Pontifícia Universidade Católica
do Rio de Janeiro

João Gonçalves Neto

Subverting the Conventional FDD Sequence: A Diagnosis-First Approach with Deep Learning

Tese de Doutorado

Thesis presented to the Programa de Pós–graduação em Engenharia Química, de Materiais e Processos Ambientais, do Departamento de Engenharia Química e de Materiais da PUC-Rio in partial fulfillment of the requirements for the degree of Doutor em Engenharia Química, de Materiais e Processos Ambientais.

Advisor : Profa. Amanda Lemette Teixeira Brandão
Co-advisor: Profa. Karla Tereza Figueiredo Leite

Rio de Janeiro
April 2025

Pontifícia Universidade Católica
do Rio de Janeiro

João Gonçalves Neto

Subverting the Conventional FDD Sequence: A Diagnosis-First Approach with Deep Learning

Thesis presented to the Programa de Pós–graduação em Engenharia Química, de Materiais e Processos Ambientais da PUC-Rio in partial fulfillment of the requirements for the degree of Doutor em Engenharia Química, de Materiais e Processos Ambientais. Approved by the Examination Committee:

Profa. Amanda Lemette Teixeira Brandão
Advisor
Pontifícia Universidade Católica do Rio de Janeiro – PUC-Rio

Profa. Karla Tereza Figueiredo Leite
Universidade Estadual do Rio de Janeiro - UERJ

Profa. Andréa Pereira Parente
Universidade Federal do Rio de Janeiro - UFRJ

Prof. João Batista de Paiva Soares
University of Alberta - UofA

Profa. Manoela Rabello Kohler
Pontifícia Universidade Católica do Rio de Janeiro – PUC-Rio

Prof. Maurício Bezerra de Souza Junior
Universidade Federal do Rio de Janeiro - UFRJ

Rio de Janeiro, April the 24th, 2025

João Gonçalves Neto

Majored in Chemical Engineering by the Pontifical University of Rio de Janeiro (Rio de Janeiro, Brazil). Masters in General Engineering by the École Centrale de Marseille (Marseille, France) and in Materials, Metallurgical and Chemical Processes Engineering focused on Environmental and Processes Engineering by the Pontifical University of Rio de Janeiro (Rio de Janeiro, Brazil).

Bibliographic data

Neto, João Gonçalves

Subverting the Conventional FDD Sequence: A Diagnosis-First Approach with Deep Learning / João Gonçalves Neto; advisor: Amanda Lemette Teixeira Brandão; co-advisor: Karla Tereza Figueiredo Leite. – 2025.

116 f: il. color. ; 30 cm

Tese (doutorado) - Pontifícia Universidade Católica do Rio de Janeiro, Departamento de Engenharia Química e de Materiais, 2025.

Inclui bibliografia

1. Engenharia Química e de Materiais – Teses. 2. Estrutura de aprendizado profundo. 3. Otimização de hiperparâmetros. 4. Classificação de falhas. 5. Abordagem de aprendizado por transferência. 6. Monitoramento de processos industriais. I. Brandão, Amanda Lemette Teixeira. II. Tereza Figueiredo Leite, Karla. III. Pontifícia Universidade Católica do Rio de Janeiro. Departamento de Engenharia Química e de Materiais. IV. Título.

CDD: 620.11

To those who believe in me,
even when I struggle to believe in myself.

Acknowledgments

To Amanda, who has been my advisor since the very beginning of my academic path. Thanks to her example and mentorship, I discovered a genuine passion for teaching. Her dedication and encouragement have been a constant source of motivation, and I am deeply grateful for the trust she placed in me.

To Karla, whom I had the privilege of meeting partway through my PhD. As her student, I was inspired by her remarkable ability to connect ideas across different fields to create new perspectives. She encouraged me to challenge conventional approaches and broaden my intellectual horizons, for which I am immensely thankful.

To João Soares, whom I met during an exchange program in Canada. I have great admiration for how supportive he is of students, even those who are no longer under his direct supervision. His warm welcome and generosity made me feel truly included, and I greatly appreciated the opportunity to be part of his research environment.

I am also thankful to my research group, GAAP. United by our shared love for coffee (for most of us) and a passion for exchanging ideas, GAAP's collaborative spirit made even the most challenging moments more enjoyable.

To my family, whose unconditional love and support have carried me through every stage of this journey. Their constant presence reminds me to take life one day at a time and gives me the strength to keep moving forward.

To my friends in the GGG, thank you for embracing my peculiarities and for keeping my spirits high throughout the many ups and downs. Your friendship is an invaluable source of joy and laughter. I am lucky to have you by my side.

I also want to thank Annita Fidalgo, a dear friend since my very first day of university. Our enduring friendship has provided me with constant encouragement, no matter the distance.

I appreciate the financial support through the Human Resources Program of the National Agency of Petroleum, Natural Gas and Biofuels – PRH-ANP, and thank FAPESP for overseeing this project. This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001.

Abstract

Neto, João Gonçalves; Brandão, Amanda Lemette Teixeira (Advisor); Tereza Figueiredo Leite, Karla (Co-Advisor). **Subverting the Conventional FDD Sequence: A Diagnosis-First Approach with Deep Learning**. Rio de Janeiro, 2025. 116p. Tese de Doutorado – Departamento de Engenharia Química e de Materiais, Pontifícia Universidade Católica do Rio de Janeiro.

Ensuring the safety and efficiency of chemical processes requires robust Fault Detection and Diagnosis (FDD) systems capable of detecting and distinguishing abnormal conditions in complex industrial environments. Traditional model-based approaches often struggle with chemical plants' nonlinear and dynamic nature, making data-driven methods a particularly appealing alternative. This thesis presents a two-stage deep learning framework for fault detection and diagnosis using the Tennessee Eastman Process as the case study. In the first stage, a Convolutional Neural Network trained on Gramian Angular Summation Fields representations is used for fault diagnosis, achieving state-of-the-art classification performance. We improved the diagnosis model's accuracy by systematically refining a single deep learning architecture instead of testing multiple alternatives. In the second stage, we extend this work to fault detection, reversing the conventional order of detection preceding diagnosis. We developed a detection model based on Siamese Neural Network architecture using transfer learning from the best-performing diagnosis model for its backbone. This approach allowed for a structured exploration of model modifications, including freezing convolutional layers to preserve transferred knowledge, optimizing hyperparameters for training stability, and analyzing gradient norms to enhance learning dynamics. The results were benchmarked against existing literature, showing the effectiveness of a focused architectural refinement strategy in industrial FDD applications and providing a foundation for further advancements in deep learning-based process monitoring.

Keywords

Deep learning framework; Hyperparameter optimization; Fault classification; Transfer learning; Industrial process monitorin.

Resumo

Neto, João Gonçalves; Brandão, Amanda Lemette Teixeira; Tereza Figueiredo Leite, Karla. **Subvertendo a Sequência Convencional de DDF: Uma Abordagem Iniciada com Diagnóstico e Aprendizado Profundo**. Rio de Janeiro, 2025. 116p. Tese de Doutorado – Departamento de Engenharia Química e de Materiais, Pontifícia Universidade Católica do Rio de Janeiro.

Garantir a segurança e a eficiência de processos químicos exige sistemas robustos de Detecção e Diagnóstico de Falhas (DDF) capazes de diferenciar condições anormais em ambientes industriais complexos. Nesse contexto, abordagens tradicionais de modelagem frequentemente tornam-se impraticáveis devido à natureza não linear, à quantidade de variáveis e à dinâmica de plantas químicas, o que torna métodos baseados em dados históricos uma alternativa particularmente promissora. Neste trabalho, é apresentado um modelo de aprendizado profundo em duas etapas para detecção e diagnóstico de falhas utilizando o Processo Tennessee Eastman como estudo de caso. Na primeira etapa, uma Rede Neural Convolucional foi treinada em representações do tipo Gramian Angular Summation Fields para diagnóstico de falhas, alcançando métricas de classificação equiparáveis aos reportados no estado da arte. Em vez de investigar múltiplas arquiteturas, foi feita uma investigação de forma sistemática buscando aprimorar uma arquitetura base da literatura. A segunda etapa focou na detecção de falhas, invertendo a ordem convencional de desenvolvimento começando por detecção seguida de diagnóstico. O modelo de detecção foi baseado na arquitetura de Redes Neurais Siamesas, utilizando aprendizado por transferência a partir do modelo de diagnóstico de melhor desempenho como backbone. Essa abordagem possibilitou uma exploração estruturada de modificações no modelo base, incluindo o congelamento progressivo de camadas convolucionais para preservar o conhecimento transferido; a otimização de hiperparâmetros para maior estabilidade no treinamento e a análise das normas dos gradientes para melhorar a dinâmica de aprendizado. Os resultados foram comparados com a literatura existente, demonstrando a eficácia da estratégia investigada para aplicações industriais de DDF e fornecendo uma base para avanços futuros no monitoramento de processos com aprendizado profundo.

Palavras-chave

Estrutura de aprendizado profundo; Otimização de hiperparâmetros; Classificação de falhas; Abordagem de aprendizado por transferência; Monitoramento de processos industriais.

Table of contents

List of figures

List of tables

List of Abreviations

AE – Autoencoder

CNN – Convolutional Neural Network

DDF – Detecção e Diagnóstico de Falhas

DL – Deep Learning

DR – Detection Rate

FDD – Fault Detection and Diagnosis

FDR – Fault Detection Rate

GAF – Gramian Angular Fields

GASF – Gramian Angular Summation Fields

JTSVA – Joint Time-Serial Variation Analysis

LSTM – Long Short-Term Memory

MOLA – Multi-Block Orthogonal Long Short-Term Memory Autoencoder

OOD – out-of-distribution

SNN – Siamese Neural Network

t-SNE – t-Distributed Stochastic Neighbor Embedding

Johnny stared at the compass that he didn't realize he was gripping rather tightly. It no longer looked silver in appearance, it started to peel and reveal a black titanium underneath the silver plating. He was given one last command, "Follow it..." before the tightening in his chest disappeared.

Unpublished Work of Chloe Williamson, *The Timekeeper.*

1
General Introduction

1.1
General Contextualization

Integrating deep learning (DL) into chemical engineering has opened new possibilities for addressing complex challenges, especially in the context of fault detection and diagnosis (FDD). Industrial processes are inherently dynamic and nonlinear, making them prone to operational anomalies compromising safety, efficiency, and product quality. Traditional FDD methods often struggle with these complexities due to the high dimensionality of chemical plants. In contrast, data-driven approaches use process data to identify patterns, offering solutions aligned with current industry characteristics [4, 5, 6, 7, 8, 9, 10, 11].

However, practical challenges persist. Many industrial datasets are imbalanced, with limited fault examples compared to normal operations. Additionally, multivariate time-series data from processes exhibit intricate temporal dependencies and cross-variable interactions, complicating fault identification [12, 13, 14, 15, 16, 17, 18]. While DL methods show promise, their effective application requires addressing data structure, computational efficiency, and adaptability to diverse fault scenarios.

1.2
Motivation and Objectives

This thesis focuses on advancing data-driven FDD methodologies for industrial processes through two interconnected studies.

The first study consists of fault diagnosis via deep learning architecture. General machine learning studies often prioritize evaluating multiple model types over deeply exploring a specific structure. This work focuses on refining a single DL architecture to further investigate its potential. We transformed multivariate time-series data into 2D representations through Gramian Angular Summation Fields (GASF). This transformation allowed us to use Convolutional Neural Networks to process high-dimensional data from common fault scenarios in chemical engineering. The key objectives included:

- Proposing a more efficient approach to investigate deep learning solutions for fault diagnosis.
- Addressing the high-dimensionality problem in fault diagnosis through feature learning.
- Focusing on an single model's structure to optimize performance.
- Improving fault classification performance in the TEP benchmark.

The second study is fault detection via Transfer Learning and Siamese Networks (SNNs). We propose an innovative detection framework using SNNs with a pre-trained diagnosis model as its backbone to overcome the inconsistent feature space distribution of the anomalous scenario class. By reformulating detection as an embedding similarity task, this method enhances generalization. The main objectives were the following.

- Investigating a new framework for the development of fault detection deep learning models.
- Evaluating the combination of the SNN architecture with the best CNN model from the first case study.
- Taking advantage of the knowledge already learned by the diagnosis model for the detection task
- Improving fault detection in the TEP benchmark.

Together, these studies aim to bridge the gap between DL advancements from different fields and practical industrial requirements, offering a framework for model development.

1.3 Outline

This thesis is structured as follows:

- **Chapter 1** briefly presents the research context of FDD in industrial processes, establishes the study's motivations, and outlines the objectives further discussed in other chapters.
- **Chapter 2** consists of the published article "Can Focusing on One Deep Learning Architecture Improve Fault Diagnosis Performance?" which focuses on the investigation of the diagnosis model [3].
- **Chapter 3** consists of the submitted article "A Diagnosis-based Siamese Network for Fault Detection Through Transfer Learning" which explores the detection part of the investigation.

– **Chapter 4** summarizes the main findings, discusses their implications, and suggests directions for future research.

The articles in Chapters 2 and 3 are presented in their complete form as developed during this doctoral research. These chapters begin with their respective abstracts to provide readers with immediate context for the work that follows. The first article was published as open access, ensuring a wide dissemination of the research findings. The second article was also submitted as open access and is being considered for publication at the time of writing this document.

2
Article 1 - Can Focusing on One Deep Learning Architecture Improve Fault Diagnosis Performance?

Machine learning approaches often involve evaluating a wide range of models due to various available architectures. This standard strategy can lead to a lack of depth in exploring established methods. In this study, we concentrated our efforts on a single deep learning architecture type to assess whether a focused approach could enhance performance in fault diagnosis. We selected the benchmark Tennessee Eastman Process dataset as our case study and investigated modifications on a reference Convolutional Neural Network-based model. Results indicate a considerable improvement in the overall classification, reaching a maximum average F1-score of 89.85 %, 7.47 % above the baseline model, which is also a considerable improvement compared to other performances reported in the Literature. These results emphasize the potential of this focused approach, indicating it could be further explored and applied to other datasets in future work.

2.1
Introduction

In machine learning studies, it is common practice to evaluate multiple models, given the vast array of available and emerging architectures. For instance, Neural Architecture Search can be used to automate the design of optimal neural networks, minimizing human involvement [19, 20, 21]. However, when adopting Deep Learning (DL) alternatives, high computational costs and long processing times can limit architecture exploration. Since no single model consistently provides an optimal solution, we focused our efforts on investigating a single type of DL architecture to address the question: can this focused approach improve performance in fault diagnosis?

Observable deviations from designed operating conditions often precede accidents in chemical processes, providing early indicators of potential failures. These abnormal behaviors are related to a series of process failures usually triggered by a sensor, actuator, or another specific system fault [22, 23]. Pinpointing the original fault is not an evident task depending on the industrial plant's complexity, nonlinearity, and process dynamics. Therefore, Fault

Detection and Diagnosis (FDD) is the field of study dedicated to detecting anomalous system behaviors and diagnosing the root fault, which is responsible for the aforementioned condition [24, 5, 25].

FDD is of extreme importance in chemical processes to ensure personnel safety, protect the environment, prevent costly shutdowns, and mitigate risks associated with abnormal situations, which can compromise operational stability and lead to cascading system failures [26, 22]. Modern chemical plants are highly complex, nonlinear, and dynamic, making FDD particularly challenging [4, 5, 6]. Traditional model-based methods often become impractical due to their dependence on expert knowledge and difficulty in accurately describing such intricate processes [27]. In contrast, data-driven approaches can use real-time monitoring and historical process data, aligning with Industry 4.0 advancements to provide scalable, robust, and adaptable solutions [7, 8]. To address these challenges, we focused on enhancing fault diagnosis performance through a data-driven technique applied to the Tennessee Eastman Process dataset[28].

The TEP plays a crucial role in FDD due to its realistic simulation of complex chemical processes. Its importance stems from its ability to provide a controlled yet intricate environment where various fault scenarios can be systematically introduced and analyzed. TEP allows researchers and practitioners to develop and test advanced FDD algorithms in a setting that mimics real-world challenges without the associated risks and costs of working with actual industrial systems. The TEP's comprehensive range of fault types, including operational disturbances and sensor failures, provides a valuable testbed for evaluating the effectiveness of different diagnostic approaches and refining methodologies to enhance their robustness and accuracy. From a practical standpoint, this case study closely resembles actual industrial processes, which are often complex, costly, and sensitive to failures. [29, 30, 31, 32] For these reasons, we selected it as the case study for our investigation.

Numerous studies have investigated methods to improve performance of fault detection and diagnosis systems [26, 33, 34]. Some approaches use Recurrent Neural Networks to process the time series, such as Long Short-Term Memory networks for FDD modeling [35, 36]. On the other hand, others focus on enhancing data representation through transformations, such as wavelet transforms combined with support vector machine classifiers [37].

Recent advancements in intelligent fault diagnosis emphasize integrating deep learning and domain-specific adaptations to enhance model robustness and trustworthiness. Sun et al. (2023) introduced a domain adaptation-based method through one-dimensional convolutional autoencoders, achieving high

diagnostic accuracy and flexibility across multiple operational conditions [9]. Xie et al. (2023) proposed a unified out-of-distribution (OOD) detection framework combining class-wise outlier detection with supervised learning using a custom loss function composed of cross-entropy loss and contrastive learning loss, improving the reliability of prognostics and health management systems in renewable energy applications [10]. Zhang et al. (2024) developed a deep ensemble approach guided by max-consistency and min-similarity to address diagnostic uncertainty and enhance fault detection in industrial systems [11]. These studies focus on OOD detection, transfer learning, and ensemble strategies in achieving promising perspectives for adaptive fault diagnosis, providing a foundation for incorporating similar methodologies into broader industrial contexts.

In our work, we focused on investigating the Gramian Angular Fields (GAF) approach [38] for time series transformation, followed by Convolutional Neural Networks (CNNs) for feature learning and fault classification based on the study performed by Sun and Ren (2021)[2]. GAF transforms time series into a structured, 2D matrix representation, enabling the application of CNNs to detect patterns across both temporal and variable dimensions. By allowing simultaneous cross-variable analysis through multi-channel inputs, this approach provides a robust framework for identifying intricate fault interactions, even when their timing and expression within the system are highly complex and variable.

We focused on CNNs due to their ability to efficiently process transformed time series data and their capacity to learn features from data. This practical balance between performance, computational cost, and feature learning aligns well with our research's dataset characteristics and goals. While alternative deep learning approaches remain promising and viable options, our decision reflects the alignment of CNNs with the scope of this work.

Following their proposed methodology, we built a baseline model and investigated a series of data preprocessing and architecture modifications to search and evaluate their effects on model performance. Our investigation resulted in a model with an improved F1-score of 89.85 % at a shorter or similar process observation window compared to existing literature.

This chapter is organized as follows: The Dataset Description section provides overviews of the Tennessee Eastman Process dataset and discusses challenges associated with time series analysis. In the Fundamentals of the Model Architecture section, we explain Convolutional Neural Network architectures, show relevant works within the Fault Detection and Diagnosis context, and detail the Gramian Angular Summation Fields data transformation.

The Methodology section outlines the data pretreatment steps, describes the baseline model architecture, details the training parameters used, and specifies the modifications investigated in our study. The Results and Discussion section comprehensively evaluates our approach, addressing its limitations and suggesting potential improvements. Finally, the Conclusion section summarizes our findings and outlines directions for future research.

2.2 Dataset Description

This section introduces the Tennessee Eastman Process dataset, a widely recognized fault detection and diagnosis research benchmark. The dataset provides comprehensive multivariate time series data for evaluating and developing our fault diagnosis models.

2.2.1 Tennessee Eastman Process

The Tennessee Eastman Process (TEP) is a widely recognized simulation of a chemical production process used extensively in process control and fault detection research. Developed by Downs and Vogel in 1993[28], the TEP models a complex chemical plant with two products involving several interconnected reactions, as shown in Figure 2.1. The process operates under varying conditions and includes key features, such as reaction kinetics, heat exchanges, and multiple process disturbances. Its complexity, with numerous operational variables and potential fault scenarios, provides a robust platform for testing and evaluating various control strategies and FDD methods.

2.2.2 Data and Software Availability

The version of the TEP raw data we selected for our study is available on the Kaggle platform [39]. It includes four reactants (A, C, D, and E), an inert compound (B), a by-product (F), and two targeted products (G and F). It contains 52 process variables, of which 12 are manipulated variables, 22 are process measurements, and 18 are component analyses. In addition to normal behavior, this version contains 20 faults as described in Table 2.1. Faults 5, 8, 9, 10, 15, and 16 have been reported as challenging to diagnose [40, 41]. It is important to comment that some databases include a possible 21st fault in the TEP, but the dataset we selected consisted of the original 20 faults.

Since the TEP dataset exhibits intricate multivariate time series characteristics, traditional statistical techniques may not fully capture the temporal

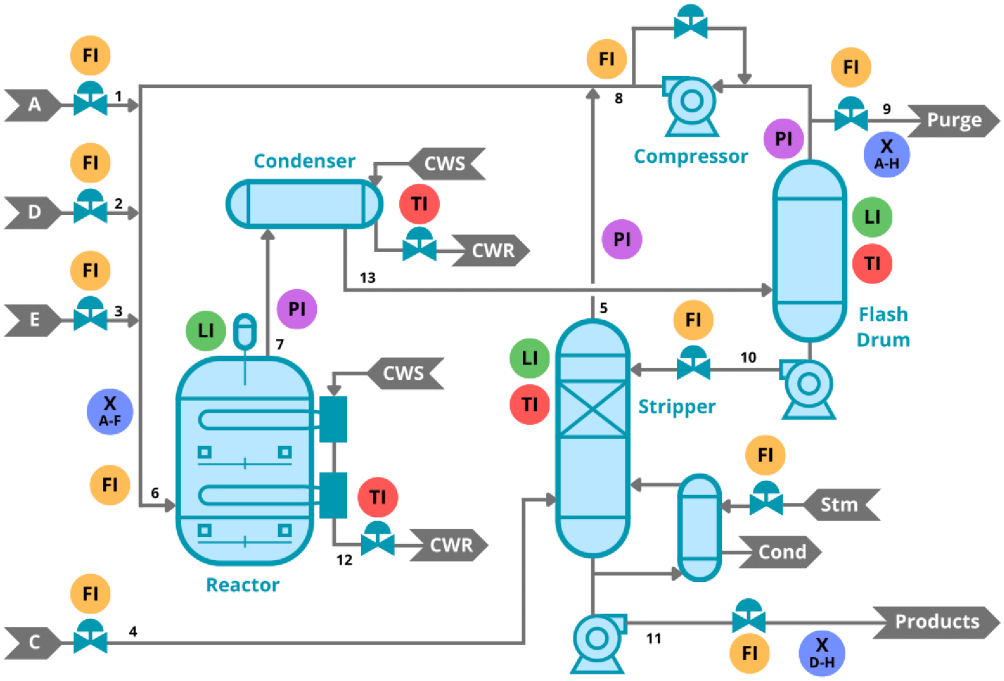

Figure 2.1: Tennessee Eastman Process Diagram. Adapted from Andris Piebalgs (2020)[1]

Table 2.1: List and Description of faults of the TEP.

No.	Description	Type
1	A/C feed ratio, B composition constant (stream 4)	Step
2	B composition, A/C feed ratio constant (stream 4)	Step
3	D feed temperature (stream 2)	Step
4	Reactor cooling water inlet temperature	Step
5	Condenser cooling water inlet temperature	Step
6	A feed loss (stream 1)	Step
7	C header pressure loss-reduced availability (stream 4)	Step
8	A, B, and C feed composition (stream 4)	Random
9	D feed temperature (stream 2)	Random
10	C feed temperature (stream 4)	Random
11	Reactor cooling water inlet temperature	Random
12	Condenser cooling water inlet temperature	Random
13	Reaction kinetics	Slow drift
14	Reactor cooling water value	Sticking
15	Condenser cooling water value	Sticking
16	Unknown	Unknown
17	Unknown	Unknown
18	Unknown	Unknown
19	Unknown	Unknown
20	Unknown	Unknown

dependencies and complex relationships within such data. As we transition to the next section, we consider these potential limitations and explore the need

for alternative approaches to analyze and interpret this type of data effectively.

2.2.3 Limitations of multivariate time series analysis

Statistical techniques such as Regression Analysis and Pearson Correlation [42] are commonly employed for feature screening by characterizing the linear relationships between pairs of variables. These traditional methods are widely used because of their simplicity and effectiveness in many scenarios. However, when applied to serialized or time-dependent data, they can lead to significant information loss or misleading conclusions, which may occur because they do not account for the temporal structure of the data, which is a critical aspect of time series analysis [14].

In the time series context, alternative methods such as autocorrelation and partial autocorrelation analysis provide valuable insights. These techniques help to understand the influence of time lags on feature behavior by examining how past values of a series affect its current state [17]. However, autocorrelation and partial autocorrelation mainly capture linear relationships and rely on the assumption that statistical properties remain constant over time. This can be overly restrictive in real-world data, where nonlinearities, trends, or structural breaks often occur. Moreover, these methods are limited to univariate analysis and do not account for interactions between multiple features.

To extend the analysis, exploring time-lagged cross-correlation can help assess relationships between features across different time delays [18]. Although this method offers insights into the relationships between variables, it can become computationally prohibitive as the number of features and the length of the observation window increase exponentially. This scalability issue poses a significant challenge when dealing with large datasets typical of real-world applications. However, how could we address this challenge?

We opted to retain all variables and investigate a type of model capable of feature learning: the process in which a machine learning model automatically extracts meaningful and relevant features from input data to improve predictions based on high-dimensionality and complex data.

2.3 Fundamentals of the Model Architecture

This section focuses on the model architecture we investigated to address our fault diagnosis challenge. We start with Convolutional Neural Networks, highlighting their role in feature extraction and classification tasks. Then, we introduce Gramian Angular Summation Fields (GASF), which serves as a data

transformation technique that we integrated into our models. By exploring this architecture in detail, we aim to provide a comprehensive understanding of its contributions to enhancing model performance and handling complex time series data.

2.3.1 Convolutional Neural Networks

A Convolutional Neural Network (CNN) is a type of deep learning model particularly well-suited for processing data with a grid-like structure, such as images[43]. It processes the spatial hierarchies in data using convolutional layers, which apply filters to input data to extract features, such as edges, textures, or complex patterns. Mathematically, a convolution operation involves sliding a filter over the input data, computing the dot product between the filter and the portion of the input it covers. The result is a feature map that highlights the presence of specific features in the input. This process can be repeated through multiple convolutional layers, progressively extracting higher-level features from the data[44].

Additionally, pooling layers are often used after convolutional layers to reduce the feature maps' spatial dimensions, which helps decrease the computational load and control overfitting [45]. There are different types of pooling operations such as max pooling, which selects the maximum value within a defined region of the feature map, or average pooling, which computes the average value. Through this combination of convolution and pooling, CNNs can effectively capture and summarize the essential patterns in the input data. This process is illustrated in Figure 2.2.

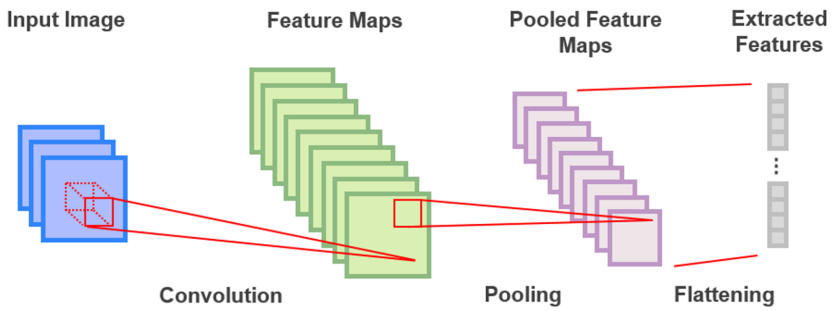

Figure 2.2: Illustration of Feature Learning in CNNs.

At the end of the feature learning process, the extracted features typically go through a fully connected layer or classifier, which maps these features to the final output, such as class labels in classification tasks[43].

Traditional CNN architectures, while powerful, face limitations primarily related to computational cost, depending on their structure. CNNs are compu-

tationally intensive, especially as the network depth and the number of filters increase, which can lead to significant resource demands, especially during training. Additionally, using CNNs with other data types, such as sequences or graphs, requires data transformation or specialized architecture variations to handle such inputs effectively [40, 41, 46].

The decision to focus on CNNs in this study, rather than exploring other deep learning models like Long Short-Term Memory (LSTM) networks, Autoencoders (AE), or Transformers, was influenced by the characteristics of the dataset and the specific advantages of using Gramian Angular Fields alongside CNNs.

CNNs are particularly well-suited to analyzing 2D representations of time series data, such as GASF, due to their ability to detect spatial patterns. This capacity for feature learning allows CNNs to automatically extract relevant patterns and relationships from data without relying heavily on handcrafted feature engineering. CNNs excel at identifying both local and global dependencies within transformed time series data, enabling robust detection of complex fault interactions[38, 44].

LSTM networks are a compelling choice for tasks involving raw time series data, as they excel in capturing sequential dependencies[47]. While LSTMs have been explored in this context, their feature learning capabilities are typically centered on temporal patterns of the individual time series, having their cross-relationships captured in the deeper layers of the network. On the other hand, the matrix structure provided by GASF allows CNN to try capturing these relationships from the beginning of the network.

AE are effective for dimensionality reduction and unsupervised anomaly detection, but their use in supervised classification tasks like fault diagnosis often requires additional layers or hybrid architectures[48]. In contrast, the CNN structure allows for direct training from input to label, with feature extraction as part of the learning process.

With their self-attention mechanisms, transformers are increasingly popular for time series analysis and offer notable flexibility in modeling complex relationships[49]. However, their inherent complexity and large model sizes can pose challenges, especially with constrained computational resources. While these architectures are highly expressive, their design may introduce added complexity for tasks that models like CNNs can potentially address.

While each alternative method offers unique strengths and could be explored in future works, CNNs' ability to automatically learn features from spatially transformed data makes them particularly compelling in the context of fault diagnosis. Combined with their computational efficiency and compat-

ibility with GASF, CNNs provided a robust framework for evaluating fault diagnosis performance while minimizing the need for extensive manual feature engineering.

Several recent publications have utilized CNNs for Fault Detection and Diagnosis in the Tennessee Eastman Process context. The studies briefly presented next were selected explicitly for comparison after our study to assess how our model performs relative to the current state-of-the-art.

Souza *et al.* (2024) explored various CNN topologies [50]. Their approach involved stacking different time series variables into a single matrix. While this technique facilitates the application of conventional CNN methods, it limits the ability of the filters to consider all process variables simultaneously.

Tao *et al.* (2024) proposed a Triage-based Convolutional Neural Network with a 1D-CNN as backbone [51]. In their investigation, They considered 15 of the 20 TEP faults.

Ren *et al.* (2023) explored an ensemble-like approach, utilizing multiple 2D transformation techniques that were stacked and fed into a traditional CNN model [52]. Simultaneously, the original time series data were processed through a 1D-CNN architecture. The final classification was a combination of the output of both models. Unfortunately, this study does not include some of the most challenging faults in its investigation.

Sun and Ren (2021) developed a fault diagnosis approach that converts multi-dimensional temporal data into multi-channel 2D images using Gramian Angular Fields, followed by multi-scale convolution modules for spatiotemporal feature extraction (GASF-MSNN) [2]. Their architecture was benchmarked against various models, including Gramian Angular Summation Fields–CNN (GASF-CNN), traditional CNN with series stacking, 1D-CNN, and LSTM networks. Although their results indicate that the GASF-MSNN model outperforms other approaches, the findings are limited by the scope of their investigation, leaving significant room for further exploration given the wide range of possible hyperparameter variations across the architectures.

Notably, their GASF-MSNN performed comparably to their GASF-CNN, prompting us to focus our investigation on the GASF-CNN due to its innovative data transformation technique and well-established classification architecture.

2.3.2 Gramian Angular Summation Fields

The Gramian Angular Summation Fields transformation is a technique used to convert time series data to account for variations for all possible time

lag combinations within the series. This transformation is a part of the broader family of Gramian Angular Fields Proposed by Wang and Oates (2015) [38]. The key idea behind GASF is to use the polar coordinate system to encode the temporal dynamics of time series data into a matrix format.

The GASF transformation is mathematically computed by first scaling the time series data using a min-max normalization ($\tilde{x}_i$). This step ensures that the data points can be mapped to angles in the polar coordinate system.

Next, the normalized values $\tilde{x}_i$ are transformed into angular values theta using the arccosine function as shown in Equation 2-1.

$$\theta_i = arccos(\tilde{x}_i) \tag{2-1}$$

We obtain the GASF matrix by calculating the pairwise summation of the cosines of these angles, represented in Equation 2-2.

$$GASF_{i,j} = cos(\theta_i + \theta_j) \tag{2-2}$$

This transformation results in a symmetric matrix where each element represents the cosine of the sum of the angles corresponding to the normalized time series values. After a set of mathematical simplifications, obtaining the same GASF matrix directly from the normalized arrays is possible by performing the calculations shown in Equation 2-3.

$$GASF(\tilde{X}) = \tilde{X}' \cdot \tilde{X} - \sqrt{I - \tilde{X}^2}' \cdot \sqrt{I - \tilde{X}^2} \tag{2-3}$$

where $\tilde{X}$ is the array containing the normalized values of the time series sample and I is the identity matrix.

These transformations can be applied to multivariate time series, creating a matrix for each variable. This collection of matrices can be stacked and processed by CNNs and other image-based models for classification tasks. We illustrated this transformation process in Figure 2.3.

In the context of FDD, problems present unique challenges due to the unpredictable and complex ways faults manifest in multivariate time series data. Faults often arise from subtle, nonlinear interactions across multiple variables, and their expressions may vary temporally, making it difficult to pinpoint the exact variable or time window where anomalies occur [24, 5, 25].

GASF transformation not only preserves the temporal sequence of data but also structures the information in a way that enhances the capabilities of Convolutional Neural Networks. CNNs are particularly effective at detecting spatial patterns in data. When applied to GASF-transformed matrices, they can search for fault patterns without being constrained to specific variables. Thus, they are especially advantageous in FDD, where anomalies may emerge in localized or dispersed regions of the time series.

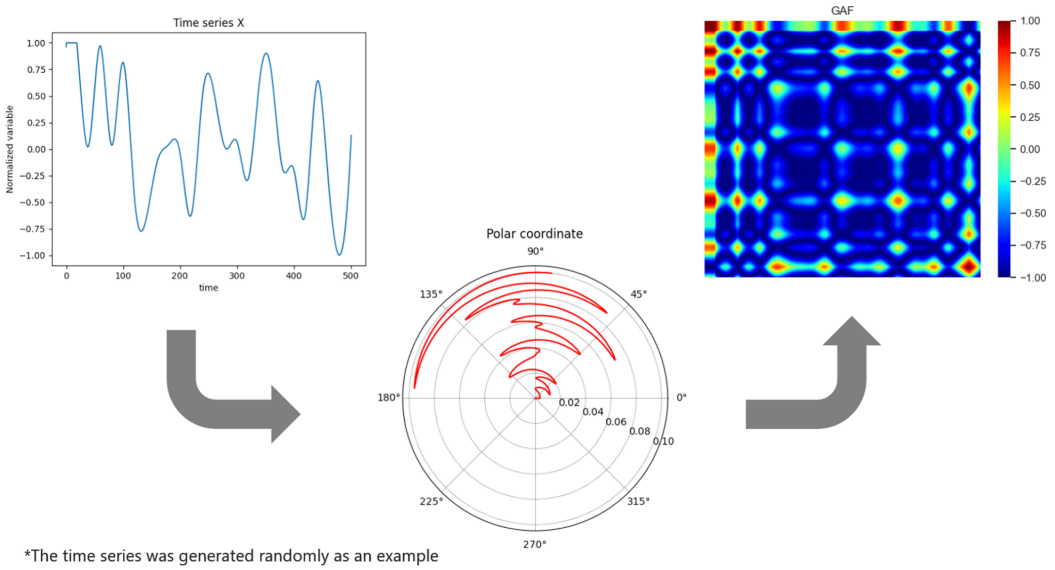

Figure 2.3: Illustration of a GAF type encoding for a general time series.

Another critical advantage of GASF is its ability to represent multivariate data as multi-channel input for CNNs. This setup enables the model to analyze interactions across all variables using filters. Kernels extract patterns that span both time (rows and columns of the matrices) and variable interactions (across channels) simultaneously.

Finally, combining GASF and CNNs enables a fault detection framework that accommodates systems with numerous variables. Knowing where faults might emerge is part of the model's specialized feature extraction capabilities. It takes advantage of CNN's inherent pattern recognition capabilities to identify anomalies wherever they occur, thus addressing the stochastic and distributed nature of fault expressions in complex systems.

2.4 Methodology

This section outlines the methodology employed in our Fault Diagnosis model investigation. The process encompasses several critical stages, including data pretreatment, baseline model architecture, training hyperparameters, and investigated modifications.

We executed all data processing, model development, and evaluation tasks using Python. Specifically, we used the Pandas[53], TensorFlow[54], and Keras[55] libraries for data manipulation and preprocessing, building and training the convolutional neural network models, and implementing custom layers and managing the model architecture.

In terms of hardware, processing was carried out using the i5-13600K CPU and 32 GB RAM, without the need for a dedicated GPU, which

we consider an advantage as it shows the accessibility of our approach. Additionally, we used the tensorflow.data.dataframe module to improve data management during processing.

2.4.1 Data Pretreatment

The original dataset [39] contains a total of 15,330,000 data samples, sampled every 3 minutes. This dataset consists of 5,250,000 samples for training and 10,080,000 samples for testing. Given the long processing times and high computational costs associated with CNN models, we decided to perform random stratified subsampling on the training data. We then split the data into three distinct datasets that include 20 possible faults since our investigation focused on developing a diagnosis model.

Following the methodology of Sun and Ren (2021) [2], we used a window size of 20 data samples. This resulted in the following datasets: the training dataset consisted of 36,600 windows (1,800 per class), the validation dataset included 9,000 windows (450 per class), and the testing dataset contained 5,000 windows (250 per class). In contrast, Sun and Ren (2021) reported using training and testing datasets of 461 and 781 samples per fault, respectively and did not mention a validation set.

After splitting the dataset, we normalized each variable based on the values from the training data. We capped all values within the training range for the validation and test datasets to prevent invalid numbers in subsequent calculations. We incorporated the Gramian Angular Summation Field transformation into the model using a custom layer. This approach is advantageous because it reduces the extensive storage space required to precompute and save the transformed data for large datasets. By performing the GASF transformation within the model, we efficiently manage memory usage and storage requirements, as the transformation is computed on the fly during training and inference.

2.4.2 Baseline Model Architecture

We adopted the GASF-CNN architecture proposed by Sun and Ren (2021) [2] as our baseline model due to its strong performance, which showed less than a 1% difference in F1-score compared to their more complex GASF-MSNN model. This model employs a custom GASF layer to transform the time series data into a set of 52 matrices, which then serve as input for a series of convolutional and pooling layers dedicated to feature extraction.

These specialized features are the input of dense layers responsible for fault classification.

In order to enhance model generalization and reduce overfitting, batch normalization is applied between the convolutional layers, and a dropout layer is introduced between the dense layer and the softmax output layer. The reference authors did not specify the padding option used in the convolutional layers. However, in our implementation, we consistently set the padding to 'same' across all convolutional layers. This choice ensures that the output dimensions of the feature maps remain identical to the input dimensions, preserving the spatial resolution of the data across all layers. Figure 2.4 provides a detailed diagram of the architecture, including specifics such as the number of kernels and kernel sizes.

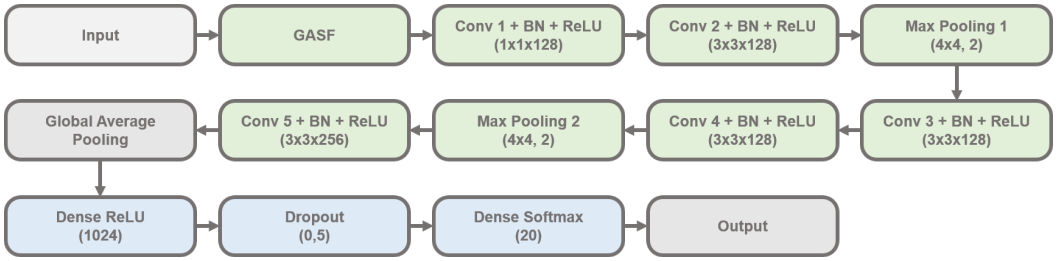

Figure 2.4: Baseline model architecture. Adapted from Sun and Ren (2021) [2].

2.4.3 Training Hyperparameters

In alignment with the methodology of Sun and Ren (2021) [2], we configured the batch size to 64 to optimize computational efficiency and model performance. We employed the Adam optimizer [56] and used categorical cross-entropy [57] as the loss function to address the multi-class nature of the fault diagnosis problem.

In order to mitigate overfitting, we implemented an EarlyStopping callback [55] with a 'patience' argument of 15 epochs, monitoring performance on the validation set. The model's weights were restored to their best state upon triggering early stopping. Furthermore, we set the maximum number of training epochs to 500 to ensure comprehensive model training.

In their work, Sun and Ren (2021) [2] do not mention the adopted learning rate, nor if they used any scheduler strategies. Therefore, in our initial investigation consisted we evaluated different learning rate decay functions as shown in Figure 2.5 applied to the baseline model.

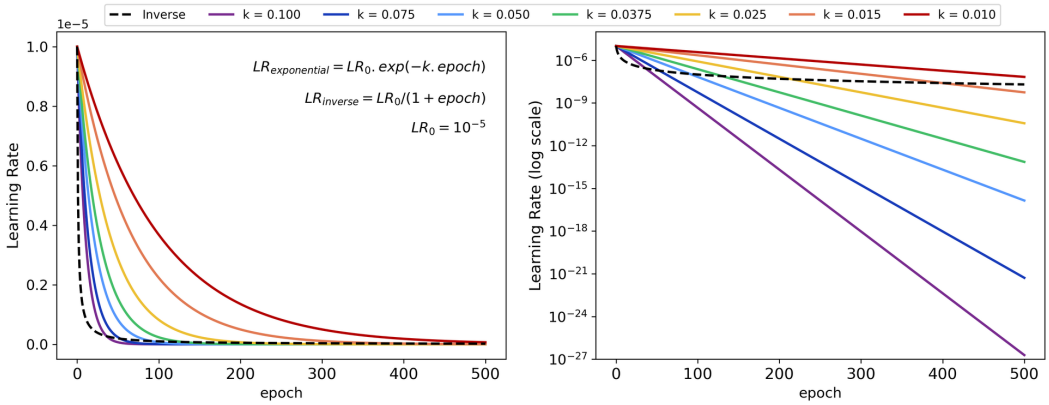

Figure 2.5: Investigated learning rate decay functions.

2.4.4 Investigated modifications

In this section, we present the modifications implemented to enhance the performance of the baseline GASF-CNN model, initially proposed by Sun and Ren (2021) [2]. We systematically explored adjustments to various architectural components to optimize the model's fault diagnosis capabilities. This detailed examination includes the rationale behind each modification and its implementation.

Our exploration focused on four specific types of modifications to assess and improve the performance change of the GASF-CNN approach:

1. Introducing an additional transformation to the matrices derived from the GASF.
2. Adjusting the kernel size of the initial convolutional layer.
3. Implementing hybrid approaches that combine the additional transformation and kernel size adjustments based on preliminary results.
4. Evaluating the impact of varying the number of filters across the convolutional layers.

2.4.4.1 Modification of type 1

The original approach following the GASF transformation involves stacking the resulting matrices to serve as input channels for the CNN. The model processes the stacked matrices of information within the observation window in localized regions determined by the kernel size of the convolutional layers. Consequently, relationships involving time samples separated by longer differences than the kernel size are typically captured only in deeper layers of the

network. To address this limitation, we explored a modification to enrich the data from the outset.

Specifically, we investigated the impact of enabling the simultaneous processing of time samples beyond the kernel size starting from the first layer.Our approach introduces additional transformations to the GASF matrices by generating rotated versions at angles of 90°, 180°, and 270°. These rotations present the model with alternative perspectives of the data's temporal and spatial relationships, enabling it to consider broader patterns during its initial layers. As illustrated in Figure 2.6. This transformation, applied within the custom GASF layer, has the potential to enrich the input data from the beginning, allowing the model to process more comprehensive temporal information right from the initial processing stages.

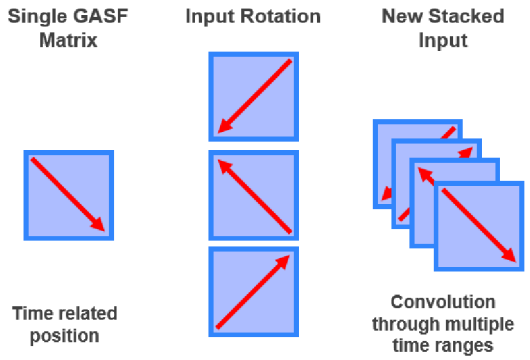

Figure 2.6: Illustration of the type 1 investigation strategy.

2.4.4.2 Modifications of type 2

Given the time-position structure of the GASF matrices, we investigated modifying the kernel size in the first convolutional layer. In the original model, using a 1x1 kernel limits the receptive field of the first layer, preventing it from capturing interactions between neighboring elements. This restriction could hinder the model's ability to fully exploit the spatial and temporal relationships encoded in the GASF matrices, particularly at the early stages of feature extraction, especially because of the number of channels.

In order to address this limitation, we replaced the 1x1 kernel with larger kernels, specifically 3x3 and 5x5, which allow the model to process information from a broader spatial and temporal context in the initial layers. This adjustment was motivated by the theoretical advantage of larger kernels in capturing local dependencies, as they can aggregate information from multiple neighboring elements, potentially leading to richer feature representations.

We also explored the impact of reducing the number of filters from 128 to 64. This adjustment aimed to mitigate the risk of overfitting, particularly when using larger kernels, which increase the number of parameters in the model. Reducing the filter count can balance the model's complexity, making it more computationally efficient while preserving its ability to learn meaningful patterns. By evaluating different kernel sizes and filter counts, we sought to balance the model's complexity and performance, ensuring that the architecture remained both computationally efficient and robust against overfitting tendencies.

2.4.4.3 Modifications of type 3

We also explored the potential benefits of combining types 1 and 2 modifications, concurrently implementing the enhanced data transformation and the adjusted kernel sizes. By stacking multiple rotated copies of the GASF matrices (type 1 modification) and using larger convolutional kernels (type 2 modification), we aimed to enrich the model's ability to capture and process spatial and temporal patterns.

The rationale behind this combined approach is that the increased data complexity from multiple GASF matrix rotations could overwhelm the point-wise convolution of the original model, limiting its effectiveness in extracting relevant features. Larger convolutional kernels might better capture the enriched data, as they can integrate information from a broader context within each layer. Thus, we decided to investigate the combined effects of these modifications aiming for a model capable of processing more comprehensive data representations and of effectively extracting features, potentially leading to improved overall performance.

2.4.4.4 Modifications of type 4

Based on the results obtained from the previous experiments, we undertook a systematic investigation into the impact of varying the number of kernels in the convolutional layers to assess its effect on model performance. This modification was motivated by the observation that the original architecture, with its fixed filter counts, might not fully capture the complexity of the temporal and spatial patterns within the GASF matrices. By systematically increasing the number of filters, we aimed to investigate the enhancement of the model's capacity for feature extraction in layers processing lower and higher-level representations.

For this exploration, we incrementally increased the filter count across different layers, as detailed in Table 2.2.

Table 2.2: Topology for the investigations of type 4.

Model ID	**Amount of Filters in Convolutional Layer**					
	Conv 1	Conv 2	Conv 3	Conv 4	Conv 5	Conv 6
Model 1	64	128	128	128	256	N/A
Model 2	128	128	128	128	256	N/A
Model 3	256	128	128	128	256	N/A
Model 4	512	128	128	128	256	N/A
Model 5	1024	128	128	128	256	N/A
Model 6	2048	128	128	128	256	N/A
Model 7	1024	512	128	128	256	N/A
Model 8	1024	512	512	128	256	N/A
Model 9	1024	512	512	256	256	N/A
Model 10	1024	512	512	256	256	128
Model 11	1024	512	512	256	256	256

It is important to note that we consistently set the kernel size of the initial convolutional layer to 3x3 during these investigations due to improvements observed from modifications of type 2. Additionally, two of the evaluated configurations included an extra convolutional layer. This addition was designed to investigate whether increased depth would enhance the model's ability to extract features and process complex patterns effectively.

The primary theoretical motivation for these modifications is that increasing the number of filters in the convolutional layers provides a greater capacity to learn diverse features from the input data. Additionally, adding depth to the network could facilitate hierarchical feature extraction, where subsequent layers build upon the representations learned by earlier ones.

For clarity and ease of reference, we only assigned specific model IDs to models resulting from modifications of type 4. This labeling approach facilitates a straightforward comparison of results and supports a clear discussion of the findings in subsequent sections.

2.4.5 Evaluation Metrics

Given that fault diagnosis in the Tennessee Eastman Process is a multiclass classification task, we use the F1-score and confusion matrices as our primary evaluation metrics. The F1-score is calculated for each class to measure the model's precision and recall balance. It is computed using the expressions shown in Eq. 3-2.

$$\begin{cases} Precision = \frac{TP}{TP+FP} \\ Recall = \frac{TP}{TP+FN} \\ F1_{Score} = 2 \cdot \frac{Precision \cdot Recall}{Precision+Recall} \end{cases} \tag{2-4}$$

In these equations, TP denotes true positives, FP represents false positives, and FN refers to false negatives.

To summarize overall model performance across all classes, we use the macro-averaged F1-score. This metric consists of the average of the F1-scores for each class, providing a balanced view of the model's performance on both majority and minority classes.

We also compare our results with those reported in the Literature to contextualize our model's performance. Previous studies, such as those by Sun and Ren (2021) [2], have reported F1-scores and confusion matrices for similar multi-class classification tasks, allowing us to benchmark our findings against established methods.

2.5 Results and discussion

In this section, we present and discuss the results of our investigation. We start by examining the impact of the learning rate on the baseline GASF-CNN model and identifying the optimal configuration. Next, we provide detailed results for the best baseline model, followed by evaluating the four types of modifications that enhance fault diagnosis. It is important to note that we prioritized keeping only the key training curves in the article's main body. We reported the entire collection of training curves of the investigated modifications in Appendix A. Finally, we compare our findings with those reported in the Literature to contextualize our results.

2.5.1 Impact of Learning Rate on Baseline Model Performance

The results reveal that the baseline model is sensitive to the choice of learning rate decay. Figure 2.7 illustrates the varying average validation F1-scores obtained under different learning rate decay strategies.

Since the exponential decay function set at a rate of $k = 0.01$ demonstrated the best validation performance (81.93 %), we adopted this configuration as the baseline for evaluating subsequent modifications. This choice ensures that all further investigations are conducted under optimal learning conditions, providing a consistent reference point for comparison.

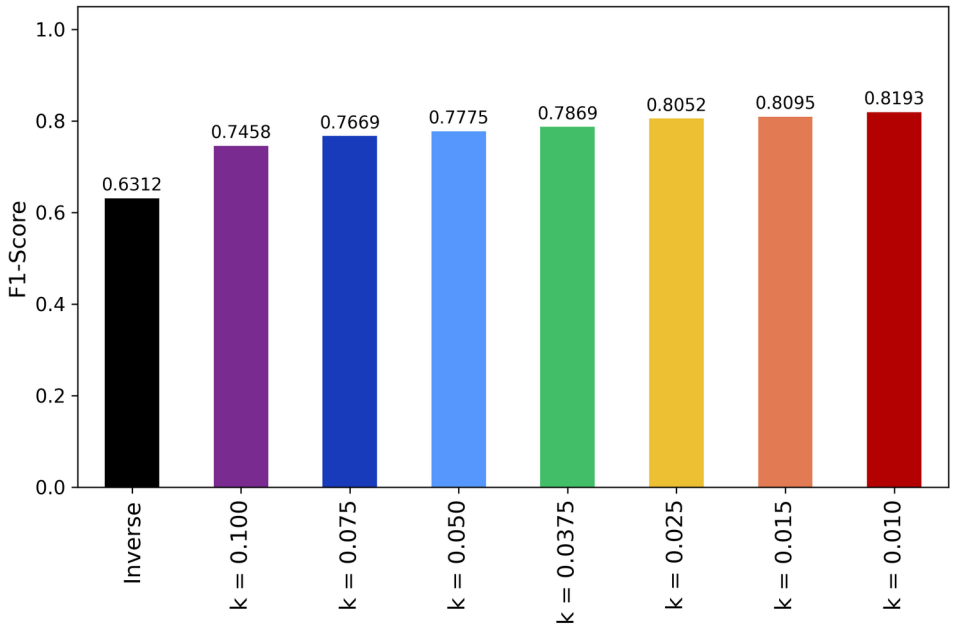

Figure 2.7: Learning rate decay functions results.

We conducted a comprehensive analysis to gain deeper insights into the baseline model's performance. Figure 2.8 details the validation F1-scores for each fault class, offering a nuanced view of the model's ability to differentiate between various fault conditions. Complementing this, Figure 2.9 presents the confusion matrix, further explaining how the model misclassifies different faults.

The model demonstrates its high capability of classifying 15 of the 20 faults, staying above 89 % F1-scores. However, the model faces challenges with faults 3, 9, 10, 15, and 16, where the F1-scores are below 52 %. These results suggest difficulties in distinguishing these faults from others, possibly due to their physical nature. Apart from fault 16, which is of an unknown type, all other faults in this group are related to feed temperature or heat transfer.

The results from the confusion matrix shown in Figure 2.9 further reinforce the hypothesis for the model's misclassification cases. It indicates that these specific faults are frequently confused with each other, indicating that their similar physical nature plays an important role in distinguishing them.

An examination of the loss curve shown in Figure 2.10 reveals no signs of overfitting in the model. The loss decreases consistently throughout training, indicating stable learning without excessive divergence between training and validation data.

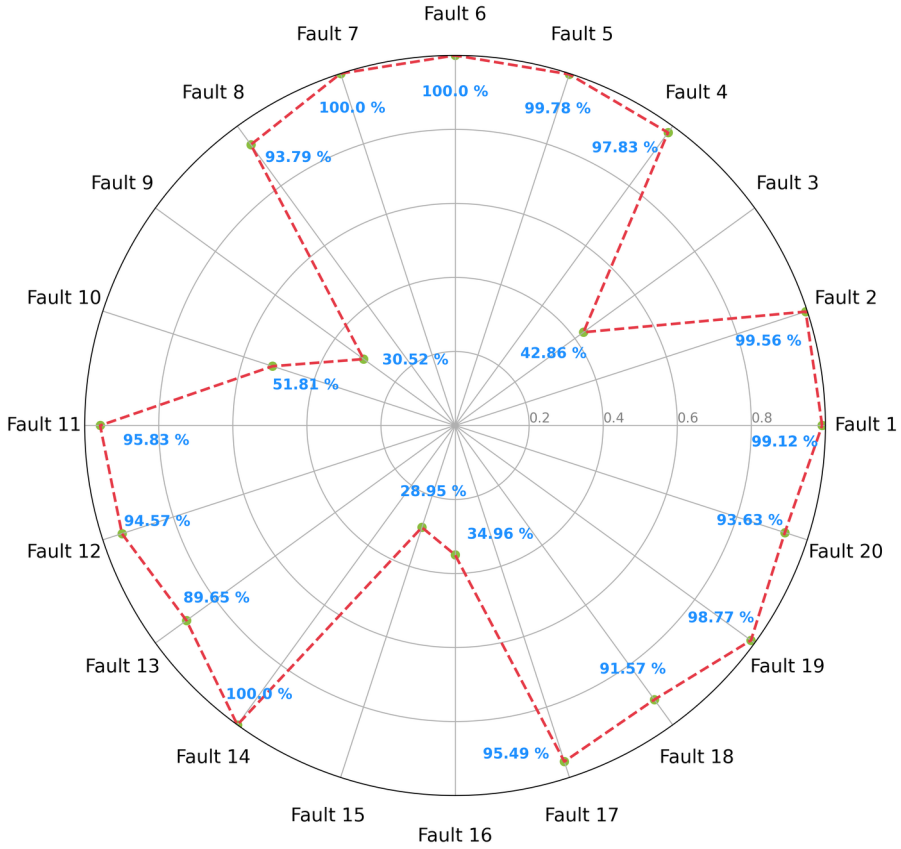

Figure 2.8: Validation F1-scores for the baseline model.

2.5.2 Influence of the Investigated Modifications

We designed the modification of type 1 to investigate the hypothesis that simultaneously processing time lags further than kernel size, which could benefit model learning. It consisted of stacking the original GASF matrices with copies rotated by 90°, 180°, and 270°.

This modification resulted in slightly lower performance than the baseline model, with an average F1-score of 81.13%. This result suggests that while the intention was to provide the model with more comprehensive temporal information, the rotated layers did not enhance, and may have even reduced, the model's ability to generalize. Adding rotated matrices may have introduced complexity or noise that outweighed the benefits of increased temporal coverage.

The modifications of type 2 focused on addressing a potential bottleneck in the baseline model's architecture, specifically in the first convolutional layer due to the filter quantity and point-wise convolutions. It involved experimenting with filter quantities of 64 and 128, using both 3x3 and 5x5 kernels. The results presented in Table 2.3 show that increasing the size

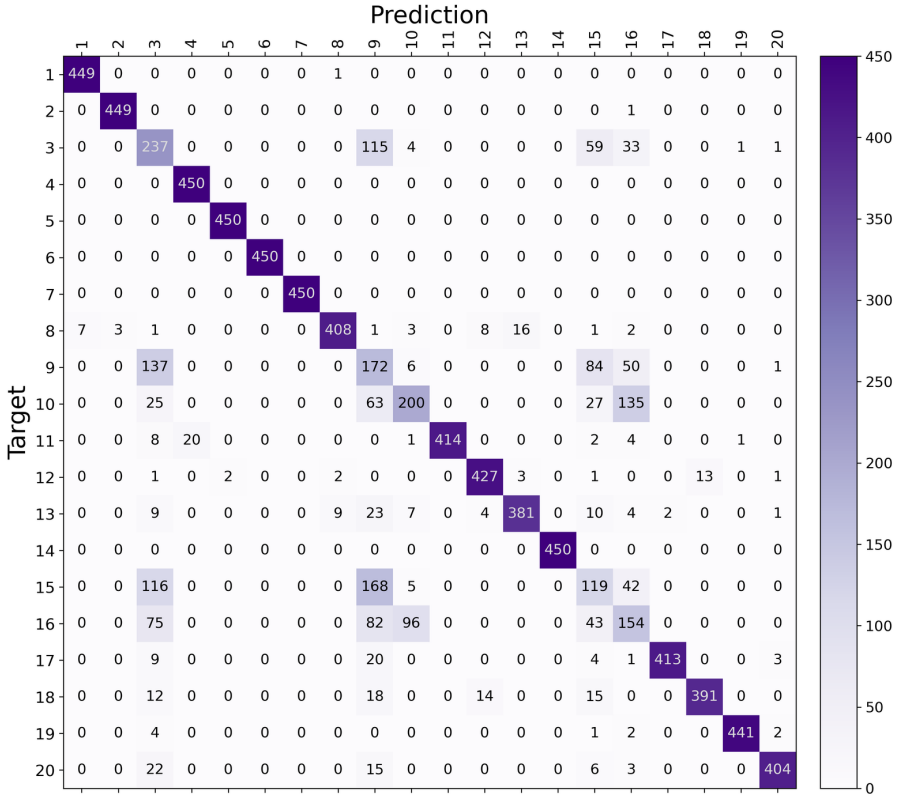

Figure 2.9: Validation confusion matrix for the Baseline Model.

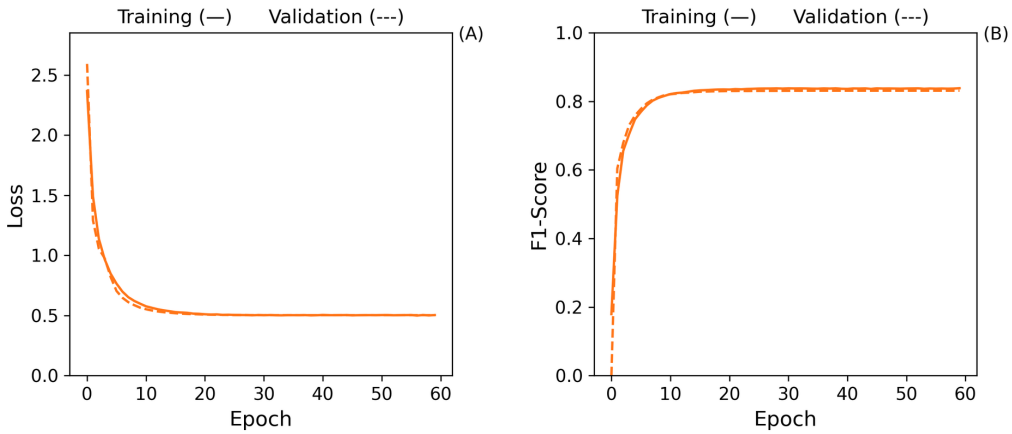

Figure 2.10: Loss (A) and F1-score (B) training curves for the Baseline Model.

of the kernels in the first convolutional layer led to improved classification performance.

The results indicate that reducing the number of filters in the initial convolutional layer negatively impacted classification performance. Larger filter sizes, particularly the 3x3 kernels, improved the model's ability to capture complex features. However, it is important to note that while increasing filters

Table 2.3: Average validation F1-scores for modifications of type 2.

Parameters	**Kernel 3x3**	**Kernel 5x5**
64 Filters	82.35 %	82.59 %
128 Filters	82.93 %	82.68 %

enhanced performance, other configurations constrained the benefit, evidenced by observing that the effects of changing the number of filters with 5x5 kernels were less impactful than in the 3x3 kernel cases, considering the current configuration.

Type 3 modifications aimed to explore potential synergistic effects by combining data augmentation with rotated layers. Specific results are shown in Table 2.4.

Table 2.4: Average validation F1-scores for modifications of type 3.

Parameters	**Kernel 3x3**	**Kernel 5x5**
64 Filters	81.51 %	56.67 %
128 Filters	82.54 %	47.89 %

Even with the increased filters, the combined approach of data enrichment through rotated layers did not yield significant improvements over the original GASF maps. For kernel size 5x5, a noticeable performance drop occurred, possibly related to overfitting. These results suggest that while individual enhancements showed promise for type 2, the combined use with type 1 modification did not provide additional benefits and may have contributed to model instability. This outcome indicates that the enriched data from rotating the matrices did not improve the model's fault classification capabilities and highlights the importance of evaluating the trade-offs between data complexity and model performance.

In light of the results, we considered the hypothesis that further increasing the number of filters in convolutional layers could enhance classification performance. It involved maintaining 3x3 kernels in the first convolutional layer and progressively increasing the number of filters throughout the convolutional layers. The average F1-score for each modification is shown in Figure 2.11, including the two 3x3 kernel size cases of type 2 modifications and cases of additional convolutional layers.

The results indicate a significant improvement in classification performance with the progressive increase in the quantity of filters. Model 9 achieved a maximum F1-score of 89.40%, which is 7.47% higher than the baseline model. Its architecture is illustrated in Figure 2.12. This increase demonstrates that systematically augmenting the number of filters was a key factor in substantially enhancing model performance in our study.

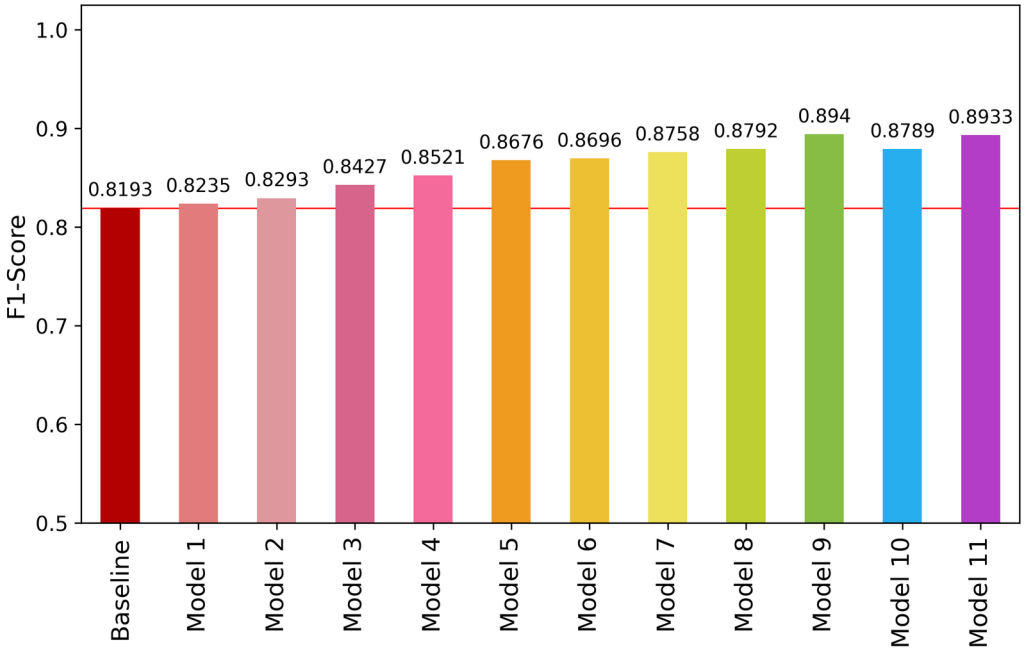

Figure 2.11: Validation average F1-score for type 4 modifications.

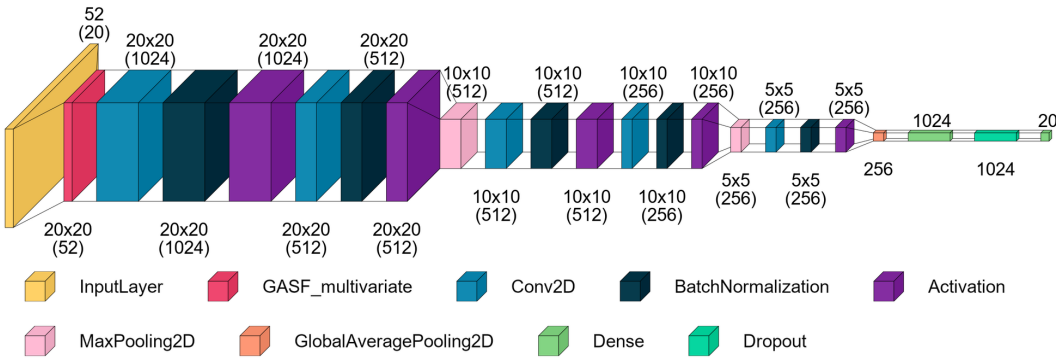

Figure 2.12: Representation of the architecture of Model 9.

Figure 2.13 displays a radar plot comparing the F1-scores for each fault class between Model 9 and the baseline model. It is possible to observe that Model 9 exhibited improved classification capabilities across all faults. The overall improvement in performance may be linked to the nature of the GASF input matrices, which are significantly more complex than typical image data. Fifty-two channels far surpass the common three channels used in standard image data. In this case, more kernels seem necessary to effectively capture and detect the more intricate and harder-to-find patterns within the data. The increase in kernel quantity allows the model to better process and extract relevant features from the extensive information embedded in the GASF matrices.

The model demonstrated significant improvement in predicting all five challenging faults, with enhancements of 28.94 % for fault 3, 9.28 % for fault

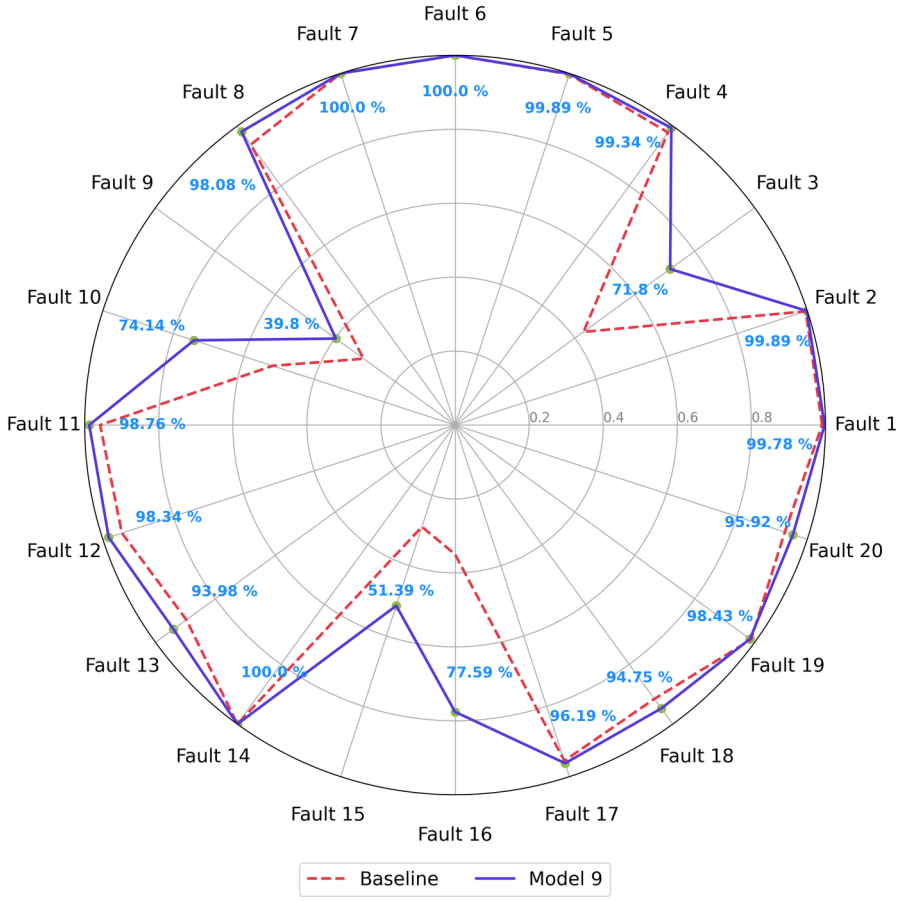

Figure 2.13: Validation F1-scores for Model 9.

5, 21.29 % for fault 10, 22.44 % for fault 15, and 43.62 % for fault 16. As shown in the confusion matrix in Figure 2.14, misclassification among these faults is less widespread. However, faults 3, 9, and 15 remain challenging to distinguish, especially for faults 3 and 9, which are both temperature-related issues on the same feed stream, making them inherently more complex to separate. Additionally, while faults 10 and 16 are frequently confused with each other, they are also closely linked to the misclassification of faults 3, 9, and 15.

Figure 2.15 illustrates the training curves, where the validation loss initially exhibits some volatility but eventually stabilizes as training progresses. While the validation loss curve did not exactly converge with the training loss, this is not a concern because the model's performance on unseen data remained robust. Early stopping was activated, indicating the model achieved optimal performance without overfitting.

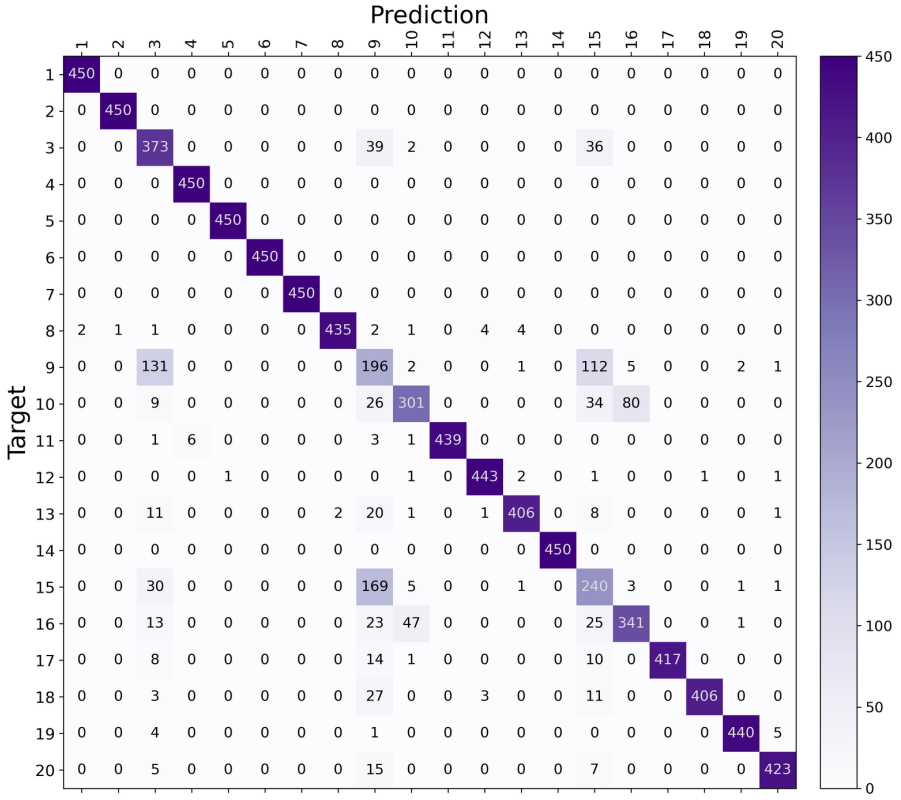

Figure 2.14: Validation confusion matrix for Model 9.

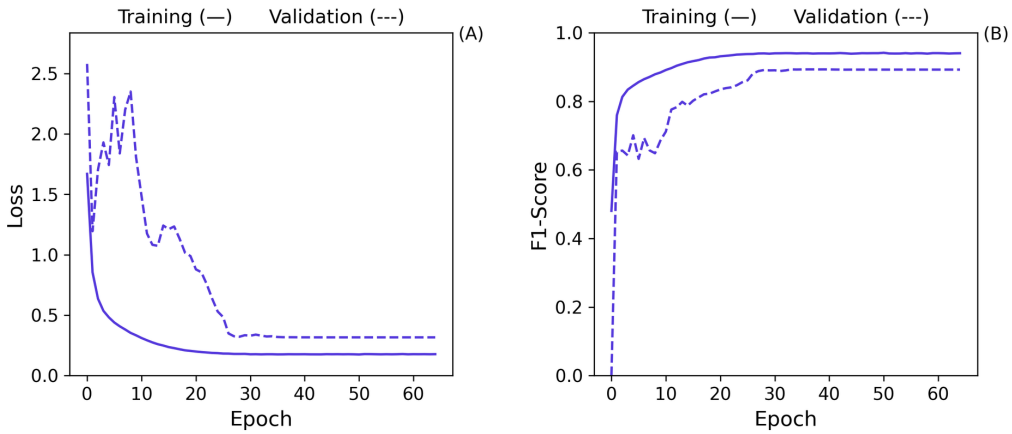

Figure 2.15: Loss (A) and F1-score (B) training curves for the best model.

2.5.3 Testing results and Comparative Analysis with the Literature

The performance of Model 9 on the testing dataset is illustrated in Figure 2.16. The F1-scores for each fault class demonstrate that the model's effectiveness extends beyond the validation set.

The average F1-score for the test dataset is 89.85%, showing only

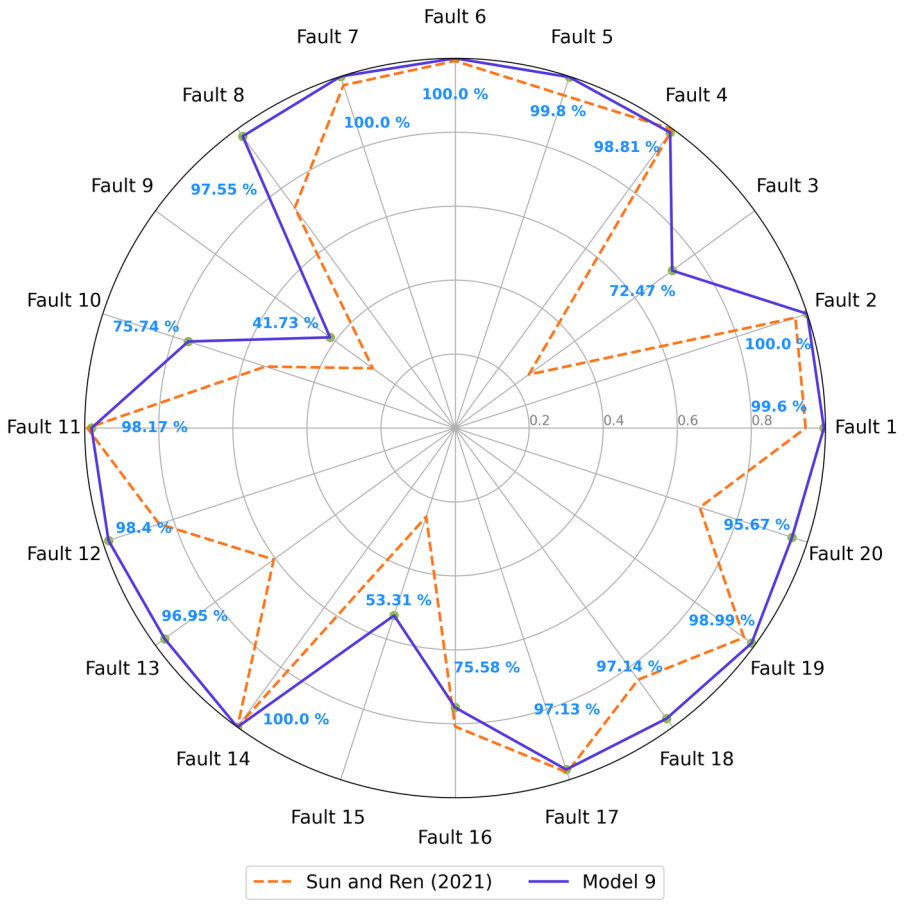

Figure 2.16: Test F1-scores for Model 9 and the reference GASF-CNN[2].

a negligible difference from the validation average. This minimal decrease suggests that Model 9 generalizes effectively to new, unseen data, maintaining consistent performance. Compared to the results reported by Sun and Ren (2021) [2] for their GASF-CNN model, which served as the basis for our architecture, Model 9 demonstrates comparable or even significantly improved performance in fault diagnosis of the Tennessee Eastman Process, except for fault 16. Additionally, Model 9 achieved an average F1-score that is 11.86% higher than the 77.99% reported by Sun and Ren (2021) [2].

For a broader perspective, in Table 2.5, we compare Model 9's performance against other state-of-the-art models from the literature. While Model 9 demonstrates competitive F1-scores across most fault classes, it notably outperforms Fault 18 with F1-scores of 97.14 %. However, the performance in Fault 9 and Fault 15 suggests room for further refinement, as indicated by the lower scores compared to Tao *et al.* (2024)[51].

It is also worth noting that some references do not include results for all fault classes, as observed in Ren *et al.* (2023)'s[52] and Tao *et al.* (2024)'s[51] work. It allows models to become more specialized at classifying the subgroup

Table 2.5: Model 9 test F1-scores comparison with state-of-art works from the literature.

Class	**Model 9**	**Ref.1**[a]	**Ref.2**[b]	**Ref.3**[c]	**Ref.4**[d]
Fault 1	99.60 %	94.67 %	**100.0 %**	99.81 %	**100.0 %**
Fault 2	**100.0 %**	96.63 %	**100.0 %**	99.40 %	99.80 %
Fault 3	72.47 %	24.62 %	—	**96.98 %**	58.00 %
Fault 4	98.81 %	99.87 %	99.00 %	99.97 %	**100.0 %**
Fault 5	**99.80 %**	94.67 %	89.00 %	93.87 %	80.30 %
Fault 6	**100.0 %**	99.30 %	90.00 %	99.46 %	**100.0 %**
Fault 7	**100.0 %**	97.50 %	99.00 %	**100.0 %**	**100.0 %**
Fault 8	97.55 %	73.61 %	82.00 %	**98.45 %**	75.40 %
Fault 9	41.73 %	27.45 %	—	**87.59 %**	0.00 %
Fault 10	75.74 %	53.84 %	77.00 %	**98.61 %**	94.20 %
Fault 11	98.17 %	99.29 %	99.00 %	99.40 %	**99.90 %**
Fault 12	**98.40 %**	83.99 %	83.00 %	97.83 %	86.70 %
Fault 13	96.95 %	60.52 %	57.00 %	**98.41 %**	87.80 %
Fault 14	**100.0 %**	99.94 %	**100.0 %**	99.62 %	85.80 %
Fault 15	53.31 %	25.22 %	—	**83.50 %**	0.40 %
Fault 16	75.58 %	**80.74 %**	77.00 %	—	1.00 %
Fault 17	97.13 %	97.92 %	**99.00 %**	—	97.90 %
Fault 18	**97.14 %**	84.13 %	81.00 %	—	84.20 %
Fault 19	98.99 %	96.42 %	99.00 %	—	**99.80 %**
Fault 20	95.67 %	69.49 %	95.00 %	—	**98.00 %**
Average F1-score	**89.85** %	77.99 %	N/A	N/A	77.46 %

[a] Sun and Ren (2021)[2]; [b] Ren *et al.* (2023)[52]; [c] Tao *et al.* (2024)[51]; [d] Souza *et al.* (2024)[50].

of faults at the expense of not covering the original 20 faults. Additionally, the mentioned studies that specified observation window sizes used windows of similar or larger duration. Sun and Ren (2021)[52] initially investigated window lengths between 10 and 40. They developed the remaining of their study with lengths of 35 and 40. Tao *et al.* (2024)[51] have set a time length of 20 as the input of their CNN-1D, which we calculated from the output dimensions and the architecture configurations. Souza *et al.* (2024)[50] chose a window size of 52 to maintain a square input matrix, which further emphasizes the improvement of our approach since less historical data is necessary to provide a diagnosis.

Overall, the consistent and high F1-scores across various faults suggest that Model 9 offers a reliable and effective approach to fault diagnosis, with potential advantages over existing methods documented in the literature.

Although we obtained a generalized enhanced performance with the current state of our research, there is still space for improvement. Further modifications within the realm of CNNs can still be explored for further performance enhancement. Besides, our investigation has primarily focused

on model diagnosis, leaving the detection aspect of FDD as an area open for exploration in future investigations.

2.5.4 Conclusions

In this study, we developed and evaluated a convolutional neural network architecture for fault detection and diagnosis in the Tennessee Eastman Process. Our approach used Gramian Angular Summation Fields to transform multivariate time series data into 2D images, allowing the CNN to extract complex spatiotemporal patterns. The results demonstrated that Model 9 achieved an average F1-score of 89.85 % on the testing dataset, which remained consistent with the validation results, indicating strong generalization capabilities.

Compared to the GASF-CNN model by Sun and Ren (2021)[2], which served as the foundation for our architecture, Model 9 showed comparable or improved performance across most faults, particularly in diagnosing some of the more challenging faults in the TEP. The change in the number of kernels was a vital parameter to achieve performance improvement, which allowed the model to capture more nuanced patterns in the data, especially given the complexity introduced by the 52-channel GASF matrices.

Our focused approach to refining a single architecture, rather than exploring multiple variations, has proven effective in identifying and optimizing the key aspects of the model that directly impact its performance. This methodical focus allowed us to gain deeper insights into the architecture's specific strengths and weaknesses, leading to targeted improvements.

While our research has achieved an improved performance, there is still room for further enhancement. Future works could explore additional modifications within the CNN framework, such as experimenting with different hyperparameters and other input transformations or even integrating advanced techniques like attention mechanisms to boost performance further. Although we focused our investigation on the TEP dataset due to its industrial relevance and complexity, it would be relevant to explore how our approach holds up against other datasets in future works. Moreover, since our investigation primarily focused on fault diagnosis, future studies could expand on the detection aspect of the problem, exploring the full spectrum of FDD to develop a more comprehensive solution.

3
Article 2 - A Diagnosis-based Siamese Network for Fault Detection Through Transfer Learning

Traditional deep learning-based approaches often struggle with data imbalance and variability across fault conditions and normal scenarios, especially in industrial processes. Besides, inconsistent feature distributions from combining different fault conditions into the same category is a limitation for many data-driven algorithms. This study proposes a fault detection framework that combines Siamese Neural Networks with transfer learning, using a pre-trained fault diagnosis model as its backbone, taking advantage of knowledge related to the attribute space that characterizes individual fault patterns. Our method transforms the detection classification problem into an embedding similarity task, allowing for improved differentiation between normal and faulty operations. This approach poses an alternative for data imbalance and lack of labeled anomaly data, as it is based on the combination of normal and faulty time series. Our best model achieved an F1-score of 91.41 % on the test set, and the t-Distributed Stochastic Neighbor Embedding indicates that the knowledge transferred from diagnosis allowed the detection model to generate embeddings that discriminate between most faulty conditions. When analyzing individual fault detection rates, we observed that our model demonstrated superior performance compared to recent literature for most fault cases.

3.1
Introduction

Fault Detection and Diagnosis (FDD) are important in maintaining the efficient productivity and safety of industrial processes. Due to their complexity, such systems are prone to anomalies and faults, meaning detection is essential for mitigating risks and maintaining process reliability[25, 5, 24, 22, 23]. Traditional methods for FDD often rely on statistical techniques or analytical modeling, which may struggle with nonlinearities and high-dimensional data[27].

Recent advancements in Deep Learning (DP) have provided promising alternatives to these challenges[9, 10, 11]. However, these approaches heavily depend on abundant and high-quality data, which includes a balanced distribu-

tion of fault samples [58]. Furthermore, in industrial settings, systems typically operate under normal conditions, thereby reducing the likelihood of records of faulty behavior. This scarcity can limit the ability of data-driven FDD techniques to capture critical fault characteristics. Consequently, implementing these methods can be unviable if this problem is not addressed first[12, 13].

Imbalanced datasets are a common scenario in multiclass classification problems. In fact, there are well-established oversampling and undersampling techniques for data balancing. Nevertheless, these methods can introduce noise into the dataset, potentially leading to overfitting in the case of oversampling or loss of valuable information when undersampling is applied. As a result, the model's generalization performance may be compromised, affecting its ability to make accurate predictions on unseen data. Besides, time-series problems require more advanced applications regarding balancing techniques.[13, 15, 16].

Another alternative is using classical modeling techniques to generate synthetic data. Simulation-based approaches create new data points that adhere to the statistical properties of the original data once the model is validated. For instance, the Monte Carlo method and domain-specific mathematical systems of equations can represent chemical processes realistically[59, 60, 61, 62, 63, 64, 65]. In time-series problems, these techniques help maintain temporal dependencies and structural patterns. A notable example is the use of numerical simulation-based models for fault diagnosis, as introduced by Xiang and Zhong (2016)[66]. Their work focused on diagnosing faults in rotating machinery using FEM-based vibration signal simulation and classification. Consequently, several FDD benchmark datasets were partially or fully developed using simulations, providing standardized testing grounds for evaluating classification models in imbalanced scenarios[67].

The Tennessee Eastman Process (TEP) is a widely used benchmark that allows researchers to compare different methodologies under controlled conditions. This dataset provides a realistic simulation of complex chemical processes, enabling the systematic introduction and analysis of various fault scenarios. By offering a risk-free environment that mimics real-world challenges, TEP supports developing and evaluating advanced FDD algorithms. Its diverse fault types, including operational disturbances and sensor failures, make it a valuable reference for assessing diagnostic approaches[30, 29, 31, 32, 68]. These qualities make this dataset suitable to use as our study case.

FDD models are commonly developed by attempting simultaneous detection and diagnosis or treating these aspects sequentially. On the first option, normal behavior is usually a class among different faults, while on the second, detection and diagnosis are studied independently, with the possibility of later

aggregation. In either case, several approaches are possible, such as an ensemble of specialized models, hierarchized models, or a generalistic multiclass estimator[11, 52, 51, 50, 10].

Specifically in detection models, grouping faulty operations into a single class is standard practice, simplifying the system into a binary classification problem. However, combining multiple infrequent classes into a single category can lead to poor classification performance due to inconsistent feature distributions[69]. This study explored this issue by combining two state-of-the-art techniques: Siamese Neural Network (SNN) architecture and transfer learning.

SNNs are particularly well-suited for tasks that require similarity assessments[70, 71], making them a natural choice for detecting deviations from normal operating states. Additionally, the contrastive loss penalizes mistakes based on the distance between the embedding representations of inputs[72]. Consequently, the model has the flexibility to learn different feature distributions of the detection task independently from the differences between the varied faulty scenarios, as long as they are distinguishable from normal behavior.

While SNNs have been explored in some industrial contexts, their application to chemical process monitoring remains limited. For example, Takimoto *et al.* (2022)[73] proposed a SNN-based anomaly detection method enhanced with an attention mechanism for visual inspection tasks in manufacturing. Their approach effectively handled few-shot scenarios, demonstrating that SNNs can be used in industrial settings where abnormal data are scarce. However, to the best of our knowledge, such techniques have not yet been extended to chemical plant environments, which involve different types of data and fault characteristics. This gap highlights the novelty of our work in applying SNN-based models to fault detection in chemical processes.

Since SNNs comprise a macro-architecture that supports a substructure, we used transfer learning in our investigation. In other words, we explored models that start with knowledge about embeddings from each fault by using the best-performing CNN model from our previous TEP fault diagnosis study[3] as their backbone. Our approach reverses the conventional order of detection preceding diagnosis, providing a new perspective for developing FDD solutions.

The remainder of this article is organized as follows: Section 2 provides the theoretical framework necessary to understand the techniques adopted in our study. Section 3 describes the steps of the investigation, including data pretreatment, model architecture, fine-tuning, and evaluation methods. Section

4 critically assesses and compares our results with recent literature. Finally, Section 5 summarizes our findings and indicates possible directions for future studies.

3.2 Theoretical Framework

This section provides an overview of Siamese Neural Networks, their fundamental architecture, and their advantages for fault detection tasks. Additionally, it briefly introduces the Tennessee Eastman Process and describes the dataset version used in this study.

3.2.1 Siamese Neural Networks

Siamese Neural Networks were first introduced by Bromley *et al.* in 1993 for text verification[70], which involved comparing two handwritten signatures to determine whether they belonged to the same individual or not. Since their inception, SNNs have gained widespread popularity due to their ability to evaluate the similarity between inputs. As a result, they have become a powerful tool for tasks ranging from identity verification to anomaly detection [71, 74, 75].

SNNs possess several characteristics that make them particularly suited for fault detection tasks. Unlike traditional classification models, they can perform well with limited data, as they focus on learning relationships or similarities rather than explicit classes[71, 74]. Models based on this type of architecture tend to generalize well to unseen examples by learning a similarity function, making them versatile for new fault scenarios or operating conditions. Fault detection datasets often suffer from imbalanced data, in which normal behavior samples usually outnumber faulty ones. SNNs mitigate this challenge by comparing pairs of samples rather than relying on direct class labels, which increases the dataset by performing different pair combinations.

Additionally, they can learn different feature distributions of subcases in the same class since the representation vector can point to different regions of the embedding space. In other words, these subcases can be placed at different locations while still resulting in a distance that correctly evaluates the similarity between the compared cases.

A typical SNN comprises two identical subnetworks called twin networks or twin units, which share the same architecture and weights. Each subnetwork processes one of the two inputs in parallel and produces a feature representation for each. The model compares these embeddings by calculating a distance

metric, such as Euclidean distance or cosine similarity, to determine the relationship between the inputs. During backward propagation, the algorithm uses the gradients from both inputs to update the shared weights[70, 71]. Figure 3.1 is a graphic representation of a general SNN architecture.

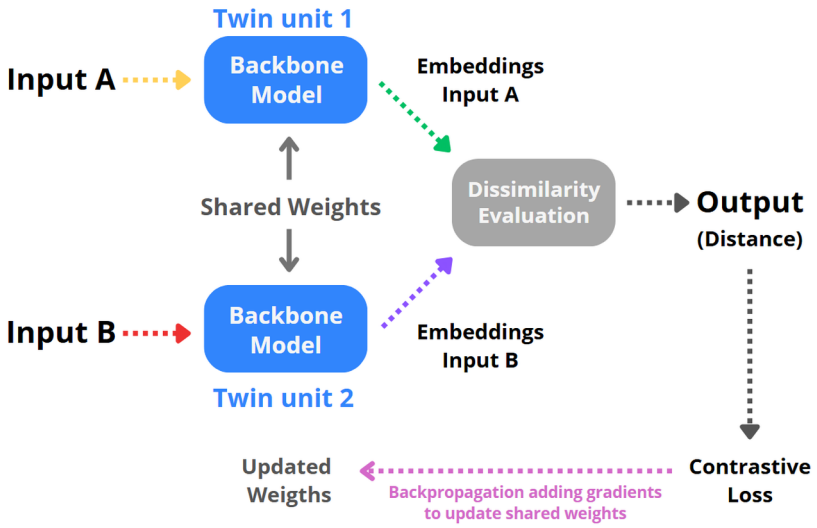

Figure 3.1: Diagram of an SNN framework.

The output of a SNN indicates a similarity measurement, which is not directly a classification. Thus, a loss function that connects such information with the correct label is required. This is the case of contrastive loss, as shown in Equation 3-1[72].

$$L = \frac{1}{N}\sum_{i=1}^{N}[(1-y_i)\cdot D^2 + y_i \cdot max(0, m-D)^2] \tag{3-1}$$

Where y is the label for the pair with zero for similar cases and one for dissimilar cases, D is the distance between the embeddings, m is a margin that defines the minimum distance required for dissimilar pairs, and N is the total number of samples in the set.

Any distance increases this loss by a quadratic factor when the target label is zero. By contrast, dissimilar cases only result in a penalty when the similarity measurement is between zero and the margin. In other words, this loss function enables the network to learn embeddings that minimize the distance for similar pairs and maximize the output for dissimilar pairs, considering a threshold and the target label. Therefore, the resulting model's distance output is directly associated with the classification task and is equivalent to a label prediction through comparison with the margin.

SNNs have been applied in various univariate detection scenarios. For instance, they have been used to monitor bridge vibration patterns indicative

of failure[76]. In power systems, SNNs have been used to identify anomalies in operating conditions by monitoring current waveforms[77].

3.2.2 Tennessee Eastman Process Overview

The Tennessee Eastman Process is a simulation of an industrial chemical process, providing comprehensive data for process control and fault detection studies. Introduced by Downs and Vogel in 1993 [28], the TEP models a dynamic chemical plant where multiple reactions interact to produce two primary products, as depicted in Figure 3.2. This simulation encompasses diverse features, including reaction kinetics, energy exchanges, and system disturbances, which collectively mirror the operational complexity of real-world chemical processes. The dataset includes various measured and manipulated variables, presenting challenges and opportunities for testing advanced fault detection methodologies.

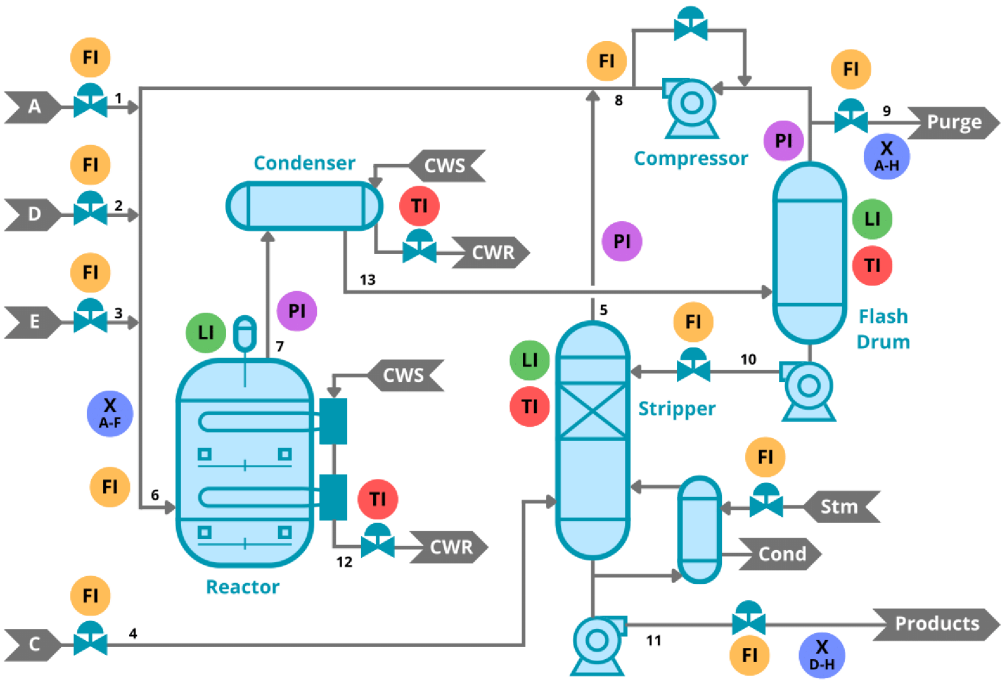

Figure 3.2: Diagram of Tennessee Eastman Process[3]. Image from Neto *et al.* 2025 [3], licensed under CC-BY 4.0, which was adapted from Andris Piebalgs (2020).[1]

3.2.3 Data and Software Availability

The dataset utilized in this study is identical to that used in our previous work and is available on the Kaggle platform

Table 3.1: List and Description of faults of the TEP.

No.	Description	Type
1	A/C feed ratio, B composition constant (stream 4)	Step
2	B composition, A/C feed ratio constant (stream 4)	Step
3	D feed temperature (stream 2)	Step
4	Reactor cooling water inlet temperature	Step
5	Condenser cooling water inlet temperature	Step
6	A feed loss (stream 1)	Step
7	C header pressure loss-reduced availability (stream 4)	Step
8	A, B, and C feed composition (stream 4)	Random
9	D feed temperature (stream 2)	Random
10	C feed temperature (stream 4)	Random
11	Reactor cooling water inlet temperature	Random
12	Condenser cooling water inlet temperature	Random
13	Reaction kinetics	Slow drift
14	Reactor cooling water value	Sticking
15	Condenser cooling water value	Sticking
16	Unknown	Unknown
17	Unknown	Unknown
18	Unknown	Unknown
19	Unknown	Unknown
20	Unknown	Unknown

(https://www.kaggle.com/datasets/averkij/tennessee-eastman-process-simulation-dataset)[39]. However, the preprocessing procedure was modified to prevent data leakage and to modify the data format to make it compatible with the SNN input. This dataset captures the behavior of a chemical system with four reactants (A, C, D, and E), an inert component (B), a by-product (F), and two products (G and H). It consists of a total of 52 variables: 12 manipulated variables, 22 process measurements, and 18 component analyses. Fault scenarios include 20 distinct types, as summarized in Table 3.1. In our previous study, we identified faults 3, 9, 10, 15, and 16 as particularly difficult to diagnose compared to the others, which is in accordance with the literature[40, 41, 3]. This dataset does not include the 21st fault, which was introduced in newer versions of the dataset.

3.3 Methodology

This section details the steps to preprocess data, investigate model architectures, train them under various configurations, and evaluate their performance. The study development was carried out in Python with the Pandas[53] and Matplotlib[78] libraries for data visualization. At the same time, Tensorflow[54], the Keras module of Tensorflow[55], and Scikit-learn[79]

were the tools used for data handling, model designing, training, and evaluation. Additionally, we executed all stages of the study on the same equipment with the following specifications: 32 GB RAM, i5-13600K CPU, and GeForce RTX4070 ti 12 GB.

3.3.1 Data Pretreatment

The data for this investigation was a subsample of the same TEP dataset we used in the fault diagnosis model[39, 3]. We exclusively used the data originally labeled training for both faulty and normal scenarios. Besides, we kept a 20-data-point observation window.

We structured the data in the expected format for a two-input SNN model. For better clarification, we refer to input one as the base time series and input two as the pair time series. Thus, a row of our dataset contains one base time series and one pair time series grouped as inputs, followed by the target label.

In order to avoid data leakage, we removed the samples present in the training, validation, and testing sets of our previous diagnosis study before performing a new random subsampling from the TEP dataset. All windows in the base time series correspond to normal behavior, ensuring that any detected fault originates from the pair time series. Additionally, we made sure to prevent the base time series from being compared to themselves, in order to avoid trivial comparisons and ensure meaningful pairwise learning.

The primary goal of the fault evaluation system is to distinguish between normal and anomalous operations, rather than to classify specific types of faults. Training on fault-fault pairs can lead to overfitting to intra-fault similarities, which weakens the discriminative signal needed to effectively separate normal from abnormal behavior. In other words, fault-fault pairs induce the model to learn subtle variations within the same fault class, rather than focusing on the critical differences between normal and anomalous behavior. Moreover, given the limited number of fault instances, the training strategy was focused on normal-normal and normal-fault pairs, as these align more closely with the objectives of fault detection.

The selection of faulty time series was stratified to represent all types of faults in equal proportions. The target labels followed the pairing process depending on the resulting combination, with zero being similar and one being dissimilar. Additionally, we kept a balanced dataset with 50 % for each label.

Despite isolating time series between datasets, we performed a random selection with replacement within the same dataset as long as the base-pair

combination remained unique. Consequently, we created datasets consisting of 70,000 samples for training, 20,000 for validation, and 10,000 for testing. Figure 3.3 summarizes the data structuring process.

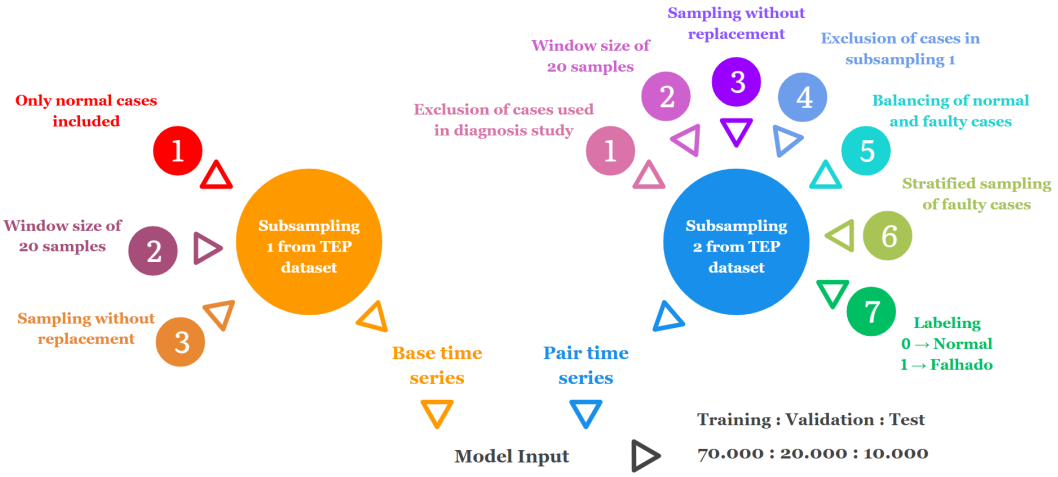

Figure 3.3: Summary of data structuring process.

3.3.2 SNN Architecture

The flexibility of the Siamese Neural Network framework allows suitable models to be used as its backbone, making it adaptable to different tasks. In this work, we applied transfer learning, a technique that uses a pre-trained model as a starting point for a new but related task [80]. By reusing and fine-tuning learned feature representations, transfer learning enables more efficient training, especially when labeled data is limited.

Transfer learning has been widely adopted in various machine learning domains due to its ability to mitigate data scarcity and distribution mismatch between training and target tasks [80, 81]. In this study, we explore the use of transfer learning not simply to reuse weights, but to strategically inherit domain-specific representations from a pre-trained fault diagnosis model. This approach is particularly relevant when the fault detection task shares underlying attribute space patterns with fault diagnosis. This enables better generalization and improved performance under fewer iterations, saving on computational resources.

Diagnosis models are explicitly trained to differentiate between various fault types, making them more adept at capturing meaningful feature representations for each condition. These rich feature representations can be adapted through transfer learning for the detection task, improving the model's ability to distinguish between normal and faulty states even for highly varied fault

feature distributions. Additionally, since a pre-trained diagnosis model already learned structured embeddings of fault conditions, the detection model benefits from a more informative input space, reducing the need for extensive labeled data and accelerating convergence during training. This approach effectively reframes detection as an embedding similarity problem rather than a direct classification task, aligning naturally with the Siamese Neural Network's strengths. For these reasons, we selected a backbone model that had already demonstrated strong diagnostic performance in our previous study[3].

Our previous model was derived from the model developed by Sun and Ren, 2021[2], which applies the Gramian Angular Summation Field (GASF) as part of the model architecture[38] to the multivariate system of input time series, resulting in a collection of matrices, one for each variable. This transformation allowed us to apply feature learning through CNN processing. Figure 3.4 illustrates the base architecture we used for the twin units.

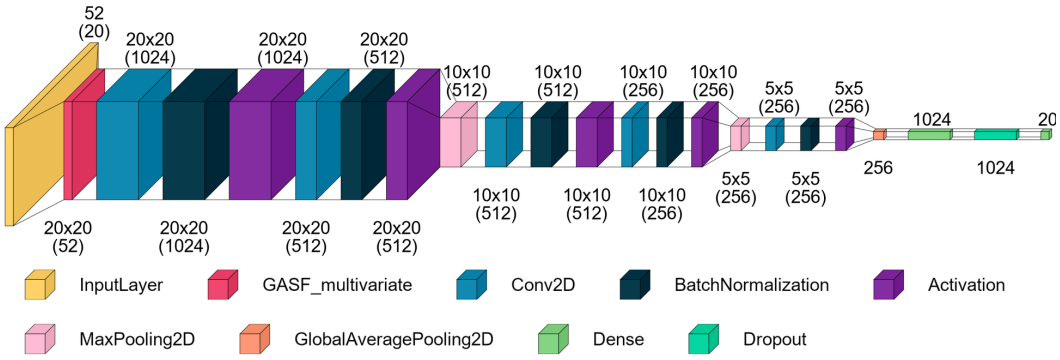

Figure 3.4: Best architecture from the diagnosis study. Image from Neto *et al.* 2025 [3], licensed under CC-BY 4.0.

Since we wanted a rich representation of the system, we removed the Softmax layer, resulting in an embedding vector of 1024 elements to describe each input. Due to the high dimensionality and interpretability, we selected the Euclidean distance to compute the similarity between embeddings, which is directly compatible with the contrastive loss formulation applied in this study. While other metrics such as cosine similarity could also be considered, incorporating them would require reformulating the loss function and fell outside the scope of this investigation.

3.3.3 Investigation Strategy

This section describes the investigation strategy used to refine the model. First, we focused on addressing the question: "How much data is enough?" We evaluated the impact of the training dataset size and explored

key training parameters to establish a baseline. Then, we analyzed stability control: overfitting in the baseline followed by underfitting in the stabilized model. Finally, we assessed the best-performing model through testing and comparison with existing literature.

We trained all models in this study using the Adam optimizer with its default learning rate of 10^{-3} and contrastive loss with a margin of 1. Following our previous work, we conducted training setting the batch size to 64 and a maximum of 500 epochs when applying early stopping. Additionally, we incorporated the Keras scheduler ReduceLROnPlateau with a reduction factor of 0.1 to dynamically adjust the learning rate.

In the first stage of our investigation, we examined the effect of training dataset size on model performance using a holdout validation approach. Models were trained on progressively larger subsets, with 2.5 % increments, until reaching the entire dataset of 70,000 samples. Initially, we set the early stopping patience to 5 and the ReduceLROnPlateau patience to 4. However, evidence of premature stopping prompted a second phase of this stage, where models were trained with a fixed amount of epochs to allow for better comparison. Since the most high-performing configurations completed training between 20 and 40 epochs for most cases, we selected the midpoint of this range (30 epochs) as a representative value for this phase to balance consistency and computational efficiency. This fixed epoch setup was used solely to support controlled comparisons across configurations and to inform adjustments to the Early Stopping and learning rate scheduler parameters in subsequent implementations. Based on the results from the second phase, we refined the patience values for early stopping and the learning rate scheduler to 15 and 10, respectively. We also included an analysis of the impact of the size if the training dataset on the training duration to evaluate computational resources used. Finally, this stage concluded with cross-validation using the selected configurations to establish a performance baseline for subsequent analyses. In all cross-validation cases in this study, we merged the training and validation sets and performed five-fold cross-validation to assess model consistency.

In the second stage of our investigation, we aimed to address model stability and mitigate overfitting, as the baseline model exhibited inconsistencies across cross-validation folds, an oscillatory behavior for the validation set, and a notable gap between training and validation performance. We froze all convolutional layers and conducted a new cross-validation under the same conditions because these behaviors could be related to the amount of trainable parameters in the model. Since the baseline model already incorporated regularization strategies, we did not introduce additional regularization techniques

at this stage.

Considering that freezing all convolutional layers solved the instability problem, but also showed evidence of performance stagnation, the third stage focused on employing strategies against underfitting. We considered three parallel investigations to enhance model performance, all using cross-validation to evaluate results. Firstly, we reduced the dropout rate in the dense layer, evaluating models with a 30 % dropout rate and no dropout, as the initial 50 % dropout may have excessively regularized the model. Secondly, we introduced an additional dense layer, testing configurations with 1024 and 512 neurons, followed by a dropout layer with a 50 % rate. All convolutional layers remained frozen in the dropout reduction and additional dense layer investigations. Thirdly, in our transfer learning framework, certain groups of layers are frozen to preserve the learned feature representations from the source fault diagnosis model, while others remain trainable to allow adaptation to the target fault detection task. In order to investigate the balance between preserving diagnosis knowledge, model adaptability, and quantity of trainable parameters, we established four groups of convolutional layers to be set as frozen in independent models, as depicted in Figure 3.5. Each group always starts at the first convolutional layer of the structure and stops before the next convolutional or pooling layer after the last layer of the previous group. Consequently, we have a total of four models to analyze the effect of progressively freezing the transferred layers.

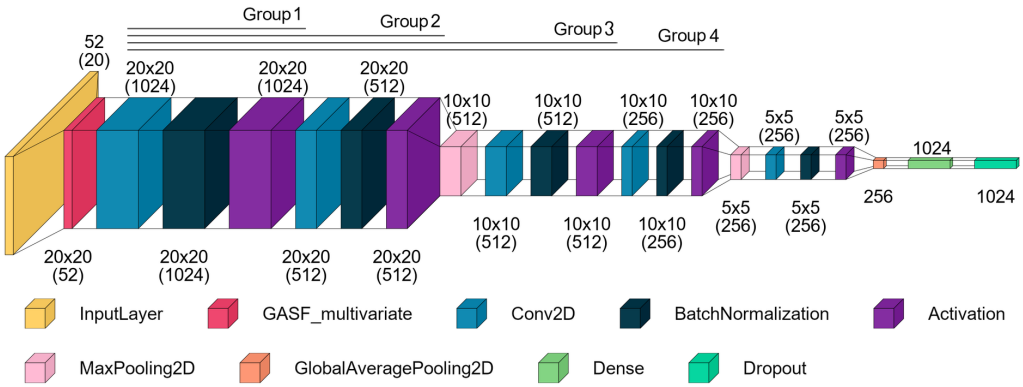

Figure 3.5: Groups of frozen convolutional layers in the twin unit. Image adapted from Neto *et al.* 2025 [3], licensed under CC-BY 4.0.

In the final stage, we trained the best-performing configuration using the holdout method with the validation dataset for early stopping and learning rate scheduling. We kept the last epoch state of trained models by setting the restore_best_weights parameter to false, as they have consistently presented the best F1-score. The resulting model was then evaluated on the testing

dataset through various analyses, including detection rate assessment, output distribution analysis, and probability equivalence evaluation for the general faulty class and individual fault types. In order to enhance detections results visualizations we performed a t-Distributed Stochastic Neighbor Embedding (t-SNE)[82] on the embeddings from the dense layer when applying the twin unit on the pair time series of the test datasets.

Finally, we compared our findings with results reported in the literature to contextualize our model's performance. Including the different folds, we trained a total of 131 models during this study, which individual training durations were recorded. Figure 3.6 shows a diagram summarizing the four investigation stages conducted in this research.

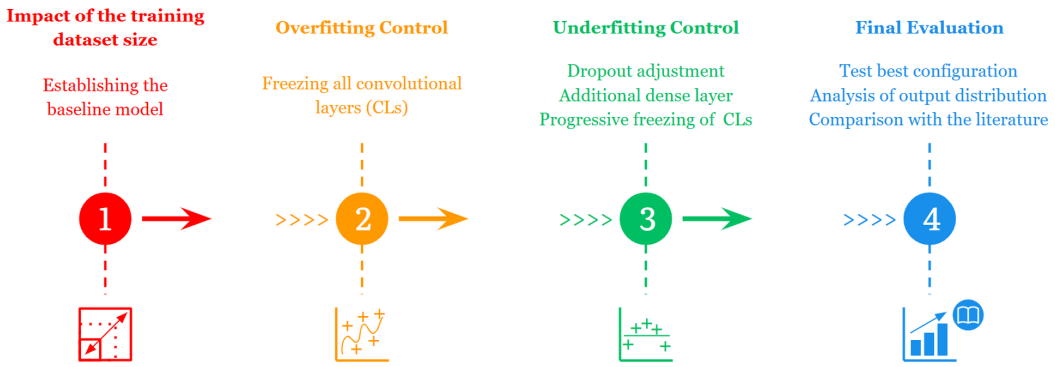

Figure 3.6: Summary of the investigation stages.

3.3.4 Model Performance and Training Evaluation Techniques

In this study, the F1-score was chosen as the primary evaluation metric, as it is well-suited for classification problems. The F1-score balances precision and recall, which makes it a robust measure for assessing model performance[83]. Equation 3-2 presents the expressions that compute precision, recall, and the F1-score. In fault detection, the recall is also referred to as the fault detection rate (FDR)[84], and we used this metric to evaluate the detection performance of individual faults in the testing stage. Additionally, we analyzed the distribution of predicted distances to investigate the model's capabilities further.

$$\begin{cases} Precision = \frac{TF}{TF+FF} \\ Recall = \frac{TF}{TF+FN} \\ F1_{Score} = 2 \cdot \frac{Precision \cdot Recall}{Precision+Recall} \end{cases} \tag{3-2}$$

Where TF is true faulty, FF is false faulty, and FN is false normal.

Additionally, we recorded the L2 gradient norm for the baseline model and all subsequent trained cases to monitor training stability and convergence. Equation 3-3 provides the expression used to compute the L2 gradient norm, which indicates optimization behavior throughout training.

$$L_{2,norm} = \sqrt{\sum_{i}^{N} gradient_i^2} \tag{3-3}$$

Since the predicted distance is not bound to a closed range, we applied a modified sigmoid function, as shown in Equation 3-4, to the model's predictions to better interpret the predicted distance distribution around the margin. This transformation sets the margin to 0.5 and limits the scale between 0 and 1. Furthermore, we used a reliability diagram[85, 86] to assess whether the Sigmoid-transformed distance output could be interpreted probabilistically using the calibration_curve function from Scikit-learn setting it to 10 bins.

$$sigmoid(D) = \frac{1}{1 + e^{(m-D)}} \tag{3-4}$$

Where D is the predicted distance and m is the margin set in the contrastive loss.

3.4 Results and Discussion

This section presents and analyzes the study findings, dividing them into two main parts. In the first subsection, we examine the impact of dataset size on model performance, establish a baseline configuration, and refine key training parameters. In the second part, we further assess the final model's performance, stability, and generalization capabilities, providing a comparative evaluation against existing literature.

3.4.1 Training dataset size, Baseline, and Hyperparameter Investigations

Figure 3.7 summarizes the results of the initial investigation. It is possible to observe a considerable increase in the resulting model performance until 5,250 samples (7.5 % of the total training dataset). The training performance steadily increases for cases of 7,000 or more samples until the entire dataset is included. However, performance on the validation set fluctuates considerably for intermediate training dataset sizes, reproaching training performance for more extensive training data. The gap observed could be related to overfitting, but it could also be related to a premature triggering of the early stopping callback.

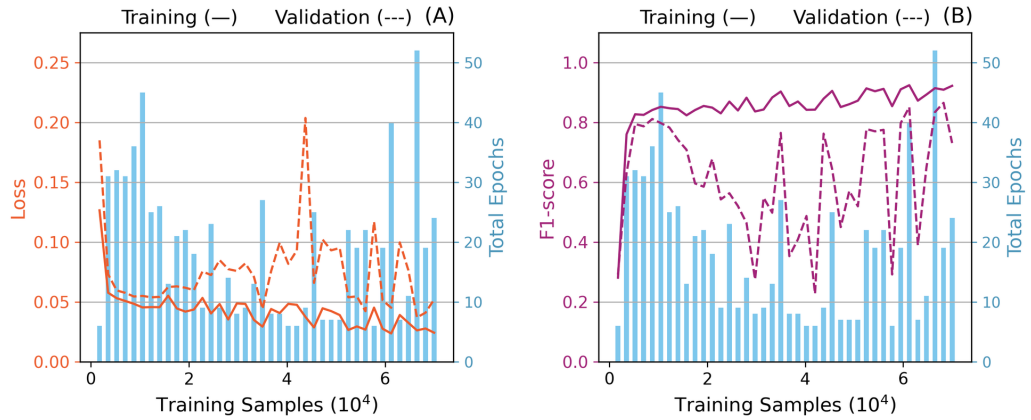

Figure 3.7: Loss (A) and F1-score (B) learning curves with early stopping and learning rate scheduler patience terms of 5 and 4, respectively.

In fact, most instances of reduced validation performance are associated with a lower number of training epochs. For example, the configuration using 68,250 training samples (97.5 % of the dataset) yielded the highest validation F1-score (86.50 %), with a modest gap of 4.38 % compared to the corresponding training F1-score, and completed after 19 epochs. In contrast, the configuration with 42,000 training samples (60 % of the dataset) resulted in the lowest validation F1-score (22.66 %), which was 61.56 % lower than its training performance and halted prematurely at just 6 epochs.

From the data presented in Figure 3.8, it is possible to observe that the increase in training F1-score is smooth and improves as the training dataset size increases. Contrarily, the same score on the validation data has shown an increasing but considerably oscillatory behavior, especially in the initial epochs.

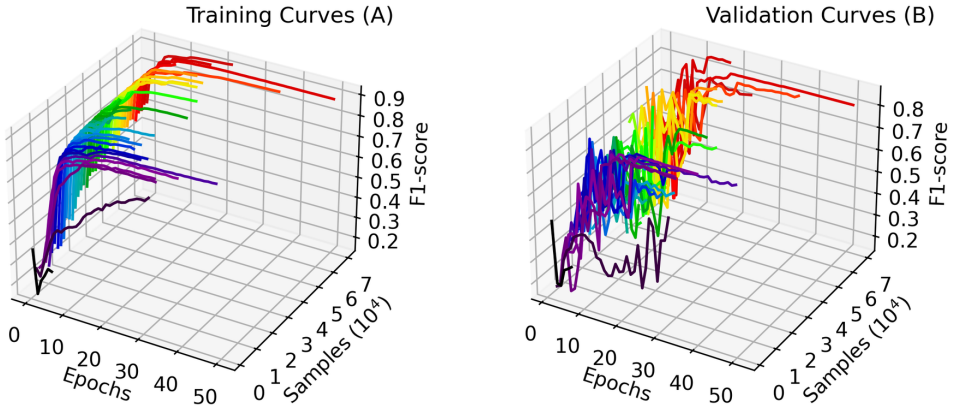

Figure 3.8: Training (A) and validation (B) learning curves with early stopping and learning rate scheduler patience terms of 5 and 4, respectively.

Such behavior is another evidence of premature early stopping, which indicates that the patience term might be too short. Since cases of better performance stopped between 20 and 40 epochs, we decided to reevaluate the effects of the training set data size with a fixed quantity of epochs equal to 30.

With the learning curve results with increasing dataset size and the fixed quantity of epochs, it was possible to build Figure 3.9. Similarly to the previous configuration, there is a spike in performance until 5,250 samples. However, the training performance increased and became more stable for cases with more data.

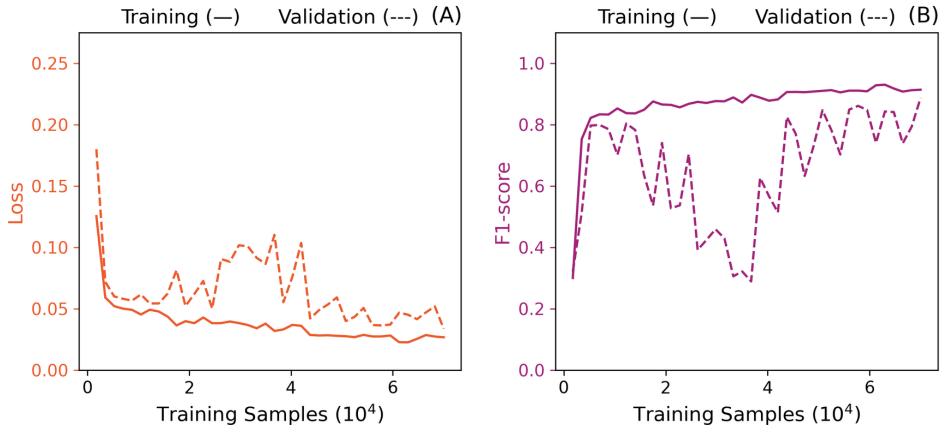

Figure 3.9: Loss (A) and F1-score (B) learning curves with 30 epochs and learning rate scheduler patience term of 4.

Although significant gaps for intermediate amounts of training data remained, validation performance has shown fewer fluctuations. Meanwhile, cases with more than 43750 training samples presented improved results.

Using the largest training dataset in the study corresponded to the highest validation F1-score of 88.89 %. This case also presented a 2.25 % gap from the training performance, which is a considerable improvement when compared to the previous configuration. These results are an indication that the amount of data we select seems to be appropriate to the problem we are trying to solve.

From observing the individual training curves shown in Figure 3.10, it is possible to conclude that intermediate cases reached an early stagnation during training. This effect is most likely related to an early triggering of the learning rate scheduler, which indicates a patience term of 4 is too short. Also, most cases have presented a more oscillatory behavior between 10 and 15 epochs. Due to these reasons, we selected *patince* terms of 10 and 15 for the lr scheduler and the early stopping, respectively, for the following implementations of this study.

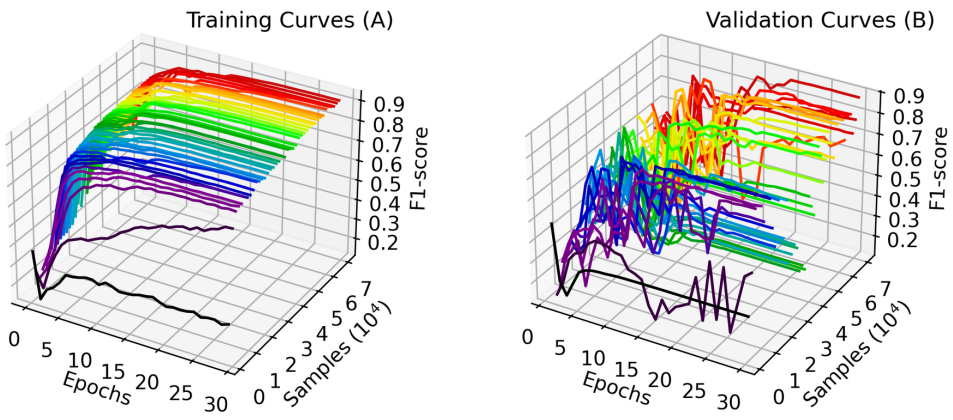

Figure 3.10: Training (A) and validation (B) learning curves with 30 epochs and learning rate scheduler patience term of 4.

Having a fixed amount of epochs allows us to analyze the impact of the amount of training data samples on the training duration. It is possible to observe a linear relationship between these two factors in Figure 3.11. In fact, we obtained an R2 of 0.9991 after applying a linear regression.

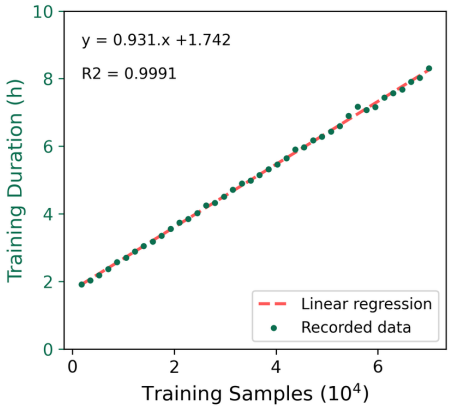

Figure 3.11: Impact of training dataset size on the training duration.

In order to evaluate stability, we performed cross-validation on the baseline model and all other hyperparameter investigations. The baseline implementation reached F1-scores of 93.99 ±0.93 % and 83.85 ±2.92 % for the training and validation datasets, respectively. Besides, the training duration was 11.3 ±2.7 hours with 46.4 ±11.3 epochs, representing considerable instability. Such behavior can also be observed in Figure 3.12, as the oscillation on the validation set seems to trigger a premature stopping on folds 2, 3, and 4. Additionally, Figure 3.13 shows the L2-norm of the gradients.

Gradients in folds 3 and 4 remained more significant than the other folds, indicating more impactful weight updates. A factor that could be

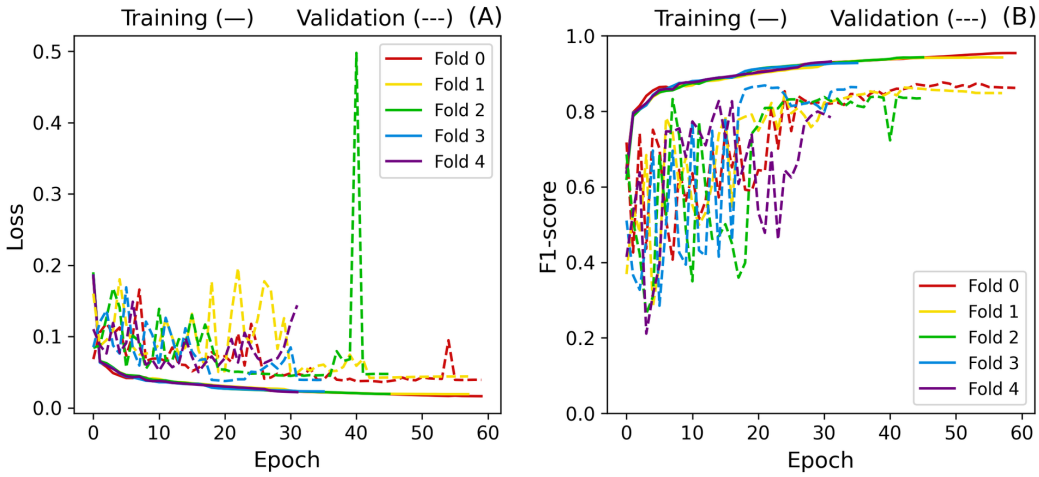

Figure 3.12: Baseline cross-validation Loss (A) and F1-score (B) curves.

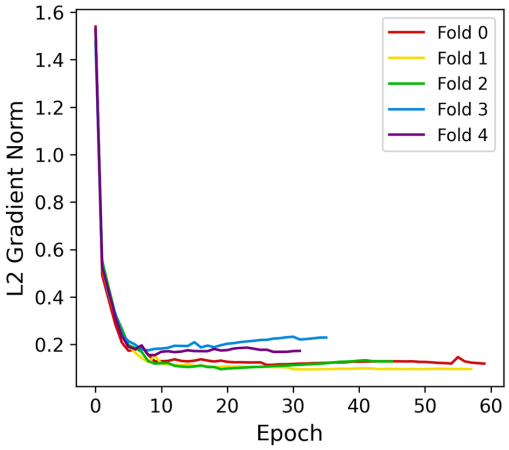

Figure 3.13: Baseline L2 gradient norm.

contributing to this instability is the amount of trainable parameters in the model (9,597,440), which could also lead to overfitting. Besides, changing the weights from the convolutional layers may shift the model away from the patterns learned from diagnosis.

Due to the reasons stated above, stage two consisted of freezing all layers before the dense layer of the backbone model. Doing so ensures the knowledge transferred is not entirely "forgotten" while making the extracted features' processing adaptable.

As expected, this procedure significantly impacted the model's stability. The cross-validation resulted in 91.41 ±0.12 % of training f1-score and 90.84 ±0.18 % of validation f1-score, representing a significant improvement in consistency and generalization. The average difference between training and validation for this metric is 0.57 % compared to the 10.14 % baseline. Furthermore, the standard deviation was reduced by 0.81 % for training and

2.76 % for validation, as shown in Figure 3.14.

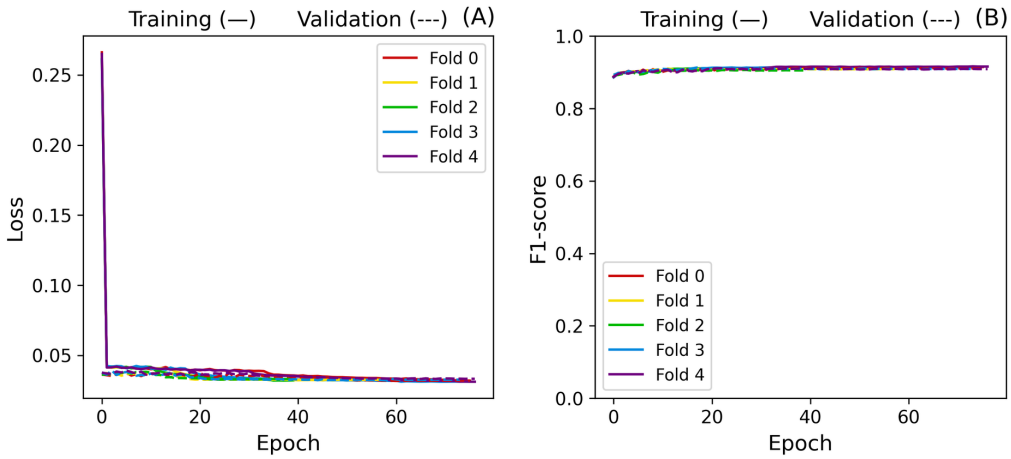

Figure 3.14: Cross-validation Loss (A) and F1-score (B) curves after freezing convolutional layers.

Not only did the model reach a higher performance on its first epoch, but validation metrics persisted smooth througout the training process. Interesting insights can also be taken from Figure 3.15. All gradients drop close to 0.22 before increasing again and stabilizing around 0.48. Since all folds starting parameters are the same, this different increasing pace observed could be due to how the optimizer navigates the loss landscape given the fold splits. Nevertheless, they seem to converge to similar final states.

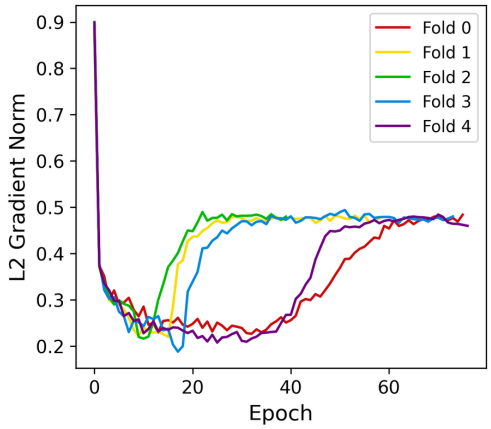

Figure 3.15: L2 gradient norm after freezing convolutional layers.

Although the number of trainable parameters was 263,168 (9,334,272 less than the baseline model) the training duration was 15.8 ±3.5 h. This duration increase results from the quantity of epochs executed before training stopped: 65.2 ±14.2.

Since the model's performance increases gradually from the first to the last training epoch, we considered the possibility of an over-regularizing due

to the 50 % dropout rate on the dense layer. For this reason, we started stage three by investigating the effects of reducing and removing the dropout effect.

There was no improvement in reducing the dropout effects on the dense layer. Contrarily, removing the dropout layer lowered the training f1-score to 90.95 ±0.13 %, with insignificant changes in validation performance. These results indicate that the 50 % dropout rate is not causing over-regularization.

The following hypothesis was that the model did not have enough complexity to process the extracted features further. Consequently, we defined the second part of stage three as evaluating the effects of making the dense part of the model deeper by adding a second dense layer with 512 and 1024 neurons.

In both cases, adding another dense layer resulted in a similar reduction of the F1-score. Besides, the average gap between training and validation for the same metric was 1.32 % for 512 neurons and 1.48 % for 1024 neurons, which indicates that a second hidden layer might increase the tendency to overfitting. Contrastingly, we observed a significant learning duration reduction in both cases: 6.3 ±1.2 hours stopping at 30.0 ±6.1 epochs for 512 neurons and 7.1 ±1.7 hours stopping at 33.0 ±8.6 epochs for 1024 neurons. These results indicate that a lack of complexity on the dense part of the model does not seem to be the limiting factor for prediction enhancement.

Finally, since freezing the entire convolutional portion of the model can limit the model's flexibility to adapt to the detection problem, the final hyper-parameter investigation consisted of progressively freezing the convolutional structure. Figure 3.16 summarizes all the performance for all configurations from stage three of this study.

It is possible to observe that freezing groups one and two - which include fewer layers - increased training performance while lowering scores on validation compared to the baseline. Such behavior indicates overfitting. On the other hand, group three displayed a considerable improvement, with an average of 4.61 % F1-score gap between training and validation. This performance is less consistent than the results for freezing all convolutional layers, as freezing group three resulted in a more significant gap between training and testing performance and larger standard deviations. Given the amount of charts, all training curves and L2 gradient norm for configurations from stage three can be found in Appendix B.

The model obtained from freezing group four presented this study's highest cross-validation average performance. It presented training and validation F1-scores of 92.48 ±0.49% and 91.38 ±0.20%, respectively, implicating an average gap of 1.10 %. Although these variation ranges are wider than freezing all

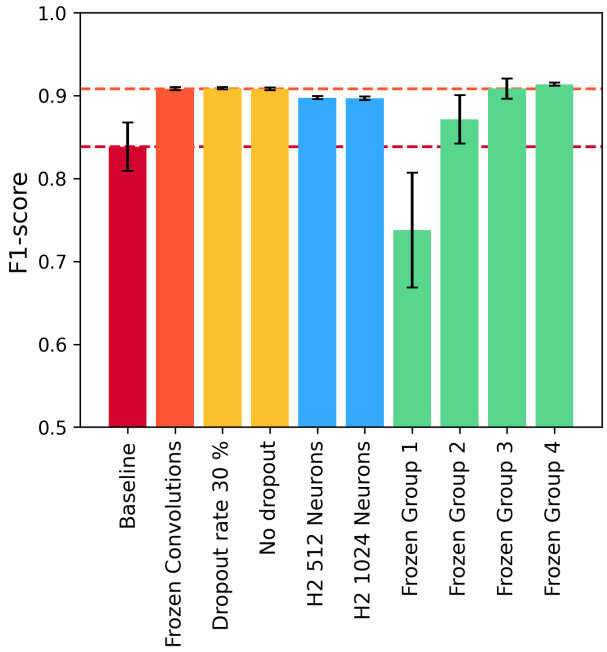

Figure 3.16: Average cross-validation performance for configurations from stage three.

convolutions, we chose this as the best configuration in the study because they are lower than 1.50 % with improved average performance. Besides, training achieved convergence in fewer iterations: 44.8 ±4.9 epochs. Consequently, the training duration was 5.1 hours faster on average, taking 10.7 ±1.1 hours, even with 853,760 trainable parameters out of 9,602,562.

In the context of stability, although the model presented fluctuations when evaluating the validation dataset, it is possible to observe in Figure 3.17 that they are more prominent in the initial learning stages. Additionally, although the increase in the validation loss triggered the early stopping, the F1-score of the last epoch was consistently the highest. This phenomenon indicates that the loss function does not directly represent our primary classification metric. This difference is expected since the contrastive loss is distance-based, while the F1-score is classification-based. Due to this trend, we kept the last state of the model for the last part of the study.

Regarding the L2 gradient norm shown in Figure 3.18, all folds presented an overlapping steady decline behavior. However, folds 0 and 4 became stable at slightly higher values than the others, which is insignificant compared to previous cases.

Since the last convolutional layers are responsible for extracting the most abstract information related to the investigated problem, it is coherent that keeping these layers trainable was beneficial to the learning process. Due to the improved performance and stability observed, we selected the freezing of the

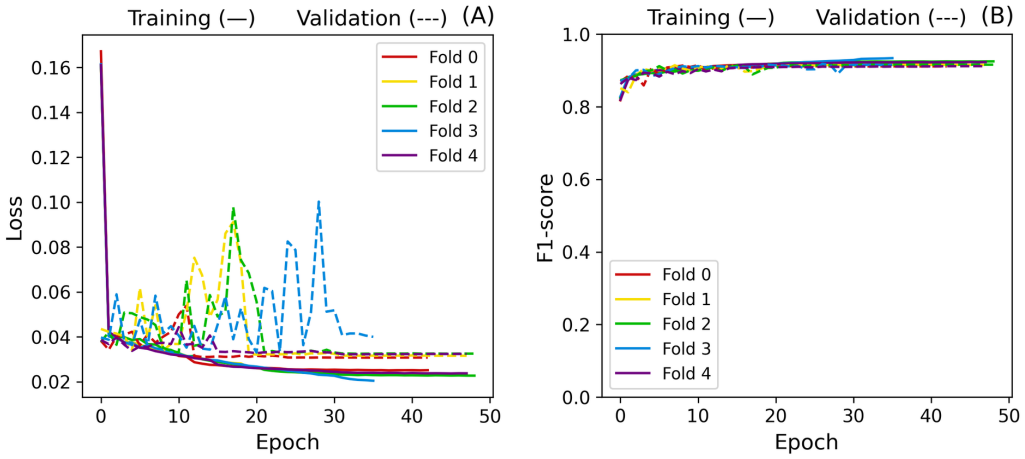

Figure 3.17: Cross-validation Loss (A) and F1-score (B) curves after freezing Group 4.

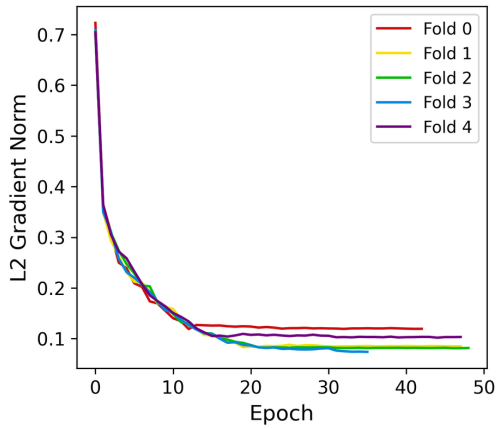

Figure 3.18: L2 gradient norm after freezing Group 4.

layers in group four as the best configuration within the scope of this study. Table 3.2 summarizes the results of all parameters that we investigated in our research.

Table 3.2: Summary of investigated hyperparameter results.

Case	**F1-score (%)**		**Epochs**	**Duration (h)**
	Training	**Validation**		
Baseline	93.99 ±0.93	83.85 ±2.92	46.4 ±11.3	11.3 ±2.7
Frozen Convolutions	91.41 ±0.12	90.84 ±0.18	65.2 ±14.2	15.8 ±3.5
30 % dropout	91.31 ±0.07	90.89 ±0.15	65.2 ±13.2	13.5 ±2.7
No dropout	90.95 ±0.13	90.82 ±0.15	62.6 ±13.0	13.1 ±2.6
H2 512 Neurons	91.07 ±0.10	89.76 ±0.18	30.0 ±6.1	6.3 ±1.2
H2 1024 Neurons	91.17 ±0.09	89.69 ±0.21	33.0 ±8.6	7.1 ±1.7
Frozen Group 1	93.91 ±0.92	73.77 ±6.95	28.0 ±0.9	6.9 ±1.5
Frozen Group 2	95.29 ±0.49	87.13 ±2.91	31 ±3.7	7.6 ±1.0
Frozen Group 3	95.46 ±0.22	90.85 ±1.25	28.2 ±3.4	6.9 ±0.8
Frozen Group 4	92.48 ±0.49	91.38 ±0.20	44.8 ±4.9	10.7 ±1.1

3.4.2
Analysis of the best configuration

We trained the model with the best configuration from scratch using the holdout method and the initial training and validation split for the testing evaluation. Figures 3.19 and 3.20 show that the holdout implementation yielded the same training behavior as observed in the cross-validation investigation. The training reached 44 epochs before early stopping, taking 8.8 hours to finish, with training and validation F1-scores of 92.74 % and 91.57 %, respectively.

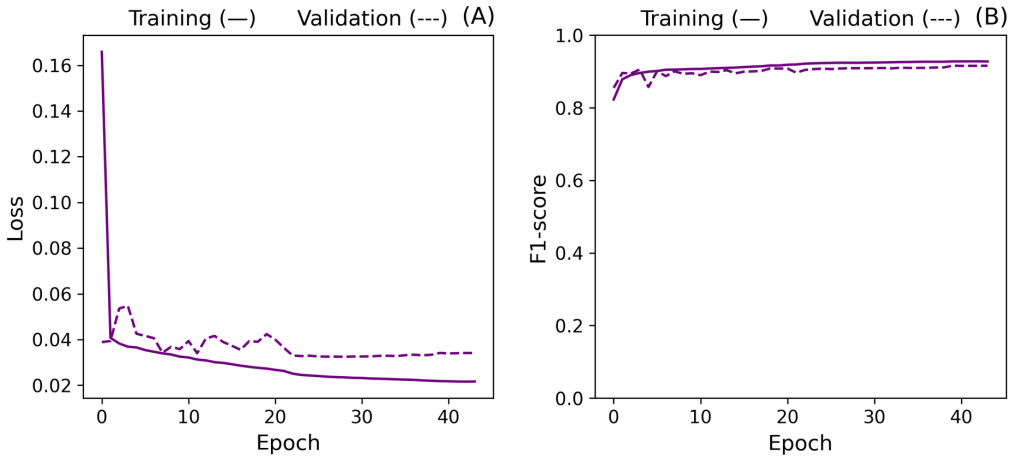

Figure 3.19: Best model's holdout Loss (A) and F1-score (B) curves.

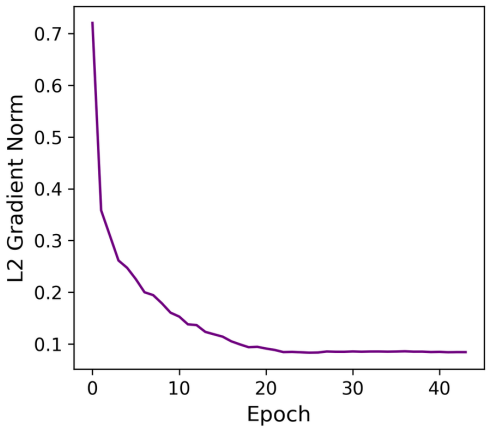

Figure 3.20: Best model's holdout L2 gradient norm.

In terms of deployment feasibility, the average inference time of the best performing model was 0.011 seconds per test sample. This is substantially faster than the dataset's sampling interval of 3 minutes, indicating that the model is capable of operating in real time with considerable margin. This suggests a strong potential for industrial implementation, as the model can deliver timely predictions with minimal computational overhead once trained.

Performance-wise, the model achieved an F1 score of 91.41 % and a contrastive loss of 0.072, which is close to the validation, indicating that no significant overfitting effect occurred.

Analyzing the distribution of distances predicted by the model is important for a better understanding of its behavior. Figure 3.21 shows a strongly skewed distribution towards zero for cases of normal behavior with an average predicted distance of 0.11 and a standard deviation of 0.16. This distribution indicates that the model was able to learn new patterns that were not directly present in the transferred knowledge. On the other hand, in the case of faulty behavior, the distance distribution is wide, with an average of 9.08 ±6.01, which is expected as multiple types of fault can result in different embedding representations. Nevertheless, the majority of faulty cases stay above the margin.

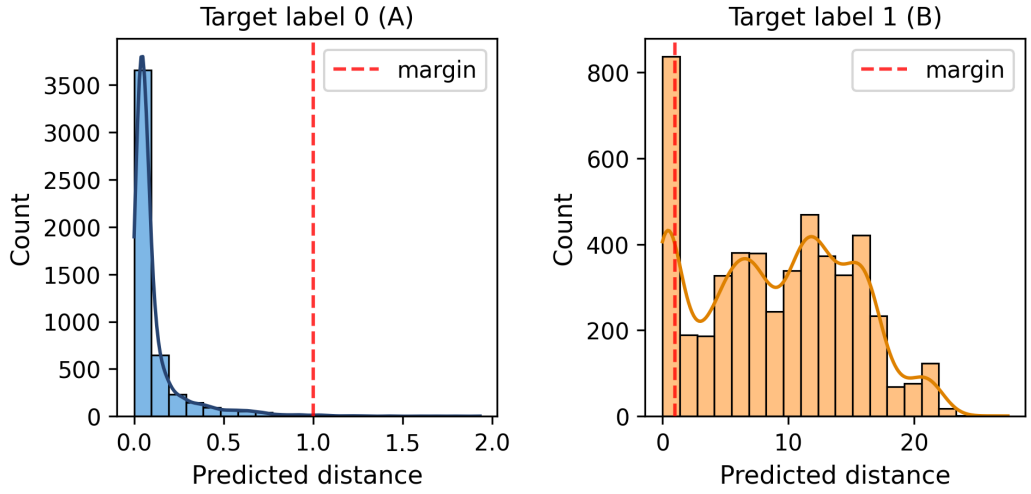

Figure 3.21: Distribution of predicted distances of test normal (A) and faulty (B) cases.

There is no limit to the model's predicted distances, which could result in misleading interpretations of the distribution charts. In contrast, the sigmoid function smoothly sets the values between zero and one. Consequently, it is possible to better represent the classification distribution using Equation 3-4. Figure 3.22 shows that most faulty cases are positioned far from the margin, except for a subset of misclassifications.

The reliability diagram from the transformed values is shown in Figure 3.23. As the curve indicates poor calibration, we conclude that this output should not be treated as a true probability. Instead, it functions as a normalized similarity score. Importantly, the model still performs reliably on normal samples, as reflected in the classification metrics and distribution analysis.

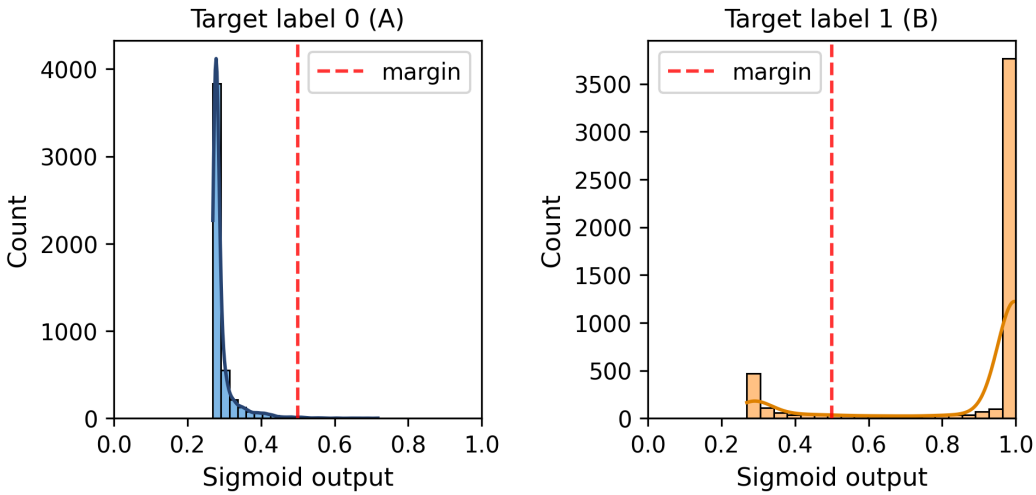

Figure 3.22: Sigmoid output of test normal (A) and faulty (B) cases.

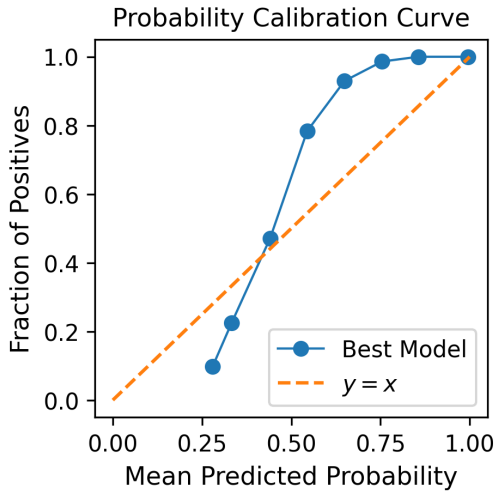

Figure 3.23: Probability calibration curve of our best model.

Although we present a probability calibration curve, our primary aim was not to achieve a calibrated probability output but rather to assess whether the transformed scores behaved like probabilities. The observed curve indicates that the model's output deviates from ideal calibration. While post-processing techniques such as temperature scaling or threshold adjustment could potentially improve calibration, these were not pursued in this study, as our objective was to analyze the inherent behavior of the model's transformed output rather than to optimize it for calibrated probabilistic predictions.

Since we have access to the specific fault label of each sample, we analyzed each fault to better understand detection predictions. It is important to mention that these results are still based on the normal versus faulty predictions from the model, not to be mistaken with fault diagnosis modeling. Table 3.3 summarizes the detection rate and statistical information of predictions.

Table 3.3: Individual Fault Results and Average Model Outputs.

Fault Case	**FDR**	**Average Distance**	**Average Sigmoid Output**
Fault 1	1.0000	11.94 ±1.64	1.00 ±0.00
Fault 2	1.0000	15.55 ±1.36	1.00 ±0.00
Fault 3	0.3911	0.93 ±0.82	0.48 ±0.18
Fault 4	1.0000	6.87 ±1.10	0.99 ±0.01
Fault 5	1.0000	6.92 ±2.30	0.99 ±0.04
Fault 6	1.0000	20.08 ±1.98	1.00 ±0.00
Fault 7	1.0000	16.96 ±1.54	1.00 ±0.00
Fault 8	0.9959	10.92 ±3.23	0.99 ±0.05
Fault 9	0.1399	0.45 ±0.70	0.37 ±0.15
Fault 10	0.8385	4.95 ±3.81	0.82 ±0.24
Fault 11	0.9958	9.08 ±3.53	0.98 ±0.07
Fault 12	0.9959	10.74 ±2.88	0.99 ±0.05
Fault 13	0.9562	10.80 ±3.07	0.97 ±0.14
Fault 14	1.0000	11.62 ±0.96	1.00 ±0.00
Fault 15	0.0075	0.14 ±0.20	0.30 ±0.04
Fault 16	0.8618	6.98 ±5.68	0.85 ±0.23
Fault 17	0.9788	14.33 ±3.63	0.98 ±0.10
Fault 18	0.9014	12.46 ±4.76	0.93 ±0.21
Fault 19	0.9919	4.73 ±1.45	0.95 ±0.09
Fault 20	0.9291	5.89 ±2.54	0.93 ±0.18

Faults 3, 9, 10, 15, and 16 were the cases with the worst performances, which is coherent with what we observed in our previous diagnosis study. Faults 10 and 16 remained above 80 %, while the others were undetectable in more than 50 % of cases with an average predicted distance below the margin. Figure 3.24 shows how the model's predictions in the three most critical cases are more skewed toward normality. All faults with FDR above 90 % resulted in distributions of the sigmoid transformation heavily leaning toward 1.0. The complete collection of histograms of distances and sigmoid transformations for each fault can be found in Appendix B.

We applied t-SNE to project the embeddings from the dense layer into a lower-dimensional representation to visualize the high-dimensional feature space. Class zero represents fault-free samples, while each other class corresponds to a fault with the same number as the class.

It can be observed in Figure 3.25 that normal cases occupy a broad region of the plot, whereas certain fault classes form more compact clusters. To improve the analysis of the relationship between detection performance and the distribution of embeddings, we separated the data from t-SNE into two visualizations.

Figure 3.26 presents the normal cases alongside fault classes according to their detection rate for better visualization. In Figure 3.26 (A), we observe that

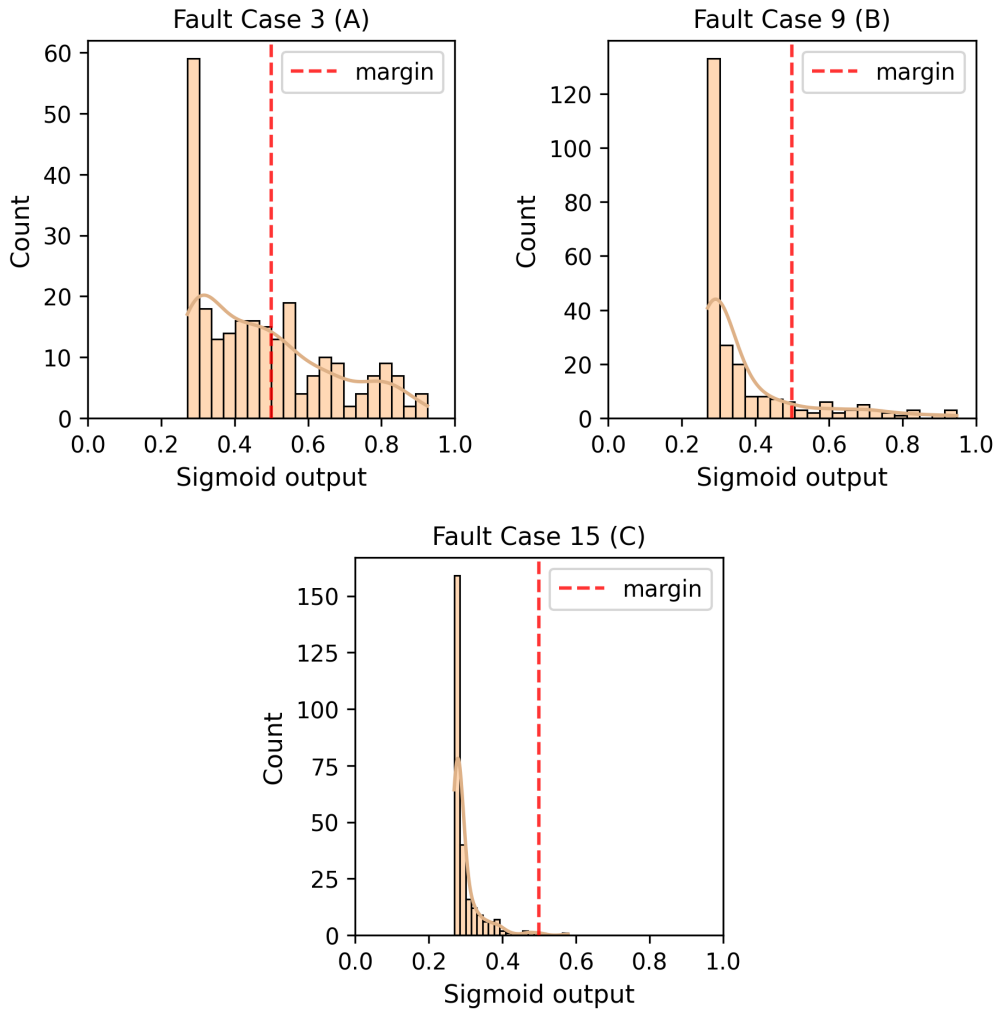

Figure 3.24: Test sigmoid output of fault cases 3 (A), 9 (B), and 15 (C).

faults 10 and 16 seem to have individual clusters with partial overlap with the normal case samples. However, Cases 3, 9, and 15 are dispersed and overlap substantially with normal behavior, indicating that the model was unable to detect patterns in these cases.

From analyzing Figure 3.26 (B), it is possible to observe that, in addition to being significantly separate from normal behavior, some cases show little overlap with other faulty conditions. This result indicates that the model could adapt characteristics from the diagnosis model to the detection task, as our contrastive loss setup does not focus on optimizing discrimination between faulty classes.

To evaluate the effectiveness of our model, we compared its detection rate with three recent studies of the TEP. Ma *et al.* (2024)[87] investigated an autoencoder architecture with a multi-block orthogonal long short-term memory backbone (MOLA). Their model used 30 input variables divided between 4 processing blocks according to specific parts of the chemical plant.

[!ht]

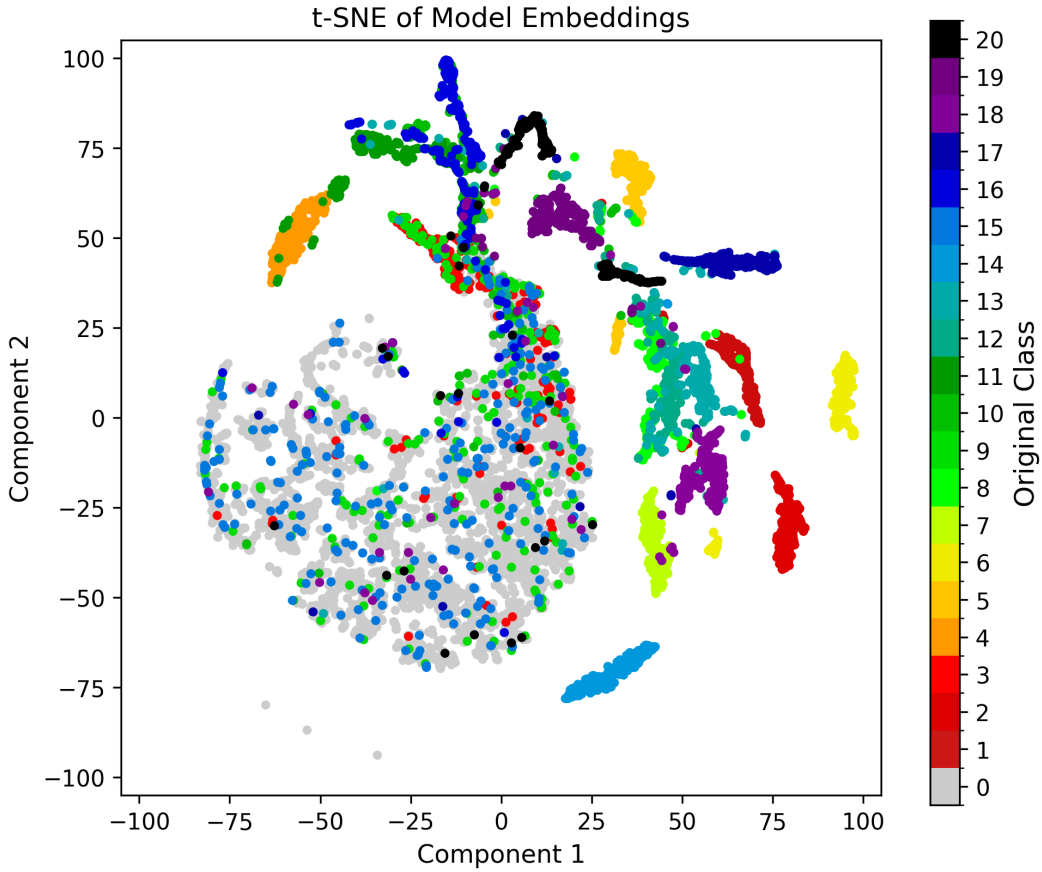

Figure 3.25: t-SNE of the embedding of the pair series from the test dataset.

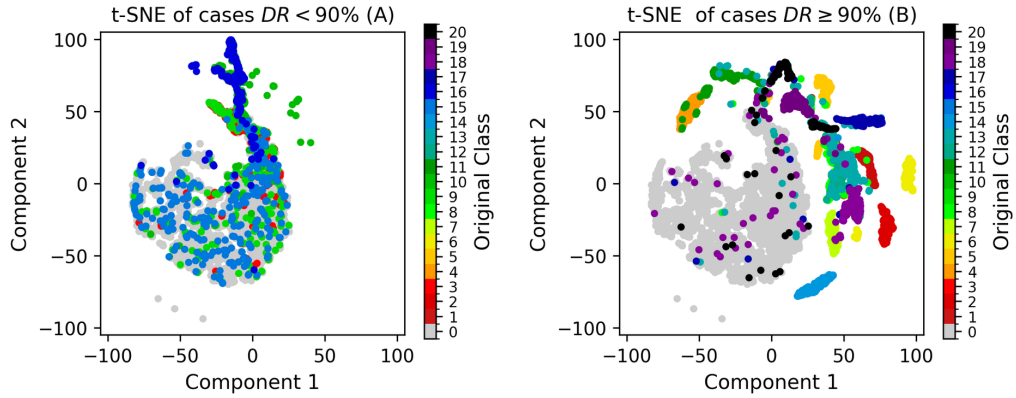

Figure 3.26: t-SNE of faults from the test dataset separated by detection rate (DR) lesser than 90% (A) and greater or equal to 90% (B).

Similar to our study, they included the original 20 fault cases.

Alternatively, Hu *et al.* (2025)[88] developed joint time-serial variation analysis (JTSVA), which integrates three sequential data feature extraction techniques, aiming to achieve enhanced discrimination of faults. This study explored 17 of the original faults in addition to the most recent one not included

in our study.

Dong *et al.* (2025)[89] proposed a hybrid model focusing on distinguishing anomalous behavior from nonstationary trends. This method uses slow feature analysis with the local outlier factor algorithm to improve model detection through dynamic updates of fault-sensitive variables. Their study considered all TEP faults.

Table 3.4 summarizes the FDRs reported in the studies mentioned alongside the results from our best model for the 20 original TEP faults. Our model achieved the highest detection performance for 65 % of faults. We have identified faults 3, 9, 10, 15, and 16 as the hardest to detect, which is also true for most references. Notably, Ma *et al.* (2024)[87] reached higher performance in these cases at the expense of lowering FDRs of easier scenarios. Finally, in cases 12 and 18, our model is on par with the best scores of the references, staying within 0.3 %.

Table 3.4: FDR comparison between our best model and state-of-art works from the literature.

Fault Case	**Our Model**	**Ref.1**[a]	**Ref.2**[b]	**Ref.3**[c]
Fault 1	**1.0000**	0.9983	**1.0000**	0.998
Fault 2	**1.0000**	0.9767	0.9838	0.984
Fault 3	0.3911	**0.9567**	N/A	0.688
Fault 4	**1.0000**	**1.0000**	**1.0000**	0.998
Fault 5	**1.0000**	0.2633	**1.0000**	0.249
Fault 6	**1.0000**	**1.0000**	**1.0000**	**1.000**
Fault 7	**1.0000**	**1.0000**	**1.0000**	**1.000**
Fault 8	**0.9959**	0.9433	0.9813	0.981
Fault 9	0.1399	**0.8850**	N/A	0.418
Fault 10	0.8385	0.8833	**0.8950**	0.808
Fault 11	**0.9958**	0.9533	0.7888	0.993
Fault 12	0.9959	0.6467	0.9988	**0.996**
Fault 13	**0.9562**	0.8733	0.9525	0.941
Fault 14	**1.0000**	0.9967	**1.0000**	0.780
Fault 15	0.0075	**0.9850**	N/A	0.418
Fault 16	0.8618	0.6450	**0.9263**	0.893
Fault 17	**0.9788**	0.9233	0.9650	0.905
Fault 18	0.9014	0.7633	**0.9038**	0.903
Fault 19	**0.9919**	0.9817	0.9163	0.870
Fault 20	**0.9291**	0.7933	0.9100	0.905
Average FDR	0.8492	**0.8734**	N/A	0.836
Average excluding 3, 9, and 15	**0.9674**	0.8613	0.9542	0.894

[a] Ma *et al.* (2024) [87]; [b] Hu *et al.* (2025) [88]; [c] Dong *et al.* (2025) [89].

Our model demonstrates strong fault detection capabilities, achieving an average FDR of 0.8492 across all 20 TEP faults, comparable to prior studies. While Ma *et al.* (2024)[87] attained a higher average FDR by 2.42 %, our

approach excelled when excluding faults 3, 9, and 15, reaching 0.9674, the highest among the analyzed references.

3.5 Conclusions

This study explored a new approach to fault detection in the Tennessee Eastman Process by combining a Siamese Neural Network architecture with transfer learning. We transformed the detection problem into an embedding similarity task using a pre-trained fault diagnosis model as the backbone, enabling the model to differentiate between normal and faulty behavior more effectively. Our results demonstrate that the best-performing configuration achieved an F1-score of 91.41 % on the test dataset, with fault detection rates competitive with recent literature.

Our investigation addressed key challenges such as overfitting and stability. The stepwise freezing of convolutional layers proved critical in maintaining feature extraction quality while allowing adaptability to the detection task. Despite the overall strong performance of our method, faults 3, 9, and 15 remain particularly difficult to detect. These faults are known to be subtle or slow-developing as they are related to heat transfer phenomena. While our model achieved state-of-the-art performance for most faults, further refinements are needed to enhance detection for these complex scenarios. Future work could explore hybrid models that integrate temporal attention mechanisms, domain-specific physical constraints, or explore the use of focal loss to enhance sensitivity to these cases.

It would benefit the literature if future works could also explore the impact of alternative distance measures on model performance, particularly in combination with modifications to the loss function. Furthermore, investigating the integration of the detection model with the diagnosis model in an ensemble framework could provide deeper insights into the full scope of the FDD problem. Finally, while our investigation focuses on the Tennessee Eastman Process dataset due to its widespread use as a benchmark, applying our approach to other industrial datasets would provide valuable information about its generalization capabilities and potential for broader applications in practical fault detection scenarios.

4
General Conclusions and Perspectives

In this doctoral research, we investigated innovative methodologies for industrial process fault monitoring by developing and evaluating two deep learning frameworks. The work establishes that strategic architectural refinements can yield significant performance improvements even within established neural network paradigms. At the same time, innovative problem reformulations can overcome persistent data challenges.

The first research phase demonstrated that concentrating efforts on optimizing a single Convolutional Neural Network architecture, rather than superficially comparing multiple models, led to gains in fault classification performance. Our best diagnosis model achieved an average F1-score of 89.85 % across all fault classes in the Tennessee Eastman Process benchmark. This result represented a significant improvement over the baseline model and outperformed several approaches from the literature.

In the second phase, we introduced a new approach to fault detection by reformulating it as a similarity problem and building upon the capabilities of the diagnosis model. The resulting Siamese Neural Network achieved a 91.41 % F1-score for fault detection. Analysis of individual fault detection rates revealed that the lowest performances happened in the same class cases of worst performances in the diagnosis model. Nevertheless, the model detection capabilities rival performances reported in current literature.

These results demonstrate several important advances in the field of FDD. The diagnosis model's performance establishes that focused architectural optimization can surpass the conventional approach of evaluating multiple models superficially. The detection framework's results show how reformulating the problem and using transfer learning can help address the feature space distribution challenge in fault detection. Together, these approaches provide complementary solutions to different aspects of the fault monitoring pipeline.

Several promising research directions can be extended from our work. The effectiveness of transfer learning between diagnostic and detection tasks suggests the potential for developing frameworks that jointly optimize both capabilities with other architectures. Since we only considered the Euclidean distance, there is an opportunity to explore how different distance metrics can

influence the embedding representation in the detection model. Future work could also develop a full FDD model by combining the detection and diagnosis models into an ensemble model. The literature would also highly benefit from investigating our proposed approach on other dataset benchmarks and real industrial data.

In conclusion, our findings emphasize the value of targeted architectural optimization and transfer learning in developing monitoring solutions for complex industrial systems. Our research makes a compelling case for rethinking conventional approaches to industrial process monitoring through innovative deep learning applications, opening new possibilities for addressing practical fault detection and diagnosis challenges.

Bibliography

[1] PIEBALGS, A.. **The tennessee eastman process: an open source benchmark**. https://keepfloyding.github.io/posts/Ten-East-Proc-Intro/. Accessed: 2024-10-07.

[2] SUN, S.; REN, J.. **Gasf–msnn: A new fault diagnosis model for spatiotemporal information extraction**. Industrial & Engineering Chemistry Research, 60(17):6235–6248, 2021.

[3] NETO, J. G.; FIGUEIREDO, K.; SOARES, J. B. P. ; BRANDÃO, A. L. T.. **Can focusing on one deep learning architecture improve fault diagnosis performance?** Journal of Chemical Information and Modeling, 65(3):1289–1304, Jan. 2025.

[4] GROUMPOS, P. P.. **Large scale systems and fuzzy cognitive maps: A critical overview of challenges and research opportunities**. Annual Reviews in Control, 38(1):93–102, 2014.

[5] SHU, Y.; MING, L.; CHENG, F.; ZHANG, Z. ; ZHAO, J.. **Abnormal situation management: Challenges and opportunities in the big data era**. Computers and Chemical Engineering, 91:104–113, 2016.

[6] DUNJÓ, J.; FTHENAKIS, V.; VÍLCHEZ, J. A. ; ARNALDOS, J.. **Hazard and operability (HAZOP) analysis. A literature review**. Journal of Hazardous Materials, 173(1-3):19–32, 2010.

[7] OZTEMEL, E.; GURSEV, S.. **Literature review of Industry 4.0 and related technologies**. Journal of Intelligent Manufacturing, 31(1):127–182, 2020.

[8] GE, Z.. **Review on data-driven modeling and monitoring for plant-wide industrial processes**. Chemometrics and Intelligent Laboratory Systems, 171(September):16–25, 2017.

[9] SUN, D.; MENG, Z.; GUAN, Y.; LIU, J.; CAO, W. ; FAN, F.. **Intelligent fault diagnosis scheme for rolling bearing based on domain adaptation in one dimensional feature matching**. Applied Soft Computing, 146:110669, Oct. 2023.

[10] XIE, W.; HAN, T.; PEI, Z. ; XIE, M.. **A unified out-of-distribution detection framework for trustworthy prognostics and health management in renewable energy systems**. Engineering Applications of Artificial Intelligence, 125:106707, Oct. 2023.

[11] ZHANG, X.; WANG, C.; ZHOU, W.; XU, J. ; HAN, T.. **Trustworthy diagnostics with out-of-distribution detection: A novel max-consistency and min-similarity guided deep ensembles for uncertainty estimation**. IEEE Internet of Things Journal, 11(13):23055–23067, July 2024.

[12] ZHANG, K.; CHEN, J.; ZHANG, T. ; ZHOU, Z.. **A compact convolutional neural network augmented with multiscale feature extraction of acquired monitoring data for mechanical intelligent fault diagnosis**. J. Manuf. Syst., 55:273–284, Apr. 2020.

[13] SEON, J.; LEE, S.; SUN, Y. G.; KIM, S. H.; KIM, D. I. ; KIM, J. Y.. **Least information spectral gan with time-series data augmentation for industrial iot**. IEEE Transactions on Emerging Topics in Computational Intelligence, 9(1):757–769, Feb. 2025.

[14] YONG LIN, XIAOPING XIN, H. Z.; WANG, X.. **The implications of serial correlation and time-lag effects for the impact study of climate change on vegetation dynamics – a case study with hulunber meadow steppe, inner mongolia**. International Journal of Remote Sensing, 36(19-20):5031–5044, 2015.

[15] MAHESHWARI, S.; JAIN, R. C. ; JADON, R. S.. **An insight into rare class problem: Analysis and potential solutions**. J. Comput. Sci., 14(6):777–792, June 2018.

[16] LEEVY, J. L.; KHOSHGOFTAAR, T. M.; BAUDER, R. A. ; SELIYA, N.. **A survey on addressing high-class imbalance in big data**. Journal of Big Data, 5(1), Nov. 2018.

[17] CHATFIELD, C.. **The analysis of time series**. Texts in statistical science. Chapman and Hall/CRC, UK, 6 edition, July 2003.

[18] SHEN, C.. **Analysis of detrended time-lagged cross-correlation between two nonstationary time series**. Physics Letters A, 379(7):680–687, 2015.

[19] LI, J.; CAO, X.; CHEN, R.; ZHANG, X.; HUANG, X. ; QU, Y.. **Graph neural network architecture search for rotating machinery fault**

diagnosis based on reinforcement learning. Mechanical Systems and Signal Processing, 202:110701, 2023.

[20] UDOVICHENKO, I.; SHVETSOV, E.; DIVITSKY, D.; OSIN, D.; TROFIMOV, I.; SUKHAREV, I.; GLUSHENKO, A.; BERESTNEV, D. ; BURNAEV, E.. **Seqnas: Neural architecture search for event sequence classification**. IEEE Access, 12:3898 – 3909, 2024.

[21] WANG, H.; SHEN, Q.; DAI, Q.; GAO, Y.; GAO, J. ; ZHANG, T.. **Evolutionary variational yolov8 network for fault detection in wind turbines**. Computers, Materials & Continua, 80(1):625–642, 2024.

[22] ARUNTHAVANATHAN, R.; KHAN, F.; AHMED, S. ; IMTIAZ, S.. **An analysis of process fault diagnosis methods from safety perspectives**, feb 2021.

[23] AVILA OKADA, K. F.; SILVA DE MORAIS, A.; OLIVEIRA-LOPES, L. C. ; RIBEIRO, L.. **A survey on fault detection and diagnosis methods**. In: 2021 14TH IEEE INTERNATIONAL CONFERENCE ON INDUSTRY APPLICATIONS (INDUSCON). IEEE, Aug. 2021.

[24] MING, L.; ZHAO, J.. **Review on chemical process fault detection and diagnosis**. In: 2017 6TH INTERNATIONAL SYMPOSIUM ON ADVANCED CONTROL OF INDUSTRIAL PROCESSES (AdCONIP). IEEE, May 2017.

[25] KIDAM, K.; HURME, M.. **Origin of equipment design and operation errors**. Journal of Loss Prevention in the Process Industries, 25(6):937–949, 2012.

[26] ALAUDDIN, M.; KHAN, F.; IMTIAZ, S. ; AHMED, S.. **A Bibliometric Review and Analysis of Data-Driven Fault Detection and Diagnosis Methods for Process Systems**. Industrial and Engineering Chemistry Research, 57(32):10719–10735, 2018.

[27] VENKATASUBRAMANIAN, V.; RENGASWAMY, R. ; KAVURI, S. N.. **A review of process fault detection and diagnosis**. Computers & Chemical Engineering, 27(3):313–326, 2003.

[28] DOWNS, J.; VOGEL, E.. **A plant-wide industrial process control problem**. Computers & Chemical Engineering, 17(3):245–255, 1993. Industrial challenge problems in process control.

[29] JI, C.; SUN, W.. **A review on data-driven process monitoring methods: Characterization and mining of industrial data**. Processes, 10(2), 2022.

[30] BAO, Y.; WANG, B.; GUO, P. ; WANG, J.. **Chemical process fault diagnosis based on a combined deep learning method**. Canadian Journal of Chemical Engineering, 100(1):54–66, 2022.

[31] MARTIN-VILLALBA, C.; URQUIA, A. ; SHAO, G.. **Implementations of the tennessee eastman process in modelica**. IFAC-PapersOnLine, 51(2):619–624, 2018.

[32] HOANG, D. T.; KANG, H. J.. **A survey on Deep Learning based bearing fault diagnosis**. Neurocomputing, 335:327–335, 2019.

[33] CHADHA, G. S.; SCHWUNG, A.. **Comparison of deep neural network architectures for fault detection in Tennessee Eastman process**. IEEE International Conference on Emerging Technologies and Factory Automation, ETFA, p. 1–8, 2017.

[34] JIA, F.; LEI, Y.; LIN, J.; ZHOU, X. ; LU, N.. **Deep neural networks: A promising tool for fault characteristic mining and intelligent diagnosis of rotating machinery with massive data**. Mechanical Systems and Signal Processing, 72-73:303–315, 2016.

[35] PARK, P.; MARCO, P. D.; SHIN, H. ; BANG, J.. **Fault detection and diagnosis using combined autoencoder and long short-term memory network**. Sensors (Basel), 19(21):4612, Oct. 2019.

[36] ZHANG, Q.; ZHANG, J.; ZOU, J. ; FAN, S.. **A novel fault diagnosis method based on stacked lstm**. IFAC-PapersOnLine, 53(2):790–795, 2020. 21st IFAC World Congress.

[37] MD NOR, N.; HUSSAIN, M. A. ; CHE HASSAN, C. R.. **Fault diagnosis and classification framework using multi-scale classification based on kernel fisher discriminant analysis for chemical process system**. Applied Soft Computing, 61:959–972, 2017.

[38] WANG, Z.; OATES, T.. **Imaging time-series to improve classification and imputation**, 2015.

[39] **Tennessee eastman process simulation dataset**. `https://www.kaggle.com/datasets/averkij/`

tennessee-eastman-process-simulation-dataset. Accessed: 2022-07-28.

[40] NIU, X.; YANG, X.. **A novel one-dimensional convolutional neural network architecture for chemical process fault diagnosis**. The Canadian Journal of Chemical Engineering, 100(2):302–316, 2022.

[41] LOMOV, I.; LYUBIMOV, M.; MAKAROV, I. ; ZHUKOV, L. E.. **Fault detection in tennessee eastman process with temporal deep learning models**. Journal of Industrial Information Integration, 23:100216, 2021.

[42] PROFILLIDIS, V.; BOTZORIS, G.. **Chapter 5 - statistical methods for transport demand modeling**. In: Profillidis, V.; Botzoris, G., editors, MODELING OF TRANSPORT DEMAND, p. 163–224. Elsevier, 2019.

[43] ALZUBAIDI, L.; ZHANG, J.; HUMAIDI, A. J.; AL-DUJAILI, A.; DUAN, Y.; AL-SHAMMA, O.; SANTAMARÍA, J.; FADHEL, M. A.; AL-AMIDIE, M. ; FARHAN, L.. **Review of deep learning: concepts, CNN architectures, challenges, applications, future directions**, volumen 8. Springer International Publishing, 2021.

[44] LORENTE, Ò.; RIERA, I. ; RANA, A.. **Image Classification with Classic and Deep Learning Techniques**, 2021.

[45] YU, D.; WANG, H.; CHEN, P. ; WEI, Z.. **Mixed pooling for convolutional neural networks**. In: Miao, D.; Pedrycz, W.; Ślęzak, D.; Peters, G.; Hu, Q. ; Wang, R., editors, ROUGH SETS AND KNOWLEDGE TECHNOLOGY, p. 364–375, Cham, 2014. Springer International Publishing.

[46] SIMONYAN, K.; ZISSERMAN, A.. **Very deep convolutional networks for large-scale image recognition**, Sept. 2014.

[47] HOCHREITER, S.; SCHMIDHUBER, J.. **Long short-term memory**. Neural Computation, 9(8):1735–1780, Nov. 1997.

[48] LU, X.; ZHANG, W. ; HUANG, J.. **Exploiting embedding manifold of autoencoders for hyperspectral anomaly detection**. IEEE Transactions on Geoscience and Remote Sensing, 58(3):1527–1537, Mar. 2020.

[49] VASWANI, A.; SHAZEER, N.; PARMAR, N.; USZKOREIT, J.; JONES, L.; GOMEZ, A. N.; KAISER, L. ; POLOSUKHIN, I.. **Attention is all you need**, 2017.

[50] E SOUZA, A. C. O.; DE SOUZA, JR, M. B. ; DA SILVA, F. V.. **Enhancing fault detection and diagnosis systems for a chemical process: a study on convolutional neural networks and transfer learning**. Evol. Syst., 15(2):611–633, apr 2024.

[51] TAO, Q.; XIN, B.; ZHANG, Y.; JIN, H.; LI, Q.; DAI, Z. ; DAI, Y.. **A novel triage-based fault diagnosis method for chemical process**. Process Safety and Environmental Protection, 183:1102–1116, 2024.

[52] REN, J.; TANG, L. ; ZOU, H.. **A comparative study of different data representations under cnn and a novel integrated fdd architecture**. The Canadian Journal of Chemical Engineering, 101(8):4571–4586, 2023.

[53] WES MCKINNEY. **Data Structures for Statistical Computing in Python**. In: Stéfan van der Walt; Jarrod Millman, editors, PROCEEDINGS OF THE 9TH PYTHON IN SCIENCE CONFERENCE, p. 56 – 61, 2010.

[54] ABADI, M.; AGARWAL, A.; BARHAM, P.; BREVDO, E.; CHEN, Z.; CITRO, C.; CORRADO, G. S.; DAVIS, A.; DEAN, J.; DEVIN, M.; GHEMAWAT, S.; GOODFELLOW, I.; HARP, A.; IRVING, G.; ISARD, M.; JIA, Y.; JOZEFOWICZ, R.; KAISER, L.; KUDLUR, M.; LEVENBERG, J.; MANÉ, D.; MONGA, R.; MOORE, S.; MURRAY, D.; OLAH, C.; SCHUSTER, M.; SHLENS, J.; STEINER, B.; SUTSKEVER, I.; TALWAR, K.; TUCKER, P.; VANHOUCKE, V.; VASUDEVAN, V.; VIÉGAS, F.; VINYALS, O.; WARDEN, P.; WATTENBERG, M.; WICKE, M.; YU, Y. ; ZHENG, X.. **TensorFlow: Large-scale machine learning on heterogeneous systems**, 2015. Software available from tensorflow.org.

[55] CHOLLET, F.; OTHERS. **Keras**. `https://keras.io`, 2015.

[56] KINGMA, D.; BA, J.. **Adam: A method for stochastic optimization**. In: INTERNATIONAL CONFERENCE ON LEARNING REPRESENTATIONS (ICLR), San Diega, CA, USA, 2015.

[57] MURPHY, K. P.. **Probabilistic Machine Learning: An introduction**. MIT Press, 2022.

[58] WANG, C.; JIANG, J.; HU, X.; LIU, X. ; JI, X.. **Enhancing consistency and mitigating bias: A data replay approach for incremental learning**. Neural Netw., 184(107053):107053, Apr. 2025.

[59] PAHIJA, E.; HWANGBO, S.; SAULNIER-BELLEMARE, T. ; PATIENCE, G. S.. **Experimental methods in chemical engineering: Monte**

carlo. The Canadian Journal of Chemical Engineering, 102(10):3308–3321, June 2024.

[60] QUEIROZ, D. G.; RODRIGUES, F. D. B.; NOGUEIRA, J. D. N. P.; MELO, P. A. ; DE SOUZA, M. B.. **Synergizing phenomenological and ai-based models with industrial data to develop soft sensors for a sour water treatment unit**. Processes, 12(9):1900, Sept. 2024.

[61] DE AZEVEDO, J. P. A.; DE SOUZA, M. B. ; PINTO, J. C.. **Process hazard analysis based on modeling and simulation tools**. Processes, 10(2):386, Feb. 2022.

[62] DO NASCIMENTO PEREIRA NOGUEIRA, J.; MELO, P. A. ; DE SOUZA JR., M. B.. **Faulty scenarios in sour water treatment units: Simulation and ai-based diagnosis**. Process Safety and Environmental Protection, 165:716–727, Sept. 2022.

[63] ABEDINI, H.; ROSTAMI, M. R. ; SHAHSAVAR, S.. **Numerical simulation of atom transfer radical polymerization of styrene by moment and population balance models**. Polymer Bulletin, 79(6):4303–4322, May 2021.

[64] KLAFKE, N.; SOUZA JR., M. B. D. ; SECCHI, A. R.. **A new benchmark for plantwide process control**. Brazilian Journal of Chemical Engineering, 33(4):985–1002, Dec. 2016.

[65] KHAN, M.; HUSSAIN, M.; MANSOURPOUR, Z.; MOSTOUFI, N.; GHASEM, N. ; ABDULLAH, E.. **Cfd simulation of fluidized bed reactors for polyolefin production – a review**. Journal of Industrial and Engineering Chemistry, 20(6):3919–3946, Nov. 2014.

[66] XIANG, J.; ZHONG, Y.. **A novel personalized diagnosis methodology using numerical simulation and an intelligent method to detect faults in a shaft**. Applied Sciences, 6(12):414, Dec. 2016.

[67] MELO, A.; CÂMARA, M. M. ; PINTO, J. C.. **Data-driven process monitoring and fault diagnosis: A comprehensive survey**. Processes, 12(2):251, Jan. 2024.

[68] NAYAK, R.; PATI, U. C. ; DAS, S. K.. **A comprehensive review on deep learning-based methods for video anomaly detection**. Image and Vision Computing, 106:104078, Feb. 2021.

[69] MULLAPUDI, R. T.; POMS, F.; MARK, W. R.; RAMANAN, D. ; FATAHALIAN, K.. **Background splitting: Finding rare classes in a sea of background**, 2020.

[70] BROMLEY, J.; BENTZ, J. W.; BOTTOU, L.; GUYON, I.; LECUN, Y.; MOORE, C.; SÄCKINGER, E. ; SHAH, R.. **Signature verification using a "siamese" time delay neural network**. Intern. J. Pattern Recognit. Artif. Intell., 07(04):669–688, Aug. 1993.

[71] VALERO-MAS, J. J.; GALLEGO, A. J. ; RICO-JUAN, J. R.. **An overview of ensemble and feature learning in few-shot image classification using siamese networks**. Multimed. Tools Appl., 83(7):19929–19952, July 2023.

[72] HADSELL, R.; CHOPRA, S. ; LECUN, Y.. **Dimensionality reduction by learning an invariant mapping**. In: 2006 IEEE COMPUTER SOCIETY CONFERENCE ON COMPUTER VISION AND PATTERN RECOGNITION - VOLUME 2 (CVPR'06), volumen 2, p. 1735–1742. IEEE, 2006.

[73] TAKIMOTO, H.; SEKI, J.; F. SITUJU, S. ; KANAGAWA, A.. **Anomaly detection using siamese network with attention mechanism for few-shot learning**. Applied Artificial Intelligence, 36(1), July 2022.

[74] AHRABIAN, K.; BABAALI, B.. **Usage of autoencoders and siamese networks for online handwritten signature verification**. Neural Comput. Appl., 31(12):9321–9334, Dec. 2019.

[75] JI, C.; MA, F.; WANG, J. ; SUN, W.. **Fault detection for industrial chemical production using siamese autoencoder**. IFAC-PapersOnLine, 58(14):823–828, 2024.

[76] HUANG, Z.; YIN, X.; LIU, Y. ; TANG, S.. **Bridge scour detection method based on siamese neural networks under bridge-vehicle-wave interaction**. Ocean Engineering, 290:116327, Dec. 2023.

[77] DE OLIVEIRA, L. M.; MENDES DA SILVA, F. E.; DA CUNHA PEREIRA, G. M.; OLIVEIRA, D. S. ; MARCELO ANTUNES, F. L.. **Siamese neural network architecture for fault detection in a voltage source inverter**. In: 2021 BRAZILIAN POWER ELECTRONICS CONFERENCE (COBEP), p. 1–4. IEEE, Nov. 2021.

[78] HUNTER, J. D.. **Matplotlib: A 2d graphics environment**. Computing in Science & Engineering, 9(3):90–95, 2007.

[79] PEDREGOSA, F.; VAROQUAUX, G.; GRAMFORT, A.; MICHEL, V.; THIRION, B.; GRISEL, O.; BLONDEL, M.; PRETTENHOFER, P.; WEISS, R.; DUBOURG, V.; VANDERPLAS, J.; PASSOS, A.; COURNAPEAU, D.; BRUCHER, M.; PERROT, M. ; DUCHESNAY, E.. **Scikit-learn: Machine learning in Python**. Journal of Machine Learning Research, 12:2825–2830, 2011.

[80] PAN, S. J.; YANG, Q.. **A survey on transfer learning**. IEEE Transactions on Knowledge and Data Engineering, 22(10):1345–1359, Oct. 2010.

[81] WEISS, K.; KHOSHGOFTAAR, T. M. ; WANG, D.. **A survey of transfer learning**. Journal of Big Data, 3(1), May 2016.

[82] VAN DER MAATEN, L.; HINTON, G.. **Visualizing data using t-sne**. Journal of Machine Learning Research, 9(86):2579–2605, 2008.

[83] GRANDINI, M.; BAGLI, E. ; VISANI, G.. **Metrics for multi-class classification: an overview**, 2020.

[84] LASISI, A.; GHAZALI, R. ; HERAWAN, T.. **Application of Real-Valued Negative Selection Algorithm to Improve Medical Diagnosis**, p. 231–243. Elsevier, 2016.

[85] NICULESCU-MIZIL, A.; CARUANA, R.. **Predicting good probabilities with supervised learning**. In: PROCEEDINGS OF THE 22ND INTERNATIONAL CONFERENCE ON MACHINE LEARNING - ICML '05, ICML '05, p. 625–632. ACM Press, 2005.

[86] GUO, C.; PLEISS, G.; SUN, Y. ; WEINBERGER, K. Q.. **On calibration of modern neural networks**. 2017.

[87] MA, F.; JI, C.; WANG, J.; SUN, W.; TANG, X. ; JIANG, Z.. **Mola: Enhancing industrial process monitoring using a multi-block orthogonal long short-term memory autoencoder**. Processes, 12(12):2824, Dec. 2024.

[88] HU, G.; TONG, C.; ZENG, J. ; LUO, L.. **Joint time-serial variation analysis for fault monitoring of chemical processes**. Process Safety and Environmental Protection, 196:106867, Apr. 2025.

[89] DONG, J.; LI, D.; CONG, Z. ; PENG, K.. **A new fault detection method based on an updatable hybrid model for hard-to-detect faults in nonstationary processes**. Reliability Engineering & System Safety, 259:110920, July 2025.

A
Supplementary results of diagnosis model investigation

A.1
Training Curves of Investigated Modifications

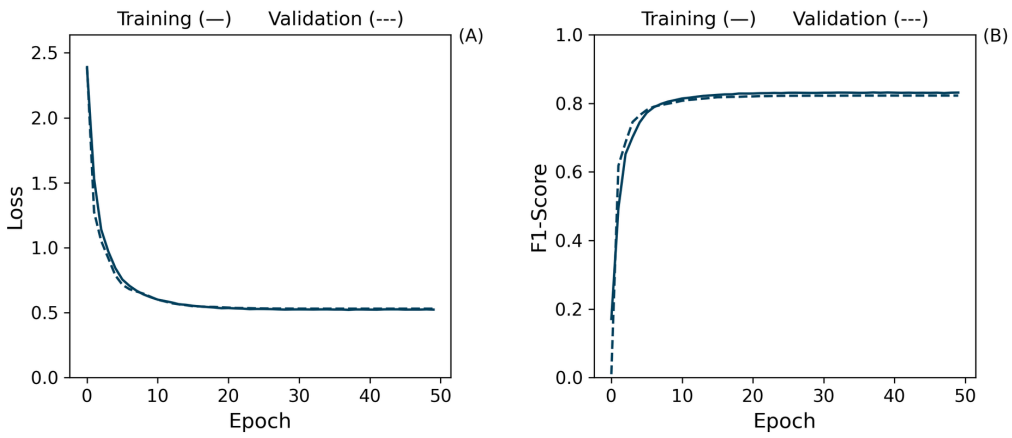

Figure A.1: Loss (A) and F1-Score (B) training curves for modification of type 1.

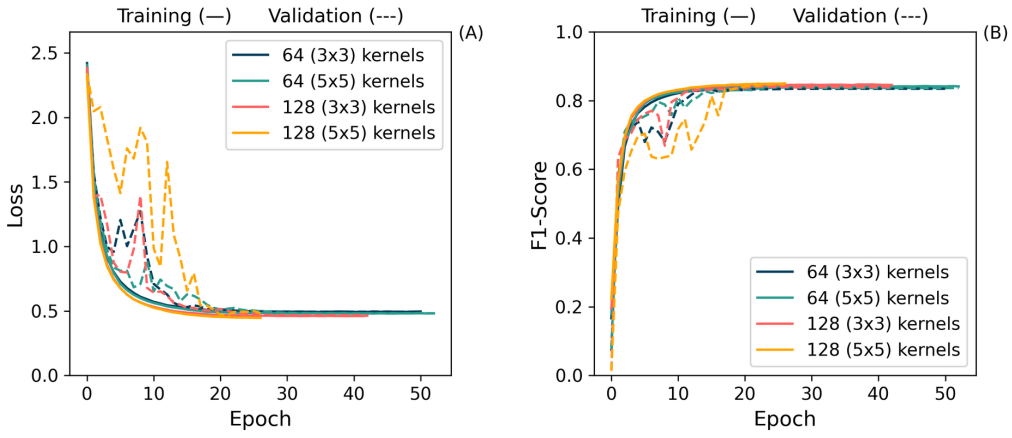

Figure A.2: Loss (A) and F1-Score (B) training curves for modifications of type 2.

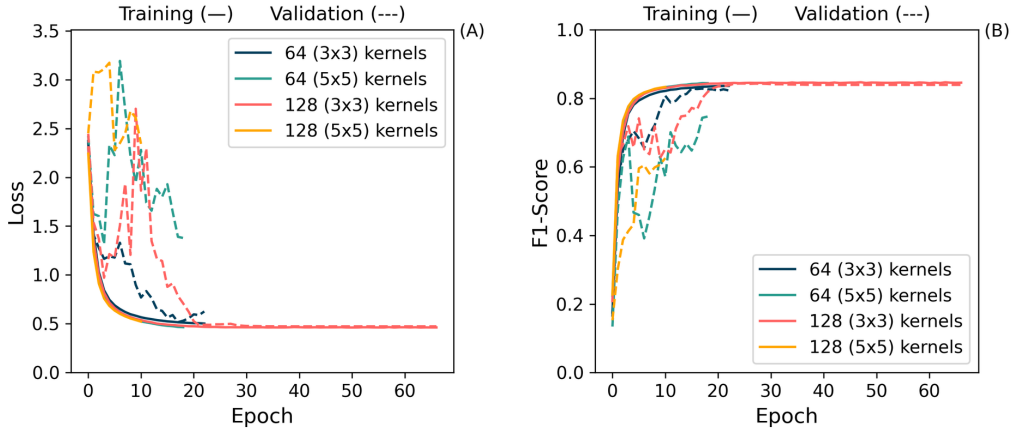

Figure A.3: Loss (A) and F1-Score (B) training curves for modifications of type 3.

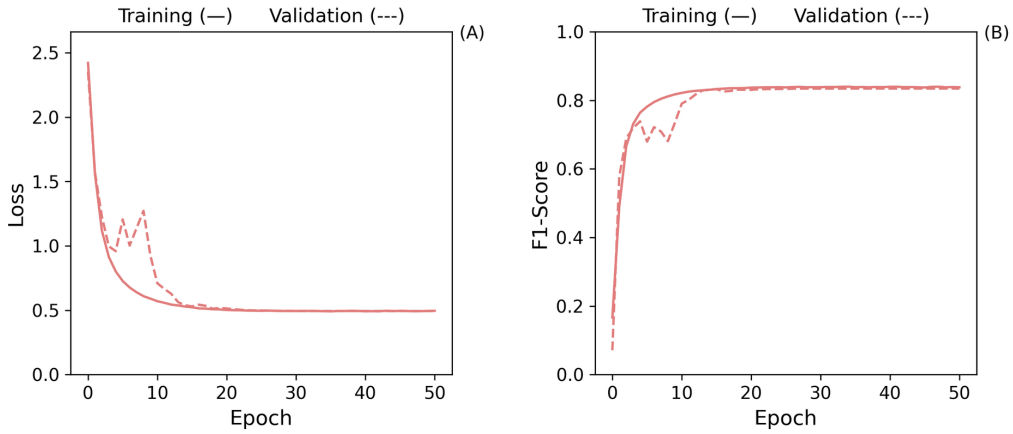

Figure A.4: Loss (A) and F1-Score (B) training curves for Model 1.

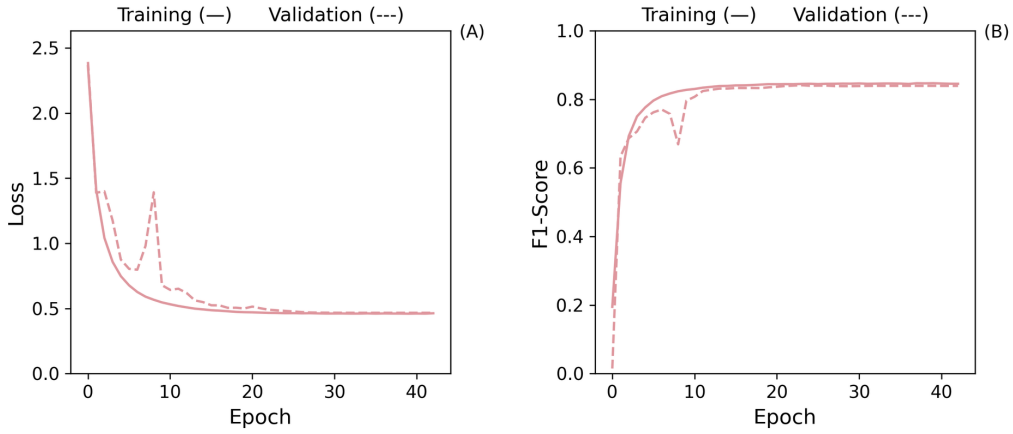

Figure A.5: Loss (A) and F1-Score (B) training curves for Model 2.

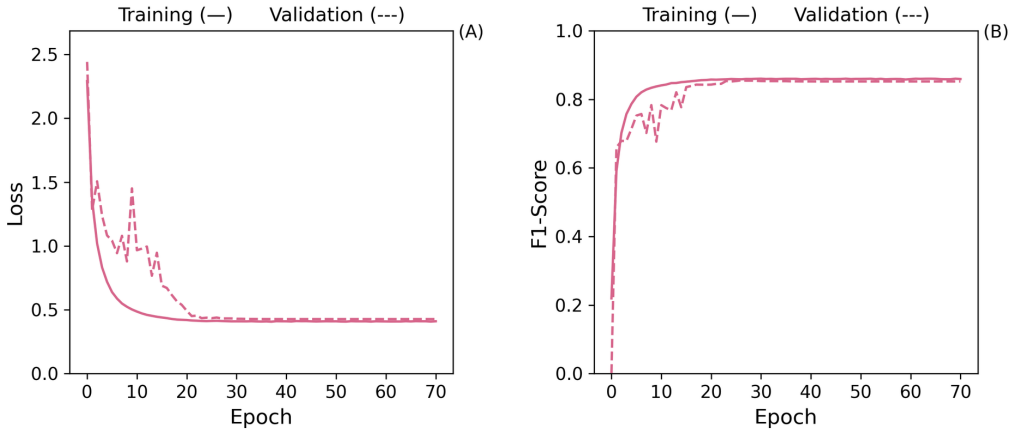

Figure A.6: Loss (A) and F1-Score (B) training curves for Model 3.

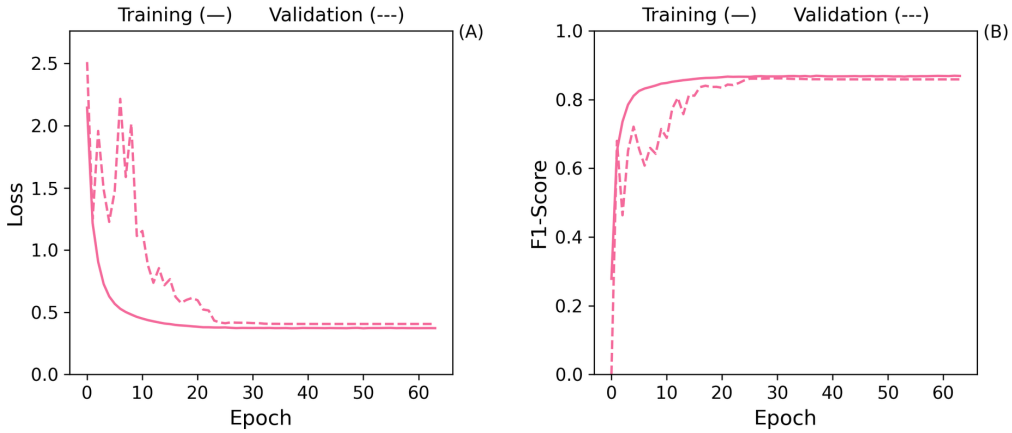

Figure A.7: Loss (A) and F1-Score (B) training curves for Model 4.

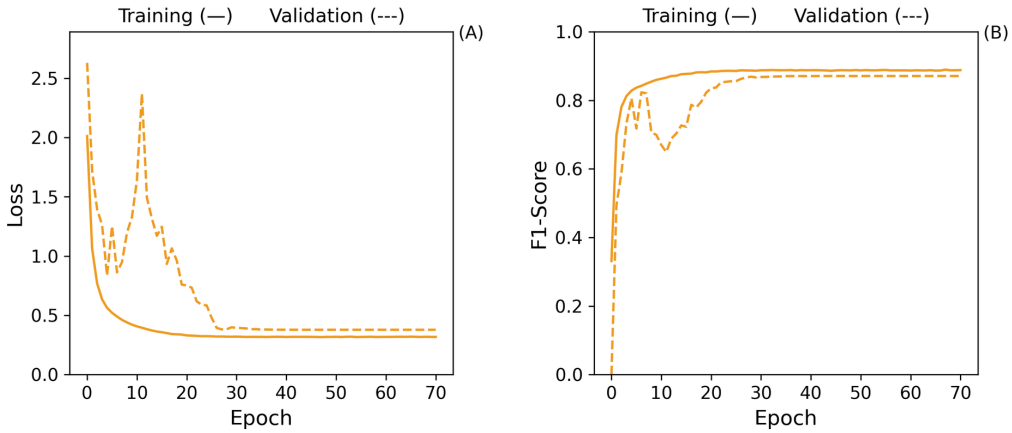

Figure A.8: Loss (A) and F1-Score (B) training curves for Model 5.

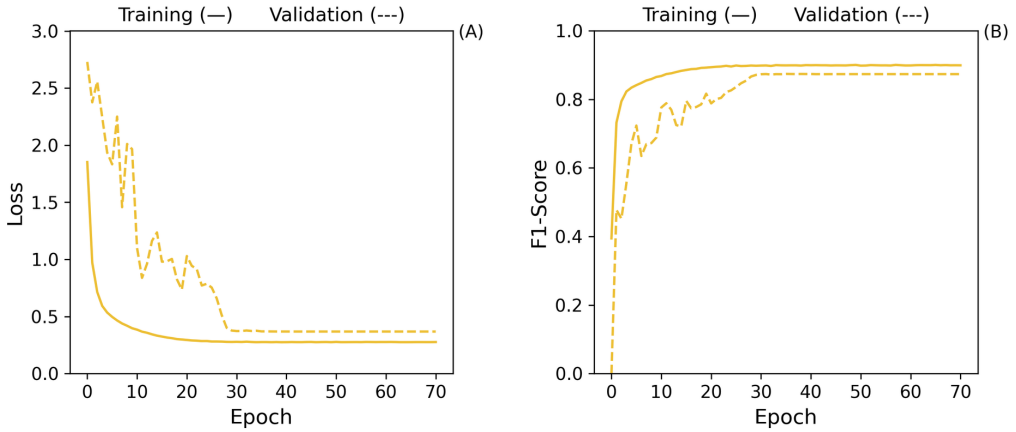

Figure A.9: Loss (A) and F1-Score (B) training curves for Model 6.

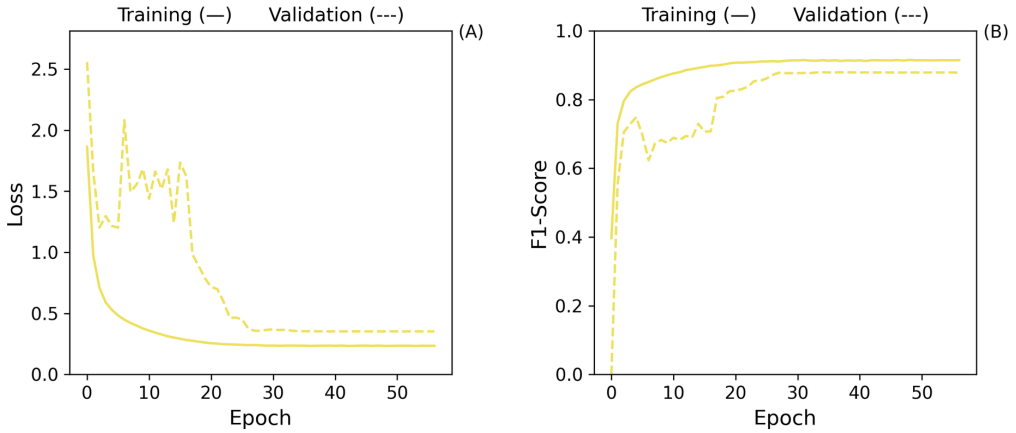

Figure A.10: Loss (A) and F1-Score (B) training curves for Model 7.

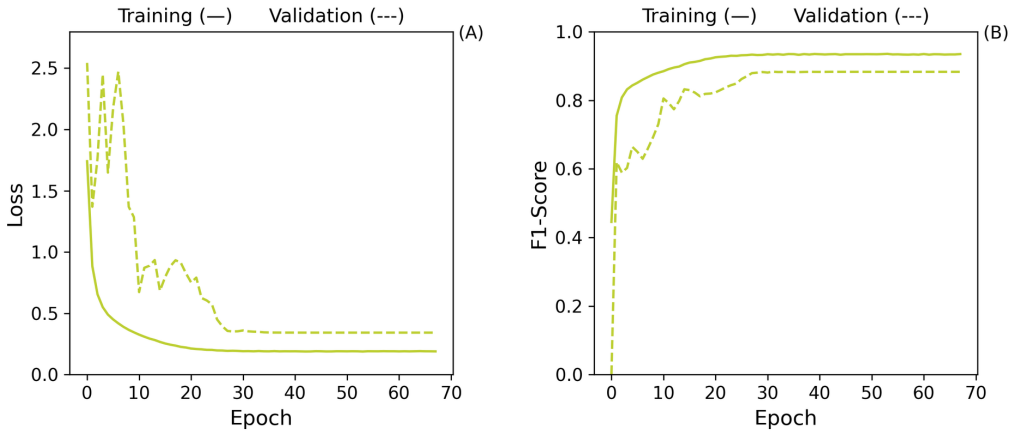

Figure A.11: Loss (A) and F1-Score (B) training curves for Model 8.

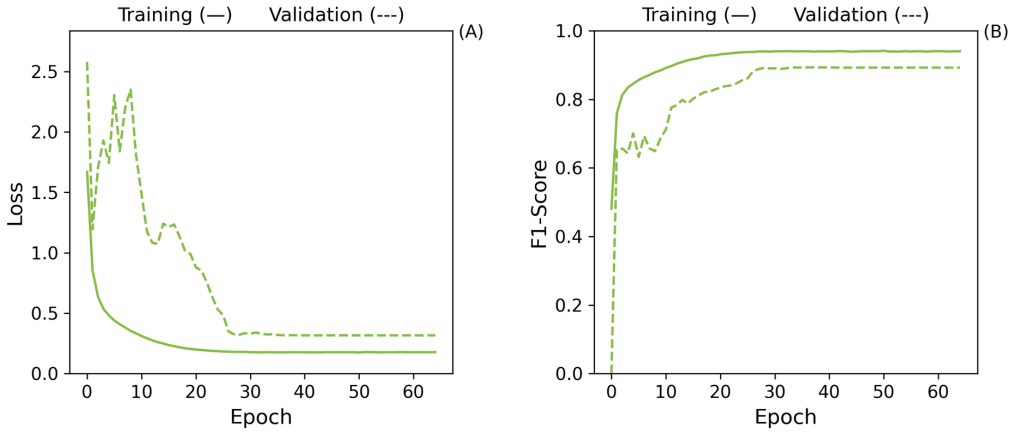

Figure A.12: Loss (A) and F1-Score (B) training curves for Model 9.

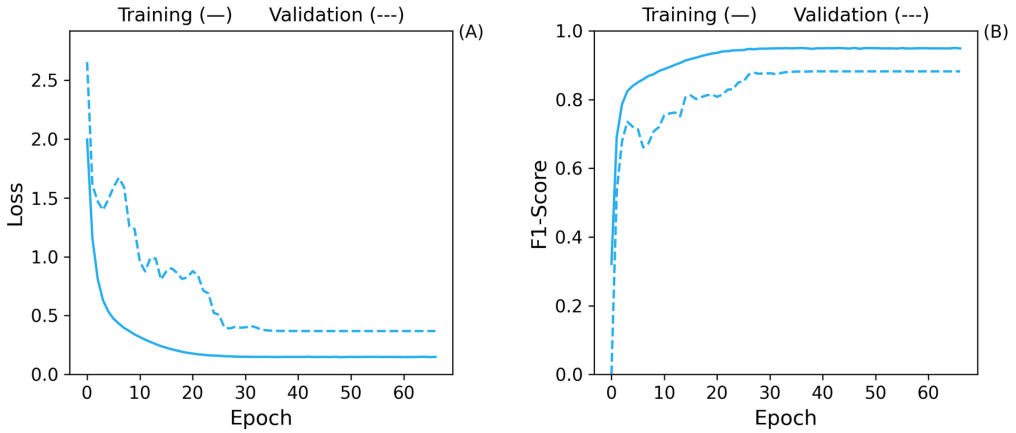

Figure A.13: Loss (A) and F1-Score (B) training curves for Model 10.

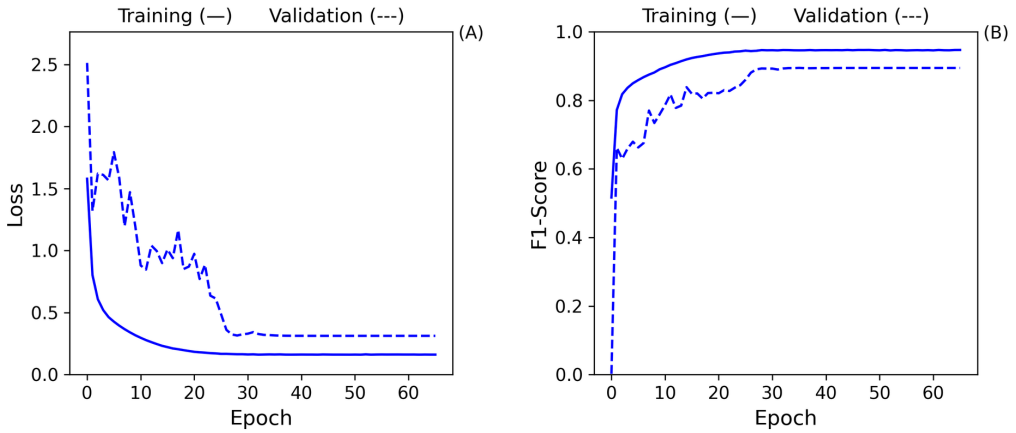

Figure A.14: Loss (A) and F1-Score (B) training curves for Model 11.

B
Supplementary results of detection model investigation

B.1
Stage 1 Supplementary Figures - Effects of training dataset size and baseline

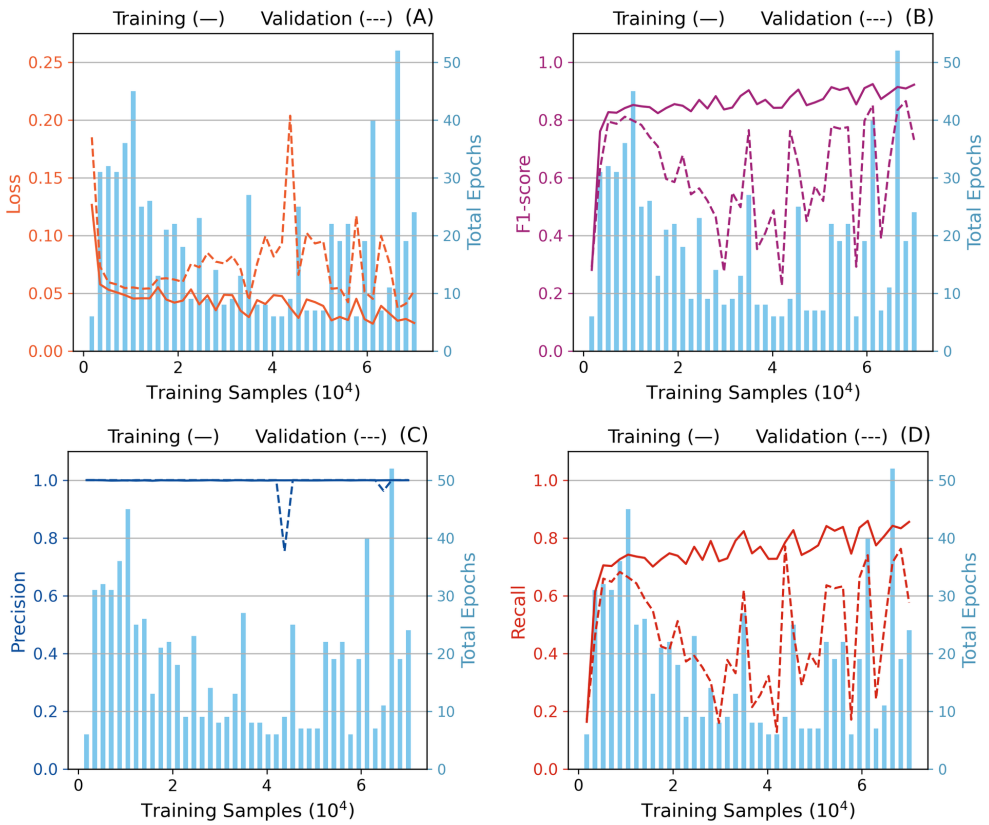

Figure B.1: Loss (A), F1-Score (B), Precision (C) and Recall (D) learning curves with early stopping.

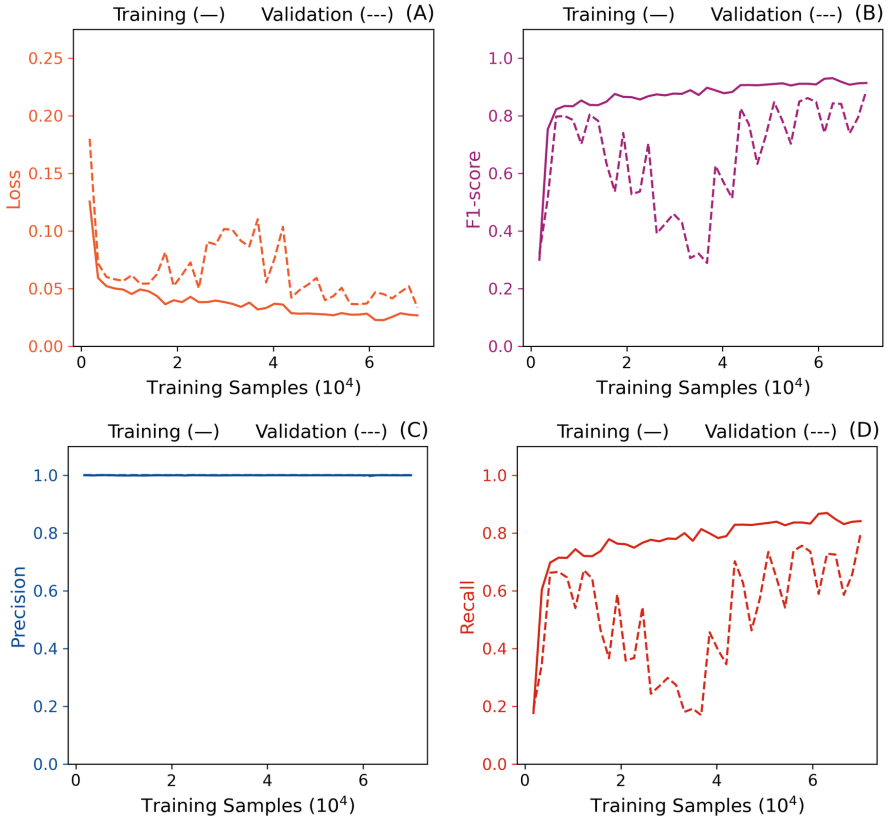

Figure B.2: Loss (A), F1-Score (B), Precision (C) and Recall (D) learning curves with 30 epochs.

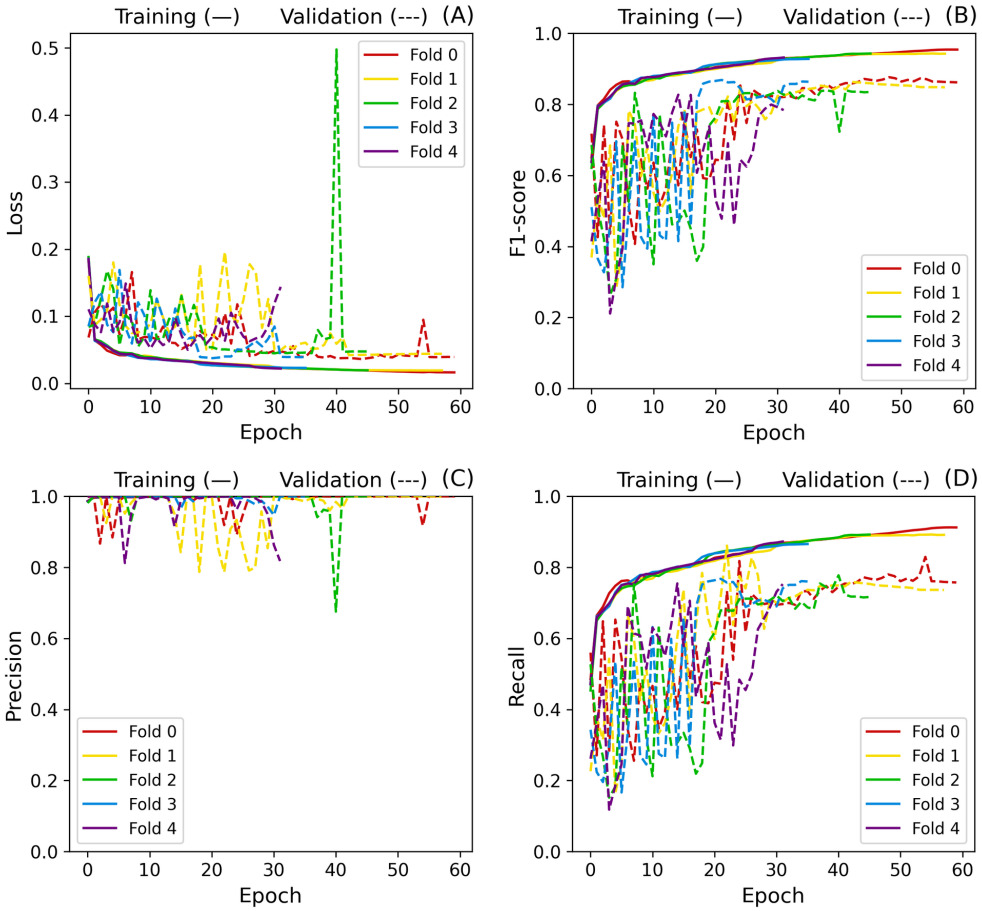

Figure B.3: Baseline cross-validation Loss (A), F1-Score (B), Precision (C) and Recall (D) curves.

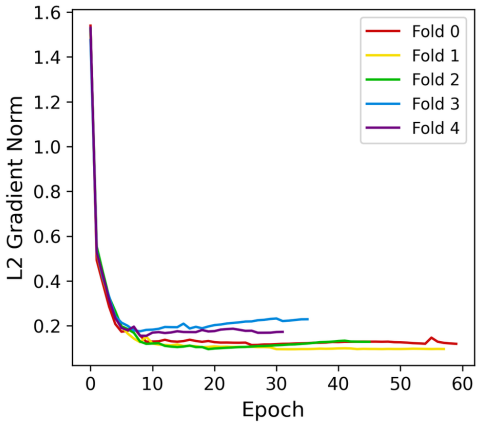

Figure B.4: Baseline L2 gradient norm.

B.2
Stage 2 Supplementary Figures - Model stability and overfitting mitigation

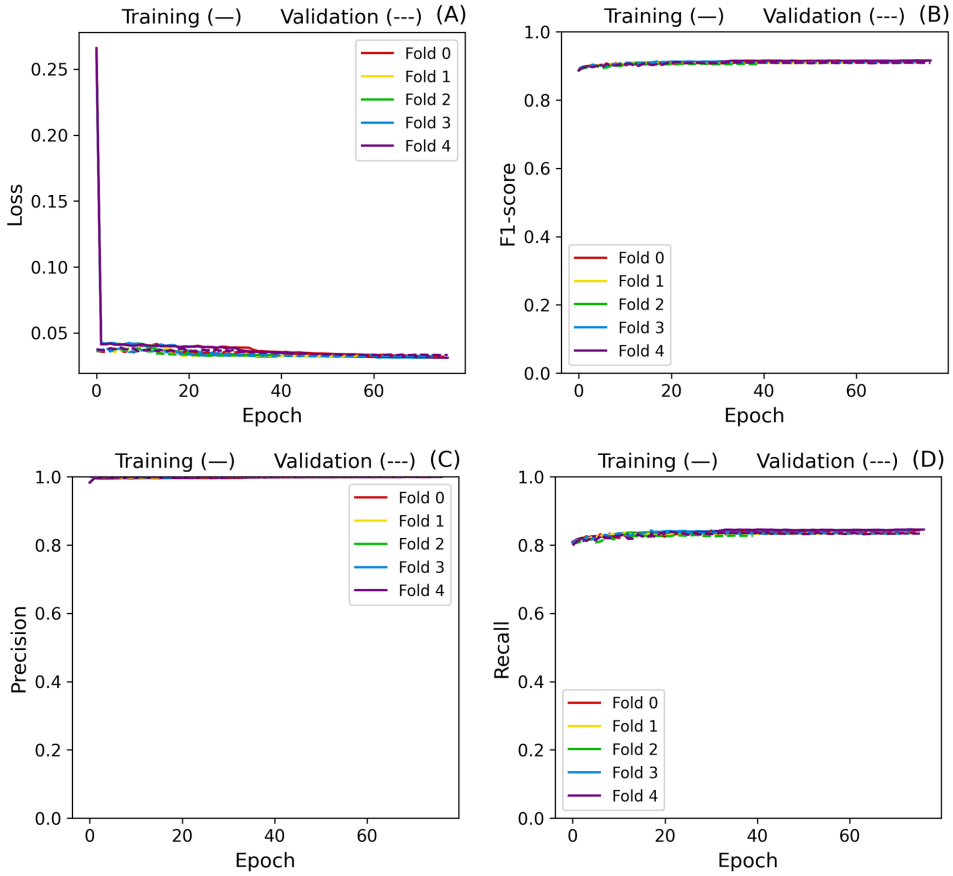

Figure B.5: Cross-validation Loss (A), F1-Score (B), Precision (C) and Recall (D) curves after freezing convolutional layers.

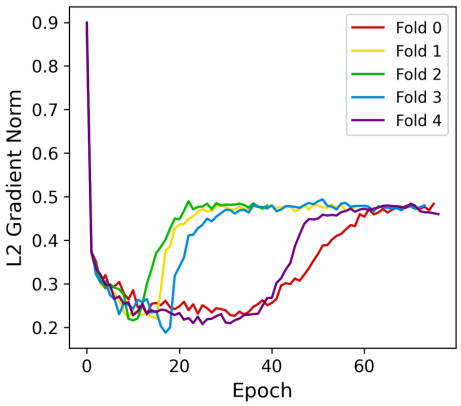

Figure B.6: L2 gradient norm after freezing convolutional layers.

B.3
Stage 3 Supplementary Figures - Model improvement investigation

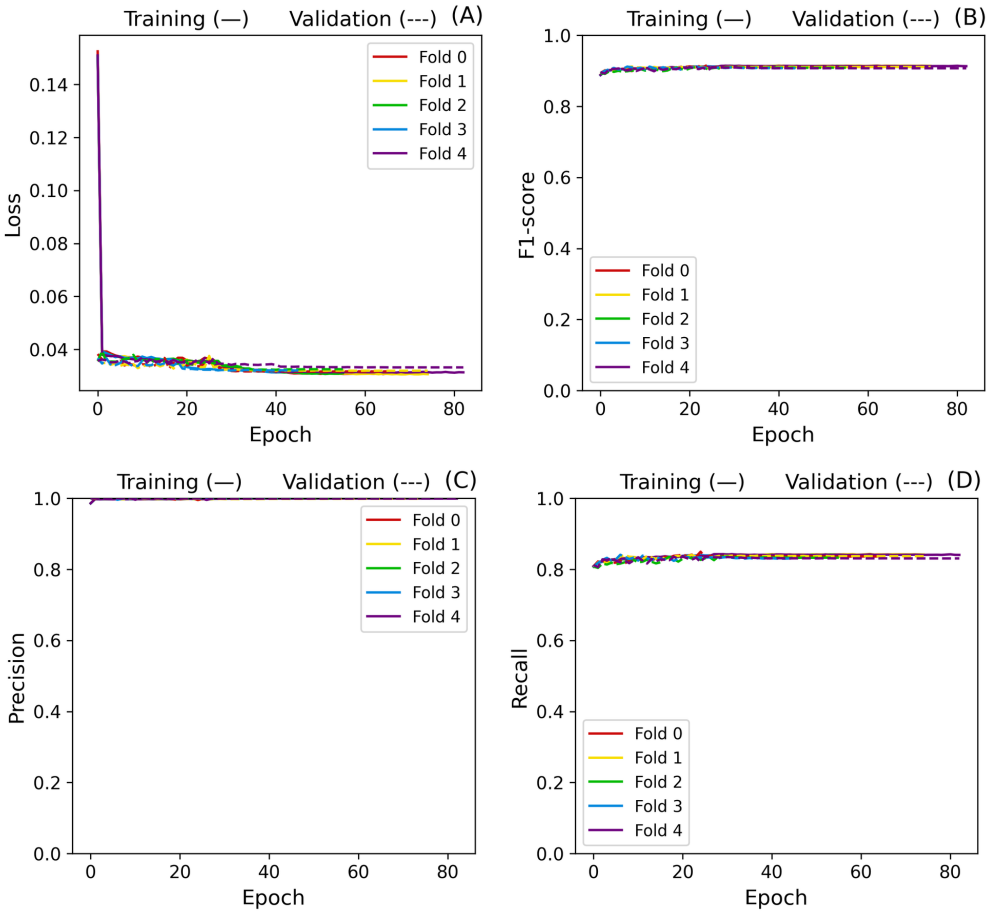

Figure B.7: Cross-validation Loss (A), F1-Score (B), Precision (C) and Recall (D) curves of model with reduced dropout rate (30 %).

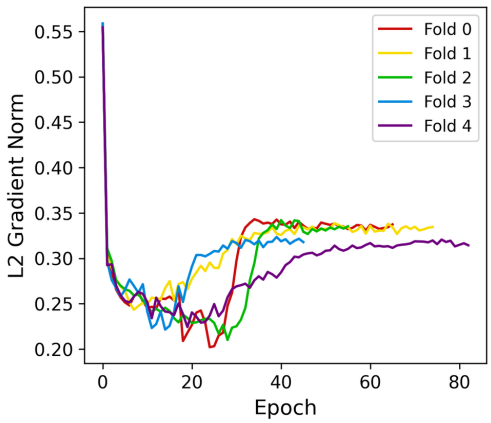

Figure B.8: L2 gradient norm of model with reduced dropout rate (30 %).

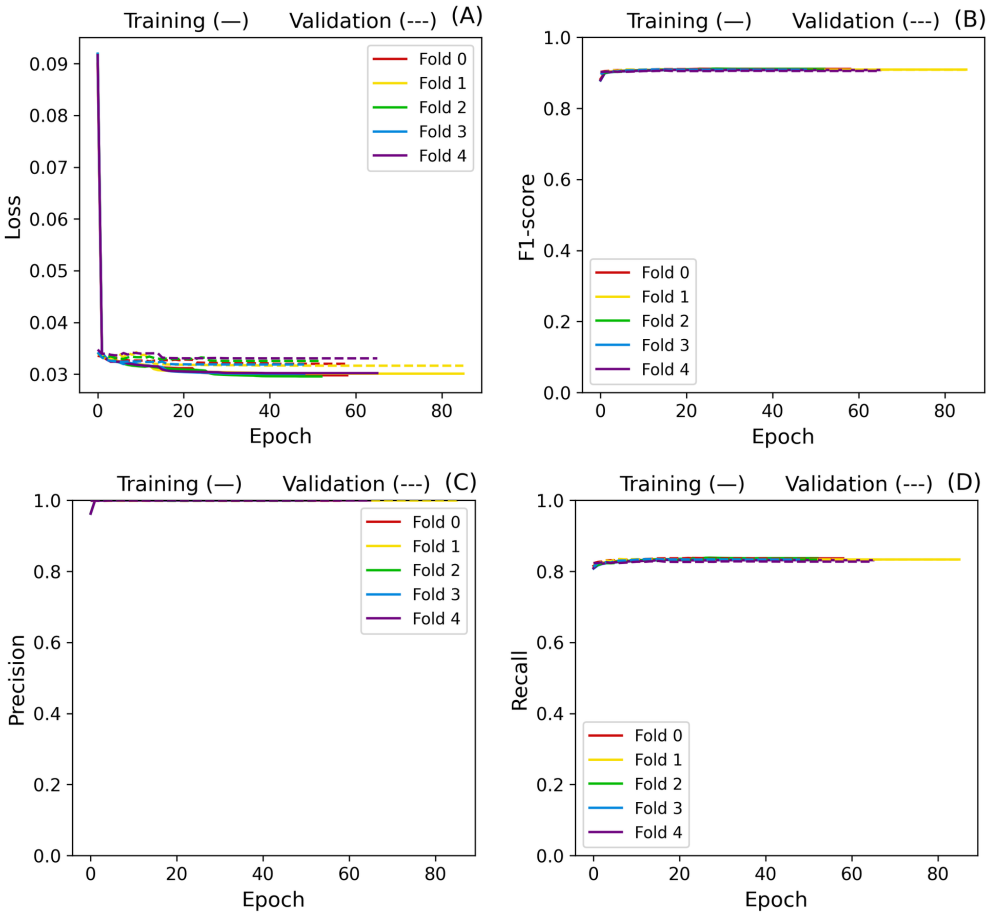

Figure B.9: Cross-validation Loss (A), F1-Score (B), Precision (C) and Recall (D) curves of model with no dropout layer.

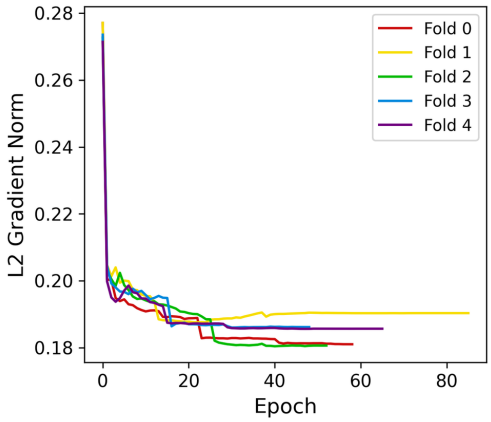

Figure B.10: L2 gradient norm of model with no dropout layer.

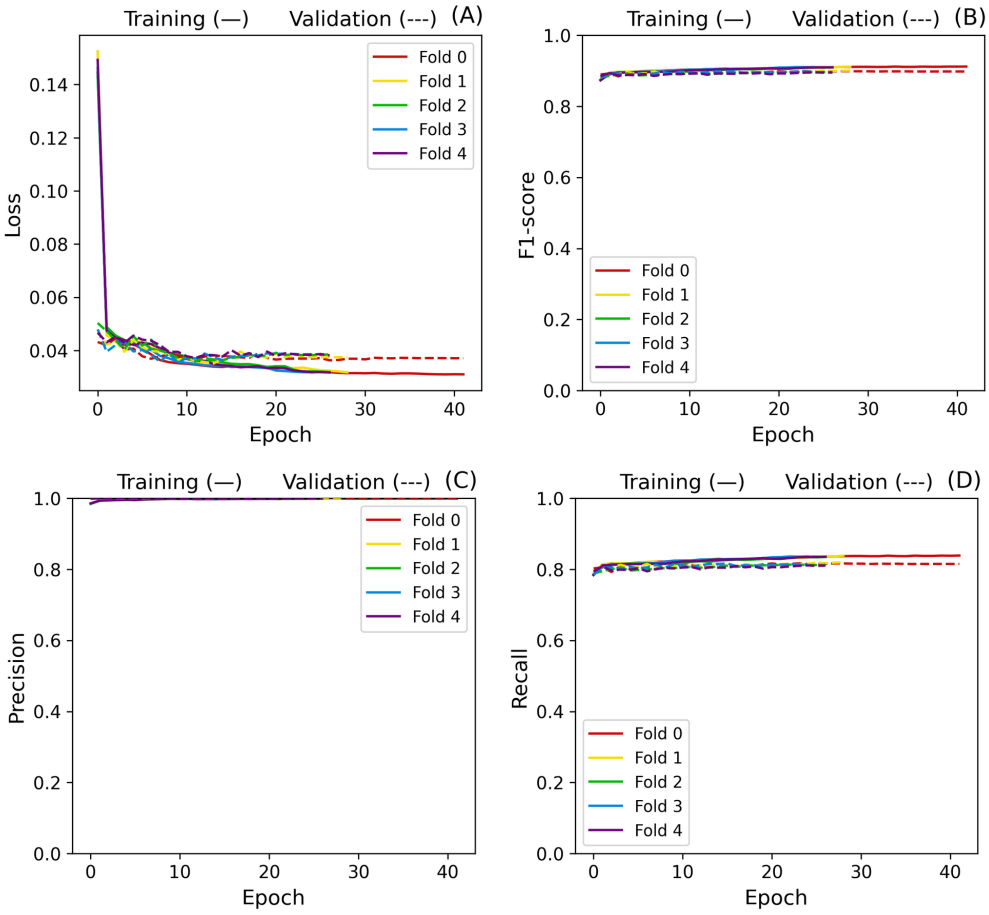

Figure B.11: Cross-validation Loss (A), F1-Score (B), Precision (C) and Recall (D) curves of model with additional dense layer with 512 neurons.

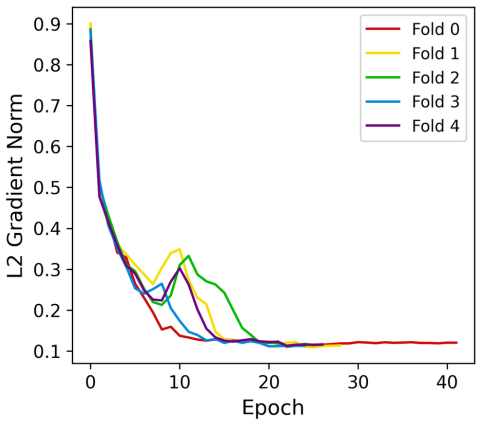

Figure B.12: L2 gradient norm of model with additional dense layer with 512 neurons.

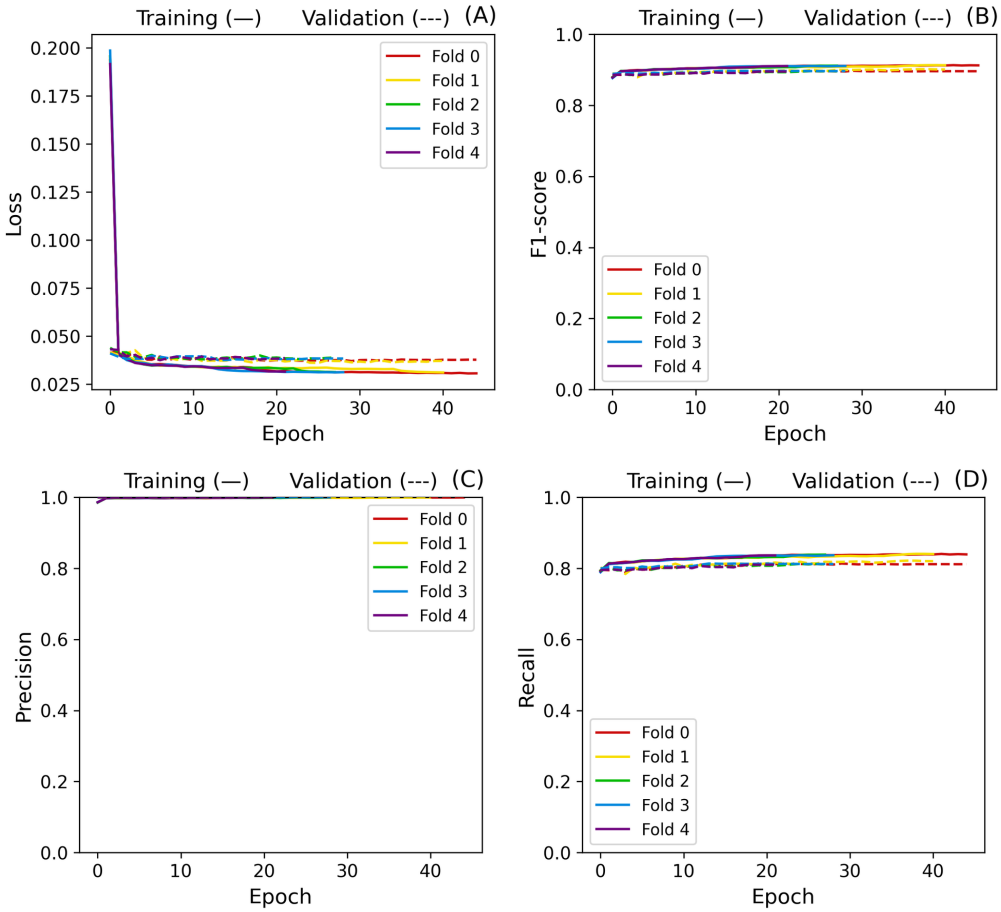

Figure B.13: Cross-validation Loss (A), F1-Score (B), Precision (C) and Recall (D) curves of model with additional dense layer with 1024 neurons.

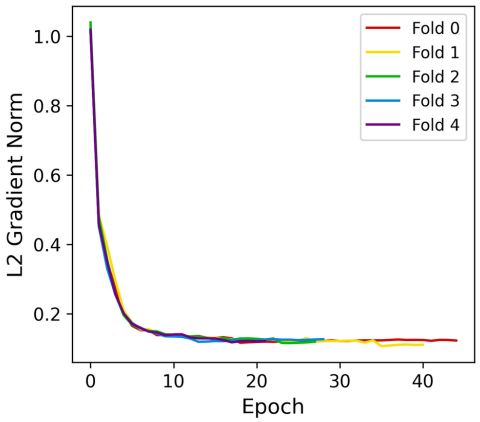

Figure B.14: L2 gradient norm of model with additional dense layer with 1024 neurons.

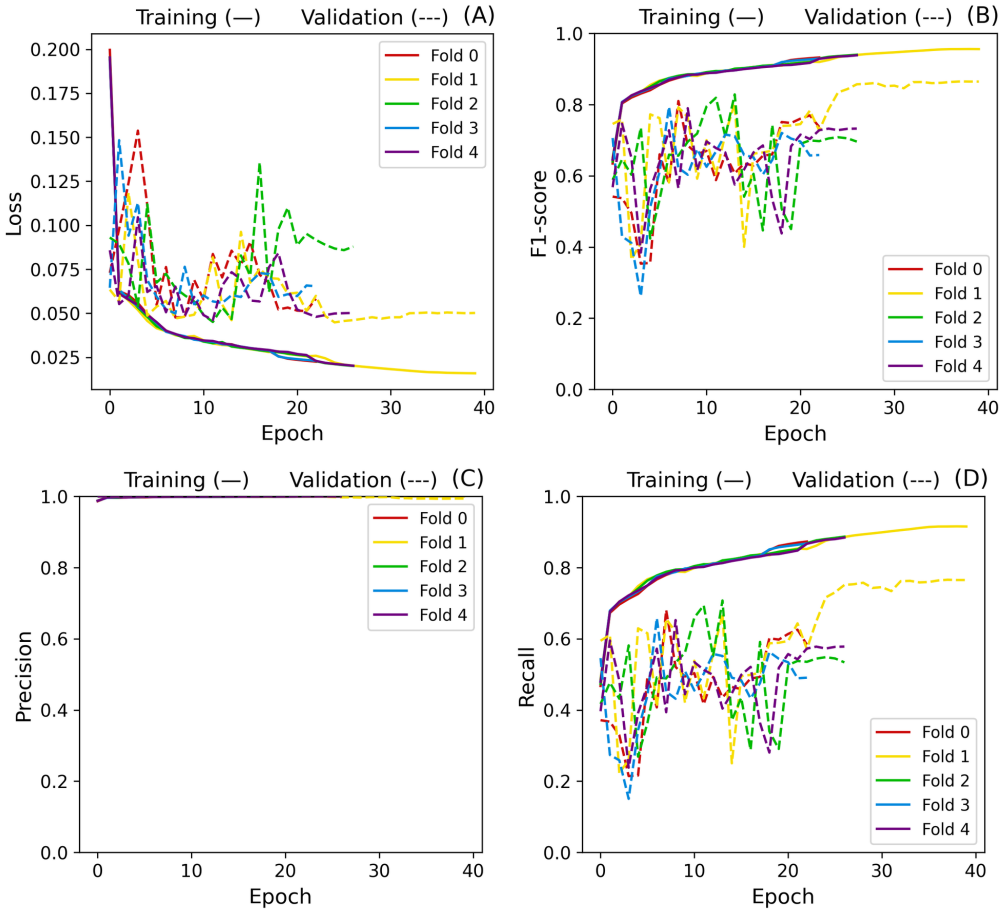

Figure B.15: Cross-validation Loss (A), F1-Score (B), Precision (C) and Recall (D) curves after freezing Group 1.

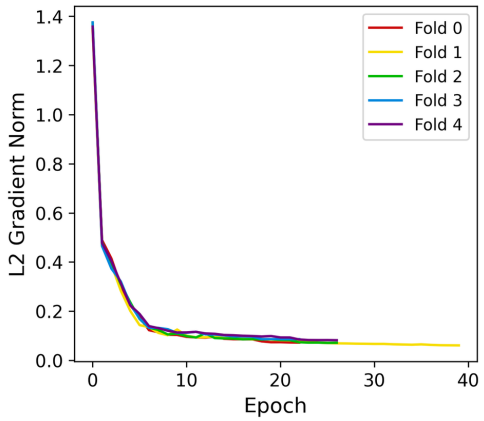

Figure B.16: L2 gradient norm after freezing Group 1.

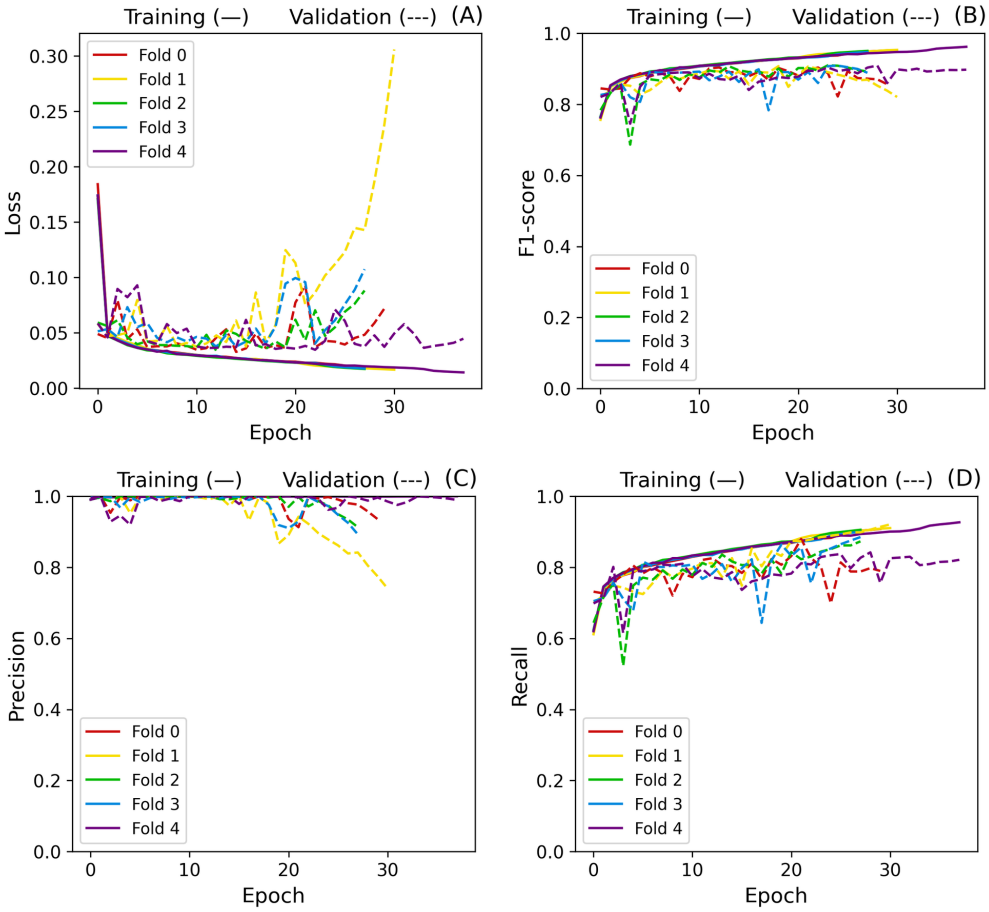

Figure B.17: Cross-validation Loss (A), F1-Score (B), Precision (C) and Recall (D) curves after freezing Group 2.

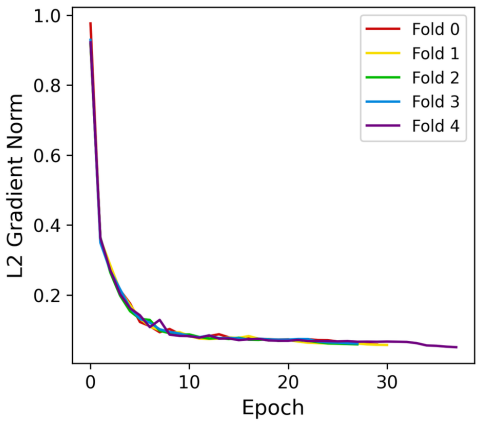

Figure B.18: L2 gradient norm after freezing Group 2.

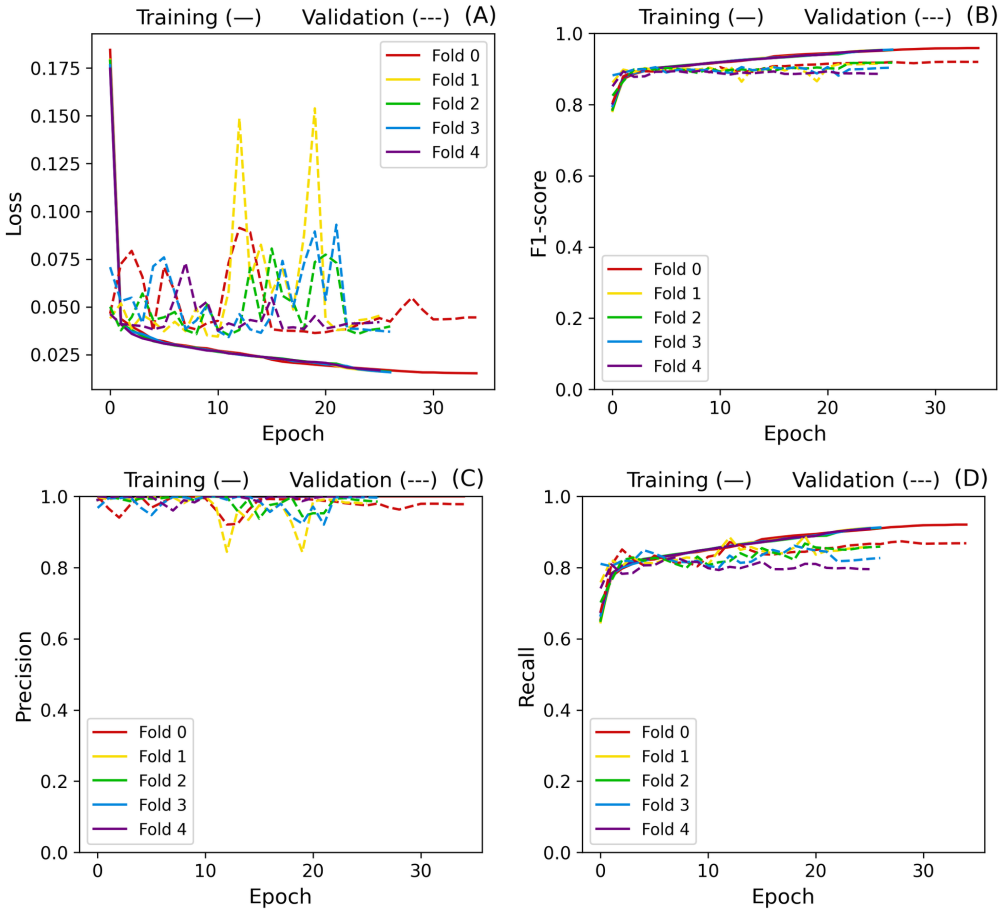

Figure B.19: Cross-validation Loss (A), F1-Score (B), Precision (C) and Recall (D) curves after freezing Group 3.

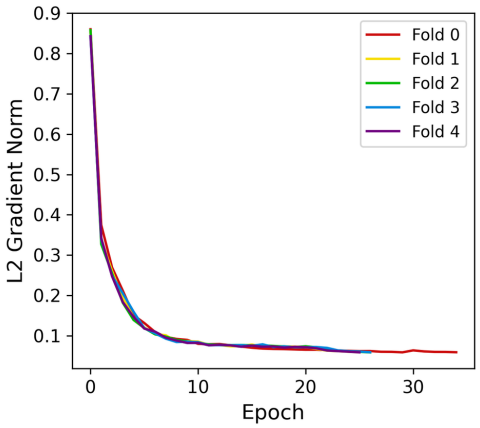

Figure B.20: L2 gradient norm after freezing Group 3.

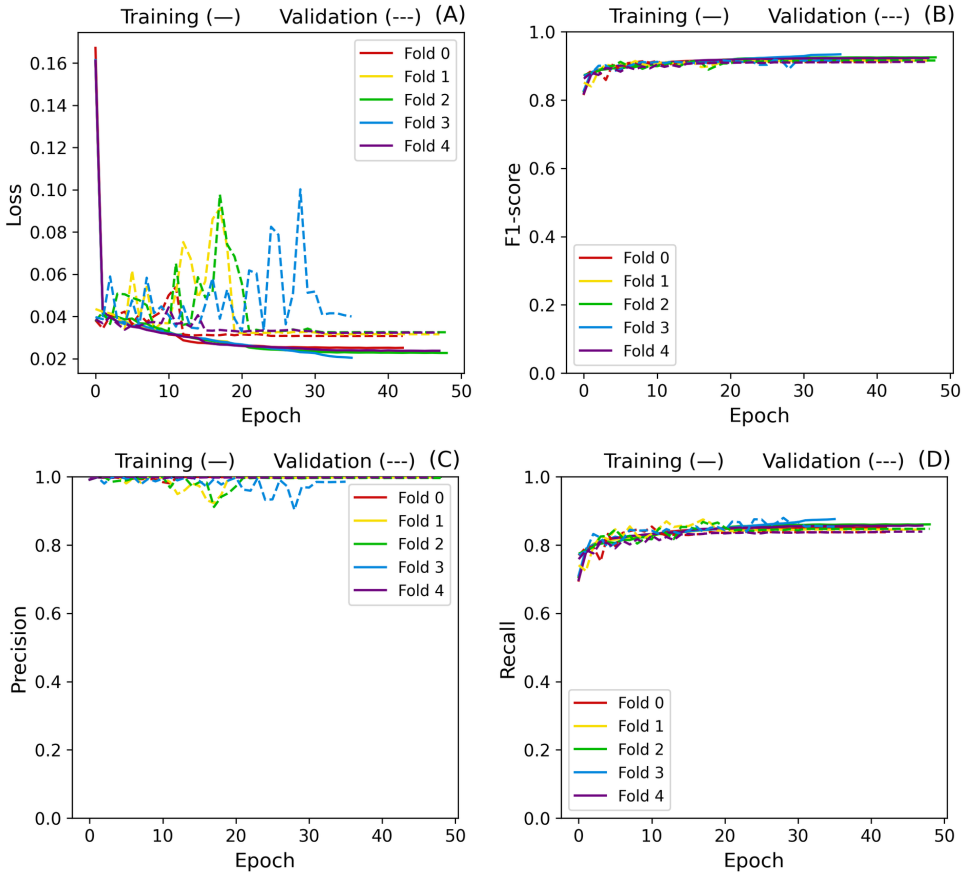

Figure B.21: Cross-validation Loss (A), F1-Score (B), Precision (C) and Recall (D) curves after freezing Group 4.

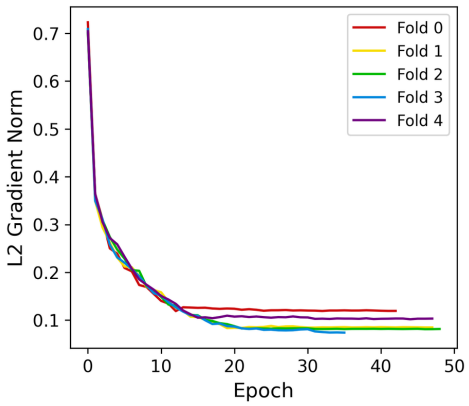

Figure B.22: L2 gradient norm after freezing Group 4.

B.4
Stage 4 Supplementary Figures - Best configuration results

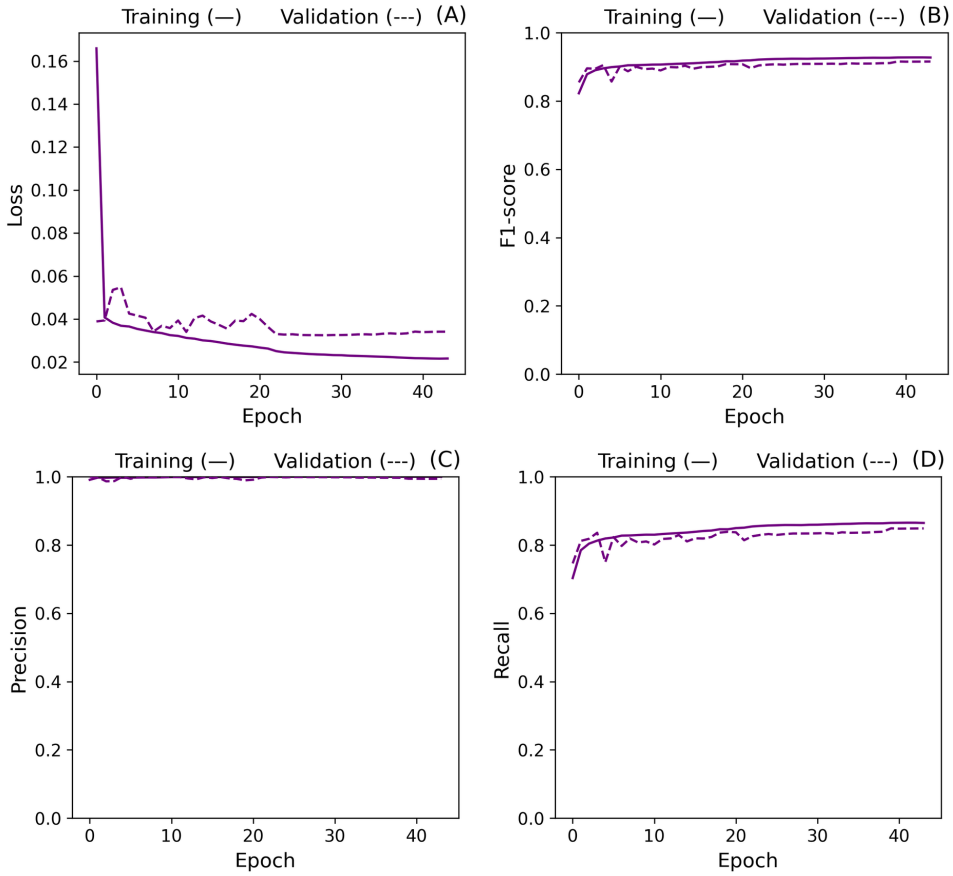

Figure B.23: Best model's holdout (A), F1-Score (B), Precision (C) and Recall (D) curves.

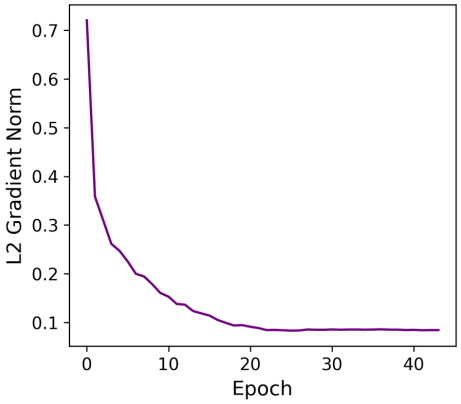

Figure B.24: Best model's holdout L2 gradient norm.

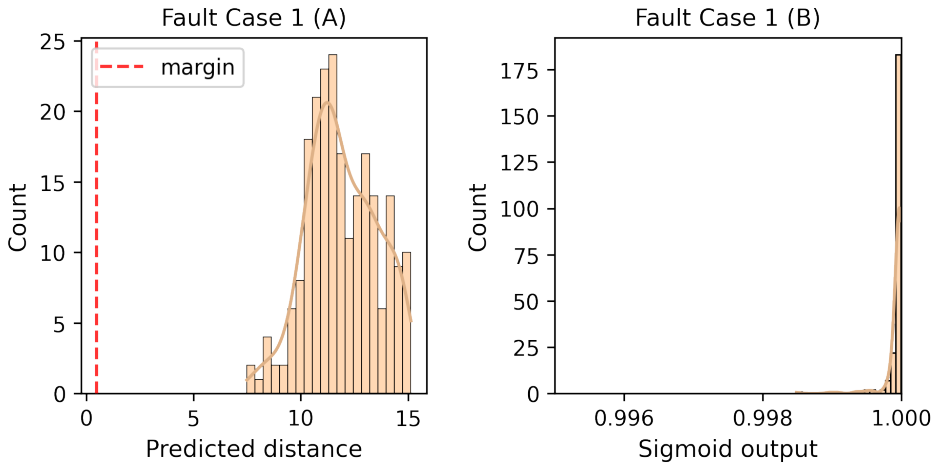

Figure B.25: Test predicted distance (A) and sigmoid output (B) distributions of fault case 1.

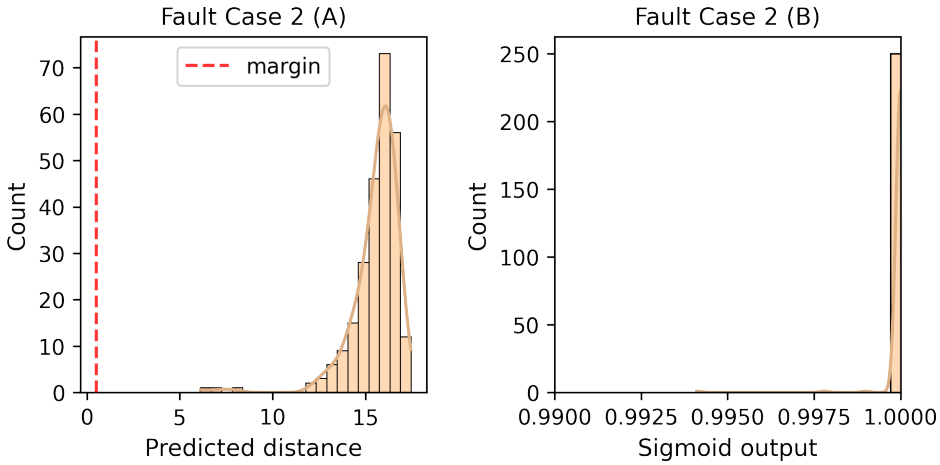

Figure B.26: Test predicted distance (A) and sigmoid output (B) distributions of fault case 2.

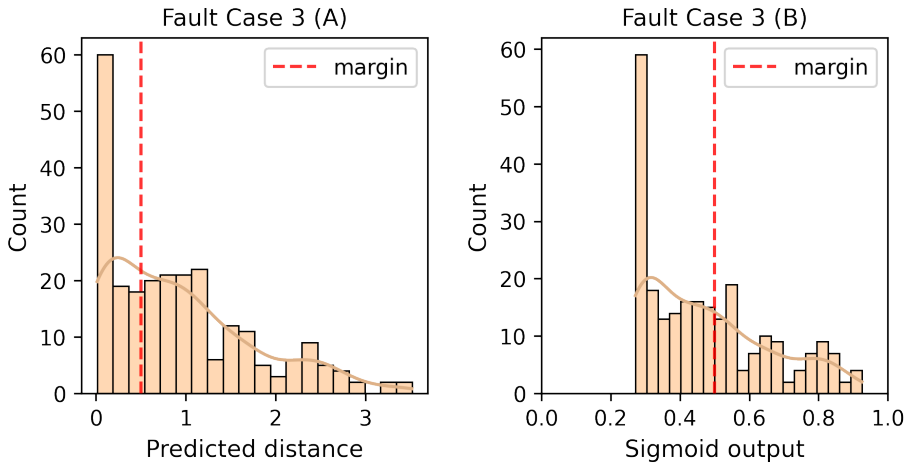

Figure B.27: Test predicted distance (A) and sigmoid output (B) distributions of fault case 3.

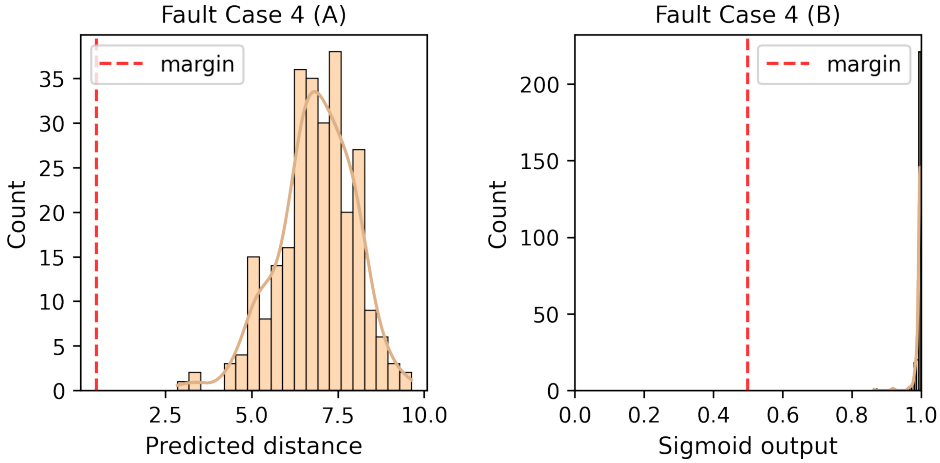

Figure B.28: Test predicted distance (A) and sigmoid output (B) distributions of fault case 4.

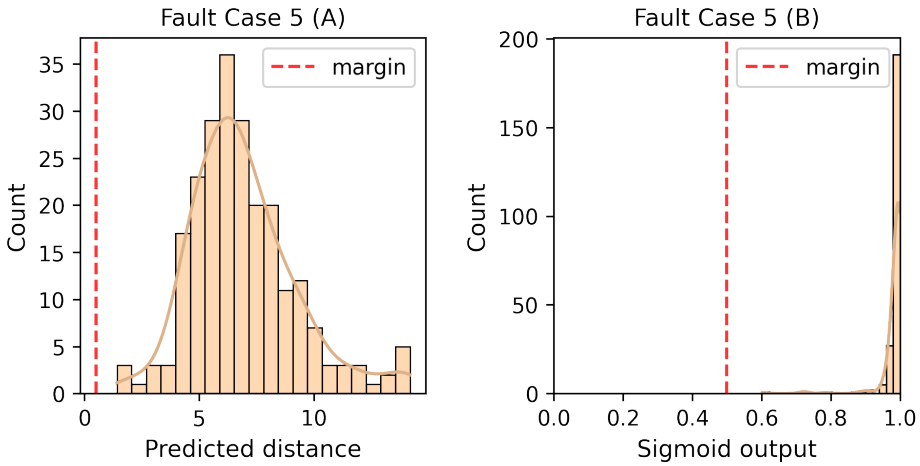

Figure B.29: Test predicted distance (A) and sigmoid output (B) distributions of fault case 5.

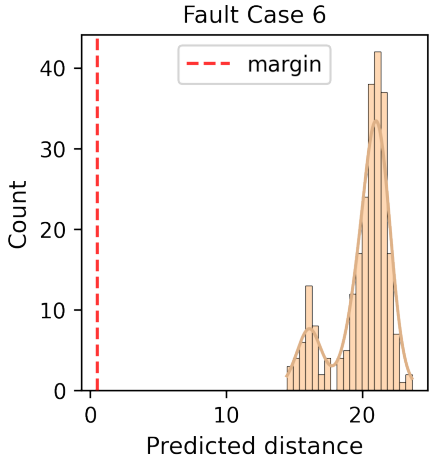

Figure B.30: Test predicted distance distributions of fault case 6. The sigmoid transformation could not be represented graphically as a distribution because all values tend to 1.0.

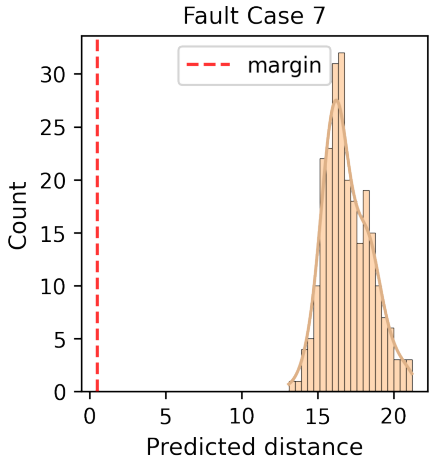

Figure B.31: Test predicted distance distributions of fault case 7. The sigmoid transformation could not be represented graphically as a distribution because all values tend to 1.0.

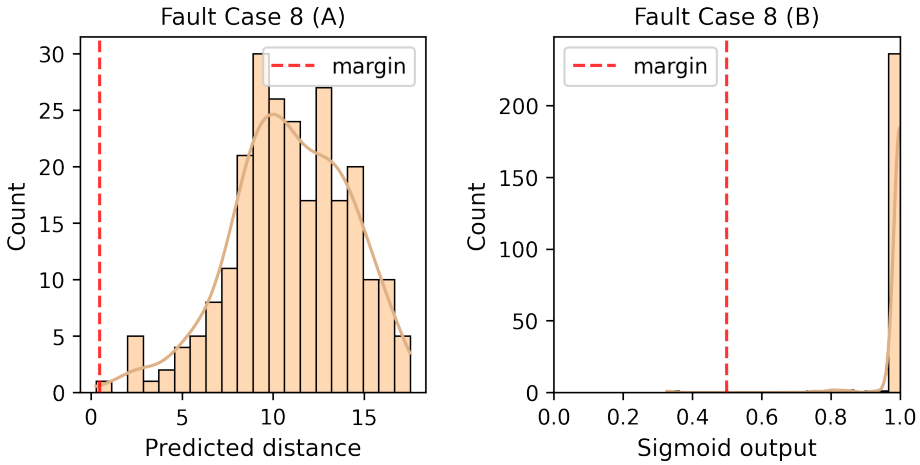

Figure B.32: Test predicted distance (A) and sigmoid output (B) distributions of fault case 8.

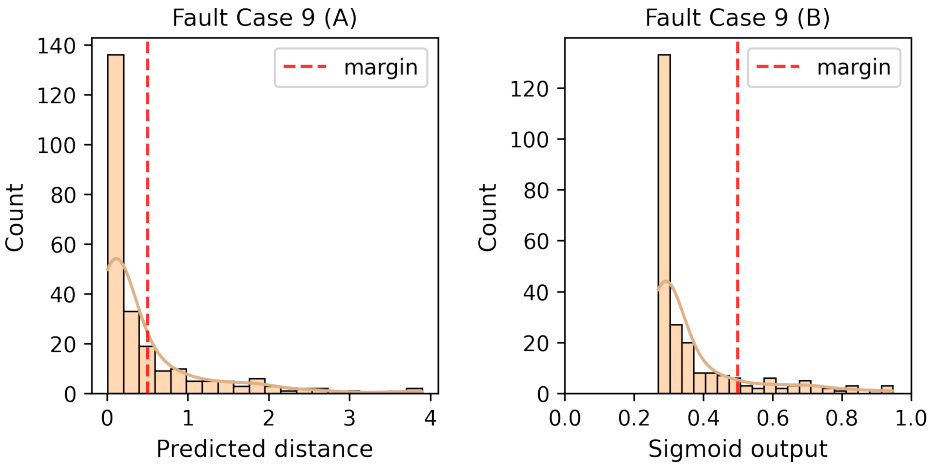

Figure B.33: Test predicted distance (A) and sigmoid output (B) distributions of fault case 9.

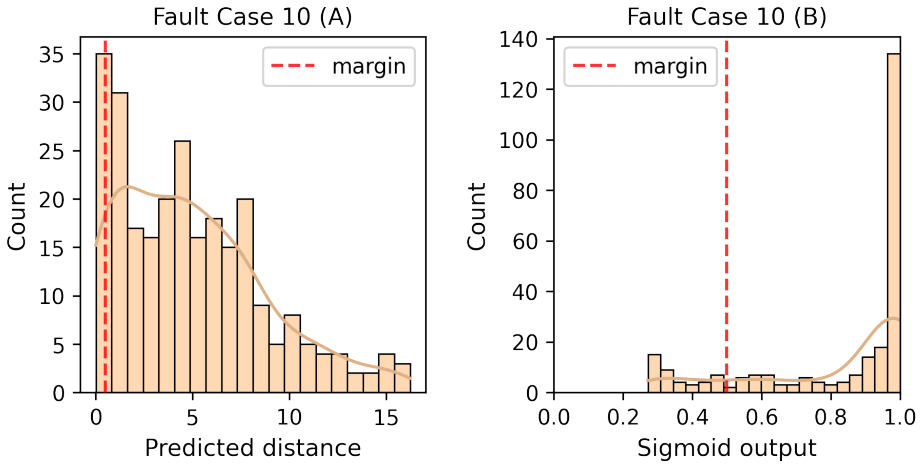

Figure B.34: Test predicted distance (A) and sigmoid output (B) distributions of fault case 10.

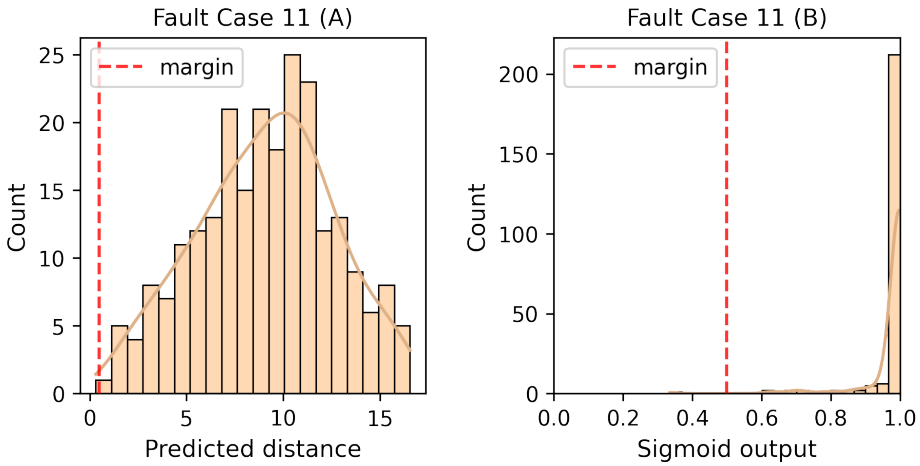

Figure B.35: Test predicted distance (A) and sigmoid output (B) distributions of fault case 11.

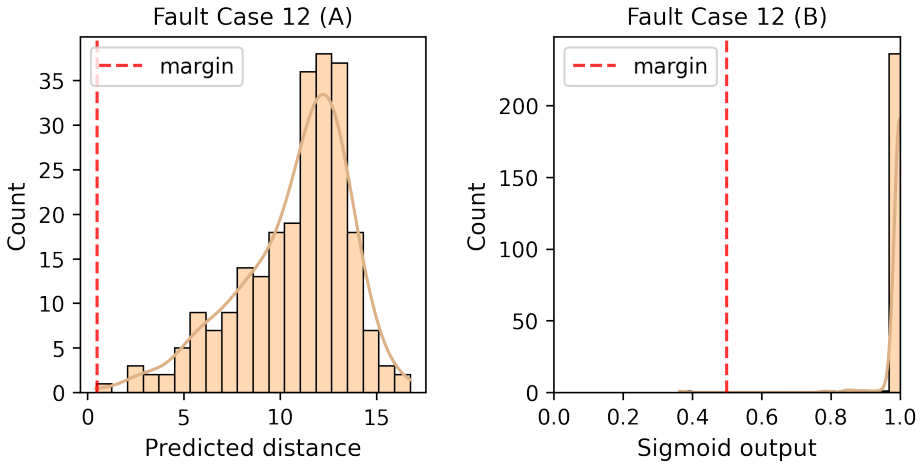

Figure B.36: Test predicted distance (A) and sigmoid output (B) distributions of fault case 12.

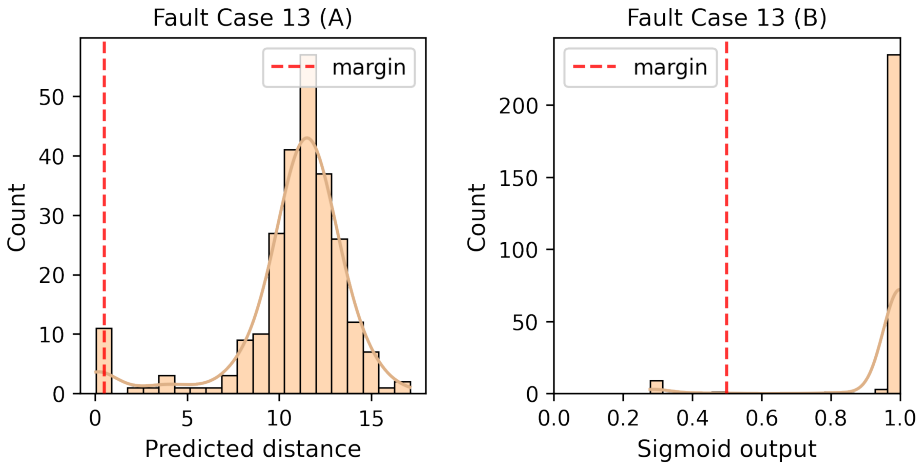

Figure B.37: Test predicted distance (A) and sigmoid output (B) distributions of fault case 13.

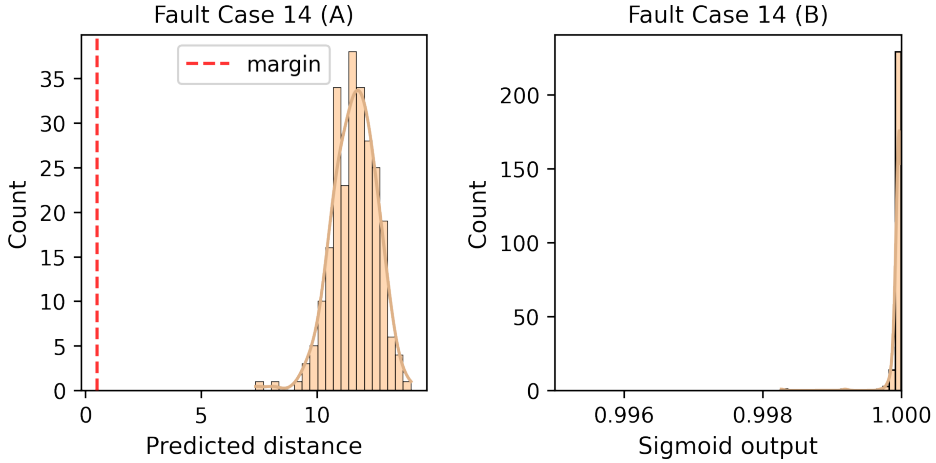

Figure B.38: Test predicted distance (A) and sigmoid output (B) distributions of fault case 14.

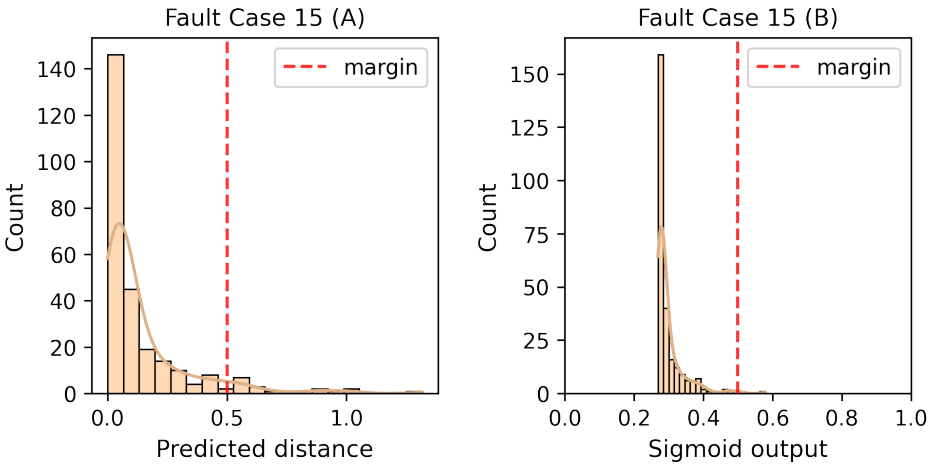

Figure B.39: Test predicted distance (A) and sigmoid output (B) distributions of fault case 15.

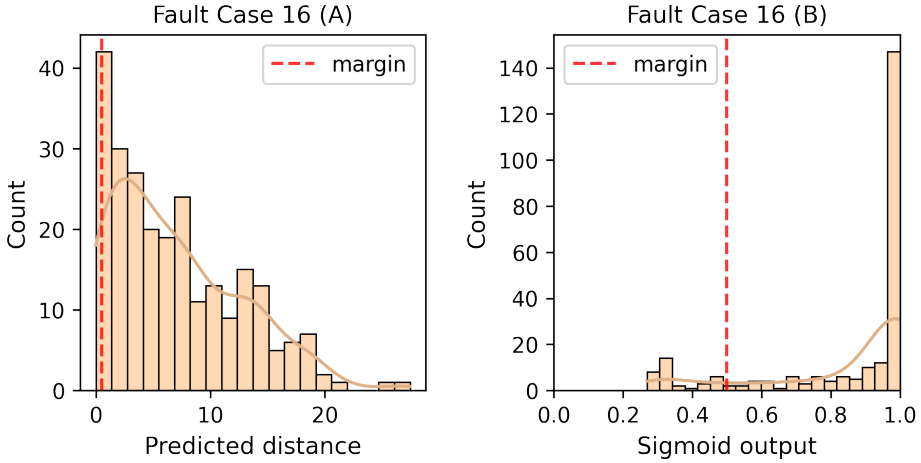

Figure B.40: Test predicted distance (A) and sigmoid output (B) distributions of fault case 16.

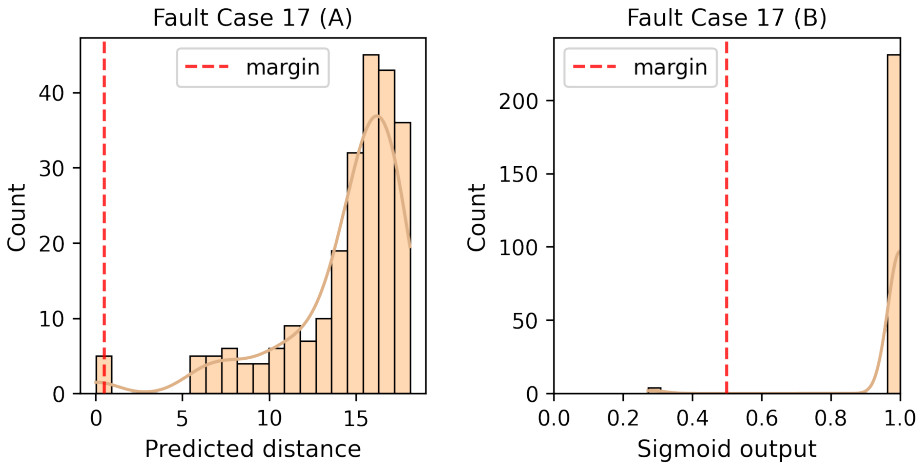

Figure B.41: Test predicted distance (A) and sigmoid output (B) distributions of fault case 17.

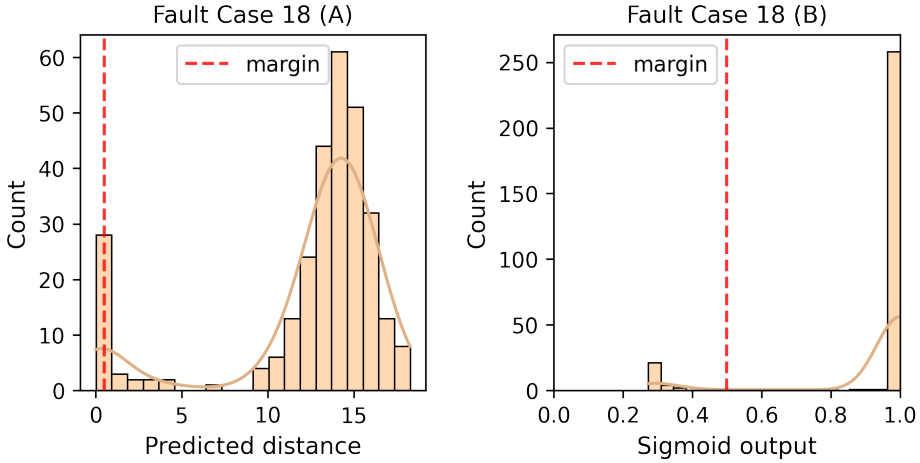

Figure B.42: Test predicted distance (A) and sigmoid output (B) distributions of fault case 18.

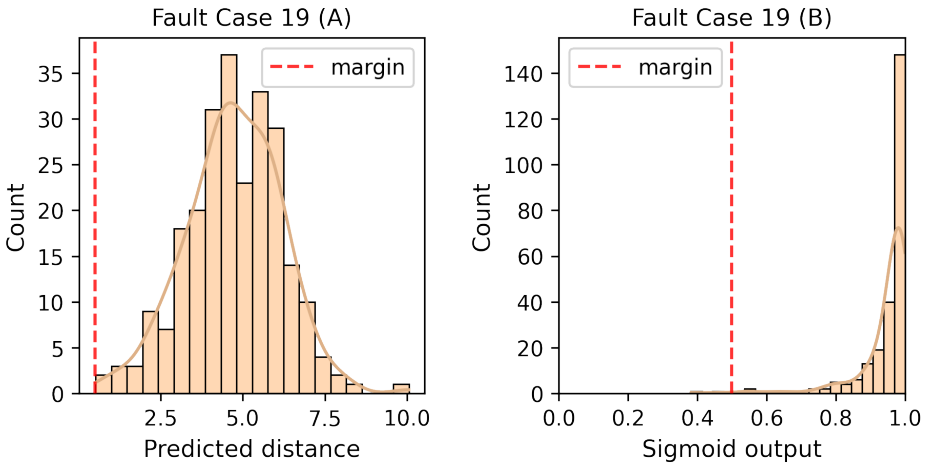

Figure B.43: Test predicted distance (A) and sigmoid output (B) distributions of fault case 19.

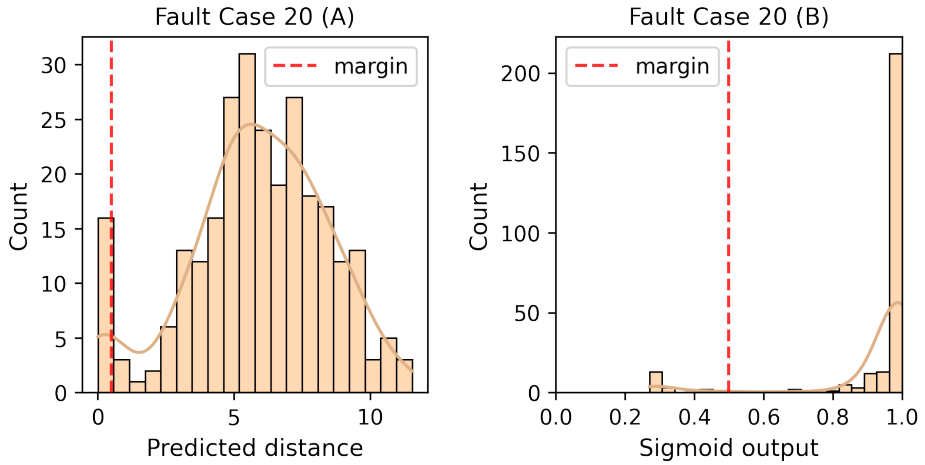

Figure B.44: Test predicted distance (A) and sigmoid output (B) distributions of fault case 20.