrent operation [53, 54]]or attention mechanism [29, 15] have been explored and comprehensively compared.

2.2.3 Recurrent-based Approaches

For time series data processing, two variants of the RNN models, Long Short-Term Memory (LSTM) [92], GRU (Gated Recurrent Unit) [16], in particular, can effectively capture temporal dependencies, thus can work efficiently on various complex time series processing tasks, such as prediction, recognition, and classification [60, 35, 64, 17, 97, 45]. For example, Lipton et al. [64]showed the ability of LSTMs to recognize patterns in multivariate time series of clinical measurements. In the meantime, RNN-based architecture is also used in combination with the CNN-based module to construct a hybrid convolutional-recurrent neural architecture that can automatically extract features and capture their short-term and long-term dependencies at the same time [60, 53, 30]. Lai et al. [60] a Long- and Short-term Time-series network (LSTNet) framework for multivariate time series forecasting. The method combines the strengths of CNN and RNN, can effectively extract short-term local dependency patterns and long-term patterns in data at the same time. In addition, they exploited an attention mechanism to alleviate the issue of nonseasonal time series prediction. Karim et al. [53] proposed LSTM FCN and ALSTM-FCN deep learning models for end-to-end time series classification, which are enhancements of a Fully Convolutional Network (FCN) with LSTM sub-module or attention LSTM sub-module. Although the enhanced models can significantly improve classification performance, the limitation is that it is designed for univariate time series. In 2019, the authors [54] subsequently introduced a squeeze-and-excitation block to augment the FCN block in the existing LSTM-FCN and ALSTM-FCN models, which can capture the contexture information and channel-wise dependencies, so that the model can be used for multivariate time series classification. Interdonato et al. [45] proposed an end-to-end DuPLO deep learning architecture for the analysis of Satellite Image Time Series data. It is also a combination of both CNNs and GRUs models, which could exploit feature information from different points of view, and then produce a more diverse and complete information representation for classification tasks [45].

RNN-based models have also been applied to spectra classification in spectroscopy. For example, in [45], the authors detected the pseudo-brain tumors via stacked LSTM neural network with Magnetic resonance spectroscopy (MRS) data, achieving high-precision binary classification of brain tumors and normal brain tissue. In [123], the RNN models were built to analyze Raman spectra to classify blood species. However, the application of RNNs to time-resolved neutron or x-ray scattering experiments is still scarce [14].

2.2.4 Attention-based Approaches

Very recently, inspired by the Transformer scaling successes in NLP [119], researchers have also successfully developed their Transformer-based or attention-based models in the tasks of time series analysis like video understanding [4], forecasting of multivariate time series data [97], satellite image time series classification [30], hyperspectral image (HSI) classification [39], and other time series classification tasks [133, 125]. Instead of processing data in an ordered sequence manner, the Transformer model processes the entire sequence of data in a parallel manner and uses self-attention mechanisms to

learn global dependencies in the sequence [127].

Ma et al. [68] first proposed a novel approach called Cross-Dimensional Self-Attention (CDSA) for the multivariate, geo-tagged time series data imputation task. The CDSA model can jointly capture self-attention across multiple dimensions (time, location, measurement), yet in an order-independent way [93]. Garnot et al. [30] proposed a spatio-temporal classifier for automatic classification of satellite image time series, in which a Pixel-Set Encoder is used to extract spatial features, and a self-attention-based temporal encoder is used to extract temporal features. This architecture has achieved significant improvements in accuracy, time, and memory consumption. Rußwurm et al. [90] explored and compared several commonly used state-of-the-art deep learning mechanisms on preprocessed and raw satellite data, such as convolution, recurrence, and self-attention for crop type identification. They pointed out that preprocessing still improved the classification performance of all models they applied, while the choice of model was less crucial [90]. In addition, their experiments showed that self-attention and RNNs outperform CNNs on raw satellite time series. They investigated this further by a gradient-based feature importance analysis and qualitatively showed that the self-attention mechanism can focus selectively on a few classification-relevant observations. But their work focuses on applying, comparing, and analyzing existing models, rather than proposing new architectures or models. Although in most cases, the attention-based architecture used for time series analysis is used as a supervised learning method, in 2020, Zerveas et al. [133] first proposes a transformer-based framework for unsupervised representation learning of multivariate time series. Even with very limited training samples, this model can still exceed the current state-of-the-art performance in the classification and regression tasks of multivariate time series, and can potentially be used for other downstream tasks, such as forecasting and missing value imputation [133]. In Section 2.2.5. we will delve into a specific area within the field of deep unsupervised learning approaches, providing a comprehensive and detailed exploration of its concepts and methodologies.

Similar to [90], our work presented in Chapter 5 also applies and compares the state-of-the-art 1D-CNN, RNN (LSTM), and self-attention mechanism. But we focus on proposing new architectures for spectral time series classification, and deal with this problem from two different perspectives, either as a 2D space segmentation problem (end-to-end binned FCNN with automatically capturing weighting factors model, as described in Section 5.1.1) or as a general 1D time series classification problem (e.g. convolutional SCT attention model, as described in Section 5.1.2). Moreover, their models are designed for crop-type classification, while our models focus on the application of spectral data classification. In Chapter 3, we used PCA as data preprocessing for dimensionality reduction, and applied unsupervised spectra clustering, and supervised LSTM-based and Transorformer-based models for 1D diffraction spectra classification. These models are not end-to-end solutions, nor can they take advantage of temporal dependencies. Another work done by Ismail et al. [46] conducted a detailed and comprehensive study of the most recent state-of-the-art performance of deep learning algorithms for time series classification. In their review work, they detailed several deep learning models based on Multi-layer perceptrons, CNN, ResNet, Encoder [93], and Time-CNN [136]. They trained and tested these models across multiple time series datasets including univariate and multivariate, and analyzed the corresponding advantages and drawbacks. But they did not include and compare the state-of-the-art Transformer-based or self-attention-based model, and RNN-based models (such as

LSTM or GRU), which are currently the most state-of-the-art structures. The only attention calculation they involved is in the Encoder model, and it is soft attention. Different from [46] our work implemented these two methods for spectra times series classification, and also focus on the attention-based architecture to leverage information from different dimensions to extract more complete and powerful feature representations.

While these different types of state-of-the-art deep supervised models outlined above can successfully characterize the spectra data and achieve good classification results, they are based on supervised learning and require extensive manual labeling, which is cumbersome and relies heavily on domain experts. In addition, compared to self-supervised or unsupervised learning, supervised learning tends to be less generalizable and more prone to overfitting, especially when your training dataset is not diverse enough.

2.2.5 Self-Supervised Learning Approaches

Recent self-supervised learning techniques have opened up a new research frontier, with deep learning architectures that can learn generalizable and useful features from unlabeled data [77], but much less research has been done on spectral time series data. The task of self-supervised learning is usually accomplished with some sort of data augmentation through which deep neural networks can extract relevant information. The most crucial point for self-supervised learning's success is that it exploits unlabeled data by imposing invariance to appropriate pretext tasks, thus largely getting rid of human supervision, while relying on the domain-specific knowledge that the relevant characteristics of the data are invariant to certain transformations. The term "pretext" implies that the task being solved is not aimed at the target task, but is only solved for the purpose of learning a good data representation [47]. The pretext task is designed according to the problem to be solved, for example, in contrastive learning [12, 13, 38], it designs a similarity metric that aims to group similar samples closer and expel different samples farther apart [47]. After the self-supervised pre-training, useful representations can be produced and used for downstream tasks, but relying on a significantly smaller number of labeled samples. In this thesis, we focus on two branches of self-supervised learning, that is self-supervised relational reasoning learning [23, 140, 82, 110] and self-supervised contrastive learning [13, 12, 38].

Relational Reasoning Learning

The relational reasoning networks are based on a key design principle, that is the use of a relation network (usually a multi-layer perception) as a learnable function to quantify the relations between entities and their properties [82]. While the relational reasoning paradigm has gained traction in the deep learning community only recently [82], it has achieved promising results in many fields, such as video processing [140], few-shot natural image recognition [110], and time series data classifcation [23]. For example, Ref. [91] trained a relational reasoning model for question answering, and reasoning about a dynamic physical system. Ref. [140] applied the relational reasoning model to videos, where a temporal relational reasoning model was presented to learn and reason about temporal dependencies between video frames at multiple time scales. Ref. [110] trained a relational reasoning model for few-shot recognition. By learning to discriminate how entities relate to themselves and other entities, their model can

learn rich and descriptive representations. A study closest to our work is shown in Ref. [23], where the authors proposed a relational reasoning framework for the time series data classification. Unlike earlier work that attempted to train a relational reasoning model for a single specific task, they designed a relational reasoning model with two pretext tasks to exploit both inter- and intra-sample relationships of unlabeled data. This strategy allows the modeling of global and local dependencies in the data, yielding better feature representations. However, their work only focuses on commonly used time series data. The application of relational reasoning learning in the natural sciences is still scarce, leaving ample room for exploration and further investigation. In this thesis, we explore relational reasoning learning for spectra data classification in diffraction experiments.

Inspired by Ref. [23], in our SpecRR-Net presented in Chapter 7, we explored the feature dependencies from different dimensions and at different scales. Unlike the data in Ref. [23], our spectral time series has a strong dependence along the temporal dimension, as shown in Figure 2.3. To model the global dependencies in this dimension, we further extend Ref. [23] by introducing an inter-temporal relational reasoning head to address the temporal dependencies of spectral data, which is a key feature of spectral time series that can greatly improve the quality of learned representations. Finally, a relational reasoning network (SpecRR-Net) with three relational reasoning branches is designed, namely the inter-sample, intra-sample, and inter-temporal relation modules. In this way, pretext tasks based on these three relational reasoning modules are designed, under the supervision of these pretext tasks, the backbone encoder can obtain inter-class, intra-class, and inter-temporal knowledge from spectra without requiring label information.

Self-Supervised Contrastive Learning

Contrastive learning [34] is a framework for learning similar/dissimilar representations from data that are organized into similar/dissimilar pairs [13]. It is closely related to metric learning [58]. The fundamental difference between contrastive learning and metric learning is that metric learning is usually based on a supervised learning architecture [82]. Usually, metric learning methods based on contrastive loss and Noise-Contrastive Estimation (NCE) [33] are referred to as contrastive learning methods [82]. The core idea of contrastive learning is to attract positive sample pairs and repulse negative sample pairs in the embedding space [130]. Since there are no labels available during the representation learning stage, with the help of data augmentation techniques, positive pairs usually consist of different views of the anchor sample, while the negative pairs consist of the anchor sample and samples randomly selected from the mini-batch [12]. This can be usually treated as a dictionary look-up problem.

Recent breakthroughs in contrastive learning, such as the MoCo framework [13, 38], and SimCLR [12], leading to state-of-the-art performance in self-supervised learning. These approaches belong to instance-discrimination-based contrastive learning, which utilizes instance-instance contrast and InfoNCE [78]. The performance of contrastive learning via InfoNCE depends heavily on a negative sampling strategy, which usually requires a large number of negatives to learn good representations from the backbone encoder. A significant focus of these methods is to perform hard positive sampling while improving the efficiency of negative sampling. SimCLR follows the end-to-end training framework, where the negative keys are from the same batch and are updated end-to-end by back-propagation [13]. It addresses the negative sampling problem by applying

a large batch size, but this is memory-intensive in practice and requires specialized optimizers to stabilize the training at scale [82]. MoCo abandons the end-to-end structure and instead introduces a memory bank to dynamically maintain a negative queue during training [38], which does not require a large training batch size, but increases the number of negatives substantially. It designs momentum contrastive learning by introducing another encoder, the momentum encoder, which uses a momentum update strategy instead of gradient backpropagation [38, 13]. The queue applies an enqueue and dequeue operation to save the recently encoded batches as negative samples, which can improve the representation consistency between the current and earlier keys. This momentum encoder and memory bank strategy significantly improves the negative sampling efficiency.

Recently, contrastive learning has attracted increasing attention in the natural sciences and has shown remarkable results on a variety of scientific problems, including molecular representation [124, 50], prediction of density-of-states of 2D photonic crystals [66], similarity search for sky surveys [102], single-particle diffraction images [142]. In particular, Ref. [66] shows that self-supervised contrast learning can greatly reduce the number of labels required to train a network, which is a tedious and time-consuming operation. These successful applications in different scientific fields demonstrate the effectiveness and versatility of contrastive learning.

In this thesis, we introduced a momentum contrastive network in our RRMoco-Net in Chapter 7. Unlike the original MoCo framework, the key encoder in our SpecMoco-Net branch, which we will introduce in Chapter 7, is updated not only by minimizing the contrastive loss function but also the loss items in SpecRR-Net.

Part I

Spectra Classification with feature engineering

3

Spectral Clustering method

Spectra classification based on traditional machine learning methods with feature engineering relies on domain expertise, which can impede the automation of data analysis. However, if properly designed, these methods may enhance the efficiency, interpretability, and performance of specific tasks. For example, feature engineering techniques like PCA can effectively reduce the curse of dimensionality.

This thesis point is related to applying machine learning methods with feature engineering to spectra classification. Two approaches are proposed in this thesis point, a clustering method incorporating PCA preprocessing (Chapter 3), and a neural network-based model incorporating a binned-weighting technique (Chapter 4). The two feature engineering techniques, PCA preprocessing, and the derivation of confidence weighting factors, encode a priori knowledge into the classification model, which helps in capturing relevant patterns in the data.

This thesis begins by analyzing the spectra data collected from sample $(Mg_{0.2}Fe_{0.8})O$ magnesiowüstite, as described in Section 2.1 and 3.1. To evaluate the experiment status, the measured spectra need to be clustered or classified so that each class is assigned to a different state of the system under investigation. In this way, the actual status of the experiment can be fed back instantly according to the classification result, and the follow-up experiment can be better guided. The dataset used for this analysis is unlabeled, and therefore, clustering methods offer a suitable approach to explore and analyze the unlabeled spectra data effectively. The two major spectral changes that we aim to capture in this study are:

- the change of intensity distribution (e.g. drop or appearance) of peaks at certain locations, or
- the shift of those in the spectrum.

In this chapter, we present an unsupervised Spectral Clustering-based method for the application of diffraction spectra data classification [106]. The description of the spectral dataset is given in Section 3.1. The PCA method is used here as data preprocessing for dimensionality reduction and speeding up the calculation, which will be

described in Section 3.3. In particular, here we will also analyze the correlations between the original variables and the principal components (PCs) from PCA, and their contribution to the PCs. Then, in Section 3.4 a combination of PCA and Spectral Clustering approach is presented to cluster the spectra and detect phase transition [106]. Finally, the implementation details and the result of the model are presented in Section 3.5. In addition, in this section, we introduce classification confidence, a proposed metric to evaluate the clustering results.

3.1 Dataset Description

The spectral data used in this work was collected from $(Mg_{0.2}Fe_{0.8})O$ magnesiowüstite sample, as shown in Fig. 3.1. The sample was compressed from 1 to above 100 GPa over a period of 50 s while maintaining reasonable diffraction image quality in consecutive 100ms exposures. In high-pressure X-ray diffraction experiments on high-density materials, changes in pressure will cause changes in spectral peaks (vanishing, shifting, broadening, or splitting) that correspond to the modification of the crystal lattice (e.g. indicating phase changes). Our diffraction spectra dataset is very representative. The dataset consists of 349 samples with each of 4023 features and it is publicly available (https://zenodo.org/record/4424866). To show more clearly how the diffraction changes while the pressure on the sample is changing, we show one for every 10 diffractograms, as displayed in Figure 3.1. Hence, the time interval between adjacent diffractograms in Figure 3.1 is 1 s. As it can be clearly seen from this figure, the intensity of the spectral peaks change (increase, decrease, or even vanish) at certain locations, and peaks also shift in their 2θ-angle position, split, or start to broaden. The changes correspond to the modification of the crystal lattice (e.g. indicating phase changes).

Although it is not easy to annotate the phase transition boundaries explicitly, some representative spectra measured at both the initial and final stages belong to two different classes and can be easily annotated. To visually depict the distinctive characteristics of different phases, several representative spectral curves are annotated. Specifically, the 16 spectral curves marked in red belong to class label 0 (before phase transition) and the 12 ones marked in blue belong to class label 1 (after phase transition). It can be clearly seen from this figure, the amplitude of spectral peaks changes (increases, decreases, vanishes) at certain locations, and peaks also shift in 2θ-angle dimension, or split, or start to broaden [106]. These changes correspond to modifications of the crystal lattice (e.g. indicating phase changes).

During the experiment, we should be able to track these changes and determine the actual state of the system in near real time. Scientifically, the most relevant question is whether the phase transition in the sample has occurred. Since there is no ground truth information, in order to determine this, we should provide a judgment with minimum ambiguity at each point during the experiment.

3.2 Spectra Classification based on Spectral Clustering Method

As an unsupervised learning algorithm, clustering is one of the common nonparametric ML techniques and is widely used for exploratory data analysis [120]. Among them,

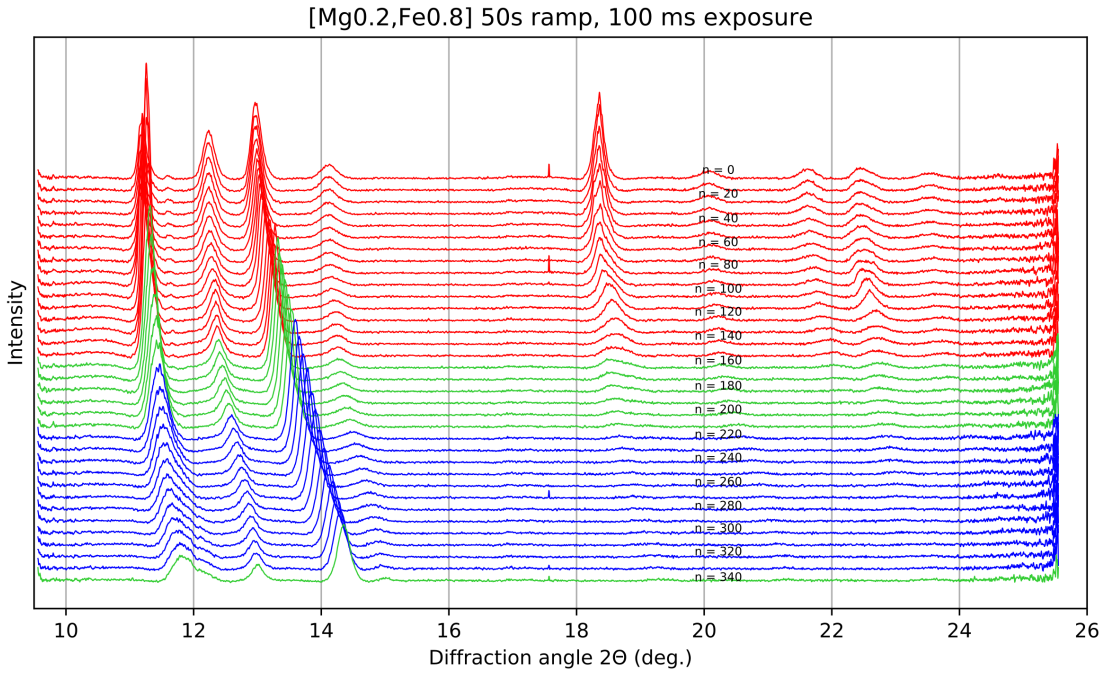


Figure 3.1: Diffraction spectra (one for every 10 diffractograms) collected during the experiment after baseline subtraction. Among them, the 16 marked in red belong to class label 0 and the 12 marked in blue belong to class label 1.

spectral clustering is a clustering method that does not make assumptions about the global structure of the data [51]. It can solve very general problems like intertwined spirals and can be implemented efficiently even for large data sets [120].

3.3 PCA for Dataset Preprocessing

As described in Section 2.1, in X-ray diffraction high-pressure experiments, pressure changes on the sample will be reflected by changes in spectral peaks. Immediate feedback on the actual status (e.g. time-resolved status of the sample) would be essential to quickly judge how to proceed with the experiment.

In spectroscopy experiments, it is very common that the number of input variables (features) is greater than the number of training samples, which will more easily lead to the problem of overfitting. Our data has the same characteristic. In order to facilitate the ML/DL training process, the PCA algorithm is applied to data dimensionality reduction while speeding up the training process. PCA uses an orthogonal transformation to convert data (of possibly correlated variables) into a set of new uncorrelated variables called principal components that successively maximize variance [52]. It is proved to be a simple and effective dimensionality reduction method for spectra data [138, 76].

3.3.1 Data Centering

Before applying the PCA algorithm, the dataset features should be centered by removing the mean. Centering is performed independently on each feature by computing the relevant statistics on the samples [61], as shown in Fig. 3.2.

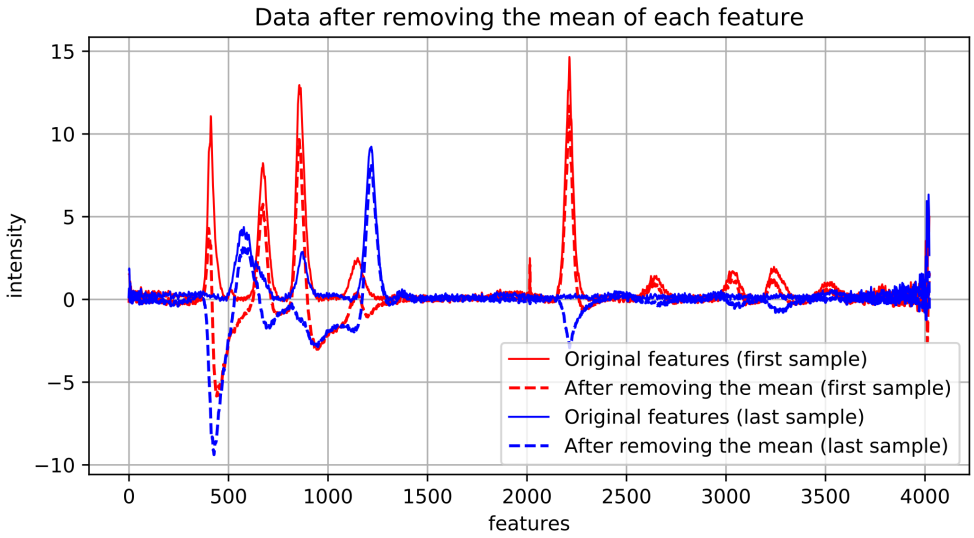
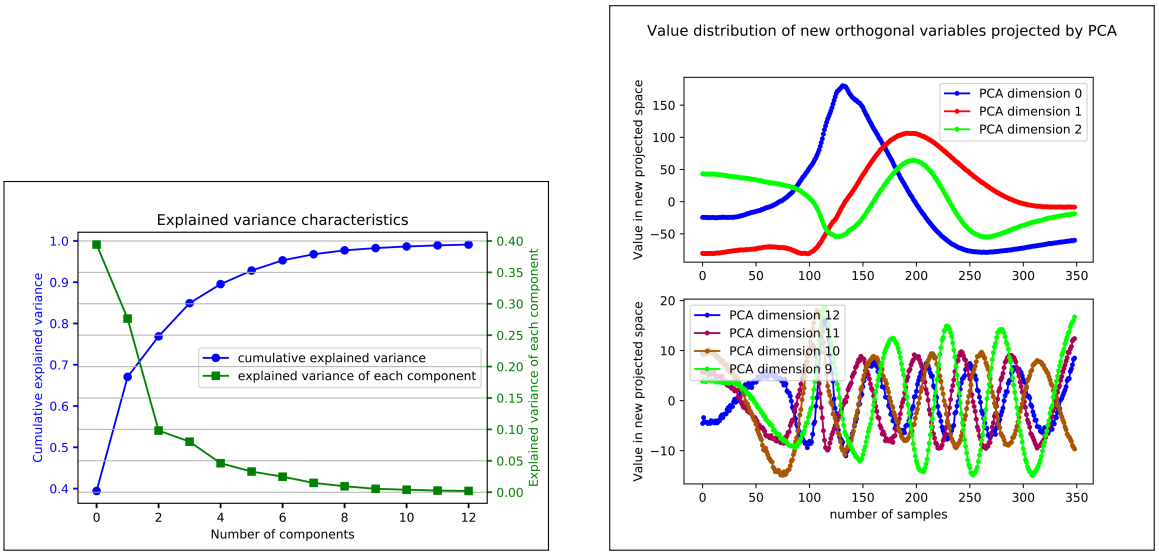


Figure 3.2: Centering the dataset (take the first and last samples are used as an example).

3.3.2 PCA Preprocessing

In the PCA method, the number of principal components (PCs) required to describe the data can be determined by looking at the cumulative explained variance ratio as a function of the number of PCs [118]. The cumulative explained variance of PCA is shown in Fig. 3.3(a), the first 2 PCs explain more than 60% of the variance. Some of the new projected orthogonal variables' (PCs) values distribution can be seen in Fig. 3.3(b). It can be clearly seen that the first PC explains the most variance in the data with each subsequent component explaining less.



(a) (b)

Figure 3.3: (a) Cumulative explained variance. (b)Value distribution of new orthogonal variables projected by PCA.

When converted back to the original space, you can see the information retained or lost by the PCA algorithm more vividly, the comparison between the inverse transfor-

mation of PCA with different explained variance and the original spectra data is shown in Fig. 3.4. We can get that the first few PCs can describe the basic distribution of the data, with other PCs providing more details. In order to retain as many features as possible, we choose 13 components that can explain 99% of the variance. Then our new projected data consist of 349 samples with each of 13 features.

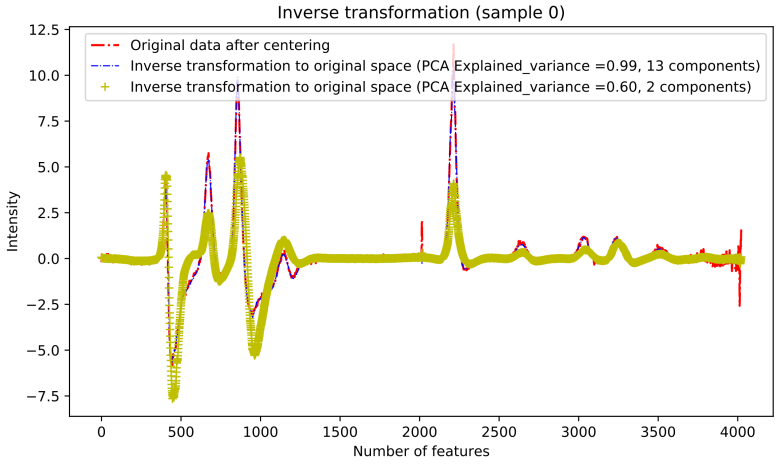


Figure 3.4: Inverse transformation with different explained variances of sample 0.

3.3.3 Contributions of Variables to PCs.

In PCA, the correlation between components and variables is called loadings, it is the element of the eigenvectors and estimates the information they share [52].

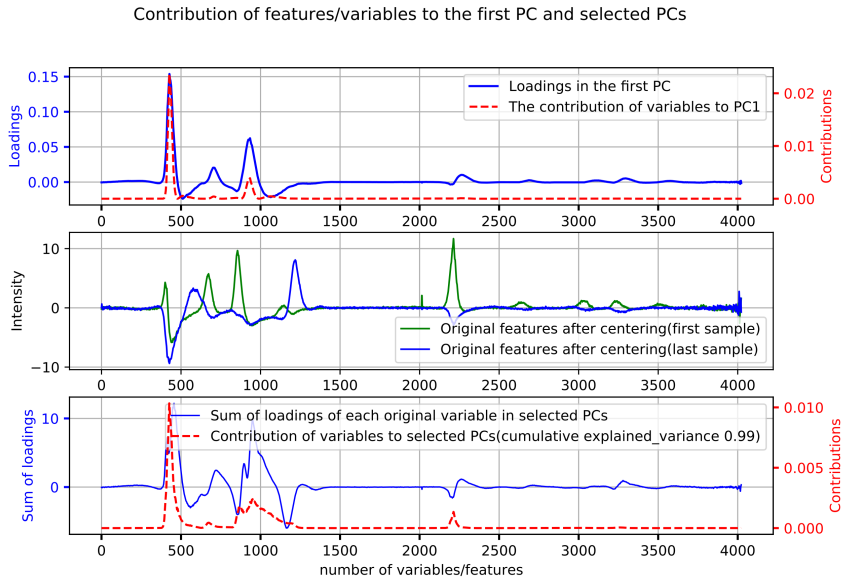


Figure 3.5: Inverse transformation with different explained variances of sample 0.

The loadings (marked with blue line) and contributions (marked with red line) of variables/features in accounting for the variability to the first PC are shown in Fig. 3.5 (top row). The sum of loadings and the contributions of each original variable/feature to selected PCs are shown in Fig. 3.5 (bottom row). It shows that the more obvious the features/variables, the greater the contribution to the selected PCs.

3.4 Spectral Clustering Method

Spectral Clustering uses information from the eigenvalues (spectrum) of special matrices (i.e., Affinity Matrix, Degree Matrix, and Laplacian Matrix) derived from the graph or the data set [69] and makes no assumptions about the form of the clusters. The method shows great clustering performance for data with non-convex boundaries. It is usually used when the dataset has a non-flat geometry and needs to be divided into a small number of clusters with even cluster size [10], which is well suitable for our case.

In this method, PCA preprocessing with the same parameters is used for dimensionality reduction, immediately followed by the standard spectral clustering algorithm. The clustering metrics used in the spectral clustering algorithm is graph distance, a graph of nearest neighbors [126], which is constructed to perform a low-dimension embedding of the affinity matrix between samples. And the K-Means label assignment strategy is applied in the approach, which is a popular choice [120].

3.5 Results and Discussions

3.5.1 Implementation Details

The model is implemented on the Jupyter Notebook platform using Scikit-learn libraries. The analysis scripts as Jupyter notebooks are publicly available on Github[1].

3.5.2 Performance Metrics

In this study, we aim to find the phase transition point, which also means classifying the spectra into 2 phases or classes during the experiment. As there is no ground-truth phase transition information, we are interested in whether there is a clear boundary or an ambiguity zone during the experiment when the classification jumps inconsistently between the phases. Hence, our performance metric shall how small this ambiguous zone is. From the physics point of view, a proper interpretation would require the phases and the ambiguity zone to be linked to specific pressure ranges. Unfortunately, the available data is not annotated and does not contain such information, so an objective measure for the classification quality is not available. This also highlights the importance of storing all relevant metadata and following FAIR principles in data management. Instead, we define the *classification confidence* as described here. To explain what an ambiguity zone is, an illustrative example is shown in Fig. 3.6. Suppose we have 24 samples, corresponding to class 0 or class 1, and their classification results are shown in Fig. 3.6, the zone marked with red for the class label jump is an ambiguity zone.

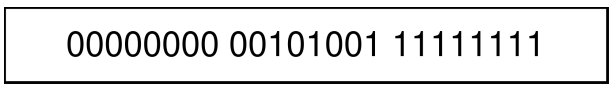


Figure 3.6: An illustrative example of an ambiguity zone.

Let N_f represent the number of spectral curves in the ambiguous region, N_t represent the number of test spectral curves, then the *classification confidence* can be defined as

[1] https://github.com/sunyue-xfel/Machine-Learning-applied-for-spectra-classification

$$P_{conf} = 1 - \frac{N_f}{N_t} \tag{3.1}$$

The clear boundary between these two types of spectra yields 100% confidence. In a particular case, if a phase change has not been found and all spectra are assigned to the same class without any boundary between the phases, it is considered that all spectra are in the ambiguous region and the classification confidence is 0.

3.5.3 Results Analysis

For the spectral clustering method, we test the classification confidence with different explained variance value which ranges from 55% to 99.99% (the corresponding number of PCs range from 2 to 301), the result shows that this method achieves consistent high-precision classification results (100% classification confidence), at the same time, the classification boundary is very stable and fluctuates only in a small range, as can be seen from Fig. 3.7. From another aspect, it also shows that the PCA algorithm can obtain the main feature information of the original data. In addition, the computation time of the spectral clustering algorithm was 0.069 s with 13 PCs pre-processed by PCA as input.

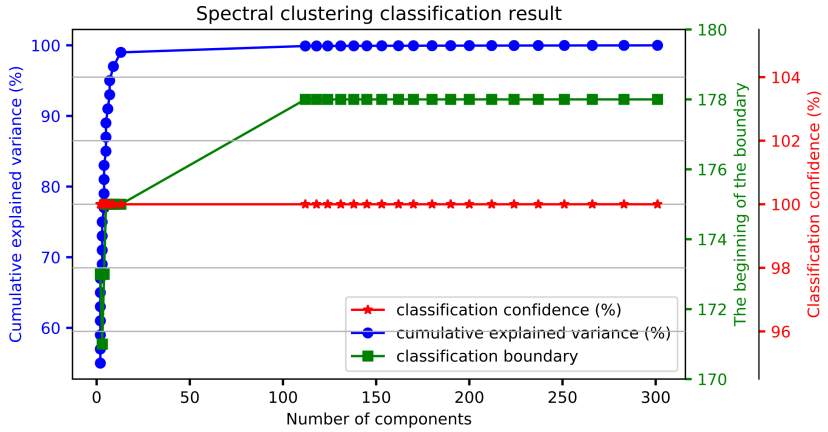


Figure 3.7: Classification confidence with different explained variance value and PCs.

3.5.4 Threats to Validity

Although we obtained nice results on ML-based spectral classification, there are still some threats to validity. It can be clearly seen that in our original spectral data set, the number of training samples is limited, and the number of features is much larger than the number of samples, which may cause over-fitting problems. In this case, the PCA method is used for dimensionality reduction. However, PCA assumes that the data follows a linear and Gaussian distribution. If the data violate these assumptions, PCA may not be a good representation of the true experimental data.

Furthermore, due to the absence of ground truth information regarding phase transitions, relying solely on the performance metric of classification confidence has limitations in accurately reflecting the actual phase transition. When the data is incorrectly labeled or lacks essential explanatory information, the effectiveness and validity of the model can be compromised.

3.6 Conclusion

In this work, we provide an unsupervised machine-learning method for time series spectra data classification. In this method, the PCA method is used as data preprocessing to reduce the dimensionality and speed up the subsequent clustering process. After that, the data were clustered by applying spectral clustering methods. Remarkably, it achieves a classification confidence level of 100% while consuming only 0.069 seconds of computation time. The findings demonstrate that the unsupervised spectral clustering approach is highly suitable for analyzing diffraction spectra data with non-flat geometries. Furthermore, to ensure reproducibility, Jupyter notebooks containing the data analysis scripts are provided alongside this work.

Contributions The author of this PhD thesis is responsible for the following contributions presented in this chapter:

- Proposal of the unsupervised machine learning method for the classification of unlabeled spectral data based on spectral clustering and PCA preprocessing.
- Studying the principle components (PCs) of the PCA method and analyzing the contribution of the original variables to the PCs.
- Implement, analyze, evaluate, and present the results. Provide the analysis script on Jupyter Notebooks for reproducibility.

4

Supervised Binned MLP model for spectral classification

Although the combination of PCA and Spectral Clustering method presented in the previous chapter (Chapter 3) already achieved high performance, there are still several limitations to be addressed. First, the model relied on PCA preprocessing rather than learning representations from raw data, which can be time-consuming for high-dimensional spectral data, thereby limiting its generality and the capacity to learn complex representations. More importantly, the tuning of parameters, such as the number of clusters may differ from dataset to dataset. The reliance on predefined assumptions about the structure and characteristics of the data requires human intervention.

With recent developments in ML, data-driven deep learning methods have turned out to be very good at discovering patterns and structures in high-dimensional data [62] during the training process. The trained models can then supply computational inexpensive decisions when applied to new data. Hence, this technique opens up new ways for data-driven analysis in diffraction and spectroscopy by offering the possibility of recognizing specific features on-the-fly during data collection. It enables fast feedback and the implementation of near real-time downstream data analysis, e.g. providing immediate information on the phase of the experiment based on the classification of the currently measured spectra.

In this chapter, we presented an example of classifying experimental spectra using Neural Network-based ML. We formulate the spectral classification problem as a two-dimensional (2D) spatial segmentation problem. The rest of this chapter is organized as follows: First, we present the proposed model (Section 4.1), i.e., a two-layer neural network combined with a classification weighting technique for spectral classification [104]. In this model, to suppress the misclassification of indistinguishable features in overlapping regions, we divide the spectrum into bins and apply an independent neural network to each bin, which, combined with the weighting factors derived from the local separability of each bin, finally allows to specify a pointwise classification result. The performance of the model is evaluated by the classification confidence metric described in Section 3.5.2. In addition, we investigate the performance of the model

using different numbers of bins and determine the appropriate number of bins [104]. Next, the classification results are presented and discussed in Section 4.2. Finally, the conclusion and contributions of this chapter are outlined in Section 4.3.

4.1 ML Model Combined with A Binned-Weighting Technique for Spectra Classification

In this work, we give an example of using experimental data to determine if our multi-crystal powder system has already passed a phase transition during a pressure ramping experiment protocol or not.

Since pressure was not recorded directly with the raw dataset, a training set containing elements from both phases could not be automatically identified, but rather the representative curves had to be selected manually. For this purpose, four representative spectral curves (2 from both classes) are included in the training set (as shown in Figure 4.2). To learn the characteristic differences between the 2 classes, Neural Network-based ML is used. To increase the robustness of the neural network model and its tolerance to small Gaussian measurement errors, 10 simulated spectral curves for each original spectrum were also included in the training set by adding some random noise, which is small enough to enable finding the characteristic features in the spectra. The random noise is generated using the Mersenne Twister[70] as the core generator. The spectral data used for the training can be seen in Figure 4.3.

As described below, with the introduction of a new metric *separability* learned from the final accuracy of the trained neural network model in each bin, we assign *believability weighting* factors to the bins to describe how much the classification prediction of the ML model in the given bin can be trusted. A weighted average of the individual predictions from all bins produces then the final classification prediction for any new spectra.

Neural Network Structure. We choose neural networks due to their ability to learn complex mappings between input and target spaces which makes them fit perfectly to our task. Neural network models have gained increased popularity recently, since they can express complex function mappings using inputs with very little or no feature engineering [31]. In this study, we explored a two-layer neural network architecture with 150 hidden units. Each layer accepts the output of the previous layer as its input, and returns a transformation function as its output [31]. We transform our problem of finding and distinguishing features in a set of 1D curves where the input is provided by the sequence of the spectral intensity values, into a 2D segmentation problem where we input every point in the spectra with their 2 coordinates (scattering angle, and azimuthally integrated intensity as on Figure 2.2). Hence, our input layer has 2 input neurons (corresponding to the coordinates of the given points in the 2D space) and the hidden layer has 150 hidden neurons (as an abstract feature map) followed by a non-linear ReLU function [75]. Their outputs are fed into the second layer (which is the output layer) with 2 output neurons (corresponding to the number of classes being measured before or after the phase transition), followed by a softmax activation function [5], as in Figure 4.1. The ReLU activation units are selected here to speed up the model learning and prevent gradient vanishing/exploding [131]. Such a neural network described above can be applied to the general 2D space segmentation

problem as discussed in Ref. [98]. Here, we investigate the application of the same NN architecture for the special case of real experimental spectra classification.

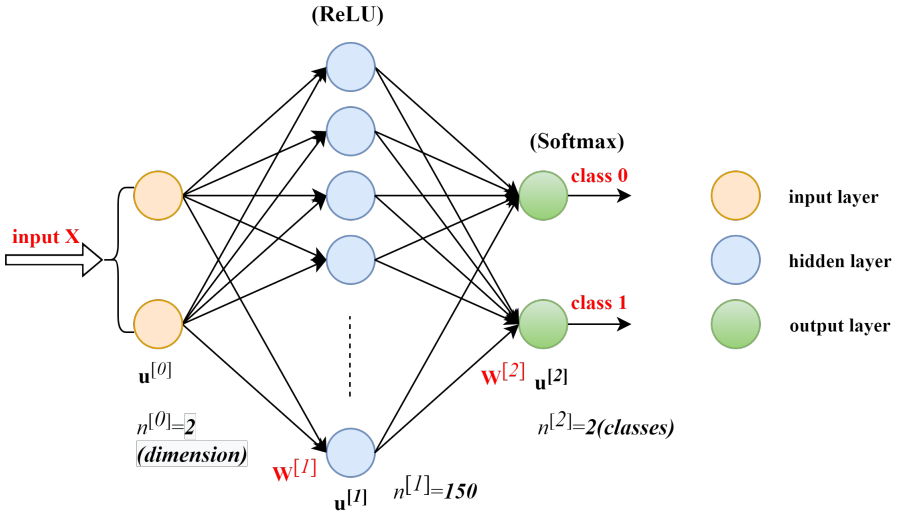
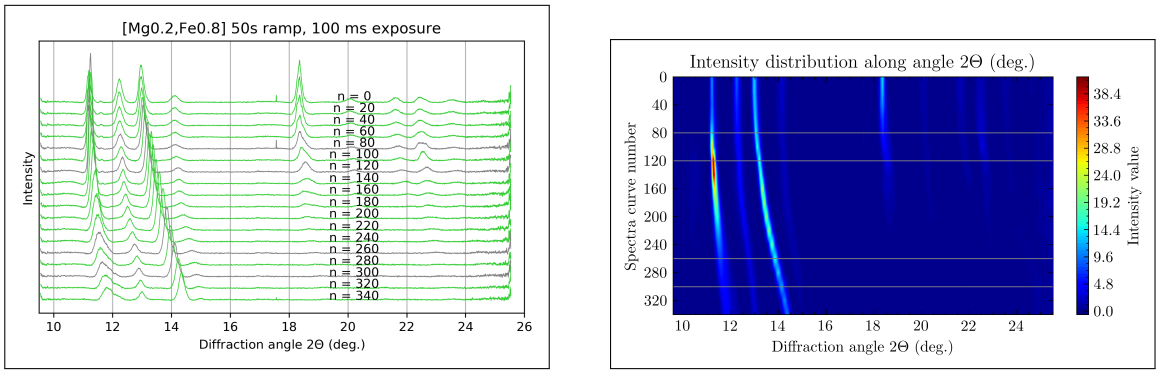


Figure 4.1: Two-layered neural network structure used for spectra classification.

Model Training and believability weighting factors. As Figure 4.3 shows, the spectra in the two training classes partially overlap. Hence, classification (or segmentation) at the overlapping parts is difficult, meaningless, and of low confidence. The more our spectra overlap, the lower the overall accuracy, independent of how precisely our model performs the segmentation at locations with well-separable features. Because of this, we divide the spectrum into several intervals, called bins, so we can learn the local separability of the spectra in each bin separately, which is provided by the classification accuracy at the end of the training. The principle of selecting the number of bins is to ensure that at least one spectral bin has a high separability indicated by the classification accuracy. Figure 4.3 shows the binning for the case of dividing the spectrum into 18 bins.



(a) (b)

Figure 4.2: (a) Spectral data (one for every 20 diffractograms) collected during the experiment after baseline subtraction, and (b) Contour maps of the Intensity distribution of each diffractogram along axis. Note that 4 spectra are grayed out which were randomly picked as the basis for the ML training set for representing the two experimental states (before and after phase transition).

After splitting the spectra, two-layered neural networks with the identical structure described above are used to perform classification training in each bin separately. The obtained classification accuracy as a separability indicator is used for automatically calculating a believability weighting factor for each bin (equation (4.1)).

The neural network is trained by backpropagation using gradient descent, with the Adam update scheme [56]. The model was implemented on a Jupyter Notebook using the Pytorch library. The analysis script is Publicly available on Github[1]. We use the classification cross-entropy as the loss function, ReLU and Softmax function as the activation function for our classification task in each bin. The statistical model is obtained by minimizing the loss function on the training data set by checking if a point is properly classified. 700 epochs are used for iteration, in which the point-wise probabilities $\hat{y}_k^{(i)}$ and then the class id $\hat{C}^{(i)} \in 0, 1, \ldots, C-1$ is predicted for each point based on the output of the softmax. The final classification (or space segmentation) model in each bin is shown in Figure 4.3 by the two background colors. As the figure inlet also highlights, the segmentation is determined by the point distribution of the spectra in each bin, and certain data will inevitably be misclassified.

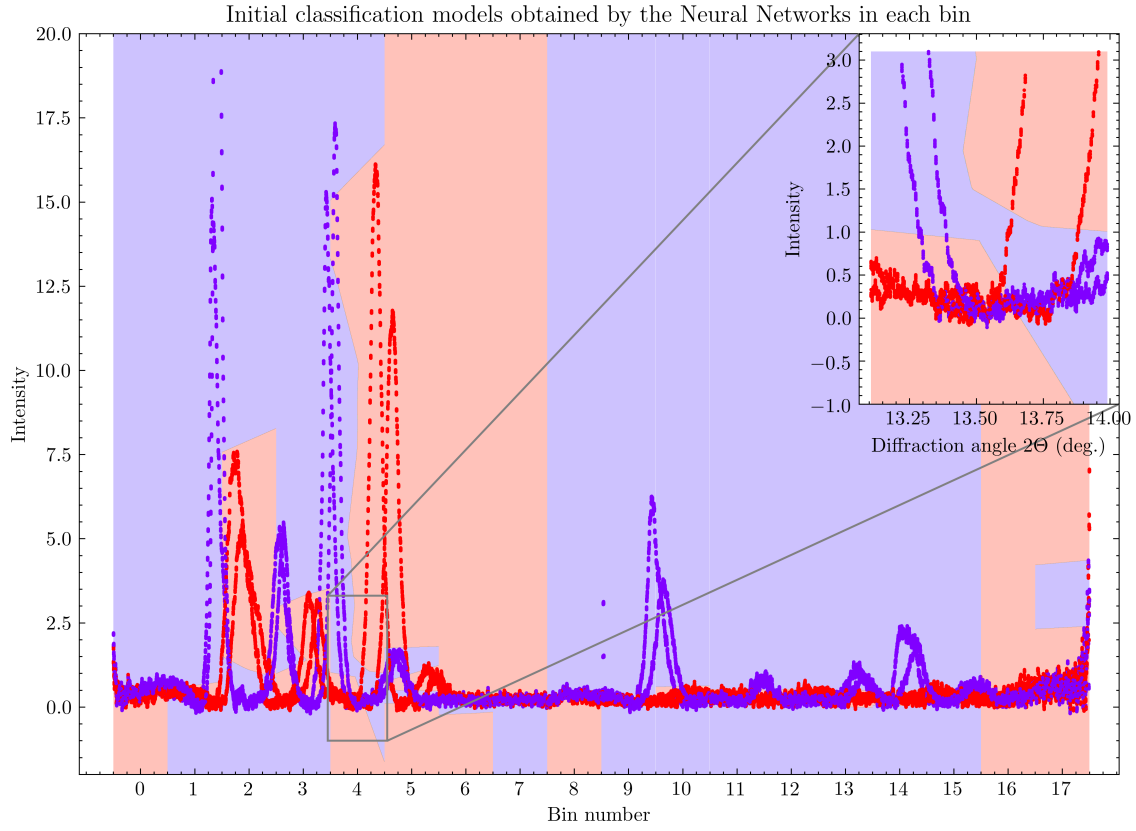


Figure 4.3: The initial classification model for each bin obtained by the respective trained two-layered network models.

In this process, the training accuracy of classifying each point in each bin is obtained. The corresponding statistics are shown in Figure 4.4. The optimized accuracy as a quality index for our neural network model is used as a measure for separability in each bin. High-quality classification is obtained in bins with high separability, whereas point-wise classification in bins with low separability is misleading.

[1]https://github.com/European-XFEL-examples/panosc-ml-spectra-classification

In this work, we use $A_b, \{b = 1, \dots B\}$ to represent the training classification accuracy and so the spectral separability in bin b, where B represents the number of bins. The values of A_b can vary between 0 and 1. Since 50% classification accuracy corresponds to maximum ambiguity, we consider that an accuracy below 55% corresponds to a confidence weighting factor of 0, and an accuracy of 100% corresponds to 1. To increase the impact of the training accuracy, a square operator is also introduced. Based on this, a believability weighting factor w_b is calculated as

$$w_b = max(0, 1.4337{A_b}^2 - 0.4337) \tag{4.1}$$

The result shows (see Figure 4.4) that the more obvious the feature difference in bins, the higher the classification accuracy (separability), thus larger weighting factors. At the same time, it can also be seen that the weighting factors corresponding to non-separable regions with accuracy less than 55% are 0.

The final classification label for each spectral curve is calculated by the weighted sum of the point-wise classification results assigned by our neural networks in each bin:

$$L_{curve} = \underset{k \in \{0,1,\dots,C-1\}}{argmax} \sum_{b=1}^{B} w_b \sum_{i=1}^{N_B} \hat{C}_k^{(i)} \tag{4.2}$$

where $\hat{C}_k^{(i)}$ equals to 1 if the i-th feature or observation $\hat{C}^{(i)}$ is labeled as k, otherwise it equals to 0. $N_B = \frac{N}{B}$ represents the number of data points in each bin, where N represents the number of data points in each spectral curve. Based on the classification prediction L_{curve} for any measured spectrum, immediate information on the phase of the system, and so on state of the experiment can be provided.

Performance metrics Next to classifying the spectra into 2 phases, our aim also includes finding a transition point during an experiment. It is difficult and inaccurate to manually determine such a boundary. Instead, we are interested in the robustness of the ML-based classification and want to minimize any ambiguity zone during the experiment when the classification prediction jumps inconsistently between the two phases. Hence, here we apply the classification confidence proposed in Section 3.5.2to measure how small such a zone of ambiguity is.

4.2 Classification Results and Discussion

The full dataset used for the evaluation of the neural network learning model consists of 349 spectral curves, each of which has $N = 4023$ data points. Part of the dataset (one for 20 diffractograms) is shown in Figure 4.2 as described above.

According to the performance metric equation (4.2), curve-wise classification confidence for the dataset can be calculated. Applying 18 bins, the ambiguous zone is small and the final curve-wise classification confidence is 98.854%. For different number of bins applied, Figure 4.5 shows the respective ambiguous zones, and Figure 10 plots the calculated values of classification confidence. Figure 4.5 shows the ambiguous zones around the phase change by displaying the spectra with colors according to the classification prediction results and applying a continuous movement of the intensity baselines for better visibility.

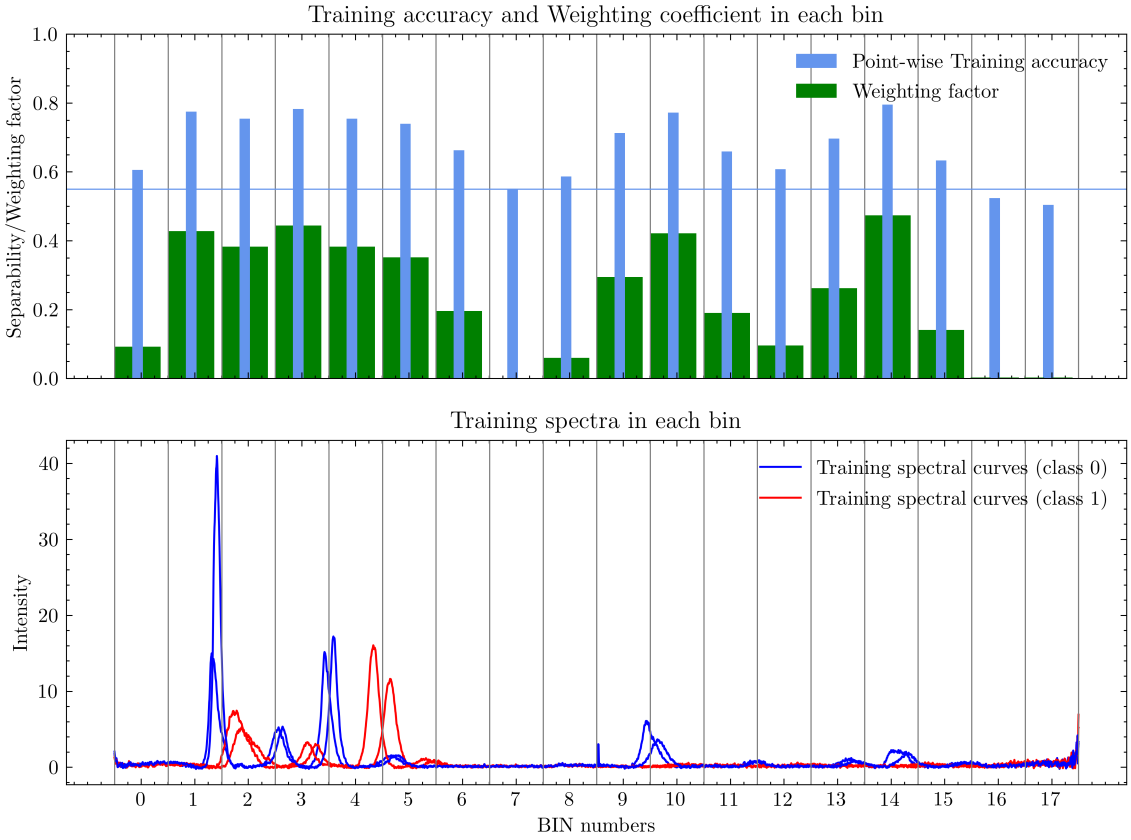


Figure 4.4: The distribution of point-wise training accuracy (separability), and the believability weighting factor for each bin.

It can be clearly seen that the more bins applied, the higher the classification accuracy, thus the smaller the ambiguous zone. This is achieved by weighting the individual classification predictions calculated at the different bins, and so making the bins containing well-separable spectral features dominant and weakening the impact of misclassifications in the bins with low spectral separability and corresponding low believability weighting factor. The key to achieving a high classification confidence is finding bins with high separability index. Therefore, the optimal number of bins should be determined by increasing the number of bins until it is ensured that bins with high separability have been found.

The relationship between the classification confidence and the number of bins applied is shown in Figure 4.6. The overall trend is that as the number of bins increases, the ambiguous region becomes smaller and smaller. The presence of bins with high separability and corresponding high believability weighting factors immediately shows that the characteristic features of the different classes have been captured and high overall performance is expectable.

Although we obtained nice results on ML-based spectral classification, in our ML algorithm, the weighting factors are calculated based on the training accuracy in each bin. As a next step, the Neural Network can be extended to form an end-to-end ML structure that can automatically learn these weighting factors and output directly the final classification label. Another development possibility is to handle the spectral data as a set of one-dimensional time series and instead of a space-segmentation approach presented here, the classification problem could check for intra-spectrum characteristics, too. In this setting, we can apply different supervised deep learning

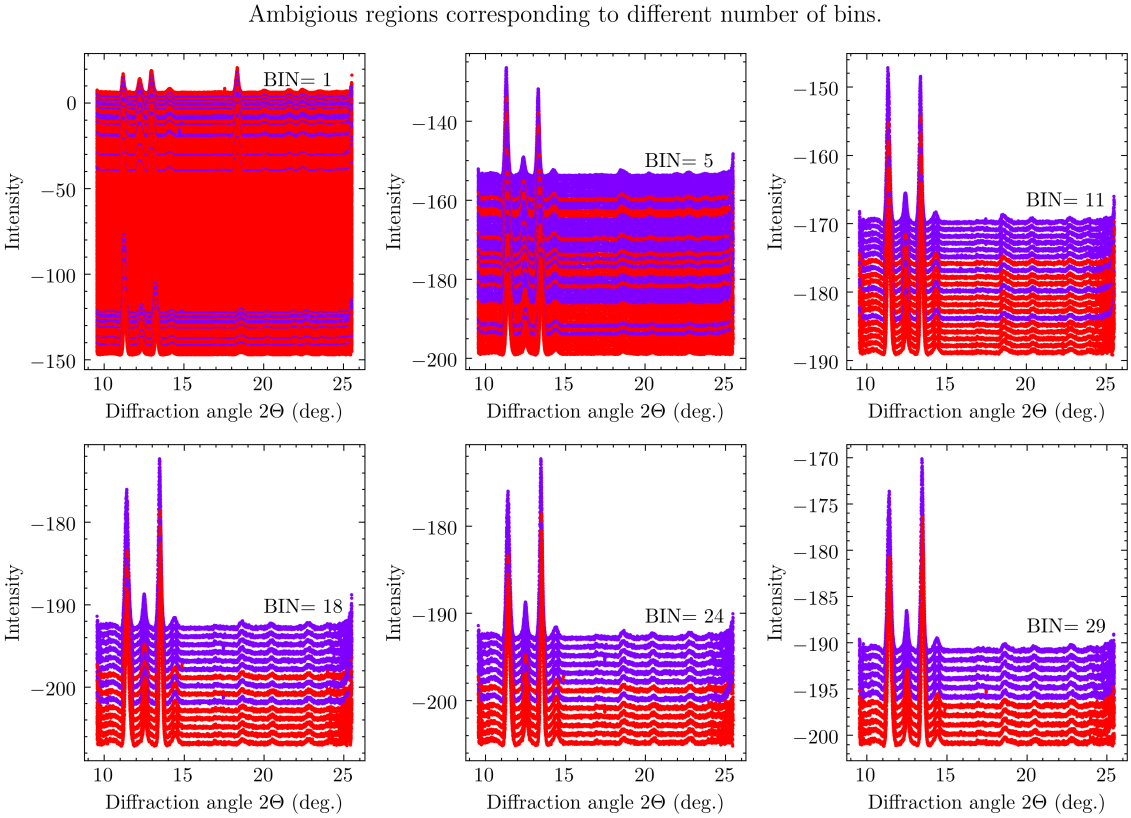


Figure 4.5: Final classification result of test spectral curves near the boundary (using 1, 5, 11, 18, 24, 29 bins). The spectrum is pulled evenly by moving the Intensity baseline for better display.

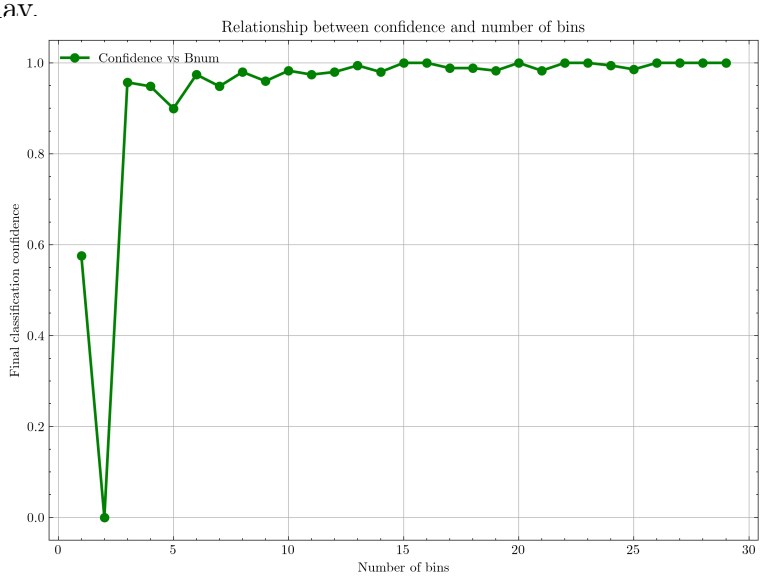


Figure 4.6: Relationship between final classification accuracy and the number of bins.

neural network architectures for the spectra classification, such as the convolutional neural network (CNN), Residual Networks (ResNets)[40], LSTM-based architecture [92], attention-based architectures[119].

4.3 Summary

In this study, we presented an example of classifying experimental spectra using Neural Network-based ML. The model aims to classify measured spectra into two classes, either before a phase transition or after it. The classification is based on spectral features specific to an experimental phase. Hence, changes in spectral peaks (increasing, decreasing, vanishing, shifting) had to be found. Since the spectra of the 2 classes are overlapping in some regions, their separability is low there. We have presented that binning the data, and down-weight the classification predictions from those bins where the classification accuracy is low results in a reliable final classification. In each bin, a two-layer neural network was trained for the classification and also for determining the separability of the classes within the bin. Hence, a weighted sum of the predicted class labels of each data point from a whole spectrum leads us to the final classification of the curve. This can minimize or even eliminate the effects of misclassifications of the data points in overlapping regions.

The result shows that our spectral classification model is robust to random noise and can identify peak intensity changes or peak shifts. The classifier can provide us with immediate feedback on the spectral class and so on the actual phase of the experiment.

Contributions. The author of this PhD thesis is responsible for the following contributions presented in this chapter:

- Design of the neural network-based ML model with a binned-weighting technique for spectra classification to minimize misclassifications of indistinguishable features in overlapping regions of different spectra.
- Design of a methodology for calculating classification weighting factors.
- Performing and evaluating the case study. Investigation of the performance of the model under different numbers of bins.
- Implement the proposed model and publish analysis scripts to facilitate the analysis of spectra and exploration of the proposed solution.

Part II

End-to-End Deep Supervised Learning Methods for Spectra Classification

5

End-to-End Machine Learning Methods for Spectra Classification and its comparison

While the previous chapter (Chapter 4) has demonstrated promising results for spectral classification, there are still some limitations to address. One of these limitations is the reliance on prior models to calculate the believability weighting factor, which hampers the automation of the entire process and restricts the generality and the ability to learn complex representations. Furthermore, there is a crucial consideration when applying a neural network-based model to spectral classification. The nature of the problem suggests that it is more naturally and generally treated as a one-dimensional spectral time series classification rather than a two-dimensional spatial segmentation problem. This perspective favors simplifying the problem, acknowledging the sequential nature of spectral data and its temporal characteristics.

To this end, in this thesis point, we aim to derive an end-to-end statistical model for the application of 1D spectra classification in order to determine the time-resolved status of the evaluated sample. The two major spectral changes that these end-to-end deep learning models aim to capture are either the changes of intensity distribution (e.g., drop or appearance) of peaks at certain locations, or the shift of those on the spectrum. We deal with the spectra classification problem from two different perspectives, one is a general two-dimensional (2D) space segmentation problem, and the other is a common 1D time series classification problem.

We focus on the two proposed classification models [105] under these two settings, namely the end-to-end binned FCNN with automatically capturing weighting factors model (Section 5.1.1), which is extended from the binned-weighting ML model proposed in Chapter 4. Another one is the convolutional Spatial-Channel-Temporal (SCT) attention model (5.1.2), which introduces self-attention mechanism performed across spatial, channel, and temporal dimensions to suppress indistinguishable features and selectively focus on obvious features with high separability. And under the setting of 1D time series classification, several other end-to-end structures based on Fully Connected Neural Network (FCNN), Convolutional Neural Network (CNN), ResNets, Long Short-Term Memory (LSTM), and Transformer are explored [105]. In this way, we pro-

vide a standard baseline to exploit several current state-of-the-art deep neural networks for end-to-end 1D spectral time series classification with a few labeled samples. We show how DL architectures can be designed and trained efficiently to learn hidden discriminative features from time-resolved spectra in an end-to-end manner. Finally, we evaluated and compared the performance of these classification models based on an example spectra dataset from multiple perspectives, and further performed the feature importance analysis to explore their interpretability.

The rest of this section is organized as follows: Section 5.1 and Section 5.2 detail the proposed or applied classification models individually. Experiment studies and discussion of results are shown in Section 5.3. In addition, an open-source interactive software program [109] on these classification models is shown in Section 5.4. Finally, the conclusion and main contributions are given in Section 5.5.

5.1 Data-driven End-to-End Neural Network Methods

In this section, we describe in detail the DL classification models we proposed or applied in different problem settings. The reasons behind our design or choice of these end-to-end methods are also explained. The first model is proposed in the setting of 2D space segmentation. The remaining models are proposed under the setting of 1D time series classification, based on FCNN, CNN, Resnet, LSTM, Transformer, and a hybrid model of CNN and self-attention architecture respectively. In the last hybrid convolution-attention model, self-attention is calculated across spatial, channel, and temporal dimensions.

5.1.1 End-to-end Binned FCNN with Automatically Capturing Weighting Factors Model

In this model, the spectra classification task is regarded as a general two-dimensional (2D) space segmentation problem. The 2D mentioned here means that instead of inputting a single variable, the values of the two coordinates x, and y of each feature point are input into the neural network model, and the model is taught which class label this point belongs to. The model aims to automatically prominent features with high separability and suppress features with low separability. Obviously, the separability of the spectra in the overlapping regions is very low, which is indicated by the low classification accuracy and reliability. Taking this into account, we divide the spectral data into several bins of the same size and use a two-layer fully connected neural network (FCNN) with the same structure to learn representative features in each local bin. The learned features are then fed into another fully connected (FC) layer to learn the local believability weighting factors of different bins, which are used to classify any spectral time series.

The structure of the end-to-end binned FCNN with automatically capturing weighting factors model is shown in Figure 5.1, which is an extension of our previous work [104]. In this model, each FCNN sub-module consists of two fully connected layers (FCs), with each one followed by a batch normalization operation and a rectified linear unit (ReLU) [75] or Sigmoid activation function [36] to speed up the training process and avoid overfitting. The input layer has 2 input neurons, corresponding to the di-

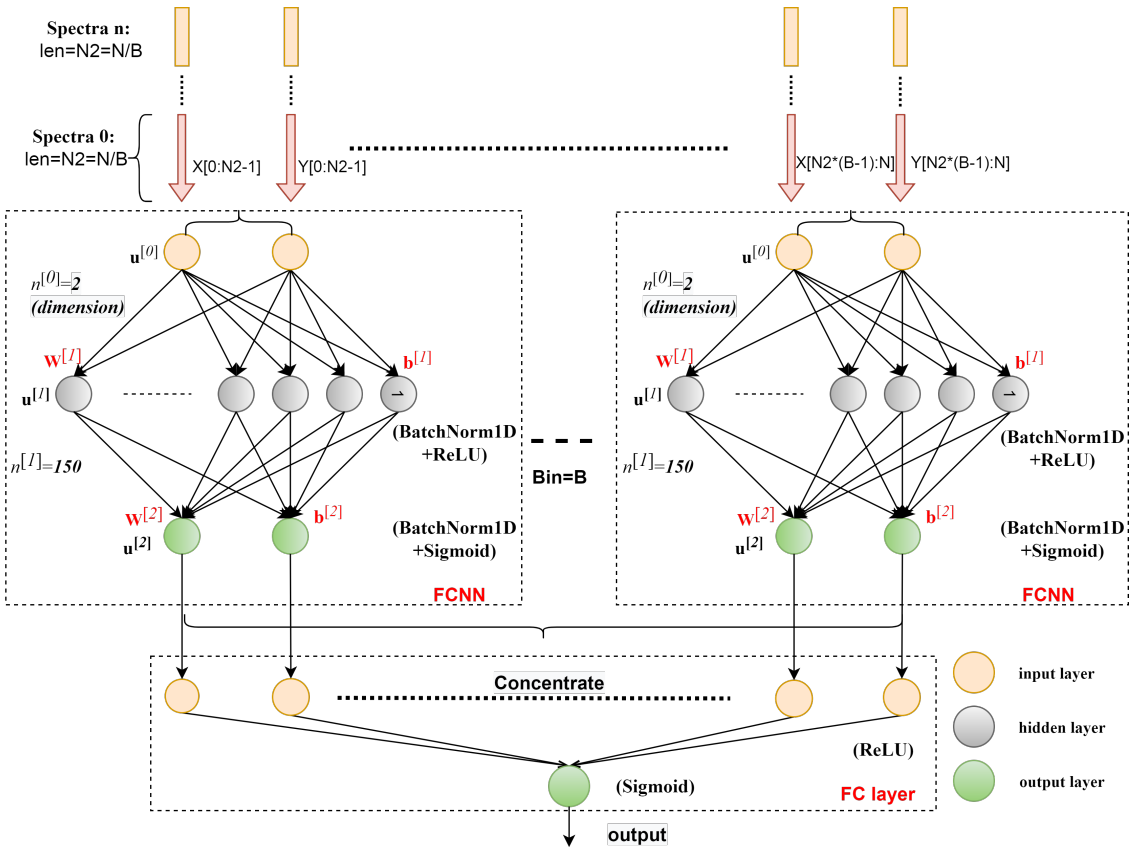


Figure 5.1: Illustration of End-to-end FCNN with automatically capturing weighting factors model.

mensions (two coordinates) of the input spectra, the hidden layer has 150 neurons, and the output layer has 2 output neurons. Then, the output of these FCNN sub-modules is concentrated and fully connected to the final output neuron. The weighting factors learned by the last FC layer can assign different weights to different bins, thereby reducing the influence of the interval of indistinguishable features, and making the features in bins with high separability and reliability more dominant, which can better extract features and learn patterns from the input data. The reason why the Sigmoid function is used instead of ReLU after the second FC layer is that we treat each FCNN sub-module as a complete classification model in each bin, so the function of the last FC layer is mainly to assign an overall classification influence weight to each bin.

When the number of bins is 1, the model becomes a standard three-layer FCNN, also known as a multi-layer perceptron (MLP). In section 5.3, we will experimentally demonstrate the advantages of the bin-weighting technique in terms of classification confidence, required training epochs, and training time.

5.1.2 Convolutional SCT Attention Network

More generally, the spectral data can be regarded as typical 1D time series data. In this setting, several other classification models can be explored. As a powerful network, CNN can learn features from different scales, but unfortunately, it cannot capture feature dependencies in a global view. To solve this problem, some methods involving attention mechanisms have been proposed [90, 119, 29, 15, 28], which investigate attention across single or multiple dimensions. Inspired by the study[119, 28], we designed

a lightweight convolutional Spatial-Channel-Temporal (SCT) attention network for the application of time-resolved spectra classification, as shown in Figure 5.2.

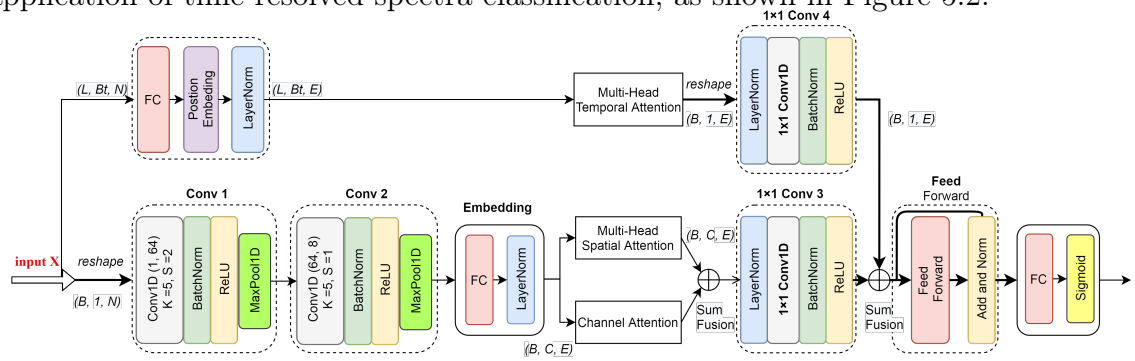


Figure 5.2: Illustration of convolutional SCT attention network architecture. In this architecture, attention is calculated across spatial, channel, and temporal dimensions.

The framework consists of two branches. The upper branch models the global contextual information in the temporal dimension. The lower branch model captures global dependencies across the channel and temporal dimensions. These two branches are then aggregated together to better represent features. Finally, a Feed Forward module and an FC module are applied for the classification task.

Spatial Attention Module and Channel Attention Module

In the lower branch (also the main branch), two convolutional modules Conv 1 and Conv 2 are first applied to extract local features. They have the same structure as the convolution modules in the CNN model, consisting of a 1D convolutional layer, a BatchNorm1d function, a ReLU activation function, and a MaxPool1D operation. The number of filters of the convolutional layer in Conv 1 and Conv 2 is 64 and 8, respectively, the kernel size is 5 and 5, respectively, and the step size is 2 and 1, respectively. The size of the kernel and the stride of the MaxPool1D operation in these two modules are 2. In order to facilitate the follow-up Multi-Head attention operation, the embedding module is added, which contains a fully connected layer, and a 1D layer normalization operation. It produces an output of dimension $d_{emb} = 256$.

We denote the input shape of this branch as of (B, C_{in}, N), where B is the batch size, C_{in} is the number of channels, equal to 1 in the input, and $N = 4023$ is the number of features. After embedding the module, the output is of shape (B, C, E), where E is the number of embedded features, $E = d_{emb} = 256$.

In this branch, two parallel attention modules are applied, a spatial attention module and a channel attention module. The spatial attention module is designed to model the spatial dependencies between any two local features. The channel attention module is designed to capture the global interdependencies over any two channel maps introduced by the Conv 1 and Conv 2 module. These two attention modules are performed concurrently and simply fused through element-wise addition.

Specifically, the Multi-Head spatial attention module is designed based on[119]. If the batch dimension is not considered, we can denote the input of the attention module as $X \in R^{C \times E}$, then Multi-Head spatial attention can be expressed as:

$$\begin{aligned} MultiHead(Q_S, K_S, V_S) &= S_S W_S^O \\ &= Concat(head_1^S, \cdots, head_i^S, \cdots, head_h^S) W_s^O, \end{aligned} \tag{5.1}$$

$$
\begin{aligned}
Mhead_i^S &= Attention(Q_S W_i^{Q_S}, K_S W_i^{K_S}, V_S W_i^{V_S}) \\
&= softmax\left(\frac{Q_S W_i^{Q_S}(K_S W_i^{K_S})^T}{d_k}\right) V_S W_i^{V_S}
\end{aligned} \tag{5.2}
$$

where $W_i^{Q_S}, W_i^{K_S} \in R^{d_{emb} \times d_k}$, $W_i^{V_S} \in R^{d_{emb} \times d_v}$ and $W_s^O \in R^{hd_v \times d_{emb}}$. In this work, we employ $h = 2$ parallel attention heads, with $d_v = d_k = 128$. At last, a residual connection is added, which can be expressed as:

$$
A_S = X + MultiHead(Q_S, K_S, V_S) \tag{5.3}
$$

Here, the query, key, value Q_S, K_S, and V_S used to calculate Multi-Head attention are obtained directly from X without transformation. The spatial attention matrix $S_S = Concat(softmax(\frac{(Q_S W_i^{Q_S}(K_S W_i^{K_S})^T)}{d_k})) \in R^{d_{emb} \times d_{emb}}$. According to Equation (5.4), the obtained spatial feature A_S is a weighted sum of all local features and original features. It achieved global spatial interdependencies by selectively aggregating spatial contexts according to the spatial attention matrix S_s. The channel attention module is designed based on [28], and we extend their method from three-dimensional image processing to a 1D time series analysis. The channel attention matrix is calculated through $S_C = softmax(Q_c K_c^T)$ which can be described as:

$$
\begin{aligned}
A_C &= X + \alpha \cdot S_C \cdot V_c \\
&= X + (\alpha \cdot softmax(Q_c K_c^T) V_c)^T
\end{aligned} \tag{5.4}
$$

where $Q_C, K_C, V_C = X^T$, which is the transpose of X.and $S_C \in R^{C \times C}$, it models the dependencies or similarity between any two channels. These two attention modules are fused together through element-wise addition. A convolution module 1×1 conv3 is performed after the parallel attention modules, which consists of a layer of normalization operation, 1×1 convolution, batch normalization, and a ReLU function. It is used for channel-wise pooling and dimensionality reduction. After this module, the output shape is $(B, 1, E)$.

Temporal Attention Module

The temporal attention module is illustrated in the upper branch of Figure 5.2. It is introduced to model the interconnections of any two spectral time series. Before applying the Multi-Head temporal attention, the input spectra are first fed into an FC embedding module and a position embedding module. Its dimension of output X_T is set to $d_{emb}^T = d_{emb} = 256$, which is the same as the output dimension of the embedding module in the spatial and channel attention path. The shape of input in the upper branch is (L, B_t, N), where L is the sequence length, and B_t is the batch size in the temporal attention branch. Similar to the Transformer-based model, the input sequence length L is set to 31 during training and 349 during testing, which corresponds to the number of original training examples and the total number of spectral curves in the test dataset. The Multi-Head temporal attention is expressed as:

$$
MultiHead(Q_T, K_T, V_T) = Concat(head_1^T, \cdots, head_i^T, \cdots, head_l^T) W_T^O \tag{5.5}
$$

$$
\begin{aligned}
head_i^T &= Attention(Q_T W_i^{Q_T}, K_T W_i^{K_T}, V_T W_i^{V_T}) \\
&= softmax\left(\frac{Q_T W_i^{Q_T} (K_T W_i^{K_T})^T}{d_k^T}\right) V_T W_i^{V_T}
\end{aligned} \tag{5.6}
$$

where $W_i^{Q_T}, W_i^{K_T} \in R^{d_{emb}^T \times d_k^T}$, $W_i^{V_T} \in R^{d_{emb}^T \times d_v^T}$ and $W_s^O \in R^{l \cdot d_v^T \times d_{emb}^T}$. In this work, we employ $l = 8$ parallel temporal attention heads. At last, a residual connection is added, which can be expressed as:

$$
A_T = X_T + MultiHead(Q_T, K_T, V_T) \tag{5.7}
$$

The temporal query, key, value Q_T, K_T, V_T used to calculate the temporal Multi-Head attention are obtained from the X_T. The temporal attention matrix $S_T = Concat\left(softmax\left(\frac{Q_T W_i^{Q_T}\left(K_T W_i^{K_T}\right)^T}{d_k^T}\right)\right) \in R^{L \times L}$. According to Equation (5.7), in our application, the shape of the obtained temporal feature map A_T is (31, 31) during training, and (349, 349) during testing.

After calculating Multi-Head temporal attention, we reshape the output into the shape of $(B, 1, E)$, and apply a 1×1 Conv 4 module. The convolutional layer in the 1×1 Conv 4 module serves as a scaling function for embedded temporal attention features, instead of changing the depth of feature maps as in the 1×1 Conv 3 module.

The temporal features with attention are also simply fused with the other two attention modules through element-wise addition. By applying the 1×1convolution modules, the features in these three attention modules can be easily fused. After that, a residual feed-forward module with the same structure as in [119] is applied to further capture the features. Finally, an FC layer with the sigmoid activation function is applied to the classification task.

In fact, the attention modules designed in this architecture can also be used individually or concurrently in combination with any other models applied in our work (i.e., FC, ResNet, LSTM, et al.). If we only model connections across spatial and channel dimensions, we can obtain another architecture, named the convolutional SC attention model, which discards the upper temporal attention branch, as illustrated in Figure 5.3. It should be noted that in this architecture if the 1×1 Conv 3 module is not used or if you exchange its position with the Feed Forward module, the modules in the red box can be performed multiple times to better represent the features However, we would better strike a balance between performance and memory and computational consumption.

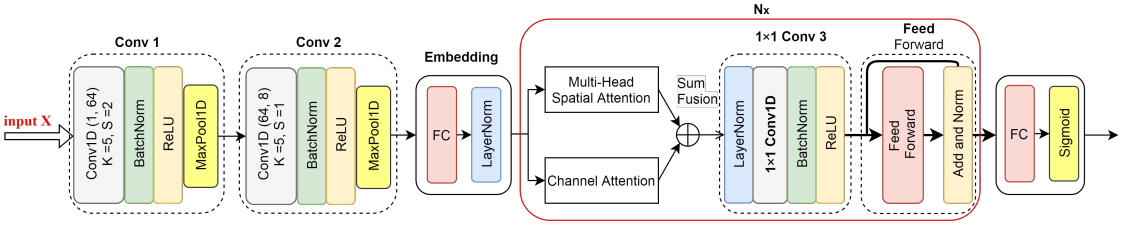


Figure 5.3: Illustration of Convolutional SC attention network architecture. In this architecture, attention is calculated across spatial and channel dimensions.

5.2 Other State-of-the-Art Deep Supervised Learning Approaches

5.2.1 1D Multi-Layer Fully Connected Neural Network (1D FCNN) Model

In this solution, we apply a 1D multi-layer Fully Connected Neural Network (1D FCNN) model for time series spectra classification, which has the most interpretable structure, as shown in Figure 5.4. It is a basic MLP by stacking four fully connected layers and is designed by following two design rules: (i) using Layer Normalization operation and dropout[101] (optionally) after each FC layer to speed up training and improve the generalization capability; and (ii) applying ReLU activation function to prevent saturation of the gradient[125].

The architecture of the 1D FCNN model is illustrated in Figure 5.4. The number of neurons in each hidden FC layer is 256, 512, 64, respectively. The output layer has 1 neuron, followed by a sigmoid activation function to convert the predicted values into probabilities. It should be noted that a dropout layer can be optionally added to prevent overfitting in this architecture.

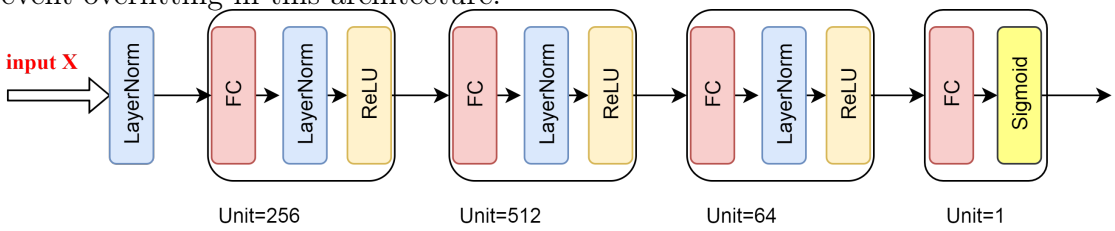


Figure 5.4: Illustration of 1D multi-layer Fully Connected Neural Network (1D FCNN) model.

5.2.2 Convolutional Neural Network (CNN) Solution

The key feature of a convolution operation is weight sharing, which makes it very suitable for automatically and adaptively learning hierarchies of features [129]. Based on this, we design a CNN-based model with reasonable layers of different sizes of kernels. This model is powerful enough for the spectral time series classification in our application. The architecture of the CNN model is shown in Figure 5.5. It is mainly composed of three sequential convolutional modules and one FC module, which are used as the feature extractor and classifier respectively.

In the CNN-based solution, the input is first normalized by the layer normalization function, and then fed into convolutional blocks. Each convolutional block consists of a 1D convolutional layer succeeded by a BatchNorm1d function, a ReLU activation function, and a 1D MaxPool operation. After that, the output of the convolution module is flattened and input into an FC block, which consists of a LayerNorm1d function and an FC layer with one output neuron. Sigmoid is used as the activation function to perform the classification task. The first and second convolutional layers have 64 filters with kernel sizes of 5 and 3, respectively. The last convolution layer is designed with 4 output channels with a kernel size of 3, and a stride of 1 to reduce the number of channels and retain the most obvious features. The size of the kernel and the stride of the max pooling operation are 2. The number of neurons in the flattened

layer (the FC layer) depends on the parameters of the three convolution modules and the number of features (data points) in the input, which is 496 in our case. In this architecture, batch normalization and layer normalization are used to speed up the training process.

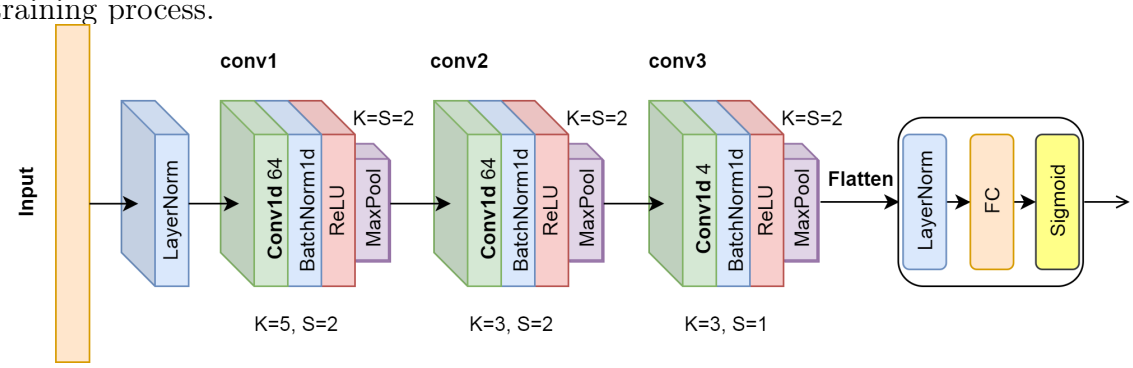


Figure 5.5: Illustration of convolutional neural network solution.

5.2.3 ResNets-Based Solution

In addition to the standard convolutional architecture, we also applied and implemented Residual Networks (ResNets)[40] to the spectral data classification. It was originally proposed to better cope with the degradation problem in deep CNNs. Residual Networks (ResNets) are introduced to cope with the degradation problem in deep convolutional neural networks. The identity shortcut connections embedded in the ResNets can aid gradient backpropagation through the network and allow the training of very deep models, as shown in Figure 5.6. Currently, ResNets are a common backbone architecture in computer vision and also have been adapted to time series classification[65] where 2D convolutions through the spatial dimensions are replaced by 1D convolutions through time [90]. They consist of residual blocks of convolution layers and batch normalization layers (optionally), followed by a Rectified Linear Unit (ReLU) activation function.

Multiscale ResNets architecture, constructed by multiple residual CNN blocks with different convolutional kernel sizes that can capture complex features at multiple scales, has shown good results on time series benchmark datasets[90, 65]. In this work, we utilize the ResNets model proposed by Liu et al. [65] where the input time series are processed in three independent streams, each stream has three residual blocks, and the convolution kernel sizes are 3, 5, and 7, respectively. After the residual blocks are in each stream, the features are then average pooled with pooling kernel sizes of 16, 11, and 6; thus, yielding multiple-scale feature representations. Because their architecture is well-designed, we adopted their architecture. One difference from the architecture[65] is that we added a LayerNorm1d layer after the original input and before the FC layer. In addition, we replaced the feature concentration operation with a simple element-wise feature addition operation after the three residual blocks. If we denote the feature representations in the three streams as $\mathcal{F}_{c1}$, $\mathcal{F}_{c2}$, $\mathcal{F}_{c3}$, then this operation can be expressed as $\mathcal{F}_c = \mathcal{F}_{c1} + \mathcal{F}_{c2} + \mathcal{F}_{c3}$. Another commonly used operation for the features fusion is to perform a weighted element-wise feature addition, which can be expressed as $\mathcal{F}_c = \alpha\mathcal{F}_{c1} + \beta\mathcal{F}_{c2} + \gamma\mathcal{F}_3$, where α, β, and Γ are learnable parameters during training. Usually learning these three parameters will greatly slow down the training process and need more training samples.

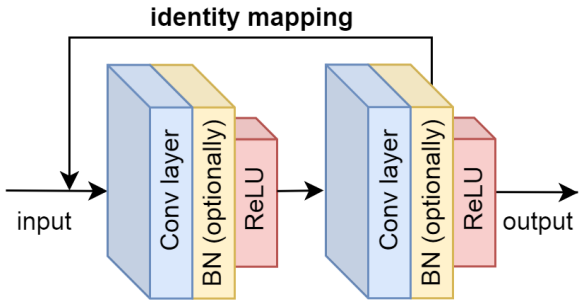


Figure 5.6: Illustration of a building residual block.

5.2.4 LSTM-Based Solution

As a variance of RNN in particular, Long short-term memory (LSTM) was originally applied in NLP tasks, and it also yielded promising results in time series analysis[90, 60, 114]. A cell and three gates (input gate, output gate, and forget gate) in the LSTM unit allow this architecture to remember values over arbitrary time intervals and regulate the flow of information[117]. The pointwise operations used to update the cell state and hidden state in the LSTM architecture can assign different weights to different features/variables in our spectral time series, thereby improving the role of distinguishable features in the classification task and weakening the impact of indistinguishable features on classification. These characteristics make it very suitable for time-resolved spectra analysis.

In the LSTM-based model, during training, we processed 31 spectral data vectors in each batch, corresponding to the number of original training examples described in Section 5.2.3, so the connections between different spectral observations at different time steps (sequential correlation of spectral time series) can be effectively leveraged. During the test, the sequence length is set to 349, which is the total number of our spectra data. The spectra data classification model based on the LSTM structure is shown in Figure 5.7.

In this model, a dense layer or a convolutional layer can be used for input embedding, followed by a LayerNorm layer and ReLU activation function. In this process, the dimensionality of the input signal can be reduced as well. For simplicity, a dense layer is applied in our implementation, with 512 output units. The learned input embeddings are then fed into the two layers LSTM module, with 512 neurons in the hidden state. Next is the classification decoder module, which consists of a LayerNorm layer and a dense layer (512 input neurons and 1 output neuron), where sigmoid is used as the activation function for the classification task. LayerNorm operations applied in this model prevent over-fitting and speed up training. Compared with our previous work [12], the PCA preprocessing process is replaced by the input embedding module, so an end-to-end LSTM-based classification model is constructed. In addition, the model also makes use of time dimension information to make the extracted features more complete.

5.2.5 Transformer-Based Solution

The Transformer model relies on the so-called self-attention mechanism and is found to be superior in quality while being more parallelizable[119]. There are many successful applications of Transformers in time series processing tasks, including classification [81, 30]. In this work, we adopted the encoder part architecture of the self-attention-

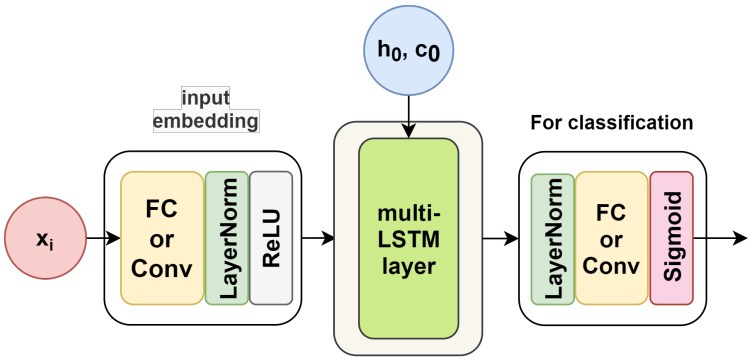


Figure 5.7: Illustration of Multi-LSTM-based model for spectra data classification.

based Transformer network as a comparison model.

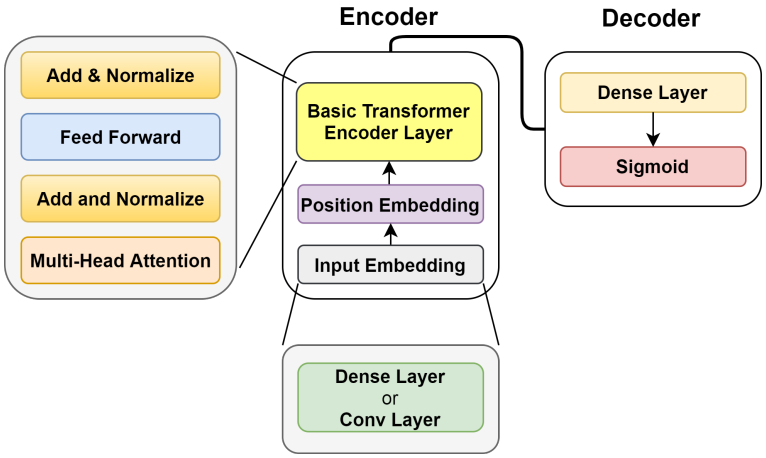


Figure 5.8: Illustration of Transformer-based Model. A single transformer layer is used for spectra data classification. In the decoder part, the dense layer with Sigmoid as the activation function is used for the classification task.

The transformer-based method for spectral data classification is illustrated in Figure 5.8 below. Since our spectral time series data lives in a continuous space of spectral intensity values, the dense layer or the convolutional layer for input embedding can be used instead of the word embedding step. In the implementation, we adopt a dense FC layer for the input embedding, it produces outputs of dimension 256. The sequence length setting in the training and testing process is the same as that of the LSTM model, so as to utilize the sequential correlation information of different spectral time series. The step of positional encoding is adopted according to Ref. [119]. In this work, we employ two Transformer encoder layers with 8 attention heads, running in parallel.

In the decoder part, similar to the input embedding, a dense layer with sigmoid as the activation function is used to predict the probability of the class label of each spectral curve. Here, the input dimension of the dense layer is 256, and the output dimension is 1.

5.3 Experiments and Results

In this section, we first introduce the implementation details of the above DL classification models and describe the specific performance metrics we used, and then evaluate and compare their classification performance from multiple perspectives. Further, we

conducted a feature importance contribution analysis to explain and interpret these DL classification models. In addition, we analyzed the self-attention scores across the spatial, channel, and temporal dimensions in the convolutional SCT attention model to better investigate its ability to selectively focus on a few classification-related information.

5.3.1 Implementation Details

All the models are performed in a supervised learning manner and implemented on the Jupyter Notebook platform using PyTorch. These models are trained by backpropagation using gradient descent, with the adaptive learning-rate method Adam [56] as the optimizer (learning rate is set to 8×10^{-3} for convolutional SCT/SC attention model and 2×10^{-3} for other models; weight decay is set to 2×10^{-4} for convolutional SCT/SC attention model and 2×10^{-5} for others). We use the binary cross-entropy loss function for our classification task. The statistical models are obtained by minimizing the loss function on the training dataset. All these models are trained on one machine with Tesla P100-PCIE-16GB GPU.

The spectral data used in this chapter was previously introduced and detailed in Section 2.1 and 3.1, as shown in Fig. 3.1. In the high-pressure experiments, the distribution of the spectra changed due to the effect of pressure changes during compression and decompression, and these changes were manifested as changes in the intensity, position, and shape of the peaks along the diffraction angular dimension and the time dimension.

For the end-to-end binned FCNN with the automatically capturing weighting factors model proposed in the 2D space segmentation problem setting, the training dataset consists of 4 original representative spectra samples and 60 simulated ones (by adding some random noise, 15 simulated spectral curves are generated based on each original spectrum), for a total of 64. The reason why we use only a small amount of training data samples in this model is that in this case, the data is only two-dimensional, so the amount of training examples is much larger than the data dimension, which is well-acceptable. In this model, we set the initial training epochs to 550. As the number of bins increases, the training examples in each bin decrease linearly, which may cause the model to easily overfit. To deal with this problem, two early training stop criteria are applied, that is, when the training accuracy is greater than 99.0% or the training loss is less than 0.15. The epochs required during training are shown in Figure 5.9.

For the applied models in the 1D time series data analysis setting, there are 31 original representative spectra samples (16 belonging to category label 0 and 15 to category label 1) with 1550 simulated ones as mentioned above (by adding some random noise, 50 simulated spectral curves are generated based on each original spectrum), a total of 1581. The small random noise in the simulated spectral data is generated using the Mersenne Twister[70] as the core generator. Their training information, such as training epochs, training time, memory consumption, etc., are listed in Table 5.1. The analysis scripts as Jupyter notebooks are publicly available at Github[1]. Further implementation details of this software program will be presented in Section 5.4.

[1]https://github.com/sunyue-xfel/Comparing-End-to-End-Machine-Learning-Methods-for-Spectra-Classification

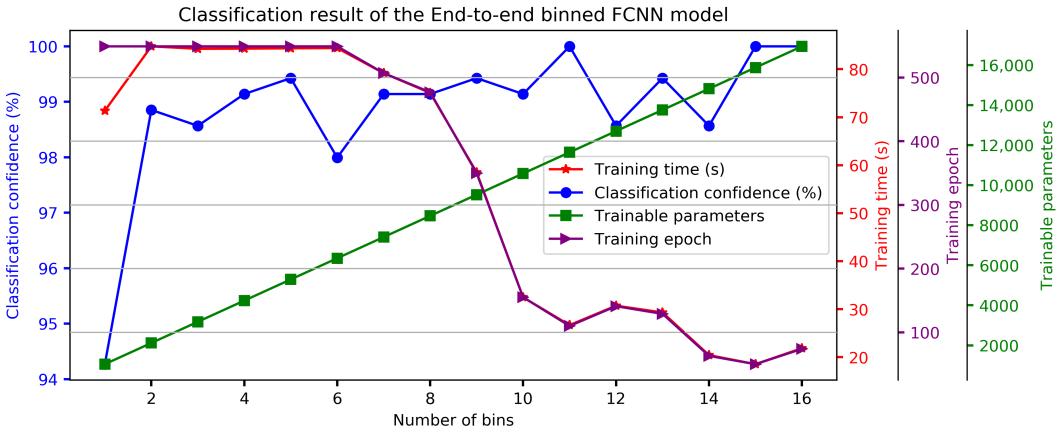


Figure 5.9: Training information and classification result of the end-to-end binned FCNN with automatically capturing weighting factors model.

Model	Trainable Parameters	Training Time (s)	Memory (GB)	Epochs	Classification Confidence	Classification Boundary
1D FCNN	1,204,335	1.576	0.1	300	100%	(188,189)
CNN	23,307	0.753	0.0	5	100%	(173,174)
ResNets	2,354,535	8.208	7.1	6	100%	(178,179)
LSTM	6,273,391	0.463	0.9	25	100%	(175,176)
Transformer	1,830,511	0.449	0.1	30	100%	(185,186)
Convolutional SCT Attention	1,953,353	1.269	0.5	5	100%	(188,189)
Convolutional SC Attention	659,525	1.652	0.4	8	100%	(182,183)

Table 5.1: Evaluation and Comparison of the state-of-the-art models on the HED spectra dataset.

5.3.2 Quantitative Model Evaluation

In the 2D space segmentation problem setting, the training information and classification result of the proposed end-to-end binned FCNN with the automatically capturing weighting factors model is shown in Figure 5.9. It can be seen from the result that the classification confidence increases as the number of bins increases, and finally achieves 100% confidence. In addition, we can also easily understand that although the trainable parameters increase linearly, as the required training epochs decrease, the required training time also decreases. The classification results show that the binned weighting technique can minimize or even eliminate the impact of misclassification of data points in the overlapping regions with low separability, and even greatly reduce the training time. When using 15 or 16 bins, the model consumes approximately 20 s training time.

When regarded as a general 1D time series data classification problem, all the applied or proposed models can achieve 100% classification confidence, and the classification boundaries obtained by different models are very stable, and only fluctuate in a small range, that is $((188 - 173)/349 = 4.3\%)$, as shown in Figure 5.10 and Table 5.1. Among these models, the CNN-based model requires the least training parameters. The LSTM-based model has the most trainable parameters, which is mainly introduced by the LSTM units, but its convergence speed is very fast, the training time is short, and the memory consumption is low. Regarding the training time, the

Transformer-based model requires the least training time (0.449 s), followed by the LSTM-based model, which needs slightly more training time (0.463 s). At the same time, it can be clearly seen that the curve of memory usage during training has a similar trend to the training time curve. Although the 1D FCNN model requires 300 training epochs, it does not consume so much training time (1.576 s), which is only 3 times that of the LSTM-based model, and it does not consume much memory. From the result, it can be noted that the ResNets model consumes the most time training (8.208 s), and takes up the most memory, but the trainable parameters are still much less than the LSTM-based model. The reason for the long training time may be that there are a bunch of convolution kernels of different scales in this model. The related training information and results are listed in Table 5.1.

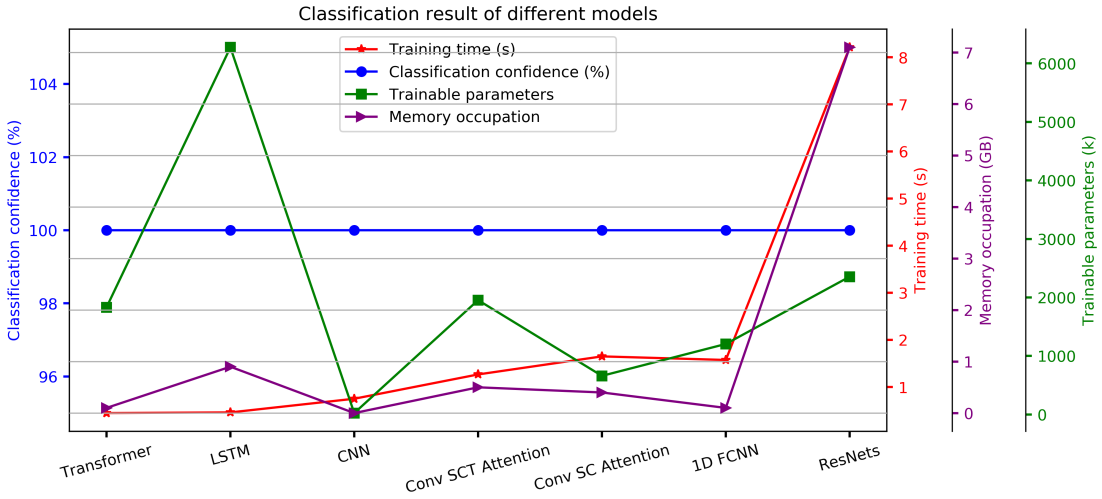


Figure 5.10: Training information and classification results of these models under the 1D time series data classification problem setting.

Nevertheless, the time consumption of ResNets is still much smaller than that of end-to-end binned FCNN with the automatically capturing weighting factors model. In the 2D space segmentation problem setting, the way the model processes data is equivalent to switching from parallel processing to serial processing. The resulting more training examples in 2D space segmentation setting greatly increases the training time. Therefore, although the binned-weighting technique greatly improved the classification performance in the 2D space segmentation problem setting, the models proposed or applied under the 1D time series classification setting are superior.

5.3.3 Feature Importance Attribution Analysis Based on Gradient Backpropagation

To better interpret the classification models, feature importance attribution analysis is usually employed to quantify the interpretability of designed networks. An explainable and interpretable classification model would be expected to have an attribution map that can dynamically highlight the highly separable features (corresponding to the peak area in our application) while suppressing the indistinguishable features as much as possible. Since gradients can reflect the influence of each input data point on the current class prediction, and this analysis is model agnostic, and does not require

ground-truth labels [90], we conducted a feature importance analysis based on gradient backpropagation. In this experiment, we estimate the contribution of each input feature on the classification prediction of each model applied under the 1D time series classification setting, as shown in Figure 5.11. The contribution map is the cumulative result of all 1581 training examples.

The results of this experiment show that in the 1D FCNN model (Figure 5.11b), the feature importance curve maintains a smooth shape, and focuses on most of the obvious features of the spectra. Among them, features close to class 0 obtain a positive contribution value, and features close to class 1 have a negative contribution value. In addition, the contribution of most features in the overlapping area is close to zero, which can reduce the misleading effect of indistinguishable features on classification, and further make the classification-relevant features dominate. Since the 1D FCNN model is only composed of FC layers, it is the most interpretable. The smooth feature importance curve also reflects this point. In order to investigate this further, we visualized the weights learned in each FC layer, as shown in Figure 5.12. It can be clearly seen that in each FC layer, only some features are assigned higher weights. The high interpretability of FC layers reveals particularly classification-relevant features. In the CNN-based model (5.11c), the obvious features are also well-highlighted, which further proves the powerful feature extraction ability of the convolution operation. However, one of the imperfections is that the curve is not well-shaped. For ResNets (5.11d), due to the large strides used in the convolution kernel and pooling operations, the feature importance attribution curve of ResNets is more discrete. Its attribution map is also a bit noisy, some indistinguishable features have considerable weight, such as features in the range of 1500 to 2000, but most of the obvious features with high separability are still emphasized.

The gradients through the LSTM-based model (5.11e) and Transformer-based model (5.11f) were a little noisy, their importance values in the overlapping regions are not very well-suppressed We can still easily see that the obtained feature importance contribution curve can well-describe the position and strength of distinguishable features. It can be seen from the importance maps that the features with higher separability achieved higher feature importance values, especially the obvious features with lower intensity values in the latter half of the spectral curve, which are also well-captured.

For convolutional attention networks, most of the obvious features are well-emphasized, and the importance contribution values of the features with low separability are much lower than that of the features with high separability. In the convolutional SCT model (Figure 5.11g), the feature importance value of some overlapping regions is even zero, which perfectly suppresses the indistinguishable features with low separability and low classification confidence. In addition, the shape of its feature importance curve is similar to that in the 1D FCNN model, indicating that it has high interpretability. Compared with the convolutional SC attention model (Figure 5.11h), it shows its advantages of introducing temporal attention. This indicates that the attention mechanism applied in this model can promote the distinguishable features and suppresses the unobvious and misleading features.

Among all these models, only some of the non-obvious features in the convolutional SCT attention model can reach an importance value of 0, which benefits from the self-attention mechanism applied across three dimensions. We further show its feature importance attribution map of the test dataset in Section 5.3.4.

The gradient-based feature importance analysis can also be used as one of the

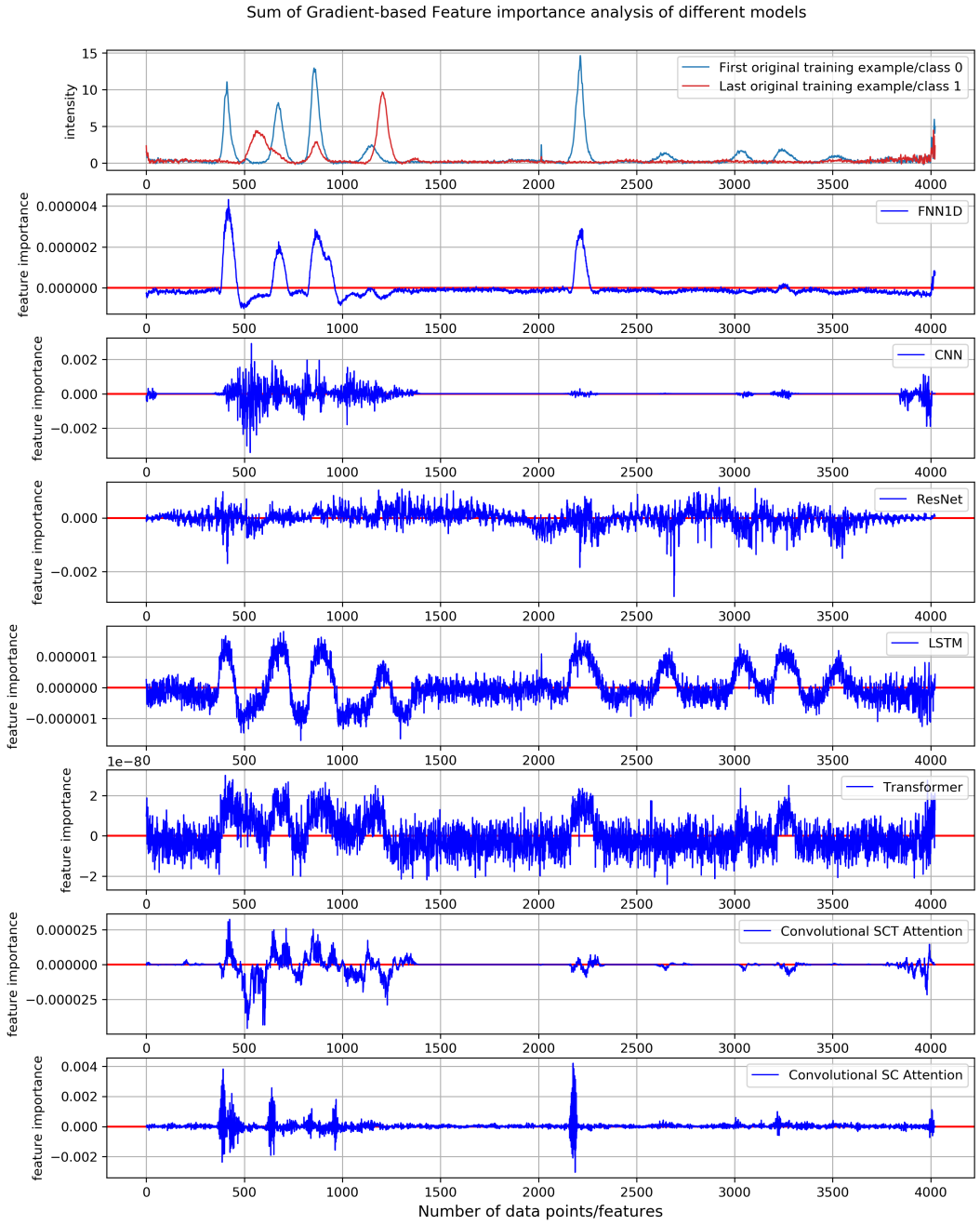


Figure 5.11: Feature importance analysis result of different models under 1D time series data analysis settings. The feature importance is calculated based on the sum of the gradients of all training examples. (a) The first and the last original training examples. (b) The feature importance map of the 1D FCNN model. (c) The feature importance map of the CNN-based model. (d) The feature importance map of the ResNets model. (e) The feature importance map of the LSTM-based model. (f) The feature importance map of the Transformer-based model. (g) The feature importance map of the Convolutional SCT attention model. (h) The feature importance map of the Convolutional SC attention model.

references for classification performance, which can be combined with the loss function for model training and evaluation. Usually, if the model is not well-trained, the feature contribution map will be rather messy, distinguishing features will not dominate the classification task, and many unobvious features will have a great impact; thereby,

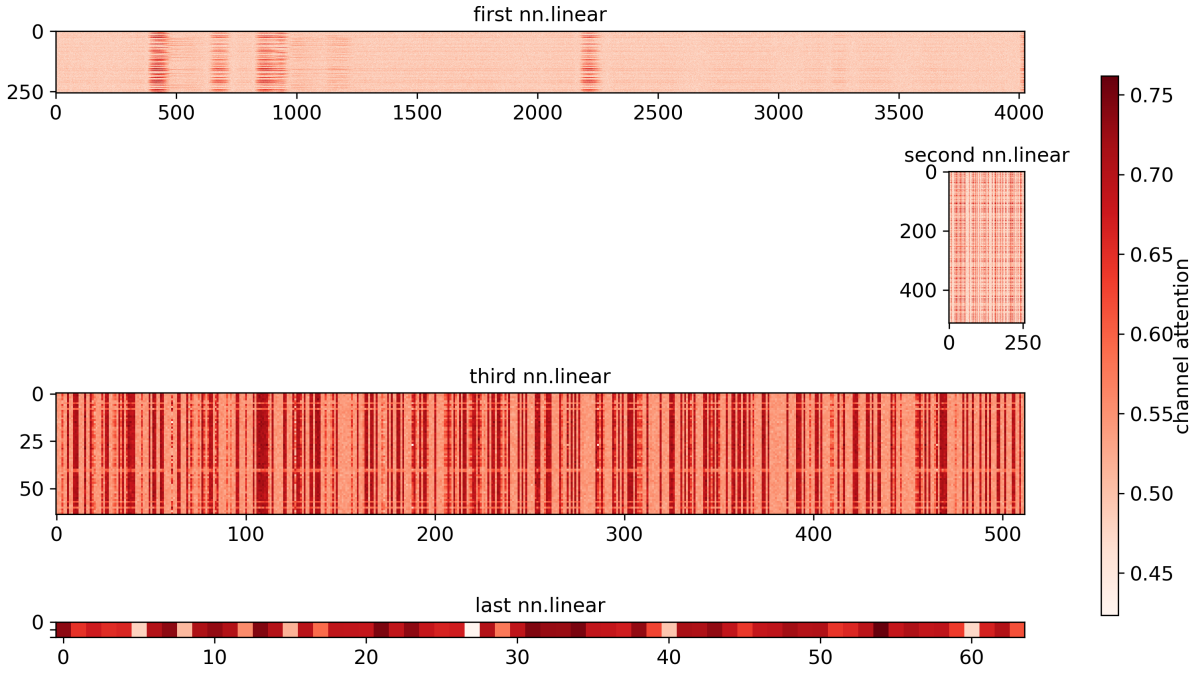
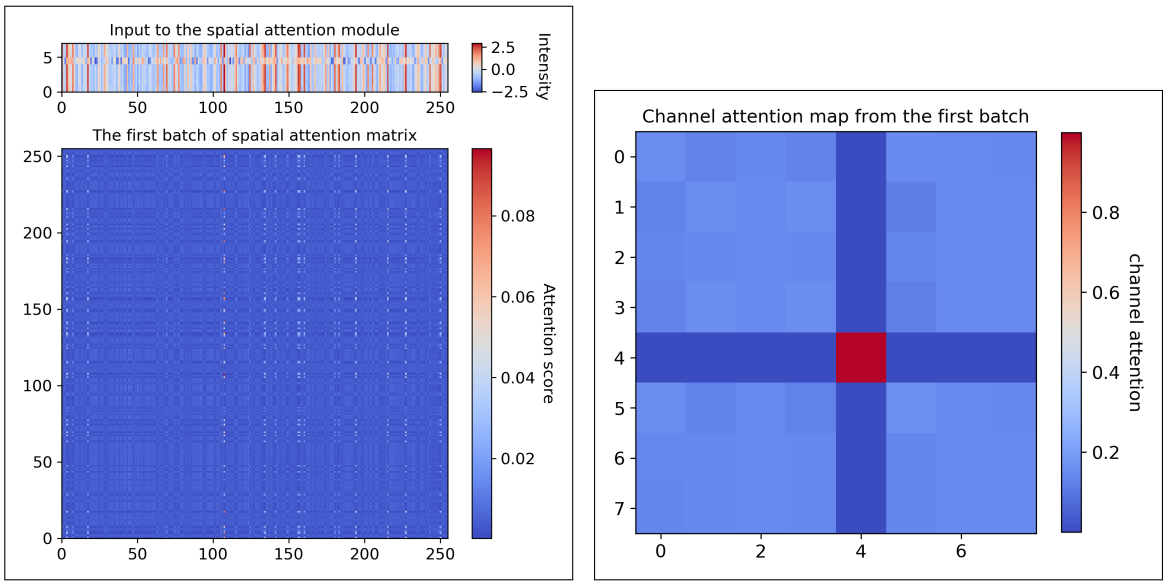


Figure 5.12: The weights learned in each FC layer of the 1D FCNN model.

reducing the credibility of the classification result.

5.3.4 Qualitative Analysis of Self-Attention Scores in Convolutional Attention Model

In the feature importance analysis part, we experimentally observed from the contribution map in the convolutional SCT attention models, the distinguishing features can be well-emphasized, which is indicated by high importance contribution values, while the importance values of features in overlapping regions with low separability is close to zero or even zero. It shows the ability to selectively focus on classification-relevant features and suppress the indistinguishable features with low classification confidence. Here, we investigate this further by focusing on the self-attention mechanism performed across spatial, channel, and temporal dimensions and analyze their attention scores. These attention matrixes can be regarded as adjacency matrices between the input feature vectors and output feature vectors. The attention scores define the dependencies between any two higher-level embedded feature vectors across different dimensions. In our showcase example, the input shape of the spatial attention module and channel attention module is (8, 256). Therefore, the shape of the resulting attention matrix in the spatial dimension and channel dimension are (256, 256) and (8, 8), respectively, as described in Section 5.1.2. In this model, the first batch of the input vectors of the spatial attention and channel attention modules are visualized in Figure 8a, top. The corresponding spatial attention matrix obtained during testing is shown in Figure 8a, bottom, and the channel attention matrix is visualized in Figure 8b. From this result, we can understand that only some obvious spatial feature vectors and channel feature vectors are emphasized, which correspond well to the attention matrices. For example, By capitalizing on the inter-dependencies among channel maps, we can accentuate the interdependent feature maps and achieve additional enhancements in feature representation[28].



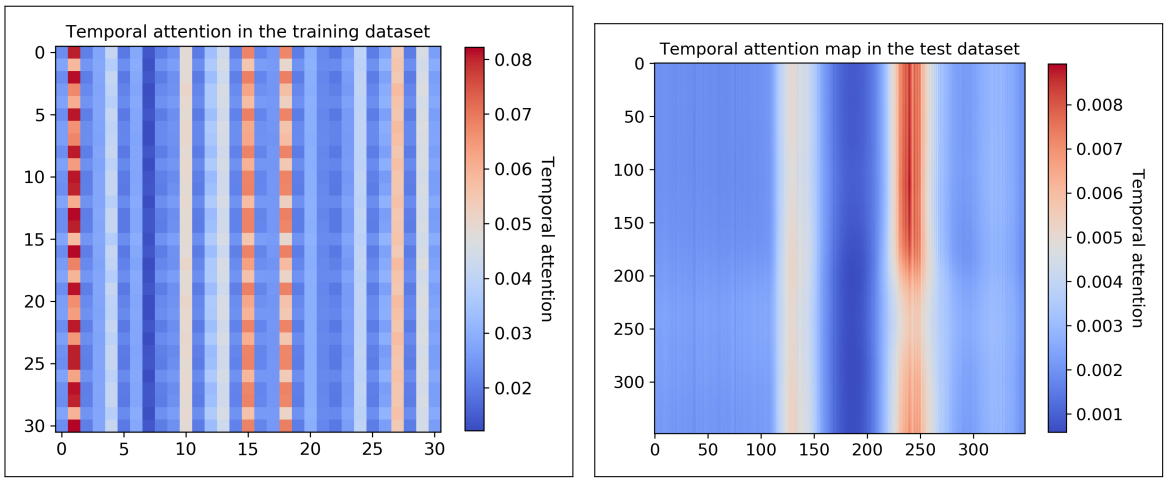
(a) t-SNE visualization of noisy data (b) Learned representations of noisy data

Figure 5.13: (a) The first batch of the spatial attention matrix during testing in the convolutional SCT attention model (top). The input to the spatial attention module (bottom). We can see clearly that the spatial attention matrix can selectively focus on obvious features. (b) The first batch of the channel attention matrix of the test dataset in the convolutional SCT attention model.

In addition, we also visualized the temporal attention matrices obtained from the training dataset and the testing dataset, and the results show that the self-attention scores only focus on the feature vectors of a few time steps, as shown in Figure 9. The temporal attention matrix of the test dataset is obtained according to the parameter matrixes learned in the Multi-Head temporal attention module, as described in Section 3.2.2. From these two temporal attention matrixes, we can find that they have similar shapes, and can selectively focus on some high-level embedded temporal feature vectors that are more relevant to the classification task. In this way, the long-range dependencies modeled across spatial, channel, and temporal dimensions feature greatly enhanced the ability of feature representation.

Further Information about Feature Importance Attribution Analysis

In Figure 5.15, we further show the feature importance map of the test dataset in the convolutional SCT attention model and visualize the feature importance attribution curve of each test spectra. It can be seen from this figure that the distribution of the feature importance map of the spectral dataset is consistent with its intensity distribution. The classification model can selectively focus on obvious features and can even capture the dependencies across temporal dimensions.



(a) t-SNE visualization of noisy data (b) Learned representations of noisy data

Figure 5.14: (a) The first batch of temporal attention matrix during training and (b) the temporal attention matrix of the test dataset in the convolutional SCT attention model.

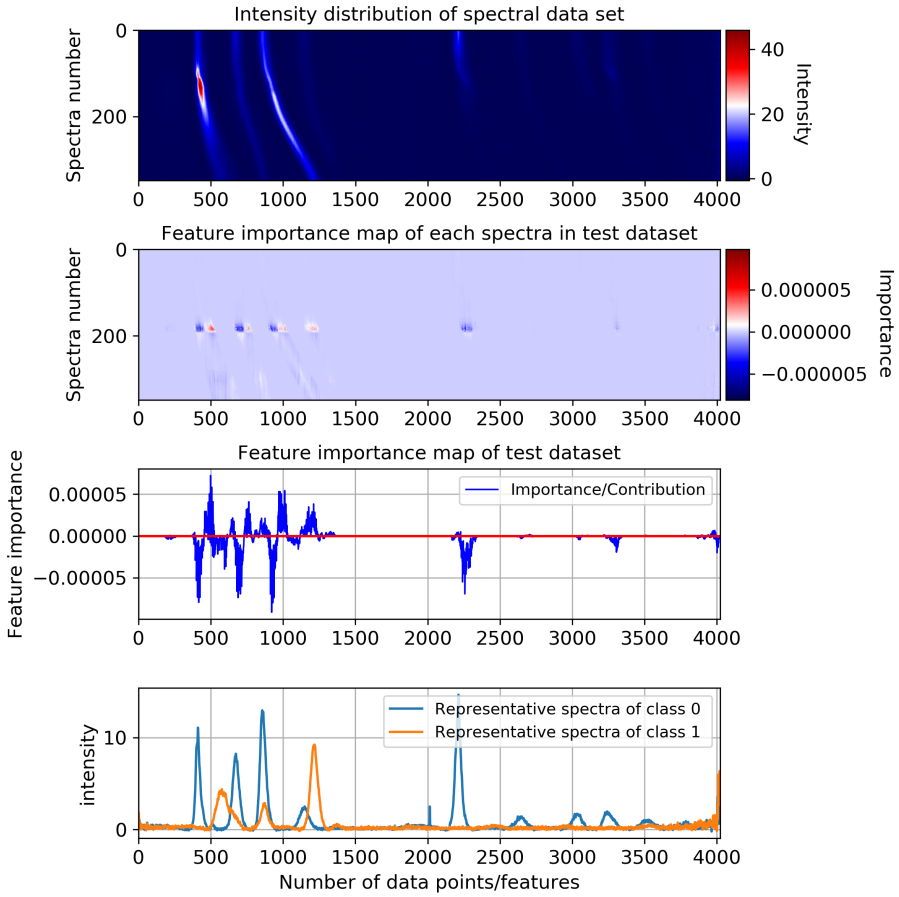


Figure 5.15: Top row: A contour plot showing the time evolution of the diffraction spectra of the HED dataset. Second row: Visualization of the feature importance map of the test dataset in the convolutional SCT attention model. Third row: The sum of the gradient-based feature importance values in the test dataset. Bottom row: Representative diffraction spectra of class 0 and class 1 (first and last spectral curves).

5.4 An Open-source Interactive Software Program for ML-based Approaches to Spectral Classification

In the preceding sections, effective end-to-end deep learning solutions have been proposed and well demonstrated, in this section, we provide details of the implementation of these theoretical contributions in an interactive software program to help software developers with spectral classification tasks based on these DL models [109]. The driving force behind this research is the need for more efficient and accurate analysis of large amounts of spectral data in an interactive computing environment.

5.4.1 Jupyter Notebook for Reproducibility

Jupyter Notebook is an interactive and exploratory computing platform [3] that combines coding, documenting, and visualizations together. In this work, all the spectra classification ML models are designed based on the PyTorch library and implemented on the Jupyter Notebook platform for better data exploration and reproducibility. The program is publicly available on GitHub. The details of this project are described below.

5.4.2 Architecture Description

The spectral analysis scripts released in the form of Jupyter notebooks mainly consist of five components:

- Read datasets from databases.
- Prepare training, validation, and test data.
- Build ML model.
- Training process.
- Results analysis.

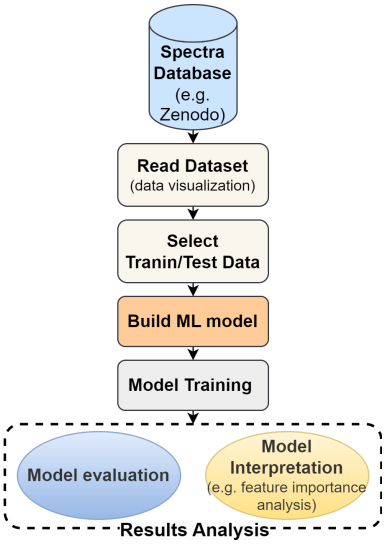


Figure 5.16: Application design architecture.

The corresponding architecture of the program is illustrated in Figure 5.16. With notebooks, you can easily visualize the data, visualize the data analysis process, as well as the analysis results. We describe each component in detail in the following section.

Read datasets from public repositories. This repository represents any publicly accessible database that stores spectral data (see Figure 5.17 for an example), which is necessary to run the program in an interactive environment. Alternatively, it can be data that is private to the user when running in its own environment, in which case not making the data publicly available has no effect on the data analysis. In addition, data reading includes data visualization, which facilitates the exploration and analysis of the data.

```
In [7]: import shutil  # unpacks in current path unless an additional path argument is provided
        ! curl --output Mg20Fe80O_ramp_xyfiles_1_349_azimuthal_integrated.7z \
        https://zenodo.org/record/4424866/files/Mg20Fe80O_ramp_xyfiles_1_349_azimuthal_integrated.7z

        import os

        path="dataset_1"

        if not os.path.isdir(path):
            print('The directory is not present. Creating a new one..')
            os.mkdir(path)
            print(path)
        else:
            print('The directory is present.')
            print(path)

        # %mkdir -p path

        import py7zr
        with py7zr.SevenZipFile("Mg20Fe80O_ramp_xyfiles_1_349_azimuthal_integrated.7z", 'r') as archive:
            archive.extractall(path)

          % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                         Dload  Upload   Total   Spent    Left  Speed
        100 6083k  100 6083k    0     0  2202k      0  0:00:02  0:00:02 --:--:-- 2202k
        The directory is present.
        dataset_1
```

Figure 5.17: An example to show how to get the diffraction spectra dataset from the open data repository ZENODO for supervised classification models.

Prepare training, validation, and test data. This procedure is designed to prepare data for training and evaluating ML models. It includes the selection of training, validation, and test data, as well as the data labeling procedure required for supervised learning methods or linear evaluation in self-supervised learning approaches (such as those in Chapter 7). In addition, it includes data simulation for supervised learning methods to scale up training data when only limited data is available, or data augmentation for defining pretext tasks in self-supervised learning methods.

Build ML model. This component is the key part of the program. In this interactive platform, these end-to-end models described in Section 5.1 are implemented and compared. In addition, in this program, the user can train a model from scratch and save it for later use or load a pre-trained model.

Training process. This component defines the training process, including the training strategy, optimizer, and hyperparameter settings. You can also easily monitor the training process in Jupyter Notebook, and Figure 5.18 shows an example of the training process. Additionally, Figure 5.18 demonstrates how Jupyter Notebook enables the

easy amalgamation of annotations, code, and real-time computational outputs, creating a user-friendly and practical environment for conducting data analysis and machine learning tasks.

Control whether you want to train the model or load an already trained model

In [40]:
```
Training_flag = 1
```

In [41]:
```
if Training_flag ==1:
    # ====================training====================
    # ====If you want to train the model yourself. ====
    from time import time
    t0= time()
    Acc =TrainingProc()
    t1= time()

    with torch.no_grad():
        y_pred= model(X2_TRN) #
        y_pred= torch.squeeze(y_pred)
    print("training done in %0.3fs" % (t1 - t0))
else:
    pass
```

```
' [EPOCH]: 29, [LOSS]: 0.000024, [ACCURACY]: 1.000'

training done in 0.274s
```

Figure 5.18: Example of the training process of the Transformer-based classification model.

Results analysis. This process consists of two parts: model evaluation as well as model interpretation. The model evaluation focuses on the testing and visualization of classification performance. In these supervised learning approaches, the classification confidence described in Section 3.5.2 is used to evaluate the performance of the ML model, which determines whether the model has clear boundaries between different categories or whether there are ambiguous phase transition regions. When the onset of the phase transition is difficult to determine, by measuring the classification confidence, we can indirectly assess the accuracy and reliability of the model predictions.

Model interpretation is a crucial process that involves various techniques such as feature importance analysis, learned representation visualization, and weight matrix visualization. This process helps improve the model by monitoring its training results and understanding its decision-making criteria. In addition, by interpreting the model, humans can increase their confidence in the model's predictions and provide explanations accordingly.

5.4.3 Interactive Environment Setup in MyBinder and Google Colab.

In this section, we set up the interactive environment for the program in two ways, MyBinder and Google Colab.

MyBinder environment. There are several ways to make an interactive environment in MyBinder based on the GitHub public repository, for example with a *requirements.txt* or *environment.yml*, more information can be found on the MyBinder

website. In this work, we established the environment by using a 'requirements.txt' file. This file outlines the specific dependencies needed by the scripts, as demonstrated in the following code snippet.

```
numpy==1.17.2
torch==1.10.0
matplotlib==3.1.3
seaborn==0.10.0
h5py==2.10.0
py7zr==0.19.2
```

Open the whole project in Mybinder or Colab:	launch binder Open in Colab
1. CNN model:	launch binder Open in Colab
2. LSTM model:	launch binder Open in Colab
3. Transformer model:	launch binder Open in Colab
4. ConvSCT Attention model:	launch binder Open in Colab
5. ConvSC Attention model:	launch binder Open in Colab
6. FNN_1D model:	launch binder Open in Colab
7. ResNets model:	launch binder Open in Colab
8. BINned_Weighting_1D model:	launch binder Open in Colab
9. Self-Supervised Classification model:	launch binder Open in Colab

Figure 5.19: MyBinder and Google Colab Lunch badges generated for the ML-based spectra classification project and the Jupyter Notebook for each ML model. By clicking on these badges, you can directly run the codes and interact with the program. It should be noted that the model will be described in Chapter 7

.

The program can be accessed through the permanent link, or by clicking on the launch badge generated for the repository in the README.md file, as shown in Figure 5.19. In addition, to facilitate access to individual ML models, we created a launch badge pointing to a specific Jupyter Notebook program for each ML model.

Colab environment. The environment setup with Google Colab is simpler than MyBinder because it comes with pre-installed packages commonly used in machine learning, such as NumPy, Torch, and Matplotlib. Compared to MyBinder, it can provide more free cloud-based computing resources and flexibility, including CPU and GPU, which makes it very suitable for deep learning programs. Packages or libraries that are not included in Colab can be easily installed using !pip install command or !apt command in Notebook cells. Finally, the corresponding badges were generated by entering the specific GitHub URL on the website.

With the user-friendly MyBinder or Colab platforms, users can run code directly in Jupyter Notebooks without any special setup or installation, greatly facilitating the exploration of data and these ML classification models, and speeding up the development process of new models. Note that the presented solution requires the download of

the data. On one side, it is nice because this solution does not require long-term data storage capacities at the computing platform, but it can also be a bottleneck when trying to work with big data which should not be downloaded all the time. Platforms, like the research data management solution NOMAD [94] which combines long-term storage with computing nodes for collaborative data analysis, can provide an ideal environment for such research.

5.5 Summary

In this work, we focus on the application of spectra classification in the fields of diffraction and spectroscopy and on developing and evaluating deep learning classification frameworks for 1D spectral time series. We regard the task as either a 2D space segmentation problem or a 1D time series data analysis problem. We focus on two proposed classification models, namely the end-to-end binned FCNN with automatically capturing weighting factors model when viewed as a 2D space segmentation problem and the convolutional SCT attention model when viewed as a 1D time series classification problem. And several other end-to-end model structures based on FCNN, CNN, ResNets, Transformer, and LSTM are explored. Finally, we evaluated and compared the performance of these classification models based on the HED spectra dataset.

In the 2D space segmentation problem setting, for the proposed binned FCNN with automatically capturing weighting factors model, a quantitative comparison experiment was carried out under different numbers of bins. The classification results show that the classification confidence increases with the increase of the number of bins, while the training time decreases, which means that the binned weighting technique can minimize or even eliminate the effects of misclassifications of indistinguishable features in overlapping regions and speed up the training.

In the 1D time series data analysis problem setting, the 1D FCNN model, CNN-based model, ResNets model, LSTM-based model, Transformer-based model, and convolutional attention models are applied, and these models were compared from multiple angles by reporting their classification performance on HED spectra dataset. Our results show that in this setting, all the models can achieve 100% confidence. Among them, the LSTM-based model and Transformer-based model architecture consume less time (0.463s and 0.449s, respectively) compared to the convolutional-based models or the FCNN-based model. Our proposed convolutional SCT attention model also achieved good performance, consuming 1.269 seconds of training time. The CNN-based model has the least trainable parameters and occupies the least memory, while the ResNet model occupies the most training memory and consumes the most training time. But as the most time-consuming model, ResNets still uses much less time than the model proposed in the 2D space segmentation problem setting. Therefore, the proposed or applied models under the 1D time series classification setting are superior.

In addition, we investigated all the models in the 1D time series classification setting further by a feature importance analysis using gradients backpropagation. We observed that the 1D FCNN model has the smoothest feature importance map, which also reflects its high interpretability. Further analysis shows that the reason is that the well-learned weights of each FC layer can focus on most of the obvious features. In addition, we found that the convolutional SCT attention model has a feature importance map similar to the 1D FCNN model, in which the importance value of some indistinguishable features can reach 0. We studied this further by visualizing the self-attention

matrices obtained during training and testing. The results show that the self-attention mechanism performed across spatial, channel, and temporal dimensions can greatly help suppress indistinguishable feature vectors and focus on obvious features with high separability.

Finally, we provided an open-source software application for spectral classification analysis using these deep learning methods. The software is developed on the Jupyter Notebook platform with interactive runtime environments such as Mybinder and Google Colab for more efficient data exploration and analysis, as well as better reproducibility and reusability.

It should be noticed that our dataset is a very nice representative. All these models presented in this work can be easily applied to other spectra datasets generated by other diffraction techniques or spectroscopy techniques, since these spectra have similar characteristics, such as peak increasing, decreasing, broadening, and shifting. In addition, the proposed classification models can also be generally applied or easily extended to the classification of other common 1D time series.

Contributions The author of this PhD thesis is responsible for the following contributions presented in this chapter:

- Design and implementation of an end-to-end binned FCNN with automatically capturing weighting factors model.
- Proposing the convolutional SCT attention model in the 1D time series classification problem setting. The proposed self-attention architecture can model dependencies across the spatial, channel, and temporal dimensions to selectively focus on a few classification-relevant features or observations.
- Introducing and implementing several other end-to-end models for time-resolved spectra classification, which are based on FCNN, convolution, LSTM, and Transformer.
- Evaluation and comparison of the classification performance of these DL models on spectral data from multiple perspectives.
- Interpretation of these classification models based on gradient feature importance analysis. Visualization of the self-attention matrix of the proposed convolutional SCT attention model in spatial, channel and temporal dimensions.
- Development and implementation of an interactive open-source software program tailored for spectral classification with better reusability and applicability, facilitating the exploration of spectral data and analysis scripts. Provide all of the trained models involved in this chapter, thus easing the burden on individuals to train models from scratch and increasing efficiency.

6

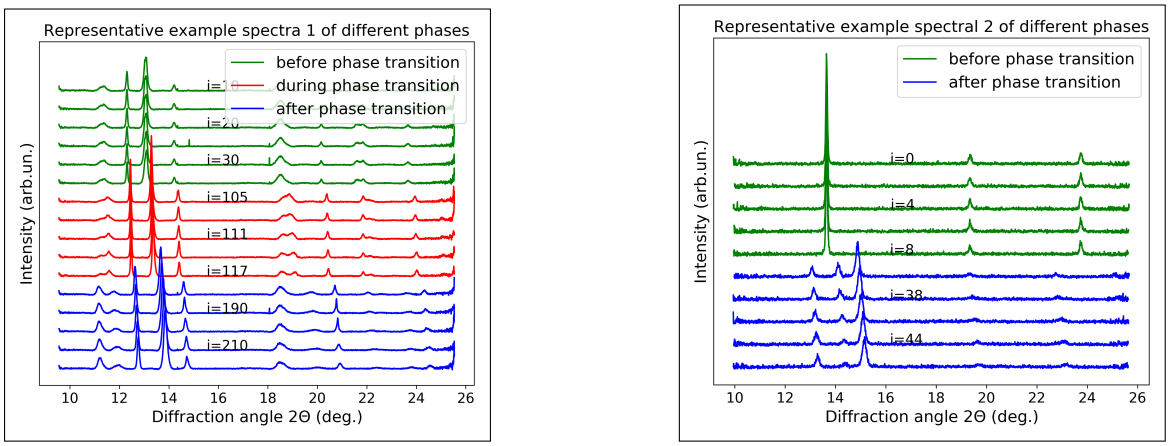
Interpretable Classification Framework for 1D Diffraction Spectra based on Supervised Contrastive Learning

The supervised end-to-end methods proposed in Chapter 5 work in a streamlined and efficient end-to-end manner, eliminating the need for data preprocessing or feature engineering, and making them powerful solutions for spectral classification. It is worth noting that the previous deep classification models are trained by minimizing the cross-entropy loss, a loss function that is widely used in deep learning models due to its efficiency and interpretability. However, there are also some limitations for models trained based on cross-entry loss. An important limitation is that it relies on accurate labels and is sensitive to noisy labels. Another limitation is that it may be prone to overfitting, especially when the dataset is small or unbalanced. To address this issue and to improve the generality of the model, a large number of simulated spectra were generated and applied to the supervised learning models introduced in the previous chapters.

Do we have a better way to solve this problem? What can we do to improve the performance of spectral classification models in terms of loss functions? In this chapter, we will explore an alternative approach to overcome the limitations of cross-entropy loss. We will introduce another loss function, namely the supervised contrast loss function, to address this problem in a different way.

In addition, it is important to note that the models presented in the previous chapters were primarily tested on a single spectra dataset. While this dataset is relatively representative, the training and validation of models based on only one dataset are limited in terms of data volume and diversity. In this chapter, we will perform experiments on multiple experimental spectra collected from samples Fe and FeO. These two samples exhibit different phase transition processes in the high-pressure experiments, reflected in the distribution of different diffraction peaks and, especially the abruptness of the phase transition. In these datasets, the abruptness of phase transition depends on several factors, such as the nature of the material, the rate of pressure change,

and the sensitivity of detectors. Fig. 6.1 shows some representative spectra of two example data sets from Fe and FeO, respectively. As can be seen from Fig. 6.1 a, for spectral curves for the 'before the phase transition' (class 0, marked in green), the peak positions have shifted and the peak intensity has changed, but these variances are not significant. The spectral curves 'during phase transition' (class 1, marked in red), are mainly manifested by the broadening and initial splitting or appearance of new peaks, while spectra for the 'after phase transition' (class 2, marked in blue) is mainly manifested by newly emerging peaks. However, in this high pressure experiment, the phase transition of the sample iron was abrupt (Fig. 6.1 b)or limited by the current time resolution and the experimental spectrum does not clearly show that it underwent a continuous phase transition. Therefore, for this type of data, we can only classify them into two categories, i.e., before- and after-phase transition processes. In general, the spectra can be classified into three categories according to the distribution of the spectral features, i.e., before, during, and after the phase transition. Furthermore, the spectral time series not only has a specific distribution along the diffraction angle dimension but also has a strong dependence on the temporal dimension.



(a) (b)

Figure 6.1: Some representative spectral curves in two example spectral datasets. a) Spectra collected from FeO with complex continuous phase transitions, which can be classified into before, during, and after phase transitions. b) Spectra collected from powder sample Fe, which have abrupt phase transitions, only need to be classified into two categories, e.g. before and after phase transition. In addition, i in both subplots corresponds to the relative acquisition time of each corresponding data set.

6.1 Supervised Contrastive Learning Framework for 1D Spectra Classification

In recent years, batch contrastive learning methods have been successfully applied to self-supervised representation learning for unsupervised training of deep images[12, 13] or time series models[19]. More recently, it has been extended to a fully supervised setting that can effectively exploit label information, namely, supervised contrastive learning. This extension has been shown to improve the generalization performance of the model and is more stable for hyperparameter settings[55]. Although it has consistently

outperformed cross-entropy in some research fields, such as image classification[55], and natural language processing [32], to our best knowledge, it has never been applied to spectral classification applications. In this method, we explore its application to the 1D spectral classification problem.

The supervised contrastive classification framework employs a two-stage training, which consists of a feature extraction backbone via supervised batch contrastive learning and a supervised single-layer linear classifier for downstream spectral classification, as shown in Figure 6.2. The first pretraining part is used to learn latent discriminative features from input 1D spectra to form useful representations in the embedding space, where the backbone encoder model applied in this approach is based on Conv SC attention model from [105], as shown in Figure 6.3. This backbone mainly consists of two convolution modules for extracting local features, and two self-attention modules performed across spatial (diffraction angle) and channel (introduced by previous convolutional channels) dimensions to build the long-range dependencies of spectra. Thus, latent dependencies and useful representations can be captured and used for classification tasks. To enable input data with different feature sizes, we apply the AdaptiveAvgPool1d function instead of the MaxPool1D operation in the second convolution module. See [105] for more details on this model. The second part is to learn a linear classifier that projects the embedded features to real-world spectral phase categories.

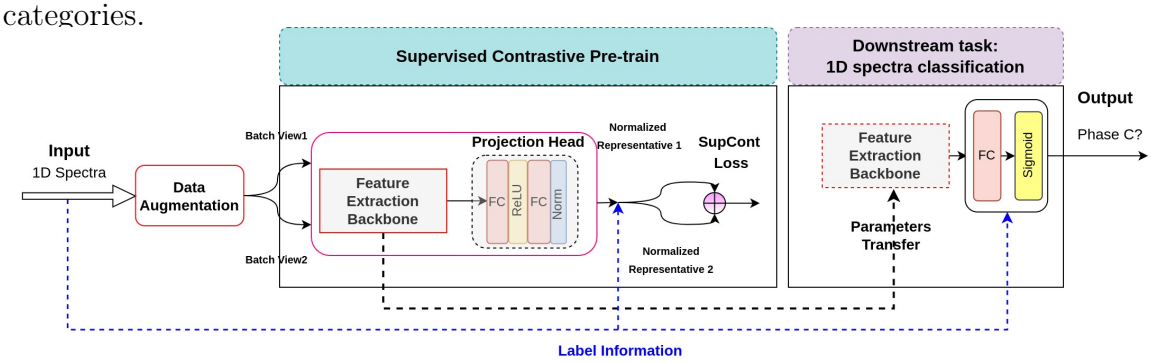


Figure 6.2: Illustration of the supervised contrastive deep learning classification framework for 1D spectra. It includes supervised pre-training and downstream spectra classification

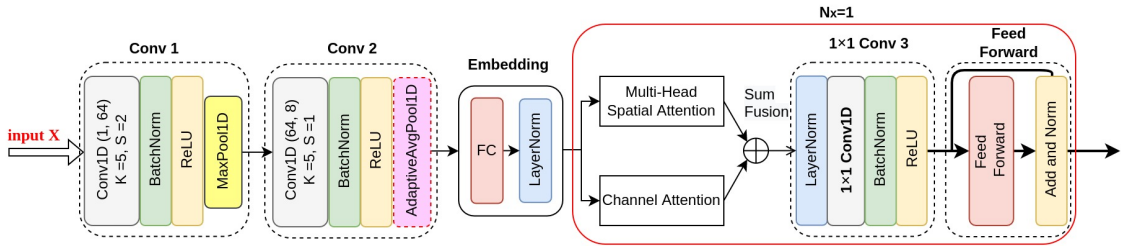


Figure 6.3: Backbone encoder in Supervised contrastive learning model.

Before training, we first apply K (here $K = 2$) augmentation $Aug(\cdot)$ to the labeled input spectra instances (x_i, y_i) to obtain multiple views of each anchor spectra, denoted by $(x_{i1}^A, \hat{y_{i1}})$ and $(x_{i2}^A, \hat{y_{i2}})$, where $\hat{y_{i1}} = \hat{y_{i2}} = y_i \in \{0 \dots C-1\}$ ($C = 3$, the number of classes in our application). Thus, given a batch of randomly sampled input 1D spectra of the batch size N, after data augmentation, the corresponding batch used for training consists of $2N$ pairs. The specific data augmentation techniques applied in this work will be introduced in the next section.

During the supervised contrastive training, the feature extraction backbone encoder $f_\theta(\cdot)$ is trained jointly with a projection head or contrastive head $p_\phi(\cdot)$. The encoder $f_\theta(\cdot)$ maps the augmented input spectra x_i^A to a representative vector r_i, $r_i = f_\theta(x_i^A) \in R^E$ ($R^E = 256$ in our implementation). The projection head applied here is Multi-Layer Perception (MLP), it projects the vector r_i to another vector z_i, $z_i = p_\phi(r_i) \in R^D$ ($R^D = 128$ in our implementation) in the projection space. Both the embedded features from the backbone encoder $f_\theta(\cdot)$ and MLP projection head $p_\phi(\cdot)$ are l2-normalized, which has been shown to speed up the training process [12, 55].

After that, the supervised contrastive loss is computed on the normalized outputs z_i of the projection network. In this method, in an augmented minibatch of size $2N$, all spectral samples from the same class are regarded as positive (similar) samples, while the rest are regarded as negatives (dissimilar samples), which can fully utilize the label information. For the random anchor spectra with index $l \in I \equiv \{0 \dots 2N-1\}$, whose class label is $\hat{y}_l$, we denote its corresponding set of positive samples as $Pos(l) \equiv \{p \in S(l), p_l = \hat{y}_l\}$. Then the loss can be expressed as:

$$L^{supcont} = \sum_{l \in A} L_l^{supcont} = \sum_{l \in A} \frac{-1}{|Pos(l)|} \sum_{p \in Pos(l)} \log \frac{exp((z_l \cdot z_p)/\tau)}{\sum_{a \in A(l)} exp((z_l \cdot z_a)/\tau)} \tag{6.1}$$

Here τ denotes an adjustable temperature parameter, $A(j) \equiv I\{l\}$, and $|Pos(j)|$ is the number of samples in the set $Pos(l)$. After pre-training, the backbone encoder can pull spectra samples belonging to the same class (positives) together while separating clusters of samples (negatives) from different classes. The two-layer projection network is discarded to get more general representations for the downstream spectra classification task. In the second stage, the pre-trained parameters in the feature extraction encoder are fully transferred to the downstream classification task in order to exploit useful representations for the classification task. Then a linear classifier is trained in a supervised manner using a cross-entropy loss based on the representation of the frozen backbone encoder. In our application, the linear classifier has 3 output neurons, which correspond to the 3 phases of the spectra data (before, during, and after the phase transition).

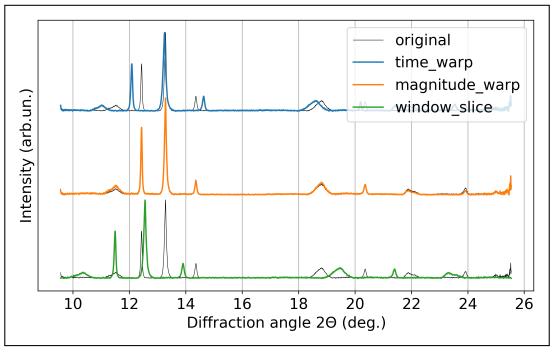
6.2 Experiments and results

6.2.1 Data Augmentation

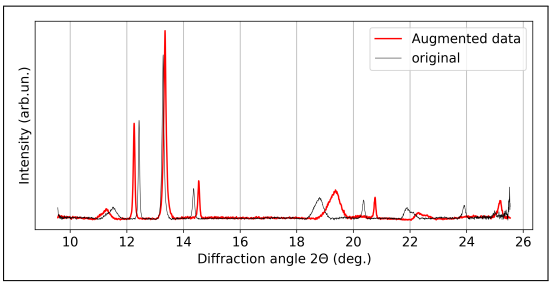
In contrastive learning, data augmentation plays a crucial role in defining the task of contrastive prediction[12], it can generate multiple views of the sample, resulting in a larger batch size, which can greatly improve the robustness and generalization ability of the model and prevent overfitting. The criterion for the selection of data augmentation techniques is to make the perturbations to the spectra large enough to capture other possible experimental complexity or noise, but not so large as to produce spectra data that is experimentally impossible.[112].

In this work, we first preprocessed data by normalizing its intensities to the $[0, 1]$ range, then we applied several data augmentation techniques, including Gaussian noise, time warping (performed in diffraction angle dimension), magnitude warping [115] and window slicing. In Figure 6.4, we give an example to illustrate the effect of each data

augmentation technique (excluding Gaussian noise) individually and the combination of these augmentations. In our implementation, to enlarge the training set, we first applied window slice operations on the spectral curve 20 times, each slice ranging in length from $0.85L$ to $0.96L$ (equidistantly spaced), where L is the original spectral length. Based on this, we apply time warping and amplitude warping subsequently by taking a random sample from a normal distribution with a probability of 20%. It should be noticed that the data augmentation techniques were chosen empirically, we donât dive deep into this aspect in this chapter.



(a) Applying data argumentation individually.



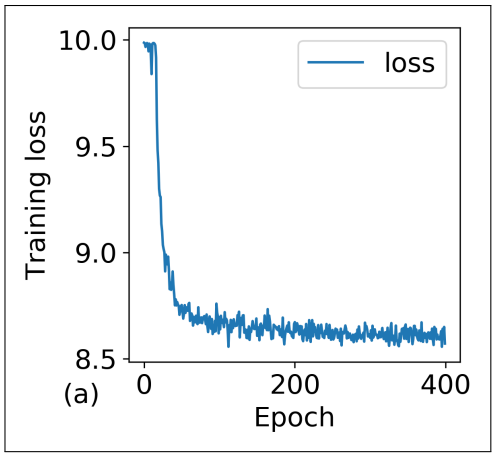
(b) Combination of Diffraction angle Warping and Magnitude Warping individually.

Figure 6.4: The effect of applying window slicing, magnitude warping, and diffraction angle warping data augmentation techniques to diffraction spectra. (a) Applying data augmentation individually. (b) The composition of these three methods of argumentation.

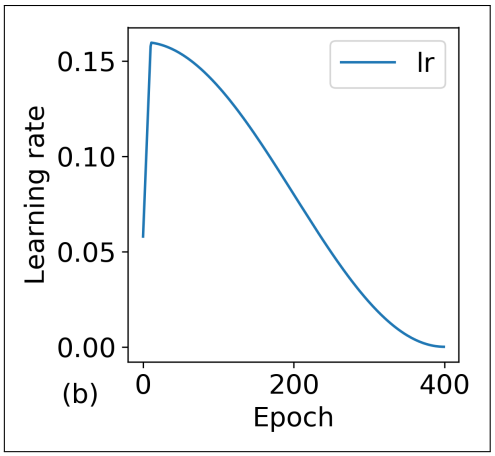
6.2.2 Implementation Details

The model is implemented on the Jupyter Notebook platform using PyTorch with a single NVIDIA A100-PCIE-40GB. Both the backbone encoder and the linear classifier are trained by minimizing loss functions (supervised contrastive loss for the backbone encoder and cross-entropy loss for the linear classifier) using gradient backpropagation, with the commonly applied Stochastic Gradient Descent (SGD) optimizer [7]. In this work, the representative spectra of each phase from four datasets were selected as the basis of the training/validation/test dataset, with a total of 286 original spectral curves. Combined with data augmentation, the total number of spectral curves involved in building the model is 6006 (5800 simulated spectra with data augmentation). The batch size is set to 1024 throughout the process. In the contrastive training stage, the initial learning rate of the optimizer is set to 0.16, with a linear warm-up for the first 10 epochs and cosine decay schedule [67], and weight decay is set to 1×10^{-4}. The pretraining epoch is set to 400, and the distribution of training loss and learning rate during this process is shown in Figure 6.5. As can be seen from this figure, the loss drops rapidly in the first 20 steps and then slowly reaches a plateau.

As mentioned earlier, in the second stage, when training the linear classifier layer for the classification task, the parameters in the backbone encoder are frozen, which means that only a small number of parameters need to be trained in this process, and the probability of model overfitting can be effectively reduced. But to be on the safe side, a train/validation/test strategy and an early stopping strategy are still employed to train



(a) Applying data argumentation individually.



(b) Combination of DiffAngle_Warp and Mag_Warp individually.

Figure 6.5: Training loss (a) and the learning rate (b) at different epochs during the supervised contrastive learning stage.

the linear classifier. We applied a stratified random split to the imbalanced data with a train/validation/test split ratio of 80%/10%/10%. The best model on the validation dataset is used for testing. The chosen idea is that the training stops early when the validation classification accuracy does not increase for a certain number of steps (we set $K = 80$ in this work). In this linear evaluation stage, the cross-entropy loss function is applied, and Adam [56] has the same initial learning rate and weight decay parameters as in the previous stage. The distribution of loss value and accuracy rate in this process are shown in Figure 6.6, respectively. As can be seen from this figure, as few as less than 10 epochs of additional training are required during this stage (According to the early stopping strategy, the training process stops at epoch 86, which means that the parameters of the linear classifier are saved at step 4 (ie $84 - 80$)). Usually, after one-epoch training, the model has reached a high classification accuracy (90.33%), and after multi-step training, the classification loss and accuracy have reached a stable optimal value. This illustrates the high performance of the encoder network, indicating that useful representations have been extracted from the spectral data by the backbone encoder.

6.2.3 Quantitative Model Evaluation

We tested the performance of the model on 9 spectral datasets, including the other five datasets that did not participate in the train/validation/test datasets used to build the classification model. We evaluate the quality of the backbone encoder $f_\theta(\cdot)$ by visualizing the representations learned of it, and test it on a downstream classification task. Figure 6.7 visualizes the learned representations or embedding vectors in the latent space learned by the encoder from two example spectral datasets. From this result, we get that distinguishable patterns are learned in the latent space. Figure 6.8 further shows an example of the t-SNE[116] visualization of learned representations of the base encoder $f_\theta(\cdot)$ with and without the projection network $p_\phi(\cdot)$, respectively. In this figure, we use classes 0, 1, and 2 to represent before, during, and after phase transition, and the class labels are predicted by the classification model. From the

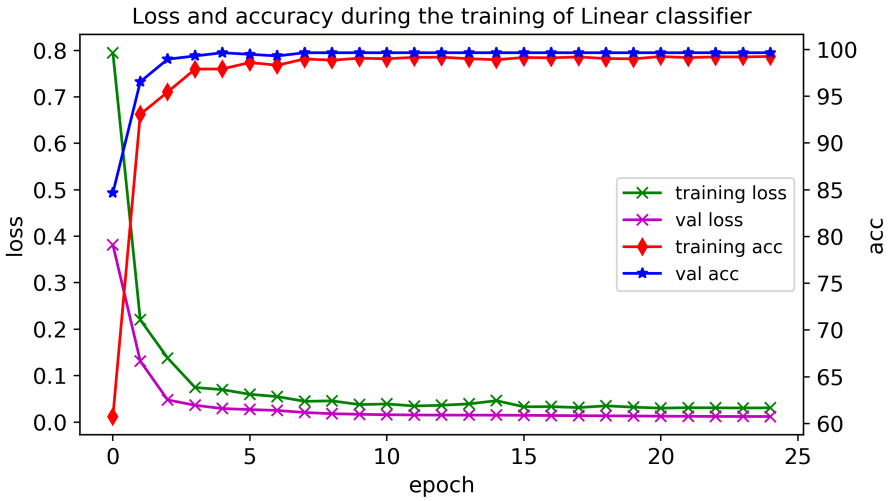
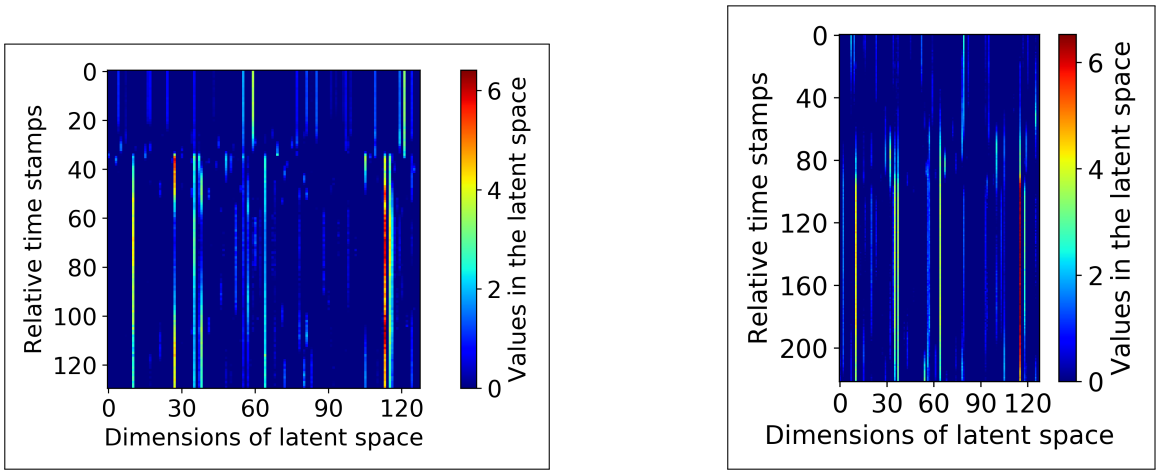


Figure 6.6: Training/validation loss and accuracy at different epochs during training of a linear classifier with cross-entropy loss function.

t-SNE visualization of the feature representation results, we can get that clusters of spectral samples belonging to the same class are pulled together in the embedding space, while clusters of samples from different classes are pushed apart. At the same time, we can get that the classes represented by the hidden vectors before the projection head $p_\phi(\cdot)$ are better separated compared to the vectors after it because $p_\phi(\cdot)$ participates in supervised contrastive learning.

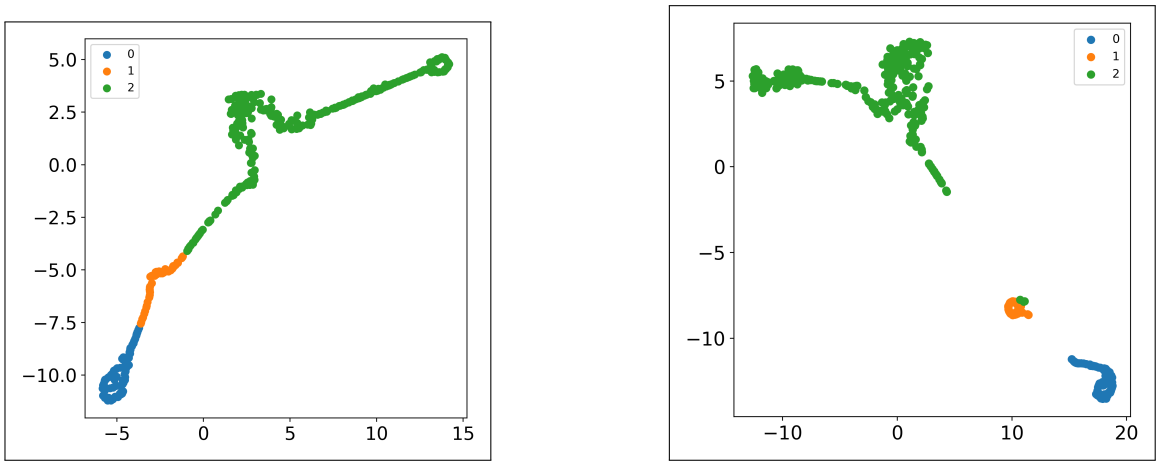


(a) (b)

Figure 6.7: Visualization of representations of (a) the D4 dataset and (b) D8 in the latent space learned from the backbone encoder.

Figure 6.9 shows the classification results on 3 of the datasets. For simplicity, we only show the result around the phase transition zone, as it can be seen that there is a clear classification boundary between different classes in these test datasets, which can capture the phase transition well. The results demonstrate the high quality of the learned representations of the backbone encoder. Among them, the green line represents phase θ (before phase transition), the red line represents phase 1 (during phase transition), and the blue line represents phase 2 (after phase transition). Note that i represents the time sequence of the number of spectral samples in each dataset,

and p represents the probability of the predicted class label. In addition, we list the classification accuracy and their average results for each dataset in Table 6.1. As can be seen from the results, although the model is trained on a relatively small dataset (using only 286 original spectra), it achieves good generalization ability due to effective training strategies and data augmentation techniques. Furthermore, the probability p provides experts with classification confidence in the predicted class label and guides them in further analysis. And it can be easily seen that the predicted class probabilities around the phase transition region have lower values. Another benefit is that a single prediction only takes about 20 μs on an NVIDIA Tesla A100-PCIE-40G, which makes it possible to provide quick feedback on the status of the experiment under evaluation.



(a) (b)

Figure 6.8: t-SNE visualization of the embedding vectors in two different latent spaces on the D9 dataset. (a) Learned representations after the MLP projection head. (b) Learned representations before the MPL projection head.

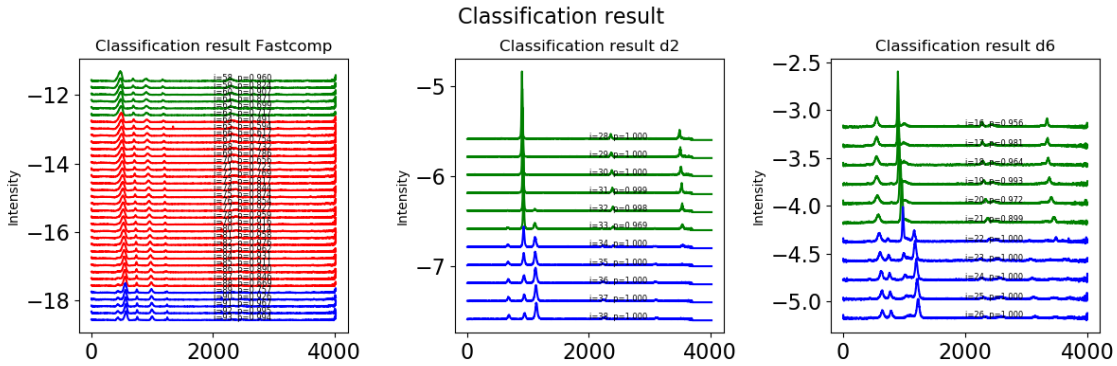


Figure 6.9: Classification results of the supervised contrastive model on 3 example experimental spectral datasets.

6.2.4 Comparision with the Supervised Model Based on Cross-Entropy Loss

In this experiment, we further compare the performance of a spectral classification model trained with a supervised contrast loss function and a model trained with a

cross-entropy loss function. To ensure a fair and unbiased comparison, we maintained consistency in the model architecture, hyperparameters, and training/validation/test datasets. The only difference introduced was the choice of the loss function. The comparison results are presented in this Table 6.1. The results show that the model trained with supervised contrastive loss exhibits better generalization than the one trained with the cross-entropy loss. However, it is worth noting that during the training process, we also observed that the model trained with the cross-entropy loss function converged faster than the model trained with supervised contrastive loss. This observation can be attributed to the simplicity of the cross-entropy loss function, which allows for efficient optimization.

Loss type	D1	D2	D3	D4	D5	D6	D7	D8	D9	**Fe**	**FeO**	**Average**
Supervised contrastive	1.0	1.0	1.0	0.99	1.0	0.99	1.0	0.95	0.97	0.997	0.960	0.989
Cross-entropy	1.0	1.0	1.0	1.0	1.0	0.84	0.96	0.98	0.94	0.972	0.962	0.970

Table 6.1: Comparison of spectral classification results trained with supervised contrast loss function and cross-entropy loss function.

6.3 More Interpretation of Classification Model

It is usually difficult for deep neural networks to afford insights toward interpreting mechanisms since they introduce the complexity of interactions and nonlinearities[122], and as such, they are considered black boxes. In this work, despite our good classification performance, the physical information underlying the classifications of interest to physicists has not been clearly revealed. It's more straightforward for physicists to analyze the experimental XRD samples according to the descriptors of their XRD patterns such as peak positions, width, intensities, and shapes[122]. Thus, it is crucial and meaningful to interpret the deep learning classification model from these physical aspects. Hence, in the next section, we interpret the classification model in terms of multiple underlying physical information, such as the number of peaks, peak positions, widths, and even the peak regions.

6.3.1 Prediction of the Number of Peaks

In this study, the appearance of new peaks is the most important signal for evaluating the new phase of the spectra. we design a simple network to check whether this information can be extracted from the learned representations. Specifically, a linear layer with P_n output neurons, followed by Softmax activation function, is applied to predict the number of spectral peaks. The model is built on the latent vectors from the frozen backbone encoder, as shown in Figure 6.10. P_n represents the maximum number of possible peaks in the tested datasets. we tested two of the experimental datasets and set P_n to 13. In this model, a linear classifier is trained using the Adam optimizer with a learning rate of 1×10^{-2} and a weight decay of 1×10^{-4}, and the model is evaluated by a cross-entropy loss function.

The results show that the classification accuracy reaches 95% in 2 training epochs. The high accuracy of the prediction results and the fast convergence of the model indicate that the learned representations of the backbone encoder are of high quality. We thus get an important conclusion, although only spectral label information without any peak information was used during encoder training. It turns out that the network can automatically retrieve physical information (number of peaks) for spectral classification tasks.

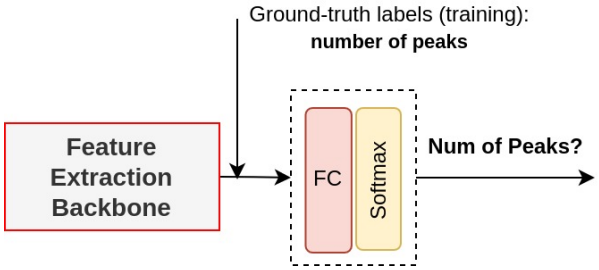
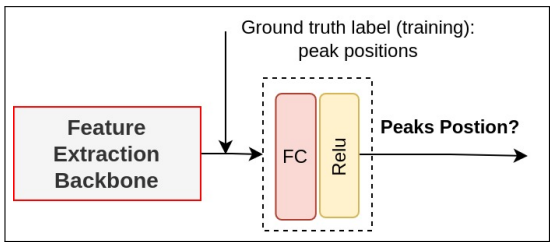


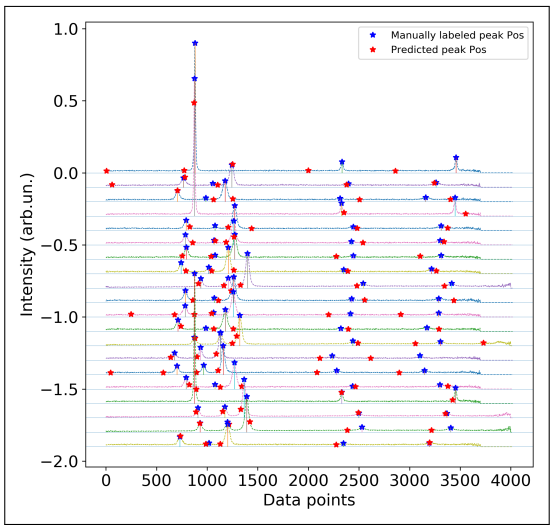
Figure 6.10: Linear classifier for predicting the number of peaks.

6.3.2 Prediction of the Peak Position

The peak position is also a key description of the spectral pattern. In our experiments, it is important to capture the peak position to evaluate the experimental state because the pressure applied to the sample is constantly changing, resulting in peak shifts throughout the process. In this subsection, we conduct a similar experiment on the same experimental datasets as the previous one to extract peak locations.



(a) Model structure of predicting peak position.



(b) Peak positions predicted by the model.

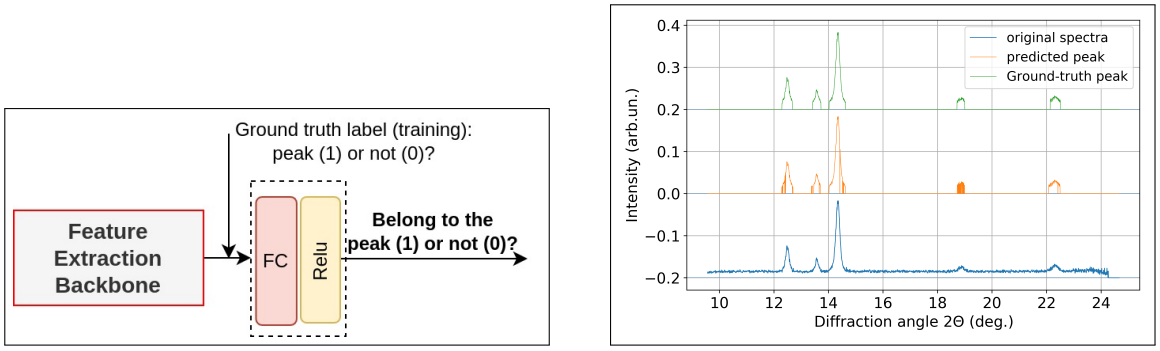
Figure 6.11: Prediction of peak position.

Similar to the neural network that predicts the number of peaks, a linear layer with P_n=13 output neurons followed by the Relu activation function is applied to predict the peak position, as shown in Figure 6.11 (a). In this model, it is formulated as a regression problem. The model is trained using the Adam optimizer with a learning rate of 1×10^{-3} and a weight decay of 1×10^{-4}. During this process, the MSE (Mean Squared Error) loss function is applied to evaluate the model. We show the prediction

result after 800 training epochs in Figure 6.11 (b). The results show that while the exact location of the peak canât be predicted, the model still tracks the approximate location of the peaks.

6.3.3 Prediction of Peaks' Region

Here, we trained a linear layer with N output neurons to reconstruct the peak region. Similarly, the linear layer is trained based on the representations from the backbone encoder. For each data point in the spectral curve, a value of 0 means not belonging to the peak, and 1 means in the peak region. It is formulated as a regression problem. During training, 15000 epochs are used, and an Adam optimizer with a learning rate of 1×10^{-3} and a weight decay of 1×10^{-4} is applied to train the linear classifier. The MSE loss function is used to evaluate the difference between the predicted output and ground-truth peak information. When the predicted value is greater than 0.5, the point is considered to belong to the peak region.



(a) Model structure of predicting peak region.

(b) Peak region predicted by the model.

Figure 6.12: Prediction of peak region.

The predicted peak regions, shown in Figure 6.12, are in good agreement with the reference value. Its ability to reconstruct peak or peak regions shows that the peak information is properly extracted. This means that the most important and classification-relevant features, e.g. the number of peaks, peak position (and the shift of peak positions), peak width, and the intensity of peaks are addressed in the encoder network.

6.4 Summary and Future work

In this chapter, we demonstrate a supervised contrastive learning method for the application of spectral classification. The deep learning classification framework consists of two training stages, e.g. supervised pre-training and linear evaluation. In the first stage, supervised contrastive learning with data augmentation is introduced to learn discriminative representations of spectra, and then in the second stage, the pre-trained encoder model is fully transferred to the specific downstream spectra classification task. The proposed method based on supervised contrastive learning is proven effective on experimental spectral datasets, clearly detecting phase transitions from each dataset, and providing very fast feedback (2×10^{-5} seconds per prediction). Furthermore, the

results on four unseen spectral datasets show that the classification model has good generalization and robustness.

In addition, we interpret the classification model in terms of traditional descriptors of spectral patterns that are straightforward to physicists, such as peak position, width, intensity, and even peak regions. It should be noted that the peak information is not included in training the spectral classification model, but obtained by training a simple linear layer neural network on top of the learned representations from the frozen encoder network. This means that the network can automatically retrieve physically meaningful peak information for spectral classification tasks.

Contributions The author of this PhD thesis is responsible for the following contributions presented in this chapter:

- Introduction of the supervised contrastive learning framework with data augmentation for 1D spectral representation and classification.
- Interpretation of the classification model from multiple perspectives in terms of traditional descriptors of spectral patterns of interest to physicists, such as peak number, peak position, width, and intensity.
- Evaluation of the effectiveness of the network on multiple experimental spectral datasets, including some unseen datasets, demonstrates the high quality and generalizability of the model.

Part III

Self-Supervised Approaches to Spectra Classification

7

Self-Supervised Classification models

With the aim of developing efficient and automated classification methods for spectral classification applications, we proposed a range of machine learning and deep learning methods in previous chapters, including traditional unsupervised learning, supervised deep learning techniques, and end-to-end supervised solutions. In addition, we introduce supervised contrastive learning to learn more general features from the data to further facilitate spectral classification. However, as previously mentioned, traditional clustering methods pose challenges when it comes to parameter tuning, particularly in determining the optimal number of clusters. On the other hand, supervised learning approaches, including the end-to-end solutions proposed in Chapter 5, require manual labeling, which hampers the automation process.

Recognizing these challenges, it is essential to strike a balance and introduce a more effective approach that overcomes the limitations of both traditional unsupervised machine learning and supervised deep learning methods. Recently, methods based on self-supervised learning have opened up a new research frontier[77]. These are based on data augmentation and appropriate pretext tasks, through which deep neural networks can learn generalizable features from unlabeled data. Self-supervised methods impose invariance to appropriate pretext tasks, which are solved with the aim of learning a good representation of the data and designed according to specific problems [12, 13, 38, 23]. While self-supervised learning requires domain-specific knowledge, the need for human supervision is largely reduced with respect to supervised learning, and the potential for automation is increased.

In this thesis point, we demonstrate that self-supervised machine learning methods can provide great opportunities to improve the scientific efficiency of experiments at large-scale X-ray facilities. We explore the application of self-supervised relational reasoning and contrastive learning to 1D spectral classification problems. In particular, we demonstrate that it can be effectively used to detect phase transitions observed in X-ray diffraction (XRD) experiments[80, 135, 57]. We introduce and discuss three self-supervised representation learning frameworks for the classification of data, namely SpecRR-Net, SpecMoco-Net, and SpecRRMoco-Net [108]. SpecRR-Net extracts discriminative features from unlabeled spectra based on relational reasoning, which at-

tempts to discover data representations by reasoning the relation among entities[23, 82] in multiple dimensions and at different scales. SpecMoco-Net is based on contrastive learning, which aims to build representations by learning similarities and dissimilarities between different objects[12, 13]. SpecRRMoco-Net benefits from both relational reasoning and contrastive learning, unifying SpecRR-Net and SpecMoco-Net. Furthermore, we discuss in detail the selection of data augmentation techniques [108], crucial to ensure that scientifically meaningful information is retained. Details of these three frameworks and applied data augmentation techniques are presented in Section 7.1. Experiment studies, the ablation study on the proposed loss function of SpecRRMoco-Net, and the ablation study on the data augmentation are shown in Section 7.2. Furthermore, a discussion of these models is provided in Section 7.3. Finally, the conclusion and main contributions are given in Section 7.4.

7.1 Self-Supervised Classification Approaches

7.1.1 Problem Definition

Given unlabeled data containing a series of spectral curves $\{x_i\}$, we aim to learn a parametrized map $f_\theta(\cdot)$, which can produce a rich and descriptive representation $zi = f_\theta(x_i)$ from unlabeled spectra for the downstream classification task. In this equation, θ are the learnable parameters of neural networks. The learned representations will be then used for downstream spectral classification tasks while using a minimal number of labels.

7.1.2 Data Augmentation

Data augmentation, which provides different views of the input data expected to be mapped to similar representation vectors, is critical in defining useful pretext tasks31 in self-supervised learning. Such augmentations produce varied spectra, possibly with simulated additional experimental complexity or noise, but still plausible and with the same target labels. The objective function, therefore, ensures that same-label variations of the input spectra must be represented similarly. Such a procedure increases the robustness and generalization capabilities of the model, as variations of the input dataset are also used to train the model.

In this work, we first preprocessed the spectra data by normalizing them to the [0, 1] range, then we sequentially applied diffraction angle warping (which is adapted from time warping[115] changing its original time dimension to the diffraction angle dimension), and magnitude warping[115] as data augmentations. Magnitude warping is used to simulate reasonable and random variations in the intensities of peaks, while not changing their positions. Diffraction angle warping is used to parallel the variation of peak positions, so to allow the model to focus more on the number of peaks rather than their location. An example of the effect of applying the augmentations is shown in Figure 7.1. It is important to note that both data augmentations are physically meaningful and specific to the case study. In fact, neither data augmentation results in changes in the number or shape of the peaks, which are the primary indicators for detecting a phase transition.

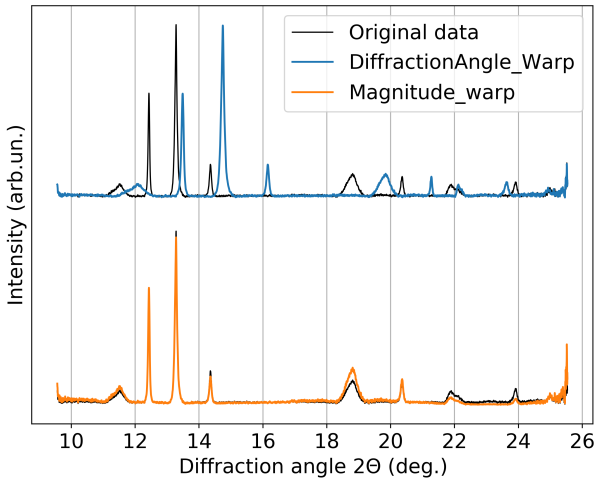


Figure 7.1: Effects of applying magnitude warping and diffraction angle warping data augmentations to diffraction spectra.

7.1.3 Self-Supervised Pre-training and Linear Evaluation on Downstream 1D Spectra Classification Task

The self-supervised classification framework adopts a two-stage training, i.e., a pre-training stage and a linear evaluation stage, as shown in Fig. 7.2. In the pre-training stage, the feature extraction backbone encoder is trained in an unsupervised manner through momentum contrastive learning and relational reasoning-based learning. During this stage, the backbone encoder projects input data into a latent space z, which provides another representation of the data. As part of the training, a transformation $g(\cdot)$ is applied on the vector z, to obtain the output used in the loss function (defined below). Such transformation is referred to as 'contrast head' or 'relational reasoning head'. The objective of this pre-training is to learn useful representations from the unlabeled spectra under the supervision of self-supervised pretext tasks, thus reducing the amount of label information needed for downstream classification tasks. After the first stage training, the contrastive head and relational reasoning heads are discarded to reduce the correlation between output variables, as it has been suggested in related self-supervised learning research, such as in Ref.[12, 13, 23], the backbone parameters are completely transferred to the second part for downstream classification tasks. In the linear evaluation stage, the feature extractor is frozen, and a single-layer linear classifier is trained using a reduced amount of labeled data, projecting the learned representations in the latent space to physically meaningful spectral phase classes.

The shared feature extraction backbone model f_q applied in this approach is the Conv SC attention model from Ref.[105], but without a feed-forward network, as shown in Figure 7.3. It consists of two convolution modules for extracting local features, and two self-attention modules performed across spatial (diffraction angle) and channel (introduced by the convolutional channels) dimensions to build long-range dependencies of spectra. In this way, latent dependencies and useful representations can be well captured. Furthermore, to accept input data with different feature sizes, we apply the 1D adaptive average pooling instead of the 1D global max pooling operation in the second convolution module, as shown in Figure 7.3). In this work, four pretext tasks are proposed to supervise the training of the backbone encoder in the first stage. These include three relational reasoning-based pretext tasks, i.e., an inter-sample relational reasoning module, an intra-sample relational reasoning module, an external-variable

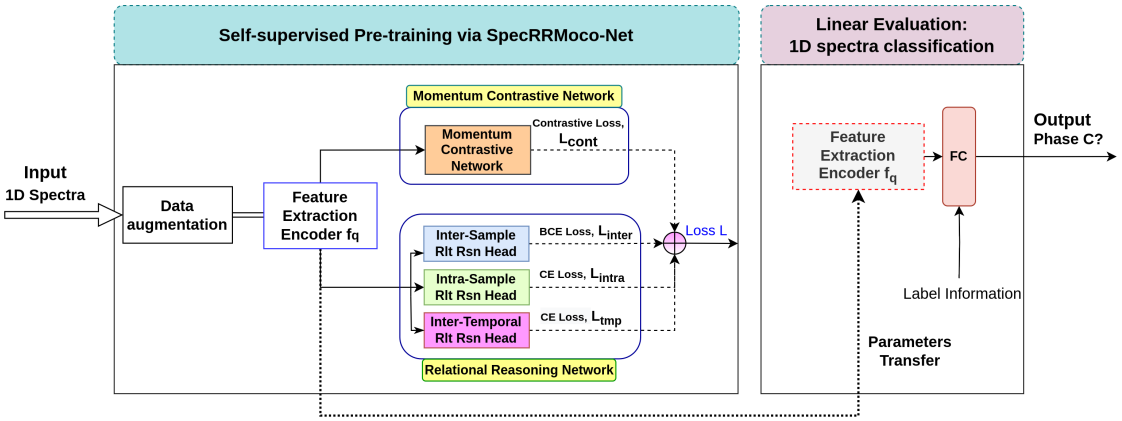


Figure 7.2: Illustration of the proposed 1D spectra classification framework based on the self-supervised SpecRRMoco-Net, which is a combination of Relational Reasoning Network (SpecRR-Net) and Momentum Contrast Network (SpecMoco-Net). The classification framework consists of two parts, namely pre-training and linear evaluation of downstream spectral classification. In the pre-training stage, the encoder f_q is trained on unlabeled data to build useful representations; in the linear evaluation stage, a a small number of labels are used to perform the downstream spectral classification task, where a linear classifier is trained on top of the frozen feature extraction encoder f_q. Specifically, the encoder is training by jointly minimizing the contrastive loss L_{cont} in SpecMoco-Net, inter-sample relational reasoning loss L_{inter}, the intra-sample relational reasoning loss L_{intra}, and the inter-temporal relational reasoning loss L_{tmp} in SpecRR-Net.

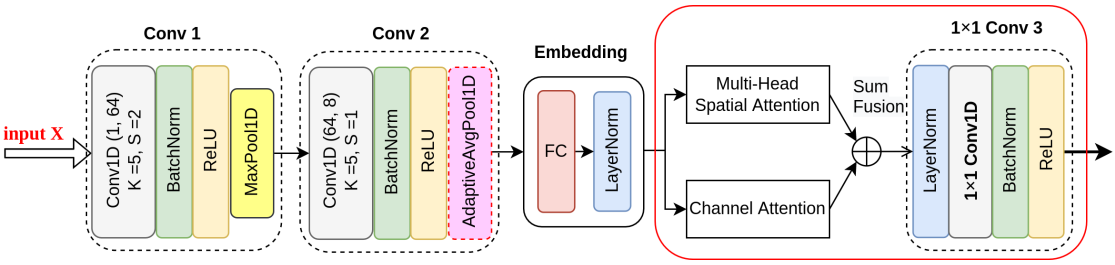


Figure 7.3: Backbone encoder network based on Conv SC attention model.

relational reasoning module, and one pretext task based on instance-level contrastive learning, as shown in Figure 7.2. We name the self-supervised classification framework based only on three relational reasoning modules as SpecRR-Net, the network based only on the contrastive module as SpecMoco-Net, and the combination of these two networks as SpecRRMoco-Net. We will describe each module in detail in the following sections.

Inter-Sample Relational Reasoning. The Inter-Sample relational reasoning module [82, 23] learns to quantify the relationships of the sampled pairs (how spectral instances are related to themselves and other instances), by formulating it as a binary classification pretext task, as shown in the upper branch of Figure 7.4.

Formally, given any spectral curve x_i, we apply K random augmentations to it to generate an augmented set $A(x_i) = \{x_i^{(k)}\}_{k=0}^{K-1}$, where $x_i^{(k)}$ is the k-th augmentation of x_i. After the forward pass of the backbone encoder $f_q(\cdot)$, the encoded representation $z_i^{(k)} = f_q(x_i^{(k)})$ can be extracted. On this basis, a positive relational representation pair $[z_i^{k1}, z_i^{k2}]$ is constructed by aggregating together representations of different views of the

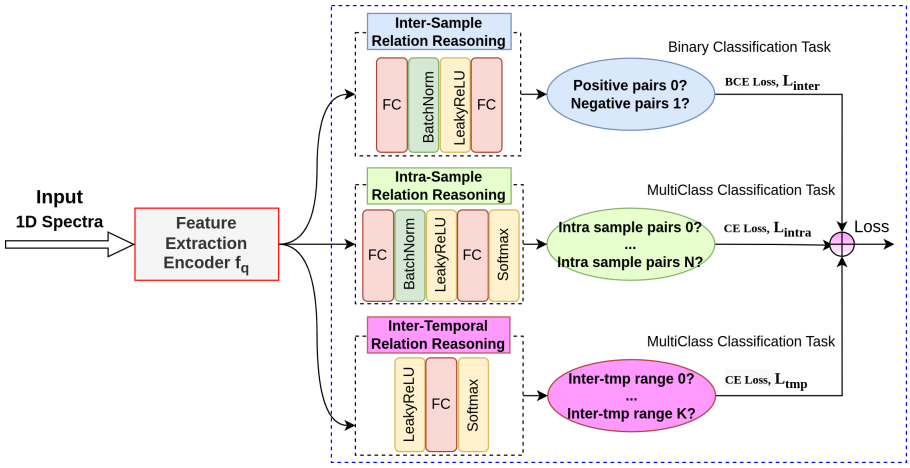


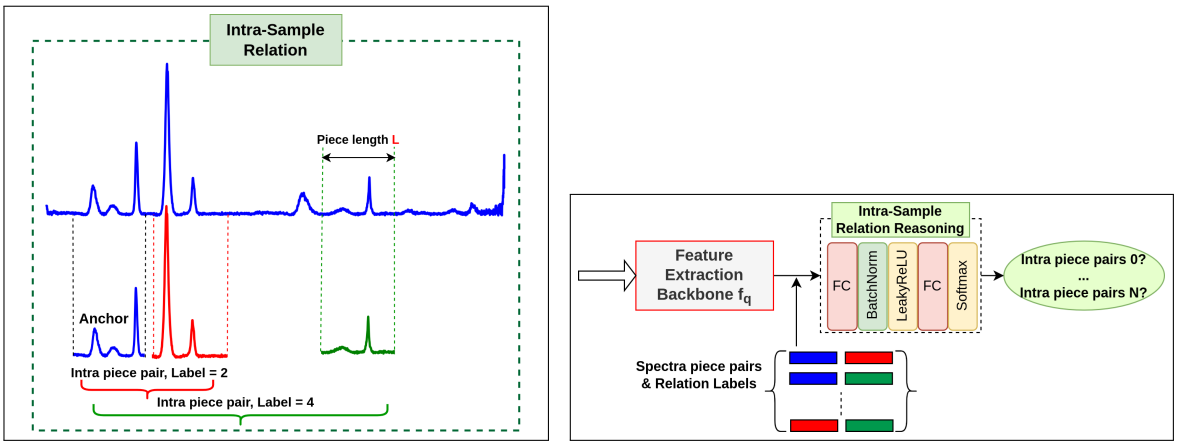
Figure 7.4: Illustration of the self-supervised relational reasoning sub-network (SpecRR-Net). It mainly consists of inter-sample, intra-sample, and external-variable relational reasoning modules. In this cartoon, L_{inter} represents inter-sample relational reasoning loss, L_{intra} represents intra-sample relational reasoning loss, L_{tmp} represents the inter-temporal relational reasoning loss.

same spectra, while a negative relational pair $[z_i^{k1}, z_j^{k2}]$ is sampled from representations of two randomly paired different spectra, where $[\cdot]$ denotes the vector concatenation operation. In this module, we can benefit from a large number of relational reasoning pairs. According to the combination strategy in Ref. [82], within a minibatch of size B, the total number of pairs $P = B(K^2 - K)$. Finally, the inter-sample relation reasoning head $r_\alpha(\cdot)$ is applied to the relation representation pairs to estimate their relation scores. Given a representation pair $[z_i, z_j]$, a relation score $s_i^{inter} = r_\alpha([z_i, z_j])$ can be predicted, whose corresponding target y_i^{inter} is equal to 1 for a positive relation pair and 0 for a negative relation pair. This pretext task is trained by minimizing the binary cross-entropy loss L_{Inter}

$$L_{Inter} = -\frac{1}{P}\sum_{i=0}^{P-1} y_i^{inter} \cdot \log(s_i^{inter}) + (1 - y_i^{inter}) \cdot \log(1 - s_i^{inter}) \tag{7.1}$$

Intra-Sample Relational Reasoning. The Intra-Sample relational reasoning module [23] models the relation between different spectral pieces within each individual diffractogram. It is adapted from the intra-temporal relational reasoning module in Ref. [23], originally proposed to model the global temporal dependencies of time series data. Here, we extend it to the diffraction angle dimension.

Formally, given any augmented spectral curve x_i^A with N data points, we randomly sample two spectral segments of length L, one starting from diffraction angle $p1$ and the other starting from diffraction angle $p2$, denoted $x_{i,1}$ and $x_{i,2}$, as shown in Figure 7.5(a). After propagating through $f_\theta(\cdot)$, their corresponding representations can be extracted. These are denoted as $z_{i,t1} = f_q(x_{i,1})$ and $z_{i,t2} = f_q(x_{i,2})$. Based on this, the intra-relational representation pair is constructed as $[z_{i,t1}, z_{i,t2}]$ and input to the intra-sample relation reasoning head $r_\beta(\cdot)$ to reason their spatial relation score $s_{i,c} = r_\beta([z_{i,t1}, z_{i,t2}])$, where c is the class index in the range $[0, C)$ (C is the number of classes). This process is illustrated in Figure 7.5(b). The intra-sample relationship between these two spectral pieces is based on their spatial distance along the diffraction angle dimension, defined as $d_{p1,p2} = |p1 - p2|$, which is based on the absolute distance between their starting



(a) Sampled intra-sample spectral piece pairs. (b) Intra-Sample relationship and the relation head network.

Figure 7.5: Illustration of the intra-sample relational reasoning module.

points. We define the number of intra-sample relation types $C = 5$, the spectral pieces length $L = 0.2N$, and the distance threshold $D = \frac{L}{2}$. Then the target intra-sample relation label y_i^{intra} is assigned by the equation $y_i^{intra} = \min(floor(\frac{d_{p1,p2}}{D}), C-1)$.

We formulate the intra-sample relational reasoning module as a multi-class classification task trained with a cross-entropy loss L_{Intra}, defined as follows:

$$L_{intra} = -\frac{1}{B}\sum_{i=0}^{B-1}\log\frac{\exp\left(s_{i,y_i^{intra}}\right)}{\sum_{c=0}^{C-1} exp(s_{i,c})} \tag{7.2}$$

The hyperparameters are consistent with those of the module in Ref. [23]. In this way, the underlying dependencies along the diffraction angle dimension can be captured.

Inter-Temporal Relational Reasoning. Although the above two relational reasoning modules can learn latent discriminative features from sampled pairs, they do not properly utilize the temporal information of spectra. The external variable related to the time could be any external condition in the experiment, e.g., temperature, pressure, or electric field. In our specific use case, this is pressure, which varies with time during compression and decompression. Guided by this, we designed and introduced a third relational reasoning branch (see the third branch of Fig. 7.4), that is the inter-temporal relational reasoning module, to build robust external dependencies from the spectral samples. This can further enable the backbone to learn useful patterns along the temporal dimension.

Formally, given any spectral curve x_i collected at time step i, the encoded representation of its augmented version x_i^A is denoted by $z_i = f_\theta(x_i^A)$. A single layer external-variable relation projection head $r_\gamma(\cdot)$ is applied to reason the inter-temporal relation score, denoted as $s_i^{tmp} = r_\gamma(z_i)$. First, the spectral curves are evenly divided into $T = 5$ inter-temporal relation categories in order of acquisition time. Then, a multi-class classification pretext task is constructed and trained with the cross-entropy loss L_{tmp} as

$$L_{tmp} = -\frac{1}{B}\sum_{i=0}^{B-1}\log\frac{\exp\left(s_{i,y_i^{tmp}}^{tmp}\right)}{\sum_{t=0}^{T-1} exp(s_{i,t}^{tmp})} \tag{7.3}$$

where y_i^{tmp} is the ground-truth class label in this module. The ablation study of hyperparameters T is left to future work. The relational scores in these three relational reasoning modules are not fixed distance metrics but rather a learnable nonlinear similarity measure for comparing the similarity of encoded spectra data.

Self-Supervised Contrastive Learning Module for 1D Spectra Classification. In the self-Supervised contrastive learning module, SpecMoco-Net, instance-wise contrastive learning [128, 12] is employed, where each spectra instance is treated as a distinct class of its own and a pretext classifier is trained to distinguish between individual instances[12]. SpecMoco-Net is based on momentum contrastive learning (MoCo)[38]. In this network, InfoNCE is used as the contrastive similarity metric. As can be seen from Figure 7.6, there are two structurally identical encoder networks in SpecMoco-Net, i.e., the encoder f_q followed by the contrastive projection head r_q and the momentum encoder f_k followed by the projection head r_k. Since self-supervised contrastive learning usually needs a large set of negatives to build useful representations, following the strategy in Ref. [38], in SpecMoco-Net we apply a queue-update dictionary to maintain a dictionary of keys for querying. As in [38], the queue of keys is updated through enqueue and dequeue operations (Figure 7.6), and the capacity of the dictionary queue, Q, is larger than the size of a batch of keys. In addition, to speed up the training, the output vector of the contrastive head is normalized by its L2-norm [12, 38].

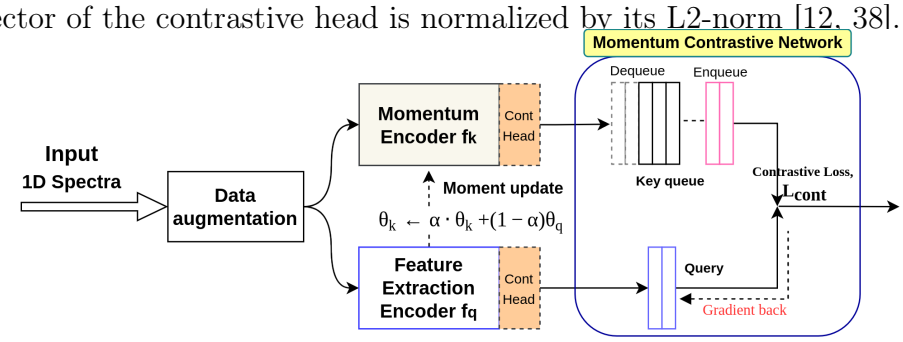


Figure 7.6: Illustration of SpecMoco-Net.

Formally, given the current mini-batch of size N, its corresponding augmented batch size is $2N$, for a random input x_i, we generate two different views of it with random data augmentations, denoted as $x_i^{(1)}$ and $x_i^{(2)}$, which are encoded by two different encoders described earlier, then through a forward pass, the query $h_i^q = r_q(f_q(x_i^{(1)}))$ and key $h_i^k = r_k(f_k(x_i^{(2)}))$ can be constructed. Given $x_i^{(1)}$ as the anchor sample, its only positive (similar) sample is $x_i^{(2)}$ in the current mini-batch, and its corresponding encoded representation h_i^k is denoted as k_i^+, while the negatives (dissimilar) are sampled from the dynamic queue of keys, denoted as k_i^-, of size Q. The model is asked to distinguish the augmented version of the anchor spectra (different views of the same spectra) from the negative spectral curves collected at different times or from different datasets. Learning aims to retrieve the corresponding "key" for a specific query. This is formulated as minimizing an InfoNCE-based contrastive loss function[78] as follows:

$$L_{cont}^q = \sum_i L_{cont}^i = \sum_i -\log \frac{\exp\left(\frac{h_i^q k_i^+}{\tau}\right)}{\exp\left(\frac{h_i^q k_i^+}{\tau}\right) + \sum_{k^-} \exp\left(\frac{h_i^q k_i^-}{\tau}\right)} \tag{7.4}$$

Here τ denotes an adjustable temperature parameter. Symmetrically, with $x_i^{(2)}$ (corresponding to the key) as the anchor, we can get L_{cont}^k. We finally obtain a symmetric

contrastive loss as follows:

$$L_{cont} = L^q_{cont} + L^k_{cont} \tag{7.5}$$

During training, the unsupervised contrastive loss brings spectra containing similar spectral peak features closer together in latent space, while spectra with different spectral features are pushed farther apart. Dissimilarities, within our case of study, are, e.g., different number of peaks, at different positions, or with different shapes.

Self-Supervised Loss Function. As can be seen from Figure 7.2, the above four modules share the same backbone encoder f_q. The training of the shared feature extraction encoder can also be viewed as multi-task learning. By jointly optimizing the inter-sample relational reasoning objective, the intra-sample relation reasoning objective, the external-variable relational reasoning objective, and the self-contrastive learning objective, the final training loss function of SpecRRMoco-Net is specified as

$$L = L_{inter} + L_{intra} + L_{tmp} + c \cdot L_{cont} \tag{7.6}$$

Here c is a coefficient to adjust the weight of the contrastive loss and is set to 0.01. Ablation studies on this coefficient are presented in Section 7.2. It is important to note here that this loss function unifies SpecRR-Net and SpecMoco-Net, i.e., a value of 0 for the coefficient c corresponds to SpecRR-Net, while a value of ∞ (achieved by retaining the L_{cont} item with a coefficient of 1, while excluding the other three relational reasoning-based loss items in the loss function). A value of c set to 0 would correspond to SpecRR-Net while setting it to a large value increases the relevance of SpecMoco-Net.

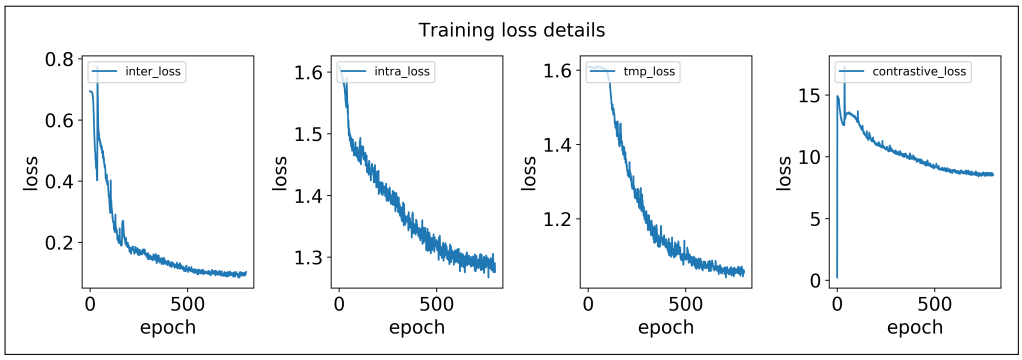
7.2 Experiments and results

7.2.1 Implementation Details

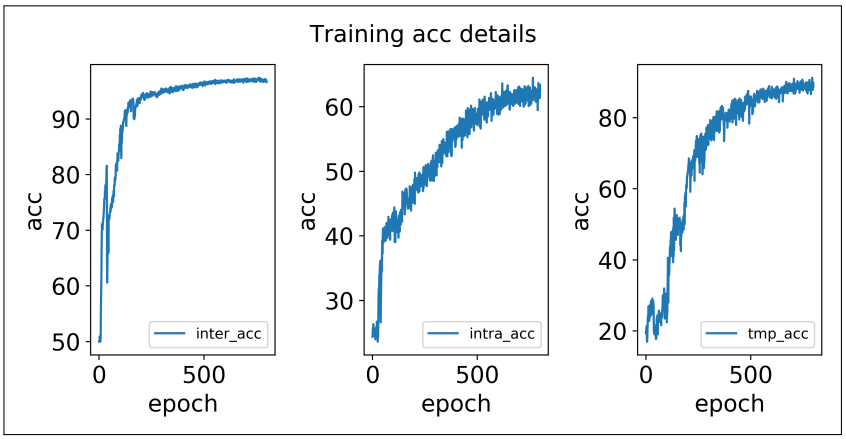
The SpecRRMoco model, SpecRR-Net, and SpecMoco-Net were trained using PyTorch on a single NVIDIA A100-PCIE-40GB. The self-supervised backbone encoder $f_q(\cdot)$ is trained by minimizing the proposed joint loss function Eq. 7.6 with a stochastic gradient descent (SGD) optimizer [7]. We named the nine data sets used as D1 to D9 for convenience, where D8 and D9 were collected from FeO powder, and D1-D7, from Fe powder sample, as described in Section 2.1. Among them, D1, D4, D8, and D9 data sets (872 spectral curves in total), were used to train the encoder network $f_q(\cdot)$ to learn feature representations during the pre-training (without label information). The batch size was set to 512, and the capacity of the queue of keys in the contrastive learning module to 872×2 (that is, twice the total number of spectral curves in the first stage training data sets). Within the SpecRRMoco-Net framework, experiments were performed with a loss factor of $c = 0.01$, unless otherwise stated. We applied data augmentations randomly 6 times in the inter-sample relational reasoning branch.

In the pre-training stage, the initial learning rate of the optimizer was set to 0.15, a linear warmup for the first 50 epochs (from a value of 0.02) followed by a cosine decay schedule was applied to adjust the learning rate during training, and the weight decay was set to 1×10^{-4}. While the relation scores in relational reasoning modules are similarity-based, we formulate each relational reasoning pretext task as a classification

task, so accuracy-based metrics can be applied to evaluate their performance on these pretext tasks. Taking SpecRRMoco-Net as an example, the pretraining details (loss and accuracy) of these four pretext tasks (3 relational reasoning pretext tasks + 1 self-supervised contrastive-based instance discrimination task) are shown in Figure 7.7. From the results, it can be seen that the training accuracies of relational reasoning pretext tasks increase with the decrease of the corresponding relational reasoning losses, and the contrastive loss decreases in the same trend.



(a)



(b)

Figure 7.7: (a) Training loss and (b) training accuracy at different epochs for each pretext task in SpecRRMoco-Net during the pre-training stage. In SpecRRMoco-Net, the encoder is trained under the supervision of inter-sample, intra-sample, and external-variable relational reasoning heads, as well as SpecMoco contrastive head. Notably, the instance discrimination pretext task based on self-contrastive learning (SpecMoco-Net) has no accuracy-based metric, it is based on a similarity measure of the learned representations rather than a classification objective. Therefore, we show the distribution of loss values for four loss function components and classification accuracy of three relational reasoning-based pretext tasks.

In the linear evaluation stage, a linear classifier was trained by minimizing the cross-entropy loss function, and an SGD optimizer with a learning rate of 0.1, and weight decay of 1×10^{-4} was applied. As mentioned earlier, spectra data is first projected to latent space to extract useful representations during the pretraining stage, and then a linear classifier is trained based on learned representations (with frozen encoder parameters), which could simplify the downstream linear classification task significantly. To further prevent overfitting, a train/validation/test strategy and an

early stopping strategy, which stops the training when the validation accuracy does not increase relative to its previous best value for $M = 20$ steps, were employed to train the linear classifier. In order to evaluate the performance of the model, all data were labeled to calculate the weighted precision and recall. However, only 42 (2.8% of the datasets) of the labeled data were used to train the linear classifier in the linear evaluation stage. These representative scattering curves were selected from D1, D4, D5, D8 and D9 data sets as the basis of the training/validation dataset. Furthermore, of the 42 labeled data 15 belonged to the "before phase transition" class, 25 to the "after phase transition" class, and 5 to the "during phase transition" class. The remaining 1,474 spectral curves (97.2% of the data) were used to test the performance of the backbone encoder and linear classifier. In the linear evaluation stage, we applied a stratified random split to the imbalanced labeled data with a train/validation split ratio of 66.7%/33.3% to ensure at least two scattering curves corresponding to each phase. The best model on the validation dataset was used for testing, updated only if higher validation accuracy was achieved in the process. Typically, after training for several epochs, the model can achieve high classification accuracy. This illustrates the high quality of the encoder network, indicating that the backbone encoder has extracted useful representations from the spectral data.

7.2.2 Linear Evaluation on Downstream Classification Task

In this subsection, we evaluated the performance of the self-supervised encoder trained by different networks on the downstream spectral classification task. To do so, we train a linear classifier on top of learned representations from the frozen backbone encoder. As described above, 42 representative labeled spectra from five data sets were used to train the linear classifier. Figure 7.8 shows the classification results together with the classification probabilities of SpecRRMoco-Net for a few example data sets using only 2.8% of the labeled data. Specifically, the figure presents the results for D6 (Fig. 7.8a), D8 (Fig. 7.8b), and D9 (Fig. 7.8c). For each sub-figure, the first column renders the contour map of the intensity distribution of the corresponding dataset, with the horizontal lines in the contour map indicating the phase transition boundary or phase transition interval. The second column shows the ground-truth labels of the dataset, where the black line represents class 0 (before phase transition), the magenta line represents class 1 (during phase transition), and the brown line represents class 2 (after phase transition). The last column shows the predicted category labels and the corresponding probabilities, where the values indicate the probabilities and the corresponding colors indicate the predicted labels. We compared the average classification precision/recall of SpecRRMoco-Net, SpecRR-Net, and SpecMoco-Net, as reported for Fe datasets (D1-D7) and FeO datasets (D8-D9) in Table 7.1. Average and standard deviation values were calculated from 20 runs. For each run, the results for the Fe data set or the FeO data set were obtained by averaging the results of all data sets within the group.

The prediction time for each spectral curve is about $40\,\mu s$, which is small enough to meet the requirement of real-time processing even at high-repetition rate facilities like the European XFEL. While this procedure would not provide feedback to the users as soon as the experiment starts, since the network training creates a delay, after the network is trained with the first data, one may reuse the model for fast feedback for the rest of the experiment.

	Model	Fe		FeO	
		Precision	**Recall**	**Precision**	**Recall**
2.8% labels	SpecSelfTime	98.6 ± 0.2	98.5 ± 0.2	78.6 ± 3.2	80.0 ± 4.3
	SpecRR-Net	99.2 ± 0.3	99.1 ± 0.3	91.6 ± 6.2	90.7 ± 5.4
	SpecMoco	98.3 ± 1.1	97.9 ± 1.8	93.8 ± 3.7	93.2 ± 4.1
	SpecRRMoco	99.6 ± 0.2	99.6 ± 0.2	96.3 ± 3.2	96.5 ± 2.4
10% labels	SpecSelfTime	98.6 ± 0.5	98.5 ± 0.5	80.5 ± 2.7	82.0 ± 3.9
	SpecRR-Net	99.1 ± 0.2	99.0 ± 0.2	96.9 ± 0.8	95.8 ± 1.1
	SpecMoco	99.3 ± 0.3	99.3 ± 0.3	94.1 ± 0.7	93.7 ± 0.7
	SpecRRMoco	99.6 ± 0.2	99.5 ± 0.2	97.1 ± 1.8	96.9 ± 1.3

Table 7.1: Classification results measured in terms of weighted precision and recall using different self-supervised methods. For each method, the classification results are reported with amounts of labels corresponding to either 2.8% or 10% of the total collected data.

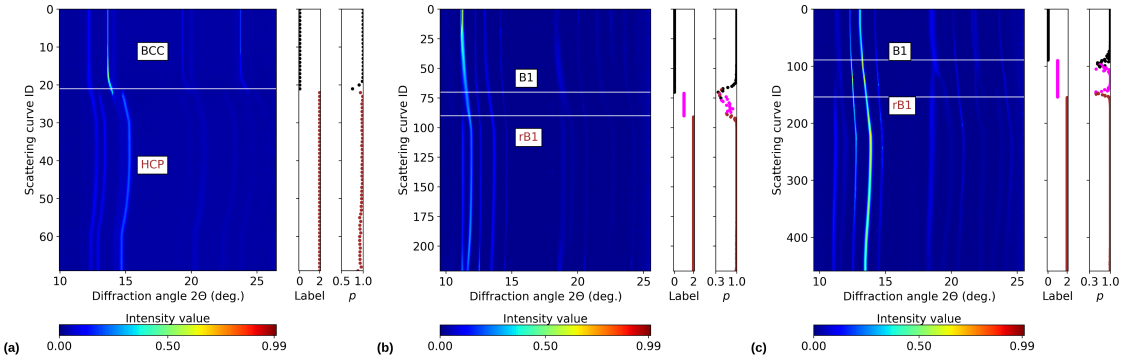


Figure 7.8: Classification results for experimental scattering curves using the proposed SpecRRMoco-Net with 2.8% labeled data. Each row of contour plots is a different scattering curve. The label as defined by an expert is also reported in the inset "Label". Black corresponds to data sets collected before the phase transition (label 0, that is BCC for Fe and B1 for FeO), magenta during (label 1), and brown after (label 2, that is HCP for Fe and rB1 for FeO). Horizontal lines in contour plots indicate the onset or end of a phase transition. The label predicted by SpecRRMoco-Net is also reported (indicated by colors in the inset "p"), together with the associated probability p. Data sets shown are (a) D6, (b) D8 and (c) D9.

The overall results show that with only 2.8% of the labels (42 spectral curves), all three models can accurately detect phase transitions in the Fe datasets, but some models do not perform well in the FeO datasets. In particular, SpecRRMoco-Net achieved better classification performance than SpecRR-Net and SpecMoco-Net under the current training strategy and hyperparameter settings, especially on FeO datasets, which are more challenging than Fe datasets due to the continuous nature of the phase transition and higher density of Bragg peaks. In SpecMoco-Net and SpecRR-Net (see Fig. 7.9), the class labels of some spectra in D8 and D9 data sets were incorrectly predicted in the 'during phase transition' region. The classification results for the 10% labeled data are also reported in Table 7.1. A clear improvement in SpecRR-Net and SpecMoco-Net performances can be seen, with SpecRR-Net achieving a slightly better result than SpecMoco-Net. For SpecRRMoco-Net, good classification performance was achieved with 2.8% of labeled data, but the classification standard deviation on the FeO dataset is further reduced with 10% of labeled data. Furthermore, SpecRRMoco-Net's

results are similar to models based on supervised contrastive learning (Table 6.1) and use fewer labels. It should be noted that the other four data sets (D2, D3, D6, and D7 data sets) were not included in the training of the backbone encoder or linear classifier in the pre-training and linear evaluation phases, but nevertheless, the self-supervised models still achieved very good classification results, meaning that the learned representation is transferable. Moreover, it also demonstrates the high quality of the learned representations of the feature extraction backbone encoder.

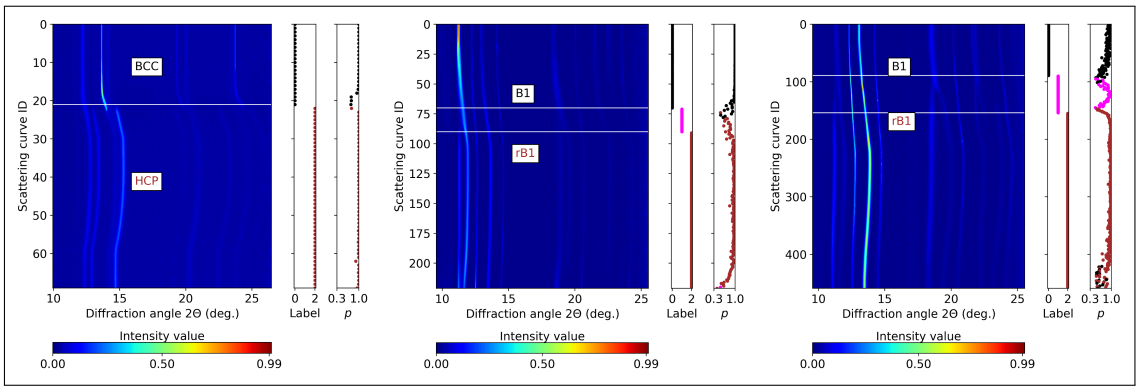
7.2.3 Comparison with Other Methods

In this section, we compare the self-supervised classification models already introduced with a modified version of the SelfTime network [23] (designed specifically for time series data), which we name SpecSelfTime. In particular, we replaced the original convolutional backbone encoder with the ConvSC attention network to better fit 1D spectral data for better performance. It should also be noted that SpecSelfTime, which is closely related to our work and the baseline of our SpecRR-Net, does not include the inter-temporal relational reasoning module we introduced in this study.

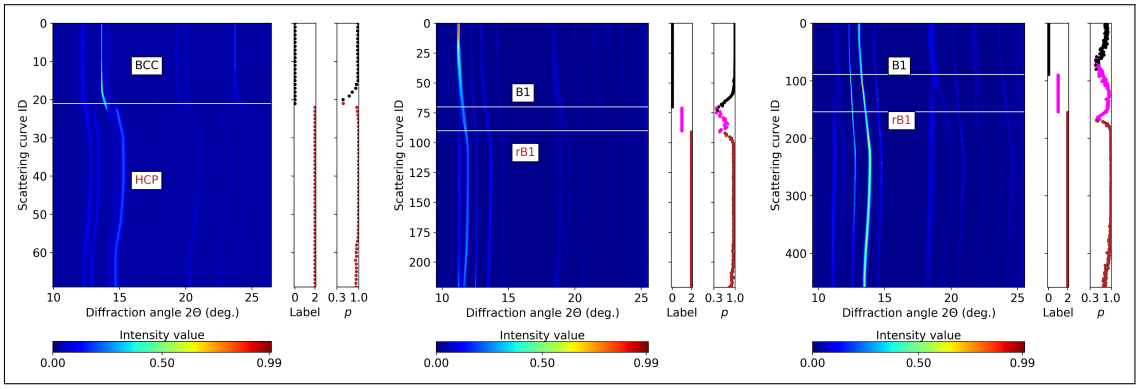
For a fair comparison, the settings of hyperparameters in the SpecSelfTime model are the same as in the SpecRR-Net and SpecRRMoco-Net models. show its classification results on experimental spectra with 2.8% labeled data (average weighted precision/recall 98.6/98.5% for Fe datasets and 78.6/80.0% for FeO datasets). In addition, for 10% of the labels, it does not have a great improvement in performance. SpecSelfTime performs poorly on several data sets, and particularly on FeO datasets, where it failed to detect the 'during phase transition' class on D8 (Fig. 7.9 (c)). This indicates the poor generalization ability of the model. As can be seen from the results, SpecSelfTime performs worse than the improved SpecRR-Net and even SpecMoco-Net, which highlights the importance of the external-variable relational reasoning module we introduced.

While the downstream classification task can evaluate the quality of the model, it cannot fully reflect the clustering ability. Therefore, as a qualitative analysis, we further evaluate the clustering power of these self-supervised classification models by visualizing the learned representations using UMAP (Uniform Manifold Approximation and Projection) [71]. Figure 7.10a renders the UMAP of SpecRRMoco-Net, while Fig. 7.10b visualizes the original data. In both cases, the class labels are ground truth.

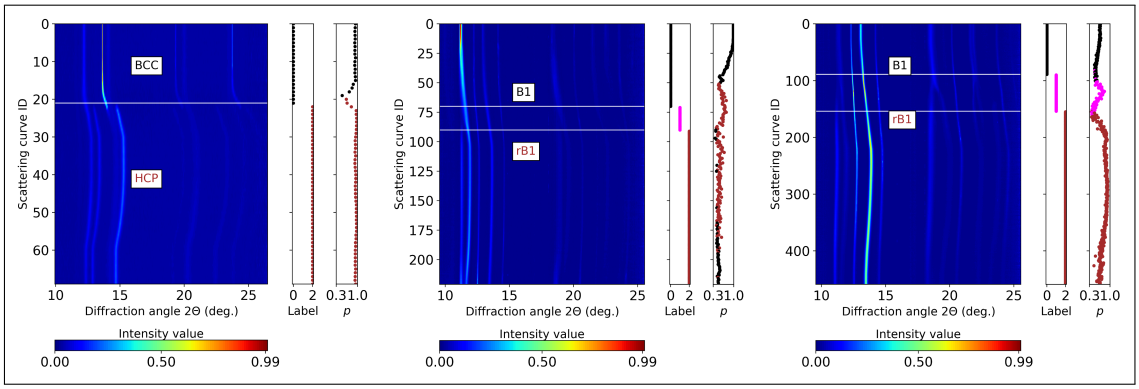
Often, there is the need to perform exploratory analysis on the acquired data, in order to understand patterns before attempting to perform any labeling. This procedure is different in scope from the direct classification and aims only at detecting similarities within the data. In this case, clustering methods are often used. For this reason, we compare, in Table 7.2, the capability of a traditionally used clustering method, Spectral Clustering, of finding similarities within each label either from the original data or starting from the representation produced by the studied self-supervised methods. To assess the clustering quality, two metrics have been used: the Silhouette Score [89]. and the Mutual Information [103]. The Silhouette Score is shown in Table 7.2, and it measures how the distances between samples within a cluster compare with distances between clusters without using the ground truth label association. The Mutual Information Score is shown in Table 7.3, and it uses ground truth information and compares the mutual agreement in the assignment using information-theoretic approaches while being invariant to permutations of the labels. These are calculated after clustering



(a)



(b)

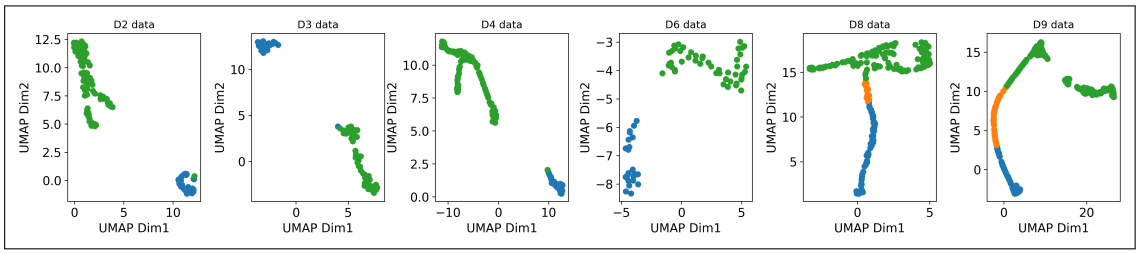


(c)

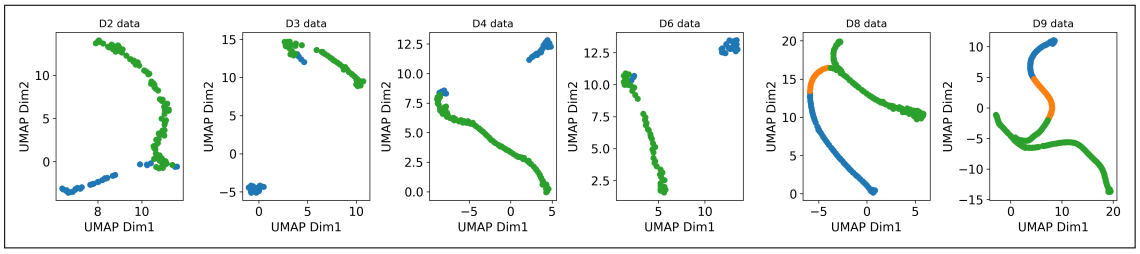
Figure 7.9: Classification results of (a) SpecMoco-Net, (b) SpecRR-Net, and (c) SpecSelfTime for experimental spectral data (D6, D8, and D9) with 2.8% labeled data.

the data by applying Spectral Clustering to the learned representations of each self-supervised encoder, or by clustering the original data itself with Spectral Clustering. The effect of varying some hyperparameters of the Spectral Clustering may also be seen in the uncertainties, as the optimal choice may not be known during the exploratory analysis phase. In addition, in spectral clustering, a K-means strategy or Discretization strategy is applied to assign labels, and the affinity matrix is constructed by computing the nearest neighbor graph or radial basis function (RBF) kernel. The mean and standard deviation values are calculated from different combinations of these parameters (four groups). As can be seen from the results, the representations learned by the relational reasoning self-supervised methods show better cluster separation ability

compared to the original data. Changing the hyperparameter settings for both SpecMoco and the clustering of the original data may lead to different results, while the relational reasoning-based methods tend to lead to representations that are less dependent on choices of the hyperparameters in Spectral Clustering. Particularly, the choice of the label assignment in Spectral Clustering leads to a high variance in SpecMoco. On the other hand, combined with the previous linear evaluation results, SpecSelfTime shows poor classification performance while achieving relatively good clustering ability. This experiment shows that the representations learned from self-supervised methods may lead to good cluster separation ability without necessarily allowing for better classification performance within the scope of the linear evaluation protocol when only a small subset of labeled data is available for training.



(a)



(b)

Figure 7.10: (a) UMAP visualization of the embedded features from some example datasets (D2, D3, D4, D6, D8, and D9, respectively) after the SpecRRMoco-Net encoder. (b) UMAP visualization of the original example data sets (D2, D3, D4, D6, D8, and D9, respectively). In both cases, the class labels are ground truth. Here, the green line represents class 0 (before the phase transition), the red line represents class 1 (during the phase transition), and the blue line represents class 2 (after the phase transition).

7.2.4 Ablation Studies on the Coefficient c in the SpecRRMoco-Net Loss Function

Here, we report on an ablation study on the coefficient c (shown in Table 7.4), performed to understand its impact on learning data representations. These experiments were performed under the same training setup described above. We varied c in the range [0.0001, 1], and also set it to 0 (that is, a pure SpecRR-Net) and infinity (that is, a pure SpecMoco-Net). For the downstream spectral classification task, 2.8% of labels were used. From Table 7.4, we can see that SpecRRMoco-Net performs well over a wide range of the coefficient (0.001-1 and infinity). This result suggests that jointly optimizing the relational reasoning-based pretext task and the contrastive learning-

Model	Average	
	Fe	**FeO**
Spectral Clustering (SC)	0.64 ± 0.10	0.52 ± 0.07
SpecSelfTime + SC	0.87 ± 0.04	0.49 ± 0.06
SpecRRMoco (c 0.01) + SC	0.85 ± 0.02	0.57 ± 0.01
SpecRR-Net + SC	0.87 ± 0.00	0.61 ± 0.03
SpecMoco + SC	0.69 ± 0.15	0.42 ± 0.02

Table 7.2: Average Silhouette coefficient obtained by applying different methods to each data set. The first row shows the effect of applying Spectral Clustering directly to the data, while the following rows show the effect of applying Spectral Clustering to the latent representation produced after the encoder trained with the respective self-supervised learning techniques. Results for Fe datasets and FeO dataset are reported. In the spectral clustering, the number of clusters was set to 3 for the Fe data sets, and 2 for the FeO data sets. In addition, Spectral Clustering hyperparameters on the label assignment and on the method for building the affinity matrix have been varied, and the average result is shown with the root-mean-squared error over different configurations.

Model	Average	
	Fe	**FeO**
Spectral Clustering (SC)	0.64 ± 0.10	0.52 ± 0.07
SpecSelfTime + SC	0.87 ± 0.04	0.49 ± 0.06
SpecRRMoco (c 0.01) + SC	0.85 ± 0.02	0.57 ± 0.01
SpecRR-Net + SC	0.87 ± 0.00	0.61 ± 0.03
SpecMoco + SC	0.69 ± 0.15	0.42 ± 0.02

Table 7.3: Mutual information between the ground truth labels and the predicted labels of different methods for each dataset. The first row corresponds to applying Spectral Clustering directly to the original data, while the following rows show the mutual information scores by applying Spectral Clustering to the latent representation produced after the encoder trained with the respective self-supervised learning techniques. Results for Fe datasets and FeO datasets are reported. Spectral Clustering hyperparameters on the label assignment and on the method for building the affinity matrix have been varied, and the average result is shown with the root-mean-squared error over different configurations.

based pretext task can improve the performance of the pure contrastive learning-based network as well as the purely self-supervised relational reasoning network under the current training setup.

7.2.5 Ablation Studies on the Data Augmentation

We report here on an ablation study on data augmentations performed in order to evaluate their impact on the SpecRRMoco-Net performances. Several commonly used data augmentation techniques were explored, including diffraction angle warping (D.A.W.), magnitude warping (M.W.), window slicing (W.S.), jittering (Jitter), and scaling. Among them, diffraction angle warping and window slicing are performed in the diffraction angle dimension, whereas jittering, scaling, and magnitude warping are performed

c	Fe		FeO	
	Precision	**Recall**	**Precision**	**Recall**
0.001	98.5 ± 0.6	98.4 ± 0.4	92.7 ± 4.0	92.8 ± 3.3
0.01	99.6 ± 0.2	99.6 ± 0.2	96.3 ± 3.2	96.5 ± 2.4
0.1	99.5 ± 0.2	99.2 ± 0.8	97.3 ± 1.0	97.0 ± 1.3
1	98.9 ± 0.6	97.9 ± 1.6	95.2 ± 2.0	95.0 ± 1.4
0 (SpecRR-Net)	99.2 ± 0.3	99.1 ± 0.3	91.6 ± 6.2	90.7 ± 5.4
Inf (SpecMoco-Net)	98.3 ± 1.1	97.9 ± 1.8	93.8 ± 3.7	93.2 ± 4.1

Table 7.4: Ablation study of the coefficient c in the loss function. Weighted precision and recall for Fe and FeO are reported as average and standard deviations over 20 runs. The best results within uncertainties are highlighted in bold.

in the magnitude domain. The variations introduced by these data augmentation techniques respect physical information contained in the data itself, with effects that resemble the realistic range of experimental effects, without changing the data labels. They generate new input with variances while keeping identical labels in the embedding space. Based on this, surrogate tasks can be formed to extract underlying patterns and build the representations. In addition to the diffraction angle and magnitude warping

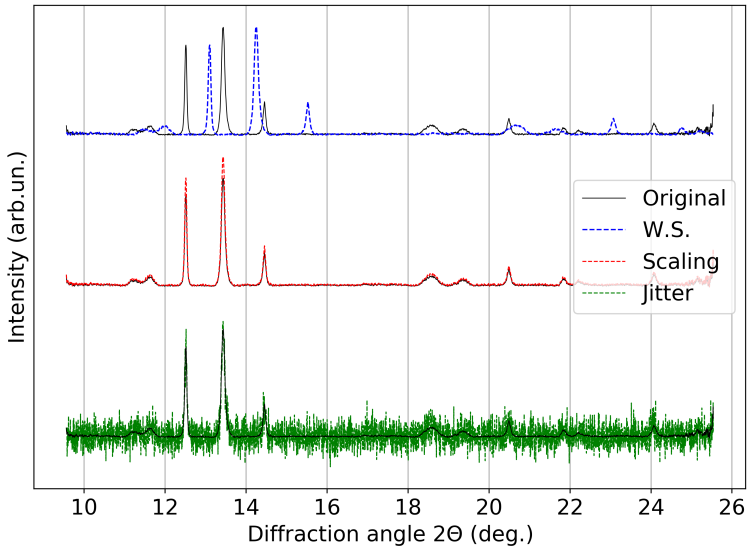


Figure 7.11: Illustration of the effect of applying window slicing (W.S.), scaling, and jittering (Jitter) data augmentation techniques to one of the scattering curves.

which were already discussed, jittering was used to introduce possible random noise in the experiment, such as additive detector noise. It was simulated by adding noise sampled from a Normal distribution with a mean value of 0 and a standard deviation of 0.1. Scaling was used to model uniform intensity variations, which is achieved by multiplying the original data by a random scalar value sampled from a Normal distribution with a mean of 1 and a standard deviation of 0.1. Window slicing was used to model small variations in diffraction angle coverage, for example, when the sample and detector are far apart, resulting in the detector covering a smaller range of diffraction angles. This is achieved by randomly cropping out a large continuous slice of the spectrum (in the implementation 80% of the original spectral length, i.e.,

randomly discarding 20% of the edge spectral segments) and interpolating it to the original length. An illustration of these three data augmentation techniques can be seen in Fig. 7.11. Scaling is used to simulate reasonable and random variations in the intensities of peaks, while not changing their positions. Diffraction angle warping is used to parallel the variation of peak positions, so as to allow the model to focus more on the number of peaks rather than their location.

Figure 7.12 shows the linear evaluation (represented by the average accuracy and standard deviation of 20 runs for all nine datasets) under different data augmentation techniques individually or in combination. The diagonal elements correspond to a single data augmentation and the non-diagonal elements represent the combination of the two consequent data augmentation techniques. The classification results show that the combination of two data augmentation techniques usually performs better than a single technique. In particular, the best result is achieved when "magnitude warping" is combined with "diffraction angle warping ". Therefore, in this study, we applied these sequentially to all models presented in this chapter.

In addition, we found that the order in which data augmentation techniques are applied also affects the results, with different orders leading to different enhanced data, and also because of the inherent randomness of each data enhancement technique. In Fig. 7.12, we use the combination of window slicing and time warping (Fig. 7.12), as an example to illustrate the potential importance of the order of applying different data augmentation techniques. The window slicing augmentation randomly removes only the edges of the spectra, while the diffraction angle warping changes the full distribution of peaks in a non-linear way. By applying the window slicing first, the edges are removed, and the remainder of the distribution is warped. It is rare in the given data, that the peaks relevant to the phase transition appear at the edge of the distribution and hence this information is rarely lost. If, on the other hand, the diffraction angle warping is applied first, relevant peaks may be warped to appear on the edges of the distributions, which may be removed when window slicing is applied. In this case, relevant information required for the phase transition identification is removed from the data and the encoder can map the augmented data to a different representation.

This experiment illustrates that data augmentations play an important role in self-supervised models [12, 13]. As it is domain-specific, it must be customized for data sets from different research areas. Once the most appropriate data augmentation techniques are identified, the ability to automatically classify the data can be effectively improved.

7.2.6 An Open-Source Interactive Software Program

Similar to the open-source interactive software program presented in Section 5.4, we provide a SpecRRMoco-Net based spectral classification program[1] [108]. For its design procedure, please see Section 5.4.

It is worth noting that since self-supervised learning includes pre-training and linear evaluation, its training process and preparation of training/test data are different from supervised learning models. For example, pre-training in self-supervised learning methods is based on unlabeled data, and their generation of training/test data and definition of excuse tasks involve data augmentation techniques.

[1] https://github.com/sunyue-xfel/Supervised_and_Self_Supervised_DL_Models_for_SpectraClassification

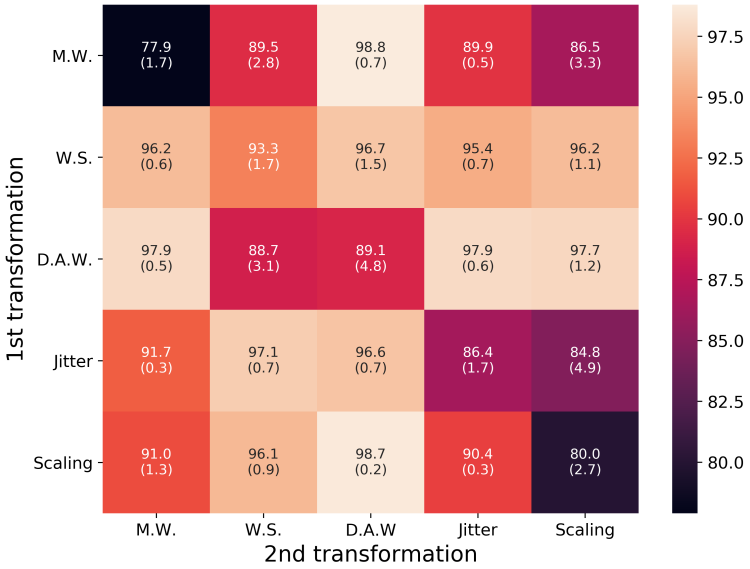


Figure 7.12: Ablation study on data augmentation techniques. Results for magnitude warping (M.W.), window slicing (W.S.), diffraction angle warping (D.A.W.), jittering (Jitter), and scaling data augmentation techniques are reported. The figure shows the average classification accuracy and standard deviation (values in parentheses) for 20 runs with 2.8% of labeled data. In addition to this, diagonal elements indicate the use of only one data augmentation technique, while other non-diagonal entries indicate the combination of two data augmentation techniques. The color scale represents the classification accuracy.

7.3 Discussion

From the above experimental results, it can be concluded that the three networks proposed in this chapter are effective in constructing data representations that can greatly improve the automation of the classifications of spectral data, and in particular the detection of phase transitions. We attribute the success of the models, consistent with the results of the ablation study, to appropriate data augmentations and pretext tasks. In fact, self-supervised learning critically relies on augmentations, which should be tailored for the scientific case object of investigation. The ones applied in this study retain physically meaningful information while simulating other plausible experimental effects. Thus, compared to traditional unsupervised clustering algorithms which require manual tuning of parameters for each dataset, self-supervised models allow the classification process to be automated once a minimal amount of labels is available.

In SpecMoco-Net, the learning process is primarily based on exploiting redundancies in the data, rather than learning to perform inference tasks based on the data itself. In addition, SpecMoco-Net is based on the instance-instance discrimination task, which cannot explicitly exploit data information at different scales, such as the global dependencies across diffraction angle dimension. Furthermore, in practice, self-supervised contrastive learning benefits from a large number of negative samples to extract meaningful representations, and while SpecMoco-Net allows a large and consistent dynamic dictionary, in our case we do not have enough spectral training examples, which is another important reason why SpecMoco-Net performed worse than SpecRR-Net and SpecRRMoco-Net in our case of study.

Networks based on relational reasoning learning can be viewed as simultaneously

learning deep embeddings and deep non-linear metrics (similarity functions) [110]. In SpecRR-Net and SpecRRMoco-Net, three relational reasoning modules are designed to capture the underlying dependencies from multiple dimensions and at different scales to build useful representations. Moreover, comparison with SpecSelfTime shows that our proposed external-variable relational reasoning module can significantly improve the performance of models by addressing the dependencies of diffraction spectra on pressure values, in this particular application. Relative to the pretext task based on contrast learning, the relational reasoning-based pretext tasks impose more supervision on the network using easily accessible sources of information. In the process of reasoning about the relations between spectral entities, irrelevant and noisy features are neglected, and non-obvious properties can be focused on, thereby gaining new knowledge. Furthermore, the difference in the structure of the two methods may also lead to some differences in the way of updating model parameters. Ablation studies on structural differences are necessary and interesting for further research, which is left to future work.

SpecRRMoco-Net benefits from both relational reasoning learning and contrastive learning, and shows better results than SpecRR-Net and SpecMoco-Net alone with the current hyperparameters and training settings, it combines SpecRR-Net and SpecMoco-Net therefore providing a flexible framework that can potentially fit a broader set of use cases. The success of each pre-text task in SpecRRMoco-Net drives the update of the encoder model, improving its feature representation ability while increasing the robustness and generality of the encoder network. Importantly, although these models are proposed for classification applications on spectral data, the architectures are general and can be easily extended to 1D time series data and various other types of data, such as image classification.

7.4 Conclusions

In this section, we propose three self-supervised frameworks to classify 1D spectral data using a minimal amount of labeled data, and we validate their performance using x-ray diffraction data of samples showing phase transitions. These frameworks are based on relational reasoning (SpecRR-Net), contrastive learning (SpecMoco-Net), or a combination of the two (SpecRRMoco-Net). They are capable of learning discriminative features and building effective representations, therefore greatly reducing the number of labels required, making a step towards automating the spectral classification process. Among them, SpecRRMoco-Net shows superior performance by benefiting from contrastive learning and relational inference learning. Moreover, as a consequence of the reduced number of labels, scientists' time is greatly optimized. To illustrate the relationship between the collected spectra and the external pressure variable, we extend the relational reasoning-based method to explicitly include it. In this work, we demonstrate the importance of a proper choice of data augmentations, which must be tailored for the specific case of study to ensure the retention of scientifically meaningful information. In particular, we discuss and validate augmentations relevant to the case study discussed, and we prove that the three methods introduced are effective in detecting phase transitions. This is the case even when data for which no labels are available are used, which demonstrates good generalizability of the approaches. We furthermore compare the three frameworks with state-of-the-art unsupervised methods.

After an initial training step, the methods proposed here can be used to accurately

and automatically screen collected data, even in real-time at high-repetition rate facilities given the inference speed, so as to provide a better understanding of the experiment and therefore enable the most effective real-time planning.

Contributions. The author of this PhD thesis is responsible for the following contributions presented in this chapter:

- Proposition of three self-supervised frameworks: SpecRR-Net, SpecMoco-Net, and SpecRRMoco-Net for 1D spectral representation and classification. These networks can greatly reduce the number of labels required and increase the automation of the spectral classification process (Section 7.1).
- The introduction of the inter-temporal relational reasoning network in the SpecRR-Net to address the temporal dependencies of time-resolved spectral data, which is a key feature of spectral time series and can greatly improve the quality of learned representations (Section 7.1.3).
- Ablation study on data augmentation. The selection of data augmentation techniques tailored to diffraction spectral time series that preserve physically relevant information and allow for the efficient definition of pretext tasks (Section 7.1.2, 7.2.4).
- Evaluate and compare the effectiveness of these networks on multiple experimental spectral datasets. Compare these methods with other state-of-the-art methods from multiple perspectives (Section 7.2).
- In-depth discussion of the results of this chapter as well as a comprehensive analysis and interpretation of the findings (Section 7.3).
- An interactive software program for spectral classification, based on these self-supervised methods.

8 Conclusion

Experimental techniques such as spectroscopy and X-ray diffraction are instrumental in investigating matter. When experiments are performed at modern X-ray facilities, such as synchrotron radiation sources, and X-ray free electron lasers (XFELs), a vast amount of data are potentially collected over short periods of time. The ability to rapidly and accurately assess the status of an experiment is essential to maximize its efficiency. Machine learning (ML) based approaches are ideal for automating repetitive tasks and identifying features and patterns in datasets, offering great potential for minimizing the use of experts' time and maximizing scientific output.

In this dissertation, we focus on the application of spectra classification in the fields of diffraction and spectroscopy and on developing and evaluating deep learning classification frameworks for 1D spectral time series. With the aim of an efficient and automated classification method, several models based on machine learning are proposed, and validated by showing their performance using X-ray diffraction data of samples showing phase transitions. The main contributions of this thesis are summarized into three main thesis points.

We start with models with manual feature engineering. First, we proposed a spectral classification model based on PCA and spectral clustering, which are typically used in the data exploration phase. In this model, we studied the principal components (PCs) of the PCA method and analyzed the contribution of the original variables to the PC. In addition, we tested the classification confidence with different cumulative explained variance values in the PCA method. However, the performance of clustering methods is usually limited and it is challenging to determine the appropriate number of clusters, which often requires fine-tuning certain hyperparameters to obtain accurate results, thus hindering automation performance. Then, we designed a neural network-based ML model combined with a binned-weighting technique for spectra classification to minimize misclassifications of indistinguishable features in overlapping regions of different spectra. In this model, we transform the spectra classification problem into a 2D segmentation problem where we input every point in the spectra with their 2 coordinates. However, the weighting factors in this model are calculated based on the training accuracy in each bin, which poses a challenge for automating the model and

may limit its generalizability when applied to diverse datasets.

Next, to achieve better classification performance and to get rid of manual feature engineering, we proposed several neural network-based solutions in an end-to-end manner, and regard the task as either a 2D space segmentation problem or a 1D time series data analysis problem. We focus on two proposed classification models, namely the end-to-end binned FCNN with automatically capturing weighting factors model when viewed as a 2D space segmentation problem and the convolutional SCT attention model when viewed as a 1D time series classification problem. And several other end-to-end model structures based on FCNN, CNN, ResNets, Transformer, and LSTM are explored. Finally, we evaluated and compared the performance of these classification models from multiple perspectives based on an example spectra dataset. Results show that the proposed models under the 1D time series classification setting are superior.

To further increase the generability of the classification model, we introduce a supervised contrastive learning framework for 1D spectral representation and classification. Additionally, we provided an interpretation of this classification model from multiple perspectives in terms of traditional descriptors of spectral patterns of interest to physicists. These descriptors encompass attributes such as peak number, peak position, width, and intensity.

However, the data labeling process in supervised learning methods is problem-specific and time-consuming, which in turn hinders automation. To this end, we turn our attention to self-supervised classification-based approaches. With the Convolutional SCT attention model as the backbone encoder, we proposed three self-supervised frameworks to classify 1D spectral data using a minimal amount of labeled data. These frameworks are based on relational reasoning (SpecRR-Net), contrastive learning (SpecMoco-Net), or a combination of the two (SpecRRMoco-Net). They are capable of learning discriminative features and building effective representations, therefore greatly reducing the number of labels required, making a step towards automating the spectral classification process. Moreover, as a consequence of the reduced number of labels, scientists' time is greatly optimized. We demonstrate the importance of a proper choice of data augmentations, which must be tailored for the specific case of study to ensure the retention of scientifically meaningful information. After an initial training step, the methods proposed here can be used to accurately and automatically screen collected data, even in real-time at high-repetition rate facilities, so as to provide a better understanding of the experiment and therefore enable the most effective real-time planning.

Finally, to facilitate the reuse of existing methods, and speed up the development process of new models, we provide a convenient software platform for spectral classification using the main deep-learning classification models presented in this thesis, including end-to-end supervised classification models and self-supervised classification models. In this platform, data analysis scripts were provided as Jupyter Notebooks for better visualization and reproducibility. In addition, the software program is implemented in the interactive runtime environment, Google Colab and Mybinder environments for easy exploration and reuse of spectral data and ML models.

Although we obtained nice result on ML-based spectral classification, the work presented in this thesis is only a small part of this area, there is still more research topics worth exploring. First, to ensure the effectiveness and generalizability of the proposed model, it is crucial to train and validate the classification models on spectral data collected from different experiments and spectroscopy techniques. In addition,

the optimization of hyperparameters and training strategies will be an important research direction for our future work, reducing the dependence on manual intervention and fostering increased automation. Another important direction of the subsequent research is the improvement of the proposed models and will continue to develop more powerful and efficient deep classification models specifically tailored for spectral classification tasks. This could involve exploring novel architectures, more efficient pretext tasks, data augmentation strategies, and loss functions in the realm of self-supervised and unsupervised classification frameworks. Finally, an essential direction for future research is to develop interpretable, explainable, and generalizable models, revealing the decision process of classification models.

Appendices

A

Summary in English

In scientific research, spectroscopy and diffraction experimental techniques are widely used and produce huge amounts of spectral data, especially when experiments are performed at modern X-ray facilities, such as synchrotron radiation sources, and X-ray free electron lasers (XFELs), where a vast amount of data are potentially collected over short periods of time. Under this circumstance, it is crucial to rapidly and accurately extract useful and meaningful features from spectra and provide immediate feedback on the actual state (e.g. time-resolved status of the sample). This capability is essential for the understanding of the underlying processes governing the experiment and enables necessary evaluation and further guidance of the experiment being evaluated. On the other hand, when analyzing data already collected –potentially up to hundreds of thousands of data sets –it is crucial to be able to employ some automated or semi-automated methods capable of extracting scientifically interesting features in the data so as to minimize the usage of experts's time and to maximize the scientific output.

This thesis focuses on analyzing 1D diffraction spectra data from high-pressure X-ray diffraction (XRD) experiments, specifically targeting the identification of phase transitions in XRD samples. In these experiments, changes in pressure will cause changes in the unit cell [86], which are reflected in the spectral peaks, such as differences in peak intensity, shifts in position, changes in peak shape (e.g. broadening), and, more importantly, changes in the number of peaks. During compression and decompression, the generated spectra can be considered as typical time series data. Detect phase transition implies classifying these time-resolved spectra into three different categories, i.e., before, during, and after the phase transition, which is based on the distribution of spectral peaks.

Recently, Machine Learning (ML), especially deep learning (DL) models, opens up new avenues for data-driven spectra analysis. They offer great potential for discovering intricate structures and learning representations from high-dimensional data. While promising results have been demonstrated in certain applications of spectra classification within the natural sciences domain, there remains ample room for further exploration and advancement in this field, such as better data representation, generalization and interpretability of models, and efficiency and automation of classification.

Given that spectral classification is the domain-specific problem to be solved, the key features of the spectral data should be considered when designing the ML-based classification models, as shown below.

- The spectra have high dimensionality, with about 4000 features (data points) in each spectral curve. Whereas the number of diffraction files is usually much smaller than the number of features, in our use case there are 60-460 spectral curves in each dataset. This can easily lead to the overfitting of ML models, especially for supervised ML models.
- The most important information is the spectral peaks, e.g., the number of peaks, their location, and their shape and intensity characterize the properties of the sample. These features are in local and global dependencies along the diffraction angle dimension and are not independent. The statistical model should focus on the information of the peaks while suppressing noisy and irrelevant features as much as possible.
- The spectral data was collected in a high-pressure experiment with a uniform compression and decompression process. In this use case, the change in the spectral peak distribution varies with the pressure, an external variable, describing the dynamic process of phase transition. Therefore, the spectral data have a strong dependence on the external variable, which we uniformly call time dependence.

In this dissertation, we will address the above problem by presenting several different types of ML/DL methods tailored for diffraction spectroscopy, with the aim of achieving efficient and automatic classification.

Spectra classification with manual feature engineering

When considering the application of machine learning methods to spectral data analysis, scientists generally have two tools at their disposal: (i) clustering the data to distinguish between different classes of samples, or (ii) labeling selected data to train a supervised classifier. We start with unsupervised spectral clustering methods, which are typically used in the data exploration phase. Specifically, we present a spectra classification model based on the Principle Component Analysis (PCA) and Spectral Clustering. In this model, PCA is applied to reduce the dimensionality, after which the spectral clustering method is applied. To better explain the model, we investigated the contribution of the original variables in spectra to the principle components in PCA and also tested the relationship between the classification results and the number of PCs. In addition, in the absence of ground-truth label information, we propose the classification confidence metric for model evaluation.

Despite the high performance we achieved on the example spectral dataset, there are still some limitations in this method. First, determining the appropriate number or density of clusters in the clustering method is a challenge, especially when dealing with complex and high-dimensional spectral data, which often requires fine-tuning certain hyperparameters to obtain accurate results, thus hindering automation performance. Second, the approach relies on PCA for preprocessing, which may lead to the loss of important discriminatory information. Furthermore, this additional step of feature engineering can impede automation, and introduce more complexity into the interpretation of the model.

To address these limitations of the clustering method, we further propose a data-driven neural network-based statistical model applied to spectra classification. Neural networks are chosen here due to their ability to learn complex mappings between input and target spaces which makes them perfect for our task. In this model, we transform our problem of finding and distinguishing features in a set of 1D curves where the input is provided by the sequence of the spectral intensity values, into a 2D segmentation problem where we input every point in the spectra with their 2 coordinates (scattering angle, and azimuthally integrated intensity). Since classifying features in overlapping regions is difficult, meaningless, and of low confidence, machine learning models with a binned-weighting technique are proposed to minimize the misclassification of indistinguishable features in overlapping regions. The believability weighting factor is calculated based on the classification accuracy (separability) of the neural network for each bin. The solution proposed here can automatically find regions (or bins) with high separability. In addition, we also investigated the performance of the model using a different number of bins. The results show that the more bins applied, the higher the classification accuracy, thus the smaller the ambiguous zone. We can conclude that the key to achieving high classification confidence in this approach is to find bins with high separability indices.

End-to-End Deep Learning Methods for Spectra Classification

Although we obtained nice results on the previous deep neural network model with the binned-weighting technique, the weighting factors in this model are calculated based on the training accuracy in each bin. The calculation of these weighting factors is carefully crafted by human intervention, which poses a challenge for automating the model and may limit its generalizability when applied to diverse datasets.

To further improve the generality and automation of the classification models, we propose several deep classification models in an end-to-end manner, eliminating the need for manual feature engineering. We first extend the neural network model with weighting techniques to an end-to-end binned FCNN (fully connected neural network) with the automatically capturing weighting factors model. With this improvement, the local believability weighting factors of each bin are learned automatically during training and are dynamically responsive to the input data, so the model can automatically prominent features with high separability and suppress indistinguishable features with low separability.

More generally, the spectra classification problem can be regarded as a 1D time series classification problem, and in this setting, the convolutional SCT (spatial-channel-temporal) Attention model is proposed. It is a hybrid model of CNN (convolutional neural network) and self-attention architecture, where CNN is used to learn local features while self-attention is calculated across spatial, channel, and temporal dimensions, enabling the model to focus on distinguishable features while suppressing noisy and misleading ones. We also designed several other classification models tailored for 1D spectra based on other state-of-the-art ML algorithms, such as FCNN, CNN, Resnet, LSTM, and Transformer.

Furthermore, we evaluated and compared the classification performance of these deep end-to-end learning models on an example experimental spectra dataset from

multiple perspectives. In addition, we conducted a feature importance analysis based on gradient backpropagation. In this experiment, we estimate the contribution of each input feature to the classification prediction of each model. In this way, this thesis provides a standard baseline for 1D spectral time series classification in an end-to-end manner. Finally, we designed a software program based on these end-to-end methods, specifically for spectral classification. The program is implemented in the interactive runtime environment Mybinder and Google Colab, which promotes reproducibility and facilitates the exploration and reuse of ML models.

To further increase the generability of the classification model, we introduce a supervised contrastive learning framework combined with data augmentation for 1D spectral representation and classification, the first attempt at its application in the field of diffraction spectra classification. To reveal the underlying and increase the credibility to physicists, we interpret our classification model in terms of traditional descriptors of spectral patterns of interest to physicists, such as peak number, peak position, width, and intensity. Furthermore, we evaluate the effectiveness of the network on multiple experimental spectral datasets to demonstrate the high quality and generality of the model.

Self-Supervised Approaches for Spectra Classification

The supervised classification model presented above shows excellent classification performance, but the reliance on annotated data poses a challenge for the automation of the classification algorithm. To address this, we proposed a self-supervised classification framework, namely, the SpecRRMoco-Net, for spectral classification. This method incorporates self-supervised relational reasoning learning [23, 140, 82, 110], and self-supervised contrastive learning [13, 12, 38], along with data augmentation techniques, enabling the extraction of generalizable features from unlabeled data. In addition, in order to account for the relation between spectra collected along the temporal dimension, we extend the relational reasoning-based method to explicitly include it by introducing an inter-temporal relational reasoning network. On this basis, the method is able to classify 1D spectral curves with only a small amount of data labeled by domain experts. While self-supervised learning requires domain-specific knowledge, the need for human supervision is largely reduced with respect to supervised learning, making a step towards automating the spectral classification process. Moreover, as a consequence of the reduced number of labels, scientists' time is greatly optimized.

It should be noticed that the data augmentation techniques presented in this thesis preserve physically relevant information, and are particularly effective for diffraction spectral time series, allowing efficient definition of pretext tasks. In this thesis, we demonstrate the importance of a proper choice of data augmentations, which must be tailored for the specific case of study to ensure the retention of scientifically meaningful information. In particular, we discuss and validate augmentations relevant to the case study discussed, and we show that the proposed method introduced is effective in detecting phase transitions. This is the case even when data for which no labels are available are used, which demonstrates the good ability of the approaches. We furthermore compare the three frameworks with state-of-the-art unsupervised methods.

Furthermore, two other networks, SpecMoco-Net based on self-supervised contrastive learning and SpecRR-Net based on relational reasoning learning, are proposed and discussed for spectral classification. Experimental results show that SpecRRMoco-

Net shows superior performance by benefiting from contrastive learning and relational inference learning. We further demonstrate on multiple experimental spectral datasets that these self-supervised machine learning methods can effectively classify spectral data, offering great opportunities to improve the scientific efficiency of experiments at large-scale X-ray facilities.

Lastly, we suggest some future research directions such as hyperparameter optimization and extension of our proposed method to other spectra collected from different diffraction and spectroscopy experiments, or even ordinary 1D time series data.

Bibliography

[1] Dameli Assalauova, Alexandr Ignatenko, Fabian Isensee, Darya Trofimova, and Ivan A Vartanyants. Classification of diffraction patterns using a convolutional neural network in single-particle-imaging experiments performed at x-ray free-electron lasers. *Journal of Applied Crystallography*, 55(3), 2022.

[2] Junwen Bai, Yexiang Xue, Johan Bjorck, Ronan Le Bras, Brendan Rappazzo, Richard Bernstein, Santosh K Suram, Robert Bruce Van Dover, John M Gregoire, and Carla P Gomes. Phase mapper: Accelerating materials discovery with ai. *Ai Magazine*, 39(1):15–26, 2018.

[3] Marijan Beg, Juliette Taka, Thomas Kluyver, Alexander Konovalov, Min Ragan-Kelley, Nicolas M Thiéry, and Hans Fangohr. Using jupyter for reproducible scientific workflows. *Computing in Science & Engineering*, 23(2):36–46, 2021.

[4] Gedas Bertasius, Heng Wang, and Lorenzo Torresani. Is space-time attention all you need for video understanding? In *ICML*, volume 2, page 4, 2021.

[5] Christopher M Bishop and Nasser M Nasrabadi. *Pattern recognition and machine learning*, volume 4. Springer, 2006.

[6] Anastasia Borovykh, Sander Bohte, and Cornelis W Oosterlee. Dilated convolutional neural networks for time series forecasting. *Journal of Computational Finance, Forthcoming*, 2018.

[7] Léon Bottou. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010: 19th International Conference on Computational StatisticsParis France, August 22-27, 2010 Keynote, Invited and Contributed Papers*, pages 177–186. Springer, 2010.

[8] Sandor Brockhauser, Raimond BG Ravelli, and Andrew A McCarthy. The use of a mini-κ goniometer head in macromolecular crystallography diffraction experiments. *Acta Crystallographica Section D: Biological Crystallography*, 69(7):1241–1251, 2013.

[9] Sandor Brockhauser, Olof Svensson, Matthew W Bowler, Max Nanao, Elspeth Gordon, Ricardo MF Leal, Alexander Popov, Matthew Gerring, Andrew A McCarthy, and Andy Gotz. The use of workflows in the design and implementation of complex experiments in macromolecular crystallography. *Acta Crystallographica Section D: Biological Crystallography*, 68(8):975–984, 2012.

[10] Ferhat Ozgur Catak, Ismail Aydin, Ogerta Elezaj, and Sule Yildirim-Yayilgan. Practical implementation of privacy preserving clustering methods using a partially homomorphic encryption algorithm. *Electronics*, 9(2):229, 2020.

[11] Hui Chen, Zan Lin, and Chao Tan. Nondestructive discrimination of pharmaceutical preparations using near-infrared spectroscopy and partial least-squares discriminant analysis. *Analytical Letters*, 51(4):564–574, 2018.

[12] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020.

[13] Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. Improved baselines with momentum contrastive learning. *arXiv preprint arXiv:2003.04297*, 2020.

[14] Zhantao Chen, Nina Andrejevic, Nathan C Drucker, Thanh Nguyen, R Patrick Xian, Tess Smidt, Yao Wang, Ralph Ernstorfer, D Alan Tennant, Maria Chan, et al. Machine learning on neutron and x-ray scattering and spectroscopies. *Chemical Physics Reviews*, 2(3):031301, 2021.

[15] Junlong Cheng, Shengwei Tian, Long Yu, Hongchun Lu, and Xiaoyi Lv. Fully convolutional attention network for biomedical image segmentation. *Artificial Intelligence in Medicine*, 107:101899, 2020.

[16] Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation, 2014.

[17] Sakyasingha Dasgupta and Takayuki Osogami. Nonlinear dynamic boltzmann machines for time-series prediction. In *Proceedings of the AAAI conference on artificial intelligence*, volume 31, 2017.

[18] Angus Dempster, François Petitjean, and Geoffrey I Webb. Rocket: exceptionally fast and accurate time series classification using random convolutional kernels. *Data Mining and Knowledge Discovery*, 34(5):1454–1495, 2020.

[19] Yifei Ding, Jichao Zhuang, Peng Ding, and Minping Jia. Self-supervised pretraining via contrast learning for intelligent incipient fault detection of bearings. *Reliability Engineering & System Safety*, 218:108126, 2022.

[20] Auralee Edelen, Christopher Mayes, Daniel Bowring, Daniel Ratner, Andreas Adelmann, Rasmus Ischebeck, Jochem Snuverink, Ilya Agapov, Raimund Kammering, Jonathan Edelen, et al. Opportunities in machine learning for particle accelerators. *arXiv preprint arXiv:1811.03172*, 2018.

[21] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *kdd*, volume 96, pages 226–231, 1996.

[22] William J Evans, Choong-Shik Yoo, Geun Woo Lee, Hyunchae Cynn, Magnus J Lipp, and Ken Visbeck. Dynamic diamond anvil cell (ddac): A novel device for studying the dynamic-pressure properties of materials. *Review of Scientific Instruments*, 78(7):073904, 2007.

[23] Haoyi Fan, Fengbin Zhang, and Yue Gao. Self-supervised time series representation learning by inter-intra relational reasoning. *arXiv preprint arXiv:2011.13548*, 2020.

[24] Hans Fangohr, Marijan Beg, V Bondar, S Aplin, A Barty, M Kuhn, V Mariani, and Thomas Kluyver. Data analysis support in karabo at european xfel. 2017.

[25] J Filik, AW Ashton, PCY Chang, PA Chater, SJ Day, M Drakopoulos, MW Gerring, ML Hart, OV Magdysyuk, S Michalik, et al. Processing two-dimensional x-ray diffraction and small-angle scattering data in dawn 2. *Journal of applied crystallography*, 50(3):959–966, 2017.

[26] Jonathan A Fine, Anand A Rajasekar, Krupal P Jethava, and Gaurav Chopra. Spectral deep learning for prediction and prospective validation of functional groups. *Chemical Science*, 11(18):4618–4630, 2020.

[27] Rebecca A Fischer, Andrew J Campbell, Oliver T Lord, Gregory A Shofner, Przemyslaw Dera, and Vitali B Prakapenka. Phase transition and metallization of feo at high pressures and temperatures. *Geophysical Research Letters*, 38(24), 2011.

[28] Jun Fu, Jing Liu, Haijie Tian, Yong Li, Yongjun Bao, Zhiwei Fang, and Hanqing Lu. Dual attention network for scene segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3146–3154, 2019.

[29] Shang Gao, Arvind Ramanathan, and Georgia Tourassi. Hierarchical convolutional attention networks for text classification. In *Proceedings of The Third Workshop on Representation Learning for NLP*, pages 11–23, 2018.

[30] Vivien Sainte Fare Garnot, Loic Landrieu, Sebastien Giordano, and Nesrine Chehata. Satellite image time series classification with pixel-set encoders and temporal self-attention. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12325–12334, 2020.

[31] Kunal Ghosh, Annika Stuke, Milica Todorović, Peter Bjørn Jørgensen, Mikkel N Schmidt, Aki Vehtari, and Patrick Rinke. Deep learning spectroscopy: Neural networks for molecular excitation spectra. *Advanced science*, 6(9):1801367, 2019.

[32] Beliz Gunel, Jingfei Du, Alexis Conneau, and Ves Stoyanov. Supervised contrastive learning for pre-trained language model fine-tuning. *arXiv preprint arXiv:2011.01403*, 2020.

[33] Michael Gutmann and Aapo Hyvärinen. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 297–304. JMLR Workshop and Conference Proceedings, 2010.

[34] Raia Hadsell, Sumit Chopra, and Yann LeCun. Dimensionality reduction by learning an invariant mapping. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 1735–1742. IEEE, 2006.

[35] Nils Y Hammerla, Shane Halloran, and Thomas Plötz. Deep, convolutional, and recurrent models for human activity recognition using wearables. *arXiv preprint arXiv:1604.08880*, 2016.

[36] Jun Han and Claudio Moraga. The influence of the sigmoid function parameters on the speed of backpropagation learning. In *From Natural to Artificial Neural Computation: International Workshop on Artificial Neural Networks Malaga-Torremolinos, Spain, June 7–9, 1995 Proceedings 3*, pages 195–201. Springer, 1995.

[37] John A Hartigan and Manchek A Wong. Algorithm as 136: A k-means clustering algorithm. *Journal of the royal statistical society. series c (applied statistics)*, 28(1):100–108, 1979.

[38] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9729–9738, 2020.

[39] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. *IEEE transactions on pattern analysis and machine intelligence*, 37(9):1904–1916, 2015.

[40] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[41] Marti A. Hearst, Susan T Dumais, Edgar Osuna, John Platt, and Bernhard Scholkopf. Support vector machines. *IEEE Intelligent Systems and their applications*, 13(4):18–28, 1998.

[42] Chi-Sing Ho, Neal Jean, Catherine A Hogan, Lena Blackmon, Stefanie S Jeffrey, Mark Holodniy, Niaz Banaei, Amr AE Saleh, Stefano Ermon, and Jennifer Dionne. Rapid identification of pathogenic bacteria using raman spectroscopy and deep learning. *Nature communications*, 10(1):4927, 2019.

[43] Xuewen Hou, Guangli Wang, Xin Wang, Xinmin Ge, Yiren Fan, and Shengdong Nie. Convolutional neural network based approach for classification of edible oils using low-field nuclear magnetic resonance. *Journal of Food Composition and Analysis*, 92:103566, 2020.

[44] Alexandr Ignatenko, Dameli Assalauova, Sergey A Bobkov, Luca Gelisio, Anton B Teslyuk, Viacheslav A Ilyin, and Ivan A Vartanyants. Classification of diffraction patterns in single particle imaging experiments performed at x-ray free-electron lasers using a convolutional neural network. *Machine Learning: Science and Technology*, 2(2):025014, 2021.

[45] Roberto Interdonato, Dino Ienco, Raffaele Gaetano, and Kenji Ose. Duplo: A dual view point deep learning architecture for time series classification. *ISPRS journal of photogrammetry and remote sensing*, 149:91–104, 2019.

[46] Hassan Ismail Fawaz, Germain Forestier, Jonathan Weber, Lhassane Idoumghar, and Pierre-Alain Muller. Deep learning for time series classification: a review. *Data mining and knowledge discovery*, 33(4):917–963, 2019.

[47] Ashish Jaiswal, Ashwin Ramesh Babu, Mohammad Zaki Zadeh, Debapriya Banerjee, and Fillia Makedon. A survey on contrastive self-supervised learning. *Technologies*, 9(1):2, 2020.

[48] Tony Jebara, Yingbo Song, and Kapil Thadani. Spectral clustering and embedding with hidden markov models. In *Machine Learning: ECML 2007: 18th European Conference on Machine Learning, Warsaw, Poland, September 17-21, 2007. Proceedings 18*, pages 164–175. Springer, 2007.

[49] Zs Jenei, HP Liermann, R Husband, ASJ Méndez, D Pennicard, H Marquardt, EF OâBannon, A Pakhomova, Z Konopkova, K Glazyrin, et al. New dynamic diamond anvil cells for tera-pascal per second fast compression x-ray diffraction experiments. *Review of Scientific Instruments*, 90(6):065114, 2019.

[50] Zewei Ji, Runhan Shi, Jiarui Lu, Fang Li, and Yang Yang. Relmole: Molecular representation learning based on two-level graph similarities. *Journal of Chemical Information and Modeling*, 62(22):5361–5372, 2022.

[51] Hongjie Jia, Shifei Ding, Xinzheng Xu, and Ru Nie. The latest research progress on spectral clustering. *Neural Computing and Applications*, 24:1477–1486, 2014.

[52] Ian T Jolliffe and Jorge Cadima. Principal component analysis: a review and recent developments. *Philosophical transactions of the royal society A: Mathematical, Physical and Engineering Sciences*, 374(2065):20150202, 2016.

[53] Fazle Karim, Somshubra Majumdar, Houshang Darabi, and Shun Chen. Lstm fully convolutional networks for time series classification. *IEEE access*, 6:1662–1669, 2017.

[54] Fazle Karim, Somshubra Majumdar, Houshang Darabi, and Samuel Harford. Multivariate lstm-fcns for time series classification. *Neural networks*, 116:237–245, 2019.

[55] Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. Supervised contrastive learning. *Advances in neural information processing systems*, 33:18661–18673, 2020.

[56] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[57] Matthew S Kirschner, Benjamin T Diroll, Peijun Guo, Samantha M Harvey, Waleed Helweh, Nathan C Flanders, Alexandra Brumberg, Nicolas E Watkins, Ariel A Leonard, Austin M Evans, et al. Photoinduced, reversible phase transitions in all-inorganic perovskite nanocrystals. *Nature communications*, 10(1):504, 2019.

[58] Brian Kulis et al. Metric learning: A survey. *Foundations and Trends® in Machine Learning*, 5(4):287–364, 2013.

[59] Markus Kuster, Djelloul Boukhelef, Mattia Donato, J-S Dambietz, Steffen Hauf, Luis Maia, Natascha Raab, Janusz Szuba, Monica Turcato, Krzysztof Wrona,

et al. Detectors and calibration concept for the european xfel. *Synchrotron radiation news*, 27(4):35–38, 2014.

[60] Guokun Lai, Wei-Cheng Chang, Yiming Yang, and Hanxiao Liu. Modeling long- and short-term temporal patterns with deep neural networks. In *The 41st international ACM SIGIR conference on research & development in information retrieval*, pages 95–104, 2018.

[61] Francesca Lazzeri. *Machine learning for time series forecasting with Python*. John Wiley & Sons, 2020.

[62] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

[63] R Letoullec, JP Pinceaux, and P Loubeyre. The membrane diamond anvil cell: a new device for generating continuous pressure and temperature variations. *International Journal of High Pressure Research*, 1(1):77–90, 1988.

[64] Zachary C Lipton, David C Kale, Charles Elkan, and Randall Wetzel. Learning to diagnose with lstm recurrent neural networks. *arXiv preprint arXiv:1511.03677*, 2015.

[65] Ruonan Liu, Fei Wang, Boyuan Yang, and S Joe Qin. Multiscale kernel based residual convolutional neural network for motor fault diagnosis under nonstationary conditions. *IEEE Transactions on Industrial Informatics*, 16(6):3797–3806, 2019.

[66] Charlotte Loh, Thomas Christensen, Rumen Dangovski, Samuel Kim, and Marin Soljačić. Surrogate-and invariance-boosted contrastive learning for data-scarce applications in science. *Nature Communications*, 13(1):4223, 2022.

[67] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016.

[68] Jiawei Ma, Zheng Shou, Alireza Zareian, Hassan Mansour, Anthony Vetro, and Shih-Fu Chang. Cdsa: cross-dimensional self-attention for multivariate, geo-tagged time series imputation. *arXiv preprint arXiv:1905.09904*, 2019.

[69] Raghvendra Mall, Rocco Langone, and Johan AK Suykens. Kernel spectral clustering for big data networks. *Entropy*, 15(5):1567–1586, 2013.

[70] Makoto Matsumoto and Takuji Nishimura. Mersenne twister: a 623-dimensionally equidistributed uniform pseudo-random number generator. *ACM Transactions on Modeling and Computer Simulation (TOMACS)*, 8(1):3–30, 1998.

[71] Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018.

[72] Bjoern H Menze, B Michael Kelm, Ralf Masuch, Uwe Himmelreich, Peter Bachert, Wolfgang Petrich, and Fred A Hamprecht. A comparison of random

forest and its gini importance with standard chemometric methods for the feature selection and classification of spectral data. *BMC bioinformatics*, 10:1–16, 2009.

[73] Puneet Mishra, Dário Passos, Federico Marini, Junli Xu, Jose M Amigo, Aoife A Gowen, Jeroen J Jansen, Alessandra Biancolillo, Jean Michel Roger, Douglas N Rutledge, et al. Deep learning for near-infrared spectral data modelling: Hypes and benefits. *TrAC Trends in Analytical Chemistry*, page 116804, 2022.

[74] Fionn Murtagh and Pierre Legendre. Wardâs hierarchical agglomerative clustering method: which algorithms implement wardâs criterion? *Journal of classification*, 31:274–295, 2014.

[75] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 807–814, 2010.

[76] Tan Nian, Sun Yi-dan, Wang Xue-shun, Huang An-min, and Xie Bing-feng. Research on near infrared spectrum with principal component analysis and support vector machine for timber identification. *Spectroscopy and Spectral Analysis*, 37(11):3370–3374, 2017.

[77] Kriti Ohri and Mukesh Kumar. Review on self-supervised image recognition using deep neural networks. *Knowledge-Based Systems*, 224:107090, 2021.

[78] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

[79] Felipe Oviedo, Zekun Ren, Shijing Sun, Charles Settens, Zhe Liu, Noor Titan Putri Hartono, Savitha Ramasamy, Brian L DeCost, Siyu IP Tian, Giuseppe Romano, et al. Fast and interpretable classification of small x-ray diffraction datasets using data augmentation and deep neural networks. *npj Computational Materials*, 5(1):60, 2019.

[80] Haruka Ozawa, Futoshi Takahashi, Kei Hirose, Yasuo Ohishi, and Naohisa Hirao. Phase transition of feo and stratification in earthâs outer core. *Science*, 334(6057):792–794, 2011.

[81] J Padarian, B Minasny, and AB McBratney. Using deep learning to predict soil properties from regional spectral data. *Geoderma Regional*, 16:e00198, 2019.

[82] Massimiliano Patacchiola and Amos J Storkey. Self-supervised relational reasoning for representation learning. *Advances in Neural Information Processing Systems*, 33:4003–4014, 2020.

[83] Charlotte Pelletier, Geoffrey I Webb, and François Petitjean. Temporal convolutional neural network for the classification of satellite image time series. *Remote Sensing*, 11(5):523, 2019.

[84] David Pennicard, S Smoljanin, F Pithan, M Sarajlic, A Rothkirch, Y Yu, HP Liermann, W Morgenroth, B Winkler, Z Jenei, et al. Lambda 2m gaasâa multimegapixel hard x-ray detector for synchrotrons. *Journal of Instrumentation*, 13(01):C01026, 2018.

[85] Vanessa K Peterson, Josie E Auckett, and W-K Pang. Real-time powder diffraction studies of energy materials under non-equilibrium conditions. *IUCrJ*, 4(5):540–554, 2017.

[86] Christian Plückthun. Investigating the effect of the compression rate on the kinetic response of diamond anvil cell experiments. Technical report, Universität Rostock, 2022.

[87] Michael S Primrose, Jay Giblin, Christian Smith, Martin R Anguita, and Gabriel H Weedon. One dimensional convolutional neural networks for spectral analysis. In *Algorithms, Technologies, and Applications for Multispectral and Hyperspectral Imaging XXVIII*, volume 12094, pages 98–108. SPIE, 2022.

[88] Oldrich Renner and FB Rosmej. Challenges of x-ray spectroscopy in investigations of matter under extreme conditions. *Matter and Radiation at Extremes*, 4(2):024201, 2019.

[89] Peter J Rousseeuw. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics*, 20:53–65, 1987.

[90] Marc Rußwurm and Marco Körner. Self-attention for raw optical satellite time series classification. *ISPRS journal of photogrammetry and remote sensing*, 169:421–435, 2020.

[91] Adam Santoro, David Raposo, David G Barrett, Mateusz Malinowski, Razvan Pascanu, Peter Battaglia, and Timothy Lillicrap. A simple neural network module for relational reasoning. *Advances in neural information processing systems*, 30, 2017.

[92] Jürgen Schmidhuber, Sepp Hochreiter, et al. Long short-term memory. *Neural Comput*, 9(8):1735–1780, 1997.

[93] Joan Serrà, Santiago Pascual, and Alexandros Karatzoglou. Towards a universal neural network encoder for time series. In *CCIA*, pages 120–129, 2018.

[94] Sherjeel Shabih, Markus Kühbach, Markus Scheidgen, Lauri Himanen, Sandor Brockhauser, Benedikt Haas, and Christoph Koch. Development of a fair data management infrastructure. *Microscopy and Microanalysis*, 28(S1):2930–2932, 2022.

[95] Jian-Li Shao, Pei Wang, Feng-Guo Zhang, and An-Min He. Hcp/fcc nucleation in bcc iron under different anisotropic compressions at high strain rate: Molecular dynamics study. *Scientific Reports*, 8(1):1–10, 2018.

[96] Guoyin Shen and Ho Kwang Mao. High-pressure studies with x-rays using diamond anvil cells. *Reports on Progress in Physics*, 80(1):016101, 2016.

[97] Shun-Yao Shih, Fan-Keng Sun, and Hung-yi Lee. Temporal pattern attention for multivariate time series forecasting. *Machine Learning*, 108:1421–1441, 2019.

[98] Sameer Singh. Quantifying structural time varying changes in helical data. *Neural Computing & Applications*, 10:148–154, 2001.

[99] Gaurav Singhal, Babankumar Bansod, Lini Mathew, Jonali Goswami, BU Choudhury, and PLN Raju. Chlorophyll estimation using multi-spectral unmanned aerial system based on machine learning techniques. *Remote Sensing Applications: Society and Environment*, 15:100235, 2019.

[100] Weiran Song, Hui Wang, Paul Maguire, and Omar Nibouche. Nearest clusters based partial least squares discriminant analysis for the classification of spectral data. *Analytica chimica acta*, 1009:27–38, 2018.

[101] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.

[102] George Stein, Peter Harrington, Jacqueline Blaum, Tomislav Medan, and Zarija Lukic. Self-supervised similarity search for large scientific datasets. *arXiv preprint arXiv:2110.13151*, 2021.

[103] Alexander Strehl and Joydeep Ghosh. Cluster ensembles—a knowledge reuse framework for combining multiple partitions. *Journal of machine learning research*, 3(Dec):583–617, 2002.

[104] Yue Sun and Sandor Brockhauser. Machine learning applied for spectra classification in x-ray free electorn laser sciences. *Data Science Journal*, 21(1), 2022.

[105] Yue Sun, Sandor Brockhauser, and Péter Hegedűs. Comparing end-to-end machine learning methods for spectra classification. *Applied Sciences*, 11(23):11520, 2021.

[106] Yue Sun, Sandor Brockhauser, and Péter Hegedűs. Machine learning applied for spectra classification. In *Computational Science and Its Applications–ICCSA 2021: 21st International Conference, Cagliari, Italy, September 13–16, 2021, Proceedings, Part IX 21*, pages 54–68. Springer, 2021.

[107] Yue Sun, Sandor Brockhauser, Péter Hegedűs, Christian Plückthun, Luca Gelisio, and Danilo Enoque Ferreira de Lima. Application of self-supervised approaches to the classification of x-ray diffraction spectra during phase transitions. *Scientific Reports*, 13(1):9370, 2023.

[108] Yue Sun, Sandor Brockhauser, Péter Hegedűs, Christian Plückthun, Luca Gelisio, and Danilo Enoque Ferreira de Lima. Application of self-supervised approaches to the classification of x-ray diffraction spectra during phase transitions. *Scientific Reports*, 13(1):9370, 2023.

[109] Yue Sun, Christian Plückthun, Sandor Brockhauser, and Péter Hegedűs. An interactive machine learning application for spectra classification. In *Computational Science and Its Applications–ICCSA 2023: 23st International Conference*, pages Accepted, to appear. IEEE Computer Society, 2023.

[110] Flood Sung, Yongxin Yang, Li Zhang, Tao Xiang, Philip HS Torr, and Timothy M Hospedales. Learning to compare: Relation network for few-shot learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1199–1208, 2018.

[111] Yuta Suzuki, Hideitsu Hino, Takafumi Hawai, Kotaro Saito, Masato Kotsugi, and Kanta Ono. Symmetry prediction and knowledge discovery from x-ray diffraction patterns using an interpretable machine learning approach. *Scientific reports*, 10(1):1–11, 2020.

[112] Nathan J Szymanski, Christopher J Bartel, Yan Zeng, Qingsong Tu, and Gerbrand Ceder. Probabilistic deep learning approach to automate the interpretation of multi-phase diffraction spectra. *Chemistry of Materials*, 33(11):4204–4215, 2021.

[113] Chang Wei Tan, Christoph Bergmeir, Francois Petitjean, and Geoffrey I Webb. Monash university, uea, ucr time series extrinsic regression archive. *arXiv preprint arXiv:2006.10996*, 2020.

[114] Chang Wei Tan, Christoph Bergmeir, Francois Petitjean, and Geoffrey I Webb. Monash university, uea, ucr time series extrinsic regression archive. *arXiv preprint arXiv:2006.10996*, 2020.

[115] Terry T Um, Franz MJ Pfister, Daniel Pichler, Satoshi Endo, Muriel Lang, Sandra Hirche, Urban Fietzek, and Dana Kulić. Data augmentation of wearable sensor data for parkinsonâs disease monitoring using convolutional neural networks. In *Proceedings of the 19th ACM international conference on multimodal interaction*, pages 216–220, 2017.

[116] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

[117] Greg Van Houdt, Carlos Mosquera, and Gonzalo Nápoles. A review on the long short-term memory model. *Artificial Intelligence Review*, 53:5929–5955, 2020.

[118] Jake VanderPlas. *Python data science handbook: Essential tools for working with data*. " O'Reilly Media, Inc.", 2016.

[119] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[120] Ulrike Von Luxburg. A tutorial on spectral clustering. *Statistics and computing*, 17:395–416, 2007.

[121] Stefan Wagner, Klaus Lochmann, Sebastian Winter, Andreas Goeba, and Michael Klaes. Quality models in practice: A preliminary analysis. In *ESEM '09 Proceedings of the 2009 3rd International Symposium on Empirical Software Engineering and Measurement*, Washington, DC, USA, 2009. IEEE Computer Society.

[122] Hong Wang, Yunchao Xie, Dawei Li, Heng Deng, Yunxin Zhao, Ming Xin, and Jian Lin. Rapid identification of x-ray diffraction patterns based on very limited data by interpretable convolutional neural networks. *Journal of chemical information and modeling*, 60(4):2004–2011, 2020.

[123] Peng Wang, Liangsheng Guo, Yubing Tian, Jiansheng Chen, Shan Huang, Ce Wang, Pengli Bai, Daqing Chen, Weipei Zhu, Hongbo Yang, et al. Discrimination of blood species using raman spectroscopy combined with a recurrent neural network. *OSA continuum*, 4(2):672–687, 2021.

[124] Yuyang Wang, Jianren Wang, Zhonglin Cao, and Amir Barati Farimani. Molecular contrastive learning of representations via graph neural networks. *Nature Machine Intelligence*, 4(3):279–287, 2022.

[125] Zhiguang Wang, Weizhong Yan, and Tim Oates. Time series classification from scratch with deep neural networks: A strong baseline. In *2017 International joint conference on neural networks (IJCNN)*, pages 1578–1585. IEEE, 2017.

[126] Scott White and Padhraic Smyth. A spectral clustering approach to finding communities in graphs. In *Proceedings of the 2005 SIAM international conference on data mining*, pages 274–285. SIAM, 2005.

[127] Neo Wu, Bradley Green, Xue Ben, and Shawn O'Banion. Deep transformer models for time series forecasting: The influenza prevalence case. *arXiv preprint arXiv:2001.08317*, 2020.

[128] Zhirong Wu, Yuanjun Xiong, Stella X Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance discrimination. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3733–3742, 2018.

[129] Rikiya Yamashita, Mizuho Nishio, Richard Kinh Gian Do, and Kaori Togashi. Convolutional neural networks: an overview and application in radiology. *Insights into imaging*, 9:611–629, 2018.

[130] Chuanguang Yang, Zhulin An, Linhang Cai, and Yongjun Xu. Mutual contrastive learning for visual representation learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 3045–3053, 2022.

[131] Jie Yang, Jinfan Xu, Xiaolei Zhang, Chiyu Wu, Tao Lin, and Yibin Ying. Deep learning for vibrational spectral analysis: Recent progress and a practical guide. *Analytica chimica acta*, 1081:6–17, 2019.

[132] Omolbanin Yazdanbakhsh and Scott Dick. Multivariate time series classification using dilated convolutional neural network. *arXiv preprint arXiv:1905.01697*, 2019.

[133] George Zerveas, Srideepika Jayaraman, Dhaval Patel, Anuradha Bhamidipaty, and Carsten Eickhoff. A transformer-based framework for multivariate time series representation learning. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 2114–2124, 2021.

[134] Shichao Zhang, Xuelong Li, Ming Zong, Xiaofeng Zhu, and Debo Cheng. Learning k for knn classification. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 8(3):1–19, 2017.

[135] YY Zhang, YX Li, D Fan, NB Zhang, JW Huang, MX Tang, Y Cai, XL Zeng, T Sun, K Fezzaa, et al. Ultrafast x-ray diffraction visualization of b 1- b

2 phase transition in kcl under shock compression. *Physical Review Letters*, 127(4):045702, 2021.

[136] Bendong Zhao, Huanzhang Lu, Shangfeng Chen, Junliang Liu, and Dongya Wu. Convolutional neural networks for time series classification. *Journal of Systems Engineering and Electronics*, 28(1):162–169, 2017.

[137] Wenbin Zheng, Xiaping Fu, and Yibin Ying. Spectroscopy-based food classification with extreme learning machine. *Chemometrics and Intelligent Laboratory Systems*, 139:42–47, 2014.

[138] Wenbin Zheng, Hongping Shu, Hong Tang, and Haiqing Zhang. Spectra data classification with kernel extreme learning machine. *Chemometrics and Intelligent Laboratory Systems*, 192:103815, 2019.

[139] Liheng Zhong, Lina Hu, and Hang Zhou. Deep learning based multi-temporal crop classification. *Remote sensing of environment*, 221:430–443, 2019.

[140] Bolei Zhou, Alex Andonian, Aude Oliva, and Antonio Torralba. Temporal relational reasoning in videos. In *Proceedings of the European conference on computer vision (ECCV)*, pages 803–818, 2018.

[141] Yulong Zhuang, Salah Awel, Anton Barty, Richard Bean, Johan Bielecki, Martin Bergemann, Benedikt J Daurer, Tomas Ekeberg, Armando D Estillore, Hans Fangohr, et al. Unsupervised learning approaches to characterizing heterogeneous samples using x-ray single-particle imaging. *IUCrJ*, 9(2):204–214, 2022.

[142] Julian Zimmermann, Fabien Beguet, Daniel Guthruf, Bruno Langbehn, and Daniela Rupp. Finding the semantic similarity in single-particle diffraction images using self-supervised contrastive projection learning. *npj Computational Materials*, 9(1):24, 2023.

[143] Julian Zimmermann, Bruno Langbehn, Riccardo Cucini, Michele Di Fraia, Paola Finetti, Aaron C LaForge, Toshiyuki Nishiyama, Yevheniy Ovcharenko, Paolo Piseri, Oksana Plekan, et al. Deep neural networks for classifying complex features in diffraction images. *Physical Review E*, 99(6):063309, 2019.

[144] Patric Zimmermann, Sergey Peredkov, Paula Macarena Abdala, Serena DeBeer, Moniek Tromp, Christoph Müller, and Jeroen A van Bokhoven. Modern x-ray spectroscopy: Xas and xes in the laboratory. *Coordination Chemistry Reviews*, 423:213466, 2020.