

Physically Augmented Training for Generalizable Deep Learning in Environmental Modeling

UNIVERSITÀ DEGLI STUDI ROMA TRE
DIPARTIMENTO DI INGEGNERIA CIVILE, INFORMATICA E DELLE
TECNOLOGIE AERONAUTICHE



PhD Final Thesis
Civil Engineering
Cycle XXXVIII

PhD Candidate:
Gianmarco Guglielmo

Supervisor:
prof. Pietro Prestininzi

Co-Supervisor:
prof. Andrea Montessori

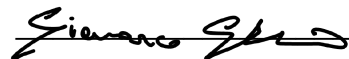


Ingegneria Civile
CORSO DI DOTTORATO DI RICERCA IN

XXXVIII
CICLO DEL CORSO DI DOTTORATO

Physically Augmented Training for Generalizable
Deep Learning in Environmental Modeling
Titolo della tesi

Gianmarco Guglielmo
Nome e Cognome del dottorando


firma

Pietro Prestininzi
Docente Guida/Tutor: Prof.


firma

Fabrizio Paolacci
Coordinatore: Prof.


firma

Abstract

In recent years, Machine Learning (ML) has attracted growing interest across virtually every scientific domain, including environmental sciences, where it is increasingly used to identify patterns in data, make predictions, and model complex physical systems with the promising advantage of reducing the computational burden of traditional numerical simulations.

However, ML methods generally require large amounts of high-quality data to achieve reliable performance. In many fields—such as hydrology and hydraulics—the limited availability of extensive training data severely restricts the applicability of data-driven approaches. Therefore, researchers have resorted to numerical models to build training datasets, as in flood hazard mapping problems. Given the limited generalization capability of data-driven models—i.e., their reduced ability to perform well on unseen data, transfer knowledge to new domains or configurations, or extrapolate beyond the bounds of the training dataset—they typically need to be retrained for each new application. This substantially diminishes the practical advantages that data-driven methods promised to offer over numerical simulations. Indeed, their proficient use is closely linked to the ability to construct reliable, albeit computationally demanding, numerical models. In short, the true bottleneck lies in their limited ability to generalize.

One possible way to endow ML approaches with this desired ability is to exploit the underlying physical laws from which environmental data originate. Physics-informed ML builds on this ambition to enhance model performance. Within this broader direction—although with notable conceptual differences from the widespread concept of Physics-Informed Neural Networks, which are discussed in this work—the methodology employed here, namely Physical Augmented Training, modifies the

loss function by introducing a physical regularization term. This term couples model inputs and outputs through physical relations, effectively augmenting the informational content available during training.

The methodology is evaluated with the objective of improving model generalization across three progressively complex testbeds: a simple physical system, a one-dimensional, highly controllable hydraulic experimental setup, and a two-dimensional flood hazard mapping problem. In the latter case, a model is trained on data from a single river catchment under multiple synthetic rainfall events and tested on a previously unseen basin in a one-to-one basin-transfer setup.

Improving catchment generalization is a key objective, as it would greatly increase the effectiveness of data-driven flood hazard mapping models, enabling them to go beyond the simple interpolation of synthetic data generated by numerical models. The proposed flood-mapping framework demonstrates promising potential for predicting flood hazard across entire river networks or multiple catchments, particularly in data-scarce regions, supporting effective knowledge transfer from data-rich basins to ungauged catchments, a long-standing chimera of hydrology. Moreover, the methodology to embed physical information appears versatile and potentially applicable to a broad range of environmental modeling tasks, offering a general strategy for incorporating physics into data-driven approaches.

Acknowledgments

I would like to express my deepest gratitude to Prof. Pietro Prestininzi, who encouraged me to undertake this PhD journey and continuously supported the development of my research through his guidance and valuable advice. I am also sincerely grateful to my co-supervisor, Prof. Andrea Montessori, whose constant availability and encouragement were never lacking. A special acknowledgment is due to Prof. Michele La Rocca for his kind support. I am also grateful to Prof. Allen Bateman Pinzon, with whom I had the pleasure of collaborating, for the fruitful scientific exchange.

I would like to particularly thank Dr. Andrea Gemma for his invaluable help and Dr. Jean-Michel Tucny for his collaboration. I also warmly thank my fellow PhD colleagues, who accompanied me throughout this journey.

My sincere gratitude goes to my teachers and mentors: Arduino Maiuri, who inspired me to strive for excellence, and Rita Migliorati, whose guidance has shaped who I am today.

My thoughts go to my grandparents, Augusta, Ippazio, and Luigi, who are no longer with us, but whose example remains in my heart. I am deeply grateful to my grandmother Adriana for her unconditional care and constant presence in everything I do. I also wish to thank my family—my mother Anna Rita, my father Cesare, my aunt Romina, and my uncle Antonio—for their encouragement and continued support.

Finally, I would like to thank all my friends and everyone who has been close to me during this journey.

Contents

List of Figures	xii
List of Tables	xv
1 State of the Art and Literature Review	5
1.1 Introduction	5
1.2 Data-driven models in hydrology and hydraulics	6
1.3 Data-driven models for flood mapping	8
1.3.1 Flood susceptibility mapping	9
1.3.2 Flood inundation mapping	9
1.3.3 Flood hazard mapping	10
1.4 Physics-Informed Machine Learning	12
1.5 Research gaps and open challenges in data-driven flood mapping . .	13
1.5.1 Generalization	13
1.5.2 Embedding physics	14
1.5.3 Probabilistic hazard assessment	16
2 Deep Learning models	17
2.1 Introduction	17
2.2 Fundamentals of Machine Learning	18
2.2.1 ML Paradigms	18
2.2.2 Dataset Partitioning	20
2.2.3 Overfitting and Underfitting	21
2.3 Neural Networks	21
2.3.1 Activation functions	24

2.3.2	Going Deep: Multi-Layer Perceptron	25
2.3.3	Training Phase	27
2.4	Convolutional Neural Networks	32
2.5	Recurrent Neural Networks and LSTM	37
2.6	Autoencoders	38
2.7	CNN encoder-decoder	40
2.7.1	U-Net architecture	41
2.8	Transformers	42
2.9	Physics exploitation	44
2.9.1	Physics-Informed Neural Networks	44
2.9.2	Proposed Approach: Physically Augmented Training	50
3	The Motion of a Pendulum as predicted by DL approaches	53
3.1	Pendulum physics	53
3.2	Dataset Creation	54
3.3	Physics exploitation	54
3.3.1	PINN approach	55
3.3.2	PAT approach	56
3.4	Conclusions	63
4	A 1D hydraulic application: varied flows in river	69
4.1	Introduction	69
4.2	Problem definition and physics	70
4.3	Dataset generation	72
4.4	Employed Neural Network architectures	75
4.4.1	Single-Point	75
4.4.2	Integrator	76
4.4.3	Vector-To-Sequence	78
4.4.4	Hyperparameters	79
4.5	PAT strategy	80
4.6	Discussion of Results	81
4.6.1	Complete training dataset	83

4.6.2	Stress tests: extrapolation, training dataset size, and model complexity	86
4.7	Conclusions	88
5	Flood mapping with catchment generalizability	91
5.1	Introduction	91
5.2	Methods	92
5.2.1	Topographic data	93
5.2.2	Patch-based approach and reconstruction methods	94
5.2.3	Loss Functions	94
5.2.4	DL Architecture	97
5.3	Simulations setup	100
5.3.1	Hydraulic model equations	100
5.3.2	Dataset generation	101
5.3.3	Hyperparameters	104
5.3.4	Model evaluation	105
5.4	Results	107
5.4.1	Comparison of Models Outcomes: patch-level	107
5.4.2	Comparison of Model Outcomes: Full basin map reconstruction	109
5.4.3	Comparison with official flood mapping	118
6	Discussion	124
6.1	Between Numerical and Deep Learning Approaches	124
6.2	Embedding physical knowledge	125
6.3	The problem of catchment generalization	127
7	Conclusions	131
7.1	Future developments	133

List of Figures

2.1	Examples of overfitting and underfitting in polynomial regression . . .	22
2.2	Learning Curve	23
2.3	Global and Local minima of a Loss Function	29
2.4	Training and Validation Loss with Early Stopping	33
2.5	Convolution illustration	34
2.6	Pooling illustration	35
2.7	Convolutional Architectures	36
2.8	A Residual block	37
2.9	RNN	38
2.10	An LSTM cell	39
2.11	Autoencoder	40
2.12	U-Net	42
2.13	PINN	46
2.14	Cylinder arrays	47
2.15	PINN for gas flow in cylinder arrays	49
2.16	Results: errors vs Knudsen number	50
3.1	Pendulum: sampled dataset	56
3.2	Pendulum: MLP prediction	57
3.3	Pendulum: PINN prediction	57
3.4	Pendulum: Single-energy level sampled dataset	59
3.5	Pendulum: Single-energy level dataset split	60
3.6	Pendulum: Single-energy level predictions	61
3.7	Pendulum: Three-energy levels sampled dataset	62

3.8	Pendulum: Three-energy level dataset split	63
3.9	Pendulum: Three-energy levels predictions	64
3.10	Pendulum: Three-energy levels predictions - phase diagrams	65
3.11	Pendulum: Three-energy levels unbalanced dataset split	65
3.12	Pendulum: Three-energy levels unbalanced dataset predictions	66
3.13	Pendulum: Three-energy levels unbalanced dataset predictions - phase diagrams	67
4.1	Schematic of a hydraulic jump	72
4.2	Sketch of the 1D physical problem	73
4.3	Visualization of a subset of the 1D dataset of water profiles	74
4.4	Profile reconstruction using the three different FFNN approaches	76
4.5	Employed MLP architectures	77
4.6	1D problem: typical outcome of the employed models	82
4.7	1D problem: metric distributions (box plot) for SP and INT approaches	84
4.8	1D problem: metric distributions (box plot) for VTS architecture	85
4.9	1D problem: Stress test on small data and extrapolation	89
4.10	1D problem: Stress test on overfitting and extrapolation	90
5.1	2D Flood Mapping experiment overview	93
5.2	Distribution of the selected patches over the training and test basins	95
5.3	DL architecture for flood mapping	98
5.4	Location of the Misa and Cesano basins	102
5.5	Distributions of input features across training, validation, and test datasets	103
5.6	Reference and predicted depth maps	110
5.7	Reference and predicted velocity maps	111
5.8	Different reconstruction methods from patches in a selected area	112
5.9	Error depth maps	115
5.10	Errors in water depth prediction as a function of rainfall intensity across different models	116
5.11	Spatial distribution of classification results	120

5.12 Spatial distribution of classification results: downstream area of test basin	121
5.13 Comparison with official flood mapping (overview)	122
5.14 Comparison with official flood mapping (details)	123

List of Tables

2.1	Weights, bias, and output for basic logical gates using a single neuron	24
2.2	XOR using a shallow NN	24
2.3	Most used activation functions	26
2.4	LSTM equations	39
2.5	Hyperparameters used for the PINN setup	50
5.1	Morphometric characteristics of the training and test basins	102
5.2	Validation and test (overlapping dataset) RMSEs for depth and velocity	108
5.3	Evaluation of the predictive capabilities of the models	113
5.4	Classification metrics for depth prediction, averaged over rainfall intensities	117

Introduction

River floods represent one of the most destructive natural hazards, challenging current forecasting systems and risk management strategies. Flood mapping plays a fundamental role in river flood risk assessment, as identifying the extent of inundated areas and the severity of flooding is crucial for estimating potential damages, planning new developments appropriately, designing safe infrastructures, managing emergencies more effectively, and supporting civil protection strategies.

Classical modeling approaches for flood mapping rely on a modeling chain [Chow, 1971]. Hydrological models quantify the expected discharge in a river reach by accounting for precipitation patterns and climate, soil types, infiltration rates differentiated by soil characteristics and land use, and evapotranspiration. Subsequently, hydraulic models use boundary conditions provided by hydrological models, along with channel geometry (such as topography, riverbed shape, and obstacles), to numerically solve the Navier-Stokes (NS) equations, typically in the form of simplified Shallow Water (SW) equations, and simulate fluid dynamics. These numerical methods for flood modeling are well established, physically based, and widely trusted.

However, the results often fall short of providing a comprehensive description of the phenomenon, given that many crucial aspects are either neglected or unknown. The outcomes feature significant uncertainty, stemming from both epistemic uncertainty due to the multitude and complexity of variables governing the phenomenon, coupled with insufficient data, and aleatoric uncertainty, intrinsic to the phenomena governing river hydrodynamics. For example, the exact geometry of the natural channel is rarely known precisely and is subject to continuous changes; parameters require calibration (often on limited observed data) within an equifinality framework, introducing further uncertainty; and additional parameters must be incorporated into the modeling

to account for local phenomena approximately. Moreover, despite advances in Computational Fluid Dynamics and available computational resources, numerical methods for river flood mapping remain impractical for large-scale or real-time applications. Even with several simplifications [Prestininzi et al., 2011], simulating environmental fluid dynamics entails a high computational burden.

Furthermore, the growing demand for reliable flood maps of ungauged basins [Cole et al., 2006, Blöschl, 2013, Hrachowitz et al., 2013] - spanning from small catchments to scarcely populated large regions in developing countries - increases the need for alternative approaches. In fact, classical modeling approaches [Kumar et al., 2023] may become unfeasible in such cases, and resorting to the exploitation of similarities with other basins has been envisaged.

Consequently, Artificial Intelligence (AI) has garnered increasing attention from the scientific community in hydrological and hydraulic research. Data-driven modeling has been proposed [Dasgupta et al., 2023] as a promising approach to overcoming the aforementioned challenges in environmental hydraulics [Kratzert et al., 2019]. The main attraction of data-driven approaches lies in their potential computational advantage over traditional numerical models, enabling applications over large spatio-temporal domains and early warning systems. These models can automatically learn complex patterns from data and provide rapid predictions once trained. Nonetheless, their black-box nature and lack of physical interpretability raise concerns about generalizability and trustworthiness [Sellars, 2018], especially in critical applications such as flood hazard assessment. Moreover, these models typically require large, high-quality datasets for effective training and validation, resources that are often scarce in hydrological contexts. Hence, if classical hydraulic models already face challenges due to the lack of high quality and/or sparse measurements, which do not allow for proper calibration of numerical models, data-driven models are even more affected [Guo et al., 2021].

A consequence closely related to overfitting and, as such, attributable to the scarcity of calibration data, is the frequent violation of conservation laws [Berkhahn et al., 2019], leading to marked non-physical results. Compounding this challenge, river flood mapping necessitates models that are able to explore scenarios not included

within the training boundaries, i.e., featuring predictive/generalization capabilities. Any ML effort aimed at predictive river flood mapping must address this fundamental requirement.

The scarcity of extensive, high-quality hydrological datasets and the proven reliability of numerical simulations make it natural to consider hybrid approaches. A common strategy is to use outputs from physically based hydraulic models as reference values for training data-driven models. While Deep Learning (DL) models are often promoted for their computational speed, this advantage is diminished when considering the time required to construct the numerical model, generate an adequate synthetic dataset, and train the DL model. Moreover, their applicability becomes even less appealing considering that these approaches mostly require retraining when applied to new configurations. However, once re-trained, these models can provide rapid flood predictions, making them attractive for real-time applications and for multiple applications with uncertainty assessment. A direct consequence of this approach is that synthetic data generated from the hydrodynamic reference models represent an upper threshold for the accuracy of DL models.

A recent paradigm shift is represented by Physics-Informed Machine Learning (PIML) [Karniadakis et al., 2021]. PIML enhances existing models by introducing physically-based constraints, seeking to reduce learning time, increase generalization capabilities [Jamali et al., 2021], and satisfy conservation equations [Jagtap et al., 2020]

The aim of this work is to assess whether the generalization capabilities of DL models for river hydraulics and flood mapping could be increased through the incorporation of physical knowledge into the training process. The incorporation of expert knowledge into DL models is achieved here by integrating physically-based information without relying on the underlying mathematical formulation (e.g., the calculation of the residuals of differential equations). Specifically, the physical information is integrated into the DL model by modifying the loss function, introducing an additional term beyond the error between the predicted and observed data. The latter employs physical principles, laws, and quantities that are not explicitly formulated in the original dataset. In this sense, a similarity can be

observed with data augmentation, a widely used technique in ML that extracts additional insights by providing alternative interpretations of the same dataset. This approach, referred to as Physical Augmented Training (PAT), appears particularly valuable for flood simulations, where hydraulically relevant distributed parameters such as roughness, lithology, topography, etc., pose significant uncertainties. The method is versatile and applicable to physical systems and scenarios in which the underlying mathematical formulation is not fully known, but expert knowledge enables the introduction of meaningful, physically inspired constraints.

The thesis is structured as follows. Chapter 1 reviews the current state of the art in the field of data-driven models for flood prediction and PIML. Chapter 2 provides an overview of DL models; it then introduces Physics-Informed Neural Networks (PINNs) and concludes by presenting the PAT strategy. The effectiveness of the proposed approach is then evaluated across three testbeds of increasing complexity: a basic physical system in Chapter 3, a highly controlled one-dimensional experiment in Chapter 4, and a two-dimensional flood mapping scenario in Chapter 5. In the latter, the model is trained on synthetic rainfall data from a single river catchment and subsequently evaluated on a distinct, unseen basin, pursuing a one-to-one basin transfer strategy. This setup poses a considerable challenge for data-driven models due to scarce training data and the requirement to extrapolate to entirely novel topographic conditions. Chapter 6 offers a critical discussion, whereas the conclusions and potential future developments are presented in Chapter 7.

Chapter 1

State of the Art and Literature Review

A critical review of the main data-driven approaches adopted in hydrology and hydraulics—with a specific focus on flood mapping—is presented, highlighting the state of the art and identifying research gaps relevant to the present study.

1.1 Introduction

Recently, data-driven models have gained increasing attention in the hydrological and hydraulic sciences, offering powerful and promising tools for analyzing complex, non-linear relationships between input and output variables. By not requiring an explicit representation of the underlying physical processes, these models can overcome some sources of epistemic uncertainty and reduce computational demands.

Data-driven approaches have been applied to a wide range of problems, often improving or even replacing single or multiple components of the modeling chain for flood prediction, including weather forecasting, rainfall–runoff modeling, flood forecasting, estimation of hydraulic parameters, and flood mapping. Within this broad spectrum of applications, river flood hazard mapping represents a particularly challenging task with significant implications for both scientific research and practical flood risk management.

Section 1.2 summarizes the main developments in data-driven approaches within

hydrology and hydraulics. Section 1.3 provides a focused review of data-driven models for flood mapping. Section 1.4 examines the growing interest in Physics-Informed approaches. Finally, Section 1.5 outlines the open research gaps and challenges.

1.2 Data-driven models in hydrology and hydraulics

Data-driven models employ diverse techniques and methodologies that utilize extensive datasets to extract meaningful patterns and generate predictions [Abrahart et al., 2008]. In recent years, their application to hydrological and hydraulic problems has grown considerably [Solomatine and Ostfeld, 2008], driven not only by advances in ML and DL methodologies, but also by computational speed-ups and the possibility of bypassing resource-intensive numerical modeling procedures and complementing traditional physically-based approaches.

ML encompasses a broad set of algorithms, ranging from classical approaches such as regression models [James et al., 2013], decision trees [Breiman et al., 2017], clustering algorithms [Rokach and Maimon, 2005], Support Vector Machines [Cortes and Vapnik, 1995], and ensemble methods such as Random Forests [Breiman, 2001], to neural networks (NNs) [Rosenblatt, 1958]. DL refers specifically to NN architectures with multiple hidden layers [LeCun et al., 2015, Goodfellow et al., 2016], following a representation learning paradigm in which higher-level features are expressed in terms of lower-level, simpler representations.

In hydrology and hydraulics applications, there has been a growing shift towards DL approaches, motivated by their ability to automatically learn hierarchical representations of data and to process high-dimensional datasets without the need for extensive feature engineering [Kutz, 2017, Shen, 2018, Sit et al., 2020]. DL architectures have shown remarkable success in many domains relevant to hydrology and hydraulics [Maier et al., 2010, Brunton et al., 2020].

Mosavi et al. [2018] provided a comprehensive summary of ML methods applied in hydrological studies, highlighting their broad applicability for flood prediction. Similarly, Xu and Liang [2021] identified key applications of ML in hydrological science, such as the processing of remote sensing data, the prediction of hydrological variables, and modeling hydrological processes, including rainfall-runoff and

streamflow.

For precipitation forecasting, data-driven models have been successfully employed to improve prediction accuracy by capturing spatiotemporal dependencies in meteorological data [Kumar et al., 2019, Barrera-Animas et al., 2022, Conti, 2024].

Hydrological time-series have been predicted using data-driven models [Nayak et al., 2004, Qin et al., 2019, Waqas and Humphries, 2024].

For rainfall–runoff modeling, earlier applications employed Multi-Layer Perceptrons (MLPs) [Hsu et al., 1995, Dawson and Wilby, 2001]. More recently, Kratzert et al. [2018] applied Long Short-Term Memory (LSTM) networks for daily runoff prediction for the first time, providing a hydrological interpretation of the LSTM by considering the cell state as a storage component. In a subsequent study, Kratzert et al. [2019] applied LSTM networks for water discharge prediction in ungauged basins, thereby addressing one of the long-standing challenges in hydrology. Significant efforts have since been devoted to rainfall–runoff modeling, with various DL architectures being explored [Van et al., 2020, Gao et al., 2020, Wang et al., 2024, Li et al., 2024b, Yin et al., 2025]. Nevertheless, the LSTM remains the most widely used architecture for several hydrological problems, including flash-flood prediction [Santos et al., 2025].

Data-driven models have also been developed to forecast hydrological variables at specific river sections [Dazzi et al., 2021, Bomers and Hulscher, 2023], enabling operational flood forecasting systems [Nevo et al., 2022, Nearing et al., 2024]. They have also been applied to predict river water quality parameters; for this application, Zanoni et al. [2022] compared deep MLPs, random forests, and linear regression models.

The use of data-driven models further extends to hydropower and dam engineering, including predictive modeling, real-time monitoring, optimization, and planning [Hariri-Ardebili et al., 2023].

Beyond hydrological applications, ML and DL have been employed in hydraulic and fluid mechanics problems [Brunton et al., 2020, Sofos et al., 2022], including the prediction of flow fields in rivers and channels [Zhang et al., 2022], the estimation of hydraulic parameters [Bindas et al., 2023], sediment transport dynamics [Gupta

et al., 2021], and the optimization of water distribution systems [Garzón et al., 2022]. Considerable effort has also been devoted to developing surrogate models aimed at reducing the computational burden associated with solving the NS equations [Jin et al., 2021] (see Section 1.4). Beyond these applications, data-driven models have been used to approximate complex fluid–structure interactions [Gao and Jaiman, 2024], investigate super-cavitation phenomena [Kamali and Erfanian, 2025] and flows over complex geometries [Tucny et al., 2025], as well as to analyze experimental data [Lee et al., 2017], representing only a subset of the applications explored to date. In turbulence studies, emerging directions include the study of turbulence statistics [Zhang et al., 2022], the development of ML-based turbulence closures [Duraismy et al., 2019, Maruyama et al., 2021], and the prediction of turbulent particle trajectories [Li et al., 2024a].

Building on the overview presented above, data-driven techniques have the potential to accelerate, reduce, or bypass numerical simulations, thereby enhancing predictive capabilities in the water sciences.

1.3 Data-driven models for flood mapping

The main interest in applying data-driven models for flood mapping lies primarily in their promising computational advantages over traditional numerical models. The latter, however, are expected to retain a crucial role in data-driven flood mapping, as they provide the training datasets [Karim et al., 2023] to overcome the data scarcity that afflicts the field. It should also be noted that, even when large numerical datasets are available, a data-driven model cannot surpass the accuracy of the numerical model that generated its training data.

A comprehensive review of works applying DL models for flood mapping was presented by Bentivoglio et al. [2022], who proposed a classification of the possible applications into three main types: flood susceptibility mapping, flood inundation mapping, and flood hazard mapping.

1.3.1 Flood susceptibility mapping

Flood susceptibility mapping qualitatively classifies a basin into areas that are more or less prone to flooding, based primarily on geographical and topographical information, along with records of past flood events. Traditionally, this task has been performed using Multi-Criteria Decision Analysis [Khosravi et al., 2019], producing large-scale regional assessments.

The adoption of DL models in this context has helped overcome the need for explicit feature extraction [Wang et al., 2020, Pradhan et al., 2023], which is often time-consuming in data-driven approaches.

DL methods have employed both MLPs and Convolutional Neural Networks (CNNs). The latter are particularly well-suited to capture spatial characteristics, thereby yielding smoother and more reliable results [Khosravi et al., 2020, Panahi et al., 2021].

Despite these advances, important limitations remain in this type of application. Reliable historical data are required, whereas the labeling of flood-prone areas often appears arbitrary. Moreover, models must generally be trained and applied separately for each region, preventing knowledge transfer across different basins. As a result, many published studies merely apply the same methodologies, sometimes with minor variations, to distinct geographic areas. These are often regions (e.g., [Das, 2020, Chaulagain et al., 2023, Ghahraman et al., 2023]) where classical approaches to hazard estimation are not feasible and where large-scale susceptibility mapping provides valuable support for land-use planning and flood risk management.

1.3.2 Flood inundation mapping

Flood inundation mapping aims to determine the spatial extent of flooded areas by processing remote sensing products. This task is a typical classification problem, distinguishing between flooded and non-flooded areas, with a possible additional discrimination between permanent water bodies (e.g., rivers and lakes) and temporarily flooded regions.

Unlike flood susceptibility studies, flood inundation mapping involves both basin-scale and local-scale areas, depending on the availability of satellite imagery. In

many cases, the limited availability of training datasets remains a major constraint for the effective application of DL to this problem [Amitrano et al., 2024].

Since the task essentially reduces to image detection, CNNs have been the predominant choice [Gebrehiwot et al., 2019, Sarker et al., 2019, Peng et al., 2019]. A natural progression has been the adoption of U-Net architectures [Ronneberger et al., 2015], as demonstrated by Nemni et al. [2020] and Li and Demir [2023].

The application of different architectures is a possible research path in this field: recently, Ahmadi et al. [2025] proposed a capsule network framework for flood inundation mapping using both radar and optical images. Their approach also introduced multiscale kernel convolution layers, which are particularly effective in capturing phenomena across different spatial scales.

Overall, despite the development of increasingly sophisticated architectures, the reliability and transferability of flood inundation mapping models are still strongly limited by data availability and quality.

1.3.3 Flood hazard mapping

Several studies have explored different DL architectures for flood hazard assessment, i.e., the prediction and quantitative evaluation of floods in terms of water depths [Liu et al., 2025] and velocities, using topographical data, hyetographs, and hydrographs as input.

One of the earliest approaches to flood inundation mapping involved MLPs trained on outputs from hydrodynamic models. Berkhahn et al. [2019] proposed an ensemble of MLPs to predict maximum flood water levels based on precipitation rates. Similarly, Lin et al. [2020] divided the study area into independent grid cells, each modeled by an MLP, to predict the maximum extent of flood inundation from multiple discharge inputs.

Convolutional architectures were then applied due to their ability to exploit spatial correlations in the data, capturing spatial patterns through convolutional kernels. Kabir et al. [2020] employed CNNs to predict fluvial flood inundations using upstream hydrographs as inputs.

Recurrent Neural Networks (RNNs) and LSTM networks have also been explored

to incorporate temporal dependencies. Zhou et al. [2021] combined deep input selection with LSTMs for rapid flood inundation prediction, using hydrograph inputs and water level outputs, applying thresholds for small depth values.

Another path of research involves coupling hydrodynamic models with DL approaches to improve predictive performance. Yang et al. [2024] explored this synergy by combining hydrodynamic simulations with real-time data from hydrological gauges and initial river water levels, thereby improving the accuracy of urban flood depth predictions.

The autoencoder architecture has recently been proven to be effective in predicting flood inundation depths and regional rainfall patterns [Kao et al., 2021], as well as in predicting the spatiotemporal evolution of river floods when formulated as a transformer-based model that employs an autoregressive procedure [Pianforini et al., 2024].

Moreover, the U-Net architecture has received significant attention, as its skip connections mitigate the problem of small kernels limiting global feature extraction. Löwe et al. [2021] used U-Net to predict urban pluvial floods, utilizing topographical data and hyetographs as inputs to generate 2D maps of the maximum water depth. A key novelty was the use of raster samples from the same city that were not included in the training data. Despite promising results, the model struggled to capture water depths along flow paths. Guo et al. [2021] proposed a Convolutional Autoencoder that integrates an MLP for rainfall processing to predict maximum urban flood depth. The study introduced a patch-based approach rather than using the entire catchment, focusing on generalization across different hyetographs. In a follow-up study, Guo et al. [2022] demonstrated that a patch-based CNN strategy allows a trained model to be reused across different catchment areas with varying topographies, provided the same rainfall event is considered. Yet, the model struggled to capture water accumulation patterns over large areas.

Finally, in recent years, several lines of work have extended the concepts of DL to non-Euclidean domains, such as graphs [Wu et al., 2020]. Bentivoglio et al. [2023] explored the use of Graph Neural Networks (GNNs) to solve SW equations on unseen topographies, offering an alternative physics-aware surrogate modeling approach.

GNNs are commonly employed as surrogate models [Taghizadeh et al., 2025]: they represent the computational domain as a graph, where each mesh element is modeled as a node, while edges encode the local connectivity between them.

1.4 Physics-Informed Machine Learning

A recent paradigm shift is represented by Physics-Informed Machine Learning (PIML) [Karniadakis et al., 2021]. PIML enhances existing models by introducing physically-based constraints, theoretically reducing learning time, increasing generalization capabilities [Jamali et al., 2021], and attempting to satisfy conservation equations [Jagtap et al., 2020]. Furthermore, it is widely reported in the literature that the inclusion of physical information into ML models is beneficial, especially in conditions of small data regimes [Zhu et al., 2019, Karniadakis et al., 2021, Eichelsdörfer et al., 2021]. Indeed, in many applications, expecting a DL model to infer a complex nonlinear relationship from only a limited number of high-dimensional input–output pairs may be overly simplistic [Raissi et al., 2019], especially considering that a successful model is ultimately expected to make reliable predictions under conditions not encountered during training.

An influential example of PIML is represented by PINNs, introduced by Raissi et al. [2019], which act as neural solvers for differential problems [Hao et al., 2022]. These networks are specifically designed to find solutions to forward and inverse problems governed by partial differential equations (PDEs), modifying the loss function with a PDE residual term computed via Automatic Differentiation [Baydin et al., 2018] and optional additional terms for initial and boundary conditions [Lu et al., 2021a].

PINNs have been applied to benchmark problems in fluid mechanics, particularly for NS equations [Jin et al., 2021]. They have demonstrated potential advantages over classical techniques by integrating empirical data and handling ill-posed problems [Cai et al., 2021, Cuomo et al., 2022]. Additionally, PINNs enable tackling problems of extremely high dimensionality, where traditional solvers would face prohibitive computational costs.

In river hydraulics, PINNs have been applied for spatial and temporal forecasting

in one-dimensional channels [Cedillo et al., 2022, Mahesh et al., 2022, Xu et al., 2024], for downscaling large-scale river models by assimilating available observations [Feng et al., 2023], and for approximating solutions to the SW equations during flood events in a specific river reach, achieving prediction accuracies comparable to Finite-Volume solvers while significantly reducing computational costs [Qi et al., 2024].

1.5 Research gaps and open challenges in data-driven flood mapping

1.5.1 Generalization

Generalization is a crucial requirement in flood mapping using data-driven methodologies, since a model acting merely as an interpolator of numerical results would offer limited practical value. The lack of sufficiently large and informative training datasets [Cache et al., 2024], combined with the high computational cost of generating numerical data, inevitably lead to overfitting. As a result, existing DL flood hazard studies are typically implemented only at local and regional scales [Bentivoglio et al., 2022], and poor generalization forces retraining whenever conditions differ from those in the original training set.

Compounding this challenge, river flood mapping necessitates models capable of exploring scenarios not comprised within the training boundaries [Fraehr et al., 2024], given that future flood conditions may differ significantly from past observations due to climate variability, land-use changes, and the need to assess rare or hypothetical extreme events. Most attempts in flood hazard simulation confirm that data-driven approaches face considerable challenges in achieving effective generalization [do Lago et al., 2023, Karim et al., 2023]. In practice, this means that such models often fail to provide reliable predictions for scenarios outside the training data [Nguyen et al., 2023].

The need for retraining in the case of predictions over new catchments is the epiphany of the broader lack of generalization affecting DL-based models [Guo et al., 2022, Manchingal and Cuzzolin, 2025]. The primary reasons for the lack of data-driven models capable of extrapolating [Nearing et al., 2021] to unseen terrains

include scaling and homogeneity issues [Guo et al., 2021].

Recent research has increasingly focused on developing DL models for flood prediction that can generalize effectively to unseen geometries [Bentivoglio et al., 2023]. For instance, Löwe et al. [2021] demonstrated promising accuracy in predicting water depth on raster samples from the same city that were not used in the training data; however, the model struggled to predict flooding outside depressions. Recently, Bartlett et al. [2025] employed dimensional analysis to derive non-dimensional input features for logistic regression, improving the model’s generalization across different terrains and confirming the growing research interest in achieving transferability across diverse geometries.

Despite all the efforts, the problem of generalization across different catchments remains largely unsolved. Indeed, as do Lago et al. [2023] highlights, there is currently no approach capable of reliably extrapolating flood predictions to different catchments and boundary conditions without retraining.

In conclusion, improved generalization would benefit all DL applications for flood mapping Lou et al. [2025], including those supporting early warning systems, which currently represent a major driver of research interests. A natural consequence of achieving catchment-level generalization would be the possibility of extending these methods to ungauged basins through transfer learning.

1.5.2 Embedding physics

The black-box nature of DL models poses further challenges, as they are not intended to incorporate physical information, leading to nonphysical predictions and reducing their reliability for hydrological and hydraulic applications [Zeng et al., 2025].

The integration of physics into DL models can occur at multiple levels [Karniadakis et al., 2021].

At the data level (observational bias), physical knowledge can be incorporated through datasets that embody the underlying physics [Kashefi et al., 2021]. Synthetic data generation, in particular, should adhere to fundamental physical principles. However, the mere use of data generated by physically based numerical models should not be considered true physics embedding.

At the architecture level (inductive bias), prior physical assumptions can be encoded into the DL architecture [Bronstein et al., 2017], for instance, by enforcing symmetries [Cohen et al., 2019].

At the loss function level (learning bias), physics can be introduced through the adoption of modified loss functions that guide training toward physically-consistent solutions [Lagaris et al., 1998]. It is typically done using soft penalty terms [Raissi et al., 2019], which allow the model to approximate, rather than exactly satisfy, these laws. In hydraulic applications, such penalties may enforce mass or energy conservation or evaluate the consistency of predicted fields.

The PINN approach, although promising for solving physical problems through DL by introducing learning biases, appears unsuitable for river flood mapping applications. Indeed, epistemic uncertainties, such as the incomplete and/or imprecise comprehension of physical processes, plague the field of hydraulics and hydrology, so that the computation of PDE residuals can encounter challenges similar to those faced by classical numerical methods. Indeed, it is well known that the classical models, i.e., numerical solutions of SW equations, are often used in an equifinality framework, meaning that an effective parameterization compensates for the absence of explicit modeling of specific phenomena [Beven, 2006].

So far, studies on PINNs have mostly focused on benchmark problems and remain distant from realistic applications in flood mapping. Given the aforementioned limitations, this line of research does not appear to offer a promising route for embedding physics into data-driven flood prediction models.

Finally, it is important to highlight that the integration of physical knowledge is closely linked to the need for developing generalizable and transferable models in river flood mapping. Indeed, a physics-aware model learns not only from the available data but also from underlying physical principles, enabling it to generalize more effectively to unseen conditions that obey the same physical laws. Nonetheless, the incorporation of physics may be further impeded by additional factors. For example, do Lago et al. [2023] highlighted that the use of patch-based methodologies has hindered the integration of physics-based constraints, as these typically rely on global quantities defined over larger spatial domains.

1.5.3 Probabilistic hazard assessment

Flood hazard is inherently probabilistic, as it results from multiple stochastic factors, including rainfall events [Cole et al., 2006] and boundary conditions [Domeneghetti et al., 2013]. Consequently, flood mapping models should be able to associate each generated map with a return period, even if in a simplified manner—for instance, by linking the map to the return period of the input, such as rainfall intensity or discharge.

Moreover, rapid predictions enable the exploration of uncertainty across different scenarios. The stochastic nature of DL training, arising from random initialization, data shuffling, and other sources [Gawlikowski et al., 2023], naturally introduces variability in the model outputs. This variability can be leveraged to quantify predictive uncertainty and support probabilistic flood risk assessment.

A promising research direction in flood mapping is the integration of probabilistic frameworks into DL architectures, with the aim of explicitly quantifying prediction uncertainties and generating probabilistic flood hazard maps. In this context, Benvivoglio et al. [2022] highlighted the potential of deep Gaussian processes to account for uncertainties arising from both the model and its inputs. Practical applications included the use of Gaussian process models as efficient emulators of numerical methods [Donnelly et al., 2022] and their application to estimate uncertainty in water depth predictions [Kohanpur et al., 2023].

Overall, advancing probabilistic hazard and risk assessment in data-driven flood mapping requires not only methodological improvements in uncertainty quantification but also a stronger integration with hydrological and hydraulic knowledge, ensuring that uncertainty estimates remain physically meaningful and useful for operational applications.

Chapter 2

Deep Learning models

The aim of this chapter is to present the theoretical fundamentals of the most widely used Deep Learning architectures, thereby providing the background required to contextualize their application within this work.

2.1 Introduction

DL has become a cornerstone of AI, providing powerful tools for the analysis and interpretation of large-scale and complex datasets. Unlike conventional ML approaches, which typically rely on extensive feature engineering, DL models are capable of automatically learning multi-level representations directly from raw data.

The automatic identification of features offers several advantages: hand-designed approaches are time-consuming and resource-intensive, whereas deep models allow for greater flexibility by adapting the training process to new tasks without the need to redesign the entire data processing pipeline.

Owing to these advantages, DL has achieved remarkable progress in fields such as computer vision, natural language processing, and sequential data modeling, and it is increasingly being applied to environmental modeling and hydrological sciences, as discussed in Section 1.

Following a concise overview of the fundamental principles of ML (Section 2.2), the chapter introduces the basic concepts of Neural Networks and Multi-Layer Perceptrons (Section 2.3). It then reviews the DL architectures most frequently discussed in

the literature—some of which are also adopted in the following chapters—namely: Convolutional Neural Networks (Section 2.4), Recurrent Neural Networks (Section 2.5), Autoencoders (Section 2.6), CNN encoder–decoder models (Section 2.7), and Transformers (Section 2.8).

Section 2.9 provides an overview of approaches for exploiting physical information in DL models. It first introduces PINNs (Section 2.9.1), including an application to complex fluid simulations, and then presents the proposed PAT strategy (Section 2.9.2), which will be employed in the subsequent chapters.

2.2 Fundamentals of Machine Learning

ML techniques have been developed to automatically acquire knowledge from data by extracting meaningful patterns. ML algorithms are able to learn from data, i.e., to improve their performance on a specific task through experience [Mitchell, 1997].

It is important to note that as the number of input variables increases, the number of potential configurations grows exponentially, a phenomenon known as the *curse of dimensionality* [Bellman, 1961]. This effect can lead to sparse data distributions and increased difficulty for models to generalize, emphasizing the need for careful feature selection or dimensionality reduction techniques.

2.2.1 ML Paradigms

ML approaches are generally categorized into supervised, unsupervised, and reinforcement learning.

Supervised Learning

In supervised learning, models are trained on labeled datasets, where each input sample x is associated with a known target output $\hat{y}$. The objective is to learn the best set of trainable parameters Ξ to establish a mapping F_{Ξ} from inputs x to outputs y :

$$F_{\Xi} : \mathbb{R}^n \rightarrow \mathcal{Y}, \quad x \mapsto y. \quad (2.1)$$

Typical supervised learning tasks include regression and classification.

-
- In regression problems, the goal is to predict continuous values, with

$$\mathcal{Y} = \mathbb{R}^m, \quad (2.2)$$

where m is the dimensionality of the continuous output.

- In classification problems, the aim is to assign discrete labels to input data, with

$$\mathcal{Y} = \{1, 2, \dots, k\}, \quad (2.3)$$

where k denotes the number of possible classes.

The model is able to perform these tasks by minimizing a suitable loss function $\mathcal{L}(\mathbf{y}, \hat{\mathbf{y}})$, where $\mathbf{y} = F_{\Xi}(\mathbf{x})$ is the model prediction and $\hat{\mathbf{y}}$ is the ground truth. Here, $\mathbf{x}$, $\mathbf{y}$, and $\hat{\mathbf{y}}$ can be vectors or, more generally, tensors. For regression, common error metrics are employed in the loss function, while for classification, cross-entropy loss is typically used.

Unsupervised Learning

In unsupervised learning, datasets lack explicit labels, and the goal is to uncover inherent structures, patterns, or groupings within the data. These tasks are often used for exploratory data analysis or feature extraction. The mapping can take different forms depending on the task. For instance, in clustering tasks, the model assigns a latent group, i.e., a group not explicitly given in the data, c , to each input x ,

$$F_{\Xi} : \mathbb{R}^n \rightarrow \{1, 2, \dots, k\}, \quad x \mapsto c, \quad (2.4)$$

trying to minimize an objective that encourages points in the same cluster to be similar and points in different clusters to be dissimilar (e.g., within-cluster variance).

In dimensionality reduction, the aim is to project the data onto a lower-dimensional latent space,

$$F_{\Xi} : \mathbb{R}^n \rightarrow \mathbb{R}^m, \quad m \ll n, \quad (2.5)$$

by minimizing a reconstruction or information-preservation loss, such as in Principal Component Analysis (PCA), so that the low-dimensional representation captures the most relevant structure of the data.

Reinforcement Learning

Reinforcement learning focuses on learning optimal decision-making policies through interactions with an environment. Unlike the previous two paradigms, it addresses sequential decision-making problems, where the action taken depends on the current state of the system and influences its future states. An agent receives feedback in the form of rewards or penalties and iteratively adjusts its actions to maximize cumulative reward.

2.2.2 Dataset Partitioning

A fundamental step in any ML workflow is the careful partitioning of the dataset into training, validation, and test sets. Unlike traditional hydrological modeling, ML workflows include an additional dataset, the validation set, alongside the usual training and test sets.

In ML models, there are two types of parameters: trainable parameters, previously denoted by Ξ , which are learned (i.e., tuned) during the training phase to minimize the loss function, and hyperparameters, which must be set prior to training, remain fixed throughout the process, and are not learned from data.

The training set is used to fit the model parameters Ξ ; the validation set is used for hyperparameter tuning and model selection, whereas the test set provides an assessment of the model's performance on unseen data. Proper partitioning is essential to ensure that the model generalizes effectively beyond the examples on which it was trained. The absence of a validation set could lead to *peaking*, a phenomenon in which hyperparameters are over-optimized on the test set, resulting in an overestimation of model performance and possible poor generalization to truly unseen data.

Traditional hydrological models do not include a dedicated validation set, and model evaluation is usually carried out by means of a calibration and test phase. Moreover, in hydrological applications, defining a separate validation set is often challenging. Available datasets are typically limited in size, and partitioning them into training, validation, and test sets may leave insufficient information for training or result in validation sets that are not representative of unseen conditions.

2.2.3 Overfitting and Underfitting

The presence of a validation set also helps monitor and prevent common issues such as overfitting and underfitting. Overfitting occurs when a model is complex enough to learn noise in the training data, achieving low training error but performing poorly on validation and test sets. Underfitting, on the other hand, arises when a model is too simple to capture the underlying patterns in the data, leading to poor performance on both the training and validation sets.

Figures 2.1 and 2.2 illustrate the concepts of overfitting and underfitting in the simple context of polynomial regression. The synthetic data were generated by sampling points along an interval and computing target values as a sinusoidal function with added Gaussian noise to mimic natural variability. The former shows three representative fits, varying the polynomial degree. The latter depicts a performance metric (the Mean Absolute Error, MAE) on both training and validation datasets as a function of polynomial degree. The discrepancy between training and validation errors provides a clear visual indication of the onset and extent of overfitting for models with higher polynomial degrees.

Model performance strongly depends on the quality and structure of the training dataset, which should be sufficiently large, diverse, and representative of the possible input states. At the same time, model complexity, in terms of the number of trainable parameters, should be properly adjusted by exploiting performance on the validation set. While underfitting is nowadays more of a theoretical concern since models with millions of parameters can be efficiently trained in a short time using parallel computation, overfitting remains a concrete risk in hydrological data-driven applications due to both the scarcity of data and the intrinsic complexity of hydrological processes.

2.3 Neural Networks

Neural Networks (NNs) are predictive ML models, originally inspired by the architecture of nerve cells [McCulloch and Pitts, 1943]. A single element of the network, called a neuron, receives a series of I inputs x_i , calculates a weighted sum of these

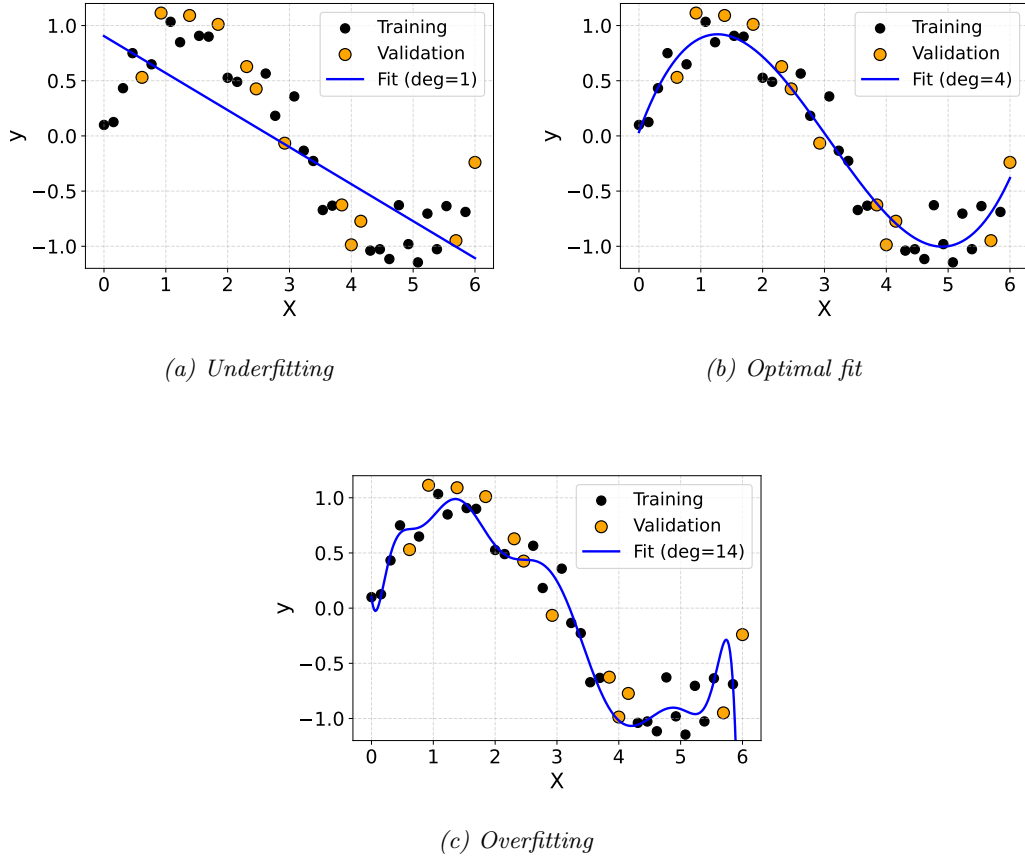


Figure 2.1: Representative examples of polynomial regression fits on synthetic data. Training samples are depicted in black, while validation samples are shown in orange. Panels (a), (b), and (c) show underfitting, optimal fitting, and overfitting, respectively.

inputs (which may include a bias value c), and then applies an activation function σ to produce the neuron's output a .

$$a = \sigma\left(\sum_{i=1}^I w_i x_i + c\right) \quad (2.6)$$

A single neuron is able to reproduce basic logical gates such as AND, OR, and NOT by employing a step (threshold) activation function:

$$st(x) = \begin{cases} 1, & x \geq 0 \\ 0, & x < 0 \end{cases} \quad (2.7)$$

and properly choosing the weights and bias (Table 2.1). These gates are linearly separable, which allows a single neuron to implement them correctly. In fact, the

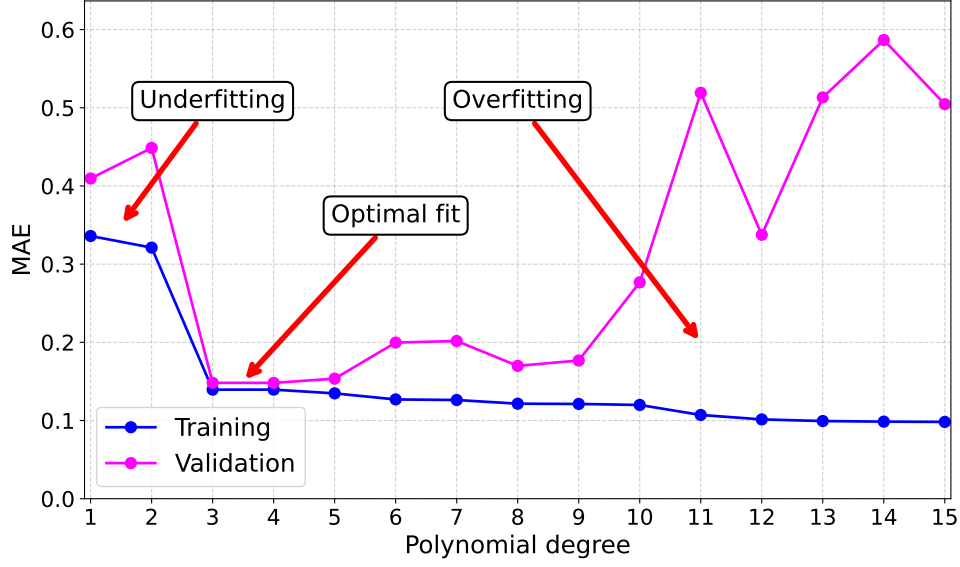


Figure 2.2: Learning curve illustrating the MAE for training and validation datasets across polynomial degrees from 1 to 15. The discrepancy between training and validation errors provides a clear indication of overfitting at higher polynomial degrees.

neuron outputs 1 only if the argument of the activation function $z = \sum_{i=1}^I w_i x_i + c$ is positive, and therefore, if the input lies on a specific side of the hyperplane defined by

$$\sum_{i=1}^I w_i x_i + c = 0.$$

In contrast, the XOR gate is not linearly separable, meaning that it cannot be represented by a single neuron and requires a multi-layer network to be implemented (Table 2.2).

A network where all inputs are directly connected to all outputs is called a perceptron. This can easily represent simple Boolean functions; however, it remains a linear separator, since it can only divide the input space with a single hyperplane.

To represent non-linearly separable Boolean functions, such as XOR, a shallow NN can be built by introducing a hidden layer (Table 2.2). For instance, the outputs a_1 and a_2 of two hidden neurons are combined with two weights w_1 and w_2 , along with a bias c ; then the activation function σ is applied to obtain the final output y :

$$y = \sigma \left(w_1 a_1 + w_2 a_2 + c \right) \quad (2.8)$$

Table 2.1: Weights, bias, and output for basic logical gates using a single neuron, with a step activation function st

Gate	w_1	w_2	c	Inputs (x_1, x_2)	$z = w_1x_1 + w_2x_2 + c$	$a = st(z)$
AND	1	1	-1.5	(0,0)	-1.5	0
				(0,1)	-0.5	0
				(1,0)	-0.5	0
				(1,1)	0.5	1
OR	1	1	-0.5	(0,0)	-0.5	0
				(0,1)	0.5	1
				(1,0)	0.5	1
				(1,1)	1.5	1
NOT	-1	-	0.5	0	0.5	1
				1	-0.5	0

Table 2.2: XOR using a shallow NN with hidden neurons z_1 (AND) and z_2 (OR) and obtaining the output as in Equation 2.8, with a step activation function ($\sigma = st$).

Gate	Inputs (x_1, x_2)	a_1 (AND)	a_2 (OR)	w_1	w_2	c	y
XOR	(0,0)	0	0	-1	1	-0.5	0
	(0,1)	0	1				1
	(1,0)	0	1				1
	(1,1)	1	1				0

Therefore, including a hidden layer enables the model to capture and learn more complex patterns in the data.

2.3.1 Activation functions

Activation functions are nonlinear to capture the non-linearity in the data. In practical NN training, the step function (2.7) is rarely used as an activation function due to its discontinuity. Instead, continuous activation functions—like Hyperbolic

Tangent, Rectified Linear Unit (ReLU), and Leaky ReLU Apicella et al. [2021]—are preferred. The choice of the activation function significantly affects the model’s performance.

The Sigmoid function maps its input to a value between 0 and 1, which can be interpreted as a probability. Its derivative is smooth and differentiable, but it suffers from the vanishing gradient problem for large positive or negative inputs. The Hyperbolic Tangent (Tanh) function provides stronger gradients than the sigmoid and maps values to the range $(-1, 1)$, but it can still experience vanishing gradients for large input magnitudes.

The Rectified Linear Unit (ReLU) mitigates the vanishing gradient problem and allows for faster training, but it can suffer from the dying ReLU issue, where neurons become inactive if they always output zero. The Leaky ReLU addresses this by introducing a small slope α for negative inputs, allowing gradients to flow even for negative values.

A summary of the most widely used activation functions is presented in Table 2.3

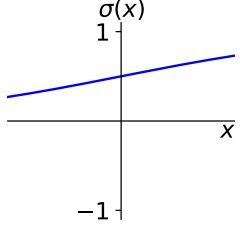
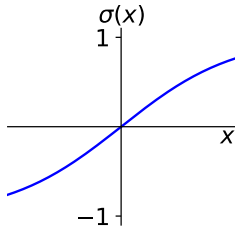
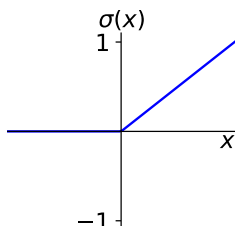
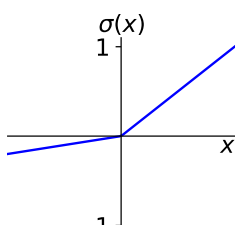
2.3.2 Going Deep: Multi-Layer Perceptron

Deep NNs LeCun et al. [2015] are obtained by stacking multiple layers of neurons. They consist of an input layer, which receives the inputs $\mathbf{x}$, H hidden layers, each with a specified number of neurons, and an output layer, whose neurons produce the predicted outputs $\mathbf{y}$.

NNs can be broadly classified into Feed-Forward Neural Networks (FFNNs), where information flows strictly from input to output, and Recurrent Neural Networks (RNNs), which contain cycles, allowing outputs to influence future inputs (see Section 2.5).

In an FFNN the signal propagates in a single direction—from the input layer, through the hidden layers, to the output layer—without any loops. The most common type of FFNN is the Multi-Layer Perceptron (MLP), which consists of one or more hidden layers with non-linear activation functions. Each neuron is connected to all neurons in the subsequent layer, and each connection has a weight that is tuned during training. The overall function F_{Ξ} can therefore be written as a composition

Table 2.3: Most used activation functions

Activation	$\sigma(x)$	Output Range	Typical Use	Plot
Sigmoid	$\frac{1}{1+e^{-x}}$	$(0, 1)$	Output probabilities	
Hyperbolic Tangent	$\frac{e^x - e^{-x}}{e^x + e^{-x}}$	$(-1, 1)$	Hidden layers	
ReLU	$\max(0, x)$	$[0, \infty)$	Hidden layers	
Leaky ReLU	$\max(x, \alpha x)$	$(-\infty, \infty)$	Hidden layers	

of layer-wise transformations f_h , ($h = 1, \dots, H$), each with its own independent parameters Ξ_h :

$$F_{\Xi}(\mathbf{x}) = f_H(\cdot; \Xi_H) \circ f_{H-1}(\cdot; \Xi_{H-1}) \circ \dots \circ f_1(\mathbf{x}; \Xi_1), \quad (2.9)$$

where $\Xi = \{\Xi_1, \dots, \Xi_H\}$ represents the full set of trainable parameters of the network.

It is important to note that a sufficiently wide and deep MLP can approximate any continuous—and in some cases even certain discontinuous—function to arbitrary precision, given enough neurons and appropriate training. This property is formalized by the Universal Approximation Theorem Hornik et al. [1989], which states that

an FFNN with a single hidden layer and a non-polynomial continuous activation function can approximate any continuous function defined on a compact domain.

However, for a given fixed network architecture, it is generally difficult to determine exactly which functions can be represented and which cannot. In other words, while the theorem guarantees the existence of a network capable of approximating any continuous function, it does not provide guidance on how to choose the architecture to achieve the desired approximation.

2.3.3 Training Phase

During the training phase, the NN parameters are learned from data by leveraging information derived from the training set, namely the couples $(\mathbf{x}, \hat{\mathbf{y}})$. However, the training process involves not only optimizing the network parameters to minimize a chosen loss function $\mathcal{L}$, but also monitoring the ability of the model to generalize to unseen data, typically through the use of the validation set (Section 2.2.2), avoiding overfitting.

The true underlying mapping to be represented by the NN can be expressed as an unknown non-linear function F , sampled by the available input data $\mathbf{x}$ and the corresponding target output $\hat{\mathbf{y}}$:

$$F(\mathbf{x}) = \hat{\mathbf{y}} \quad (2.10)$$

By adjusting the parameter vector Ξ , which includes all weights and biases, the NN learns an approximate function F_{Ξ} that mimics F :

$$F_{\Xi}(\mathbf{x}) = \mathbf{y} \quad (2.11)$$

In the training process, the network refines its weights using an optimization algorithm to minimize a loss function $\mathcal{L}$, which quantifies how much the predicted output $\mathbf{y}$ differs from the target output $\hat{\mathbf{y}}$:

$$\arg \min_{\Xi} \mathcal{L}(\mathbf{y}, \hat{\mathbf{y}}) \quad (2.12)$$

A common choice for the loss function in a data-driven model for a regression problem (where the output variables are real and unbounded) is the Mean Squared Error (MSE) between the predicted values y_i and the corresponding targets $\hat{y}_i$, with

$i = 1, \dots, N$ and N the size of the training dataset:

$$\text{MSE} = \frac{1}{N} \sum_{i=1}^N (y_i - \hat{y}_i)^2 \quad (2.13)$$

If the problem involves multiple output variables (e.g., $\hat{\mathbf{y}}_i \in \mathbb{R}^M$), the MSE is generalized by averaging across both the dataset size N and the output dimensionality C :

$$\text{MSE} = \frac{1}{NC} \sum_{i=1}^N \|\mathbf{y}_i - \hat{\mathbf{y}}_i\|_2^2 \quad (2.14)$$

Other loss functions can also be used, depending on the nature of the data and the goals of the model.

Optimization

The process of optimization is not straightforward. It is enough to note that multiple distinct choices for the parameter vector Ξ can lead to the same input-output mapping, for instance, due to weight permutations or sign flips [Chen et al., 1993].

For simplicity, let us consider a regression problem with a single target variable $\hat{y}$ and predictions $y = F(\mathbf{x}; \Xi)$, where $\mathbf{x}$ denotes the inputs and Ξ the model parameters.

Viewing the error as a surface over weight space, the optimization goal corresponds to finding the lowest point of this surface, i.e., the global minimum where $\mathcal{L}(\Xi)$ attains its smallest value. However, because of the nonlinear nature of $F(\mathbf{x}; \Xi)$, the error function $\mathcal{L}(\Xi)$ is typically non-convex (Figure 2.3). This fact implies that optimization may converge to local minima or saddle points rather than the global minimum.

Optimization algorithms proceed by choosing an initial guess $\Xi^{(0)}$ and iteratively updating the parameters from the values $\Xi^{(old)}$ to the values $\Xi^{(new)}$. The gradient indicates the direction of steepest ascent of the error function, and gradient descent follows the opposite direction to optimize the parameters. Therefore, the updates are usually based on the gradient of the loss with respect to Ξ :

$$\Xi^{(new)} = \Xi^{(old)} - \lambda \nabla_{\Xi} \mathcal{L}(\Xi^{(old)}) \quad (2.15)$$

where $\lambda > 0$ is the learning rate, which controls the step size during each update. The learning rate plays a crucial role in ensuring convergence: if it is too large, the

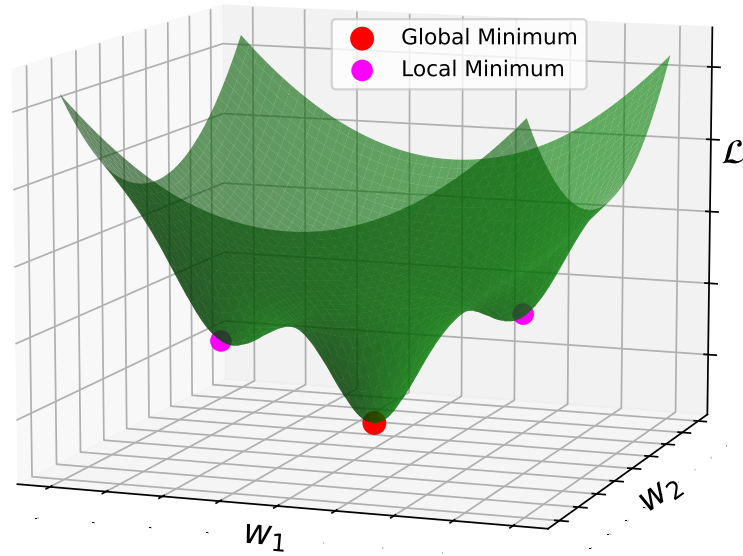


Figure 2.3: Visualization of the error surface over a bi-dimensional weight space $\Xi = (w_1, w_2)$. The global minimum (red) represents the lowest point where $\mathcal{L}(w_1, w_2)$ is minimal, while local minima (magenta) illustrate points where optimization may get stuck due to the non-convex nature of $\mathcal{L}$.

optimization may diverge; if it is too small, convergence will be excessively slow, increasing the computational time for training.

In practice, it is computationally inefficient to evaluate the gradient over the entire dataset at each iteration. Instead, the dataset is divided into smaller subsets, called mini-batches. Each update is then performed using only the examples in a mini-batch, leading to the stochastic gradient descent formulation. An epoch corresponds to one complete pass through the training dataset. Using batches introduces some noise in the gradient estimates, which can help the optimization escape from local minima.

Among the various improvements of basic gradient descent, one of the most widely adopted is the Adam (Adaptive Moment Estimation) optimizer [Kingma and Ba, 2014]. Adam adjusts the learning rate for each parameter individually using estimates of the first and second moments of the gradients, which helps achieve robust and fast convergence. It is often combined with a learning rate scheduler that reduces the global learning rate when the validation loss reaches a plateau, further improving convergence.

Backpropagation

The gradient descent algorithm requires the computation of the derivatives $\partial\mathcal{L}/\partial\Xi_k$ of the loss function with respect to every trainable parameter (weights and biases) Ξ_k in the NN. This is performed using the backpropagation algorithm Rumelhart et al. [1986], which efficiently calculates the gradients by propagating the error backward through the network using the chain rule.

In the first phase, called the *forward pass*, the output is predicted using the current weights, allowing the error to be computed. During the *backward pass*, the error is propagated to each hidden node proportionally to the strength of its connection to the subsequent layer. The resulting gradients are then used to update the weights in order to minimize the loss function. This process is repeated for all layers of the network.

The main steps of the backpropagation algorithm and weight update using the gradient descent update (Equation 2.15) are summarized here, using the following notation:

- $\mathbf{x}_i$ – input vector of the i -th training example.
- $\hat{y}_i$ – target output of the i -th training example.
- λ – learning rate used for gradient descent updates.
- σ – activation function.
- L – index of the output layer (i.e., the total number of layers).
- $\mathbf{a}^{(l)}$ – activation vector of layer l , i.e., the output of the neurons of layer l after applying the activation function σ .
- $\mathbf{z}^{(l)} = \mathbf{W}^{(l)}\mathbf{a}^{(l-1)} + \mathbf{b}^{(l)}$ – pre-activation vector of layer l
- $\mathbf{W}^{(l)}$ – weight matrix connecting layer $l - 1$ to layer l .
- $\mathbf{c}^{(l)}$ – bias vector of layer l .
- $\Delta^{(l)}$ – modified error vector of layer l .
- E – loss function computed for the current training example.

- $\odot$ – element-wise product.

Algorithm Backpropagation Algorithm and Weight Update

- 1: **Input:** Training data $(\mathbf{x}_i, \hat{y}_i)$, learning rate λ , L number of layers
- 2: **Initialize:** Trainable parameters $W^{(l)}, c^{(l)}$ for all layers $l = 1, \dots, L$
- 3: **for** each training example $(\mathbf{x}_i, \hat{y}_i)$ **do**
- 4: Set input layer: $a^{(0)} = \mathbf{x}_i$
- 5: **Forward pass:** compute activations

$$a^{(l)} = \sigma(W^{(l)}a^{(l-1)} + c^{(l)}), \quad l = 1, \dots, L$$

- 6: Compute error $E = E(a^{(L)}, \hat{y}_i)$
- 7: **Backward pass:**
- 8: Output layer: $\Delta^{(L)} = \nabla_{a^{(L)}} E \odot \sigma'(z^{(L)})$
- 9: **for** layer $l = L - 1$ down to 1 **do**
- 10: Hidden layer: $\Delta^{(l)} = (W^{(l+1)T} \Delta^{(l+1)}) \odot \sigma'(z^{(l)})$
- 11: **end for**
- 12: **Gradient computation:**

$$\frac{\partial E}{\partial W^{(l)}} = \Delta^{(l)} (a^{(l-1)})^T, \quad \frac{\partial E}{\partial c^{(l)}} = \Delta^{(l)}$$

- 13: **Weights and biases update:**

$$W^{(l)} \leftarrow W^{(l)} - \lambda \frac{\partial E}{\partial W^{(l)}}, \quad c^{(l)} \leftarrow c^{(l)} - \lambda \frac{\partial E}{\partial c^{(l)}}$$

- 14: **end for**
-

Preventing Overfitting

Various techniques have been proposed to reduce the occurrence of overfitting in NNs during the training phase.

Classical regularization approaches include the introduction of an additional term in the loss function. Returning to simple linear regression, one can apply ℓ_1 and ℓ_2 weight penalties, which directly act on the model parameters w_i . The ℓ_1 penalty (Lasso) adds a term proportional to $\sum_i |w_i|$, promoting sparsity by driving some weights exactly to zero. The ℓ_2 penalty (Ridge) adds a term proportional to $\sum_i w_i^2$

to the loss function, discouraging large weights; indeed, in linear regression, the presence of very large weights is a clear sign of overfitting.

For NNs, additional methods have been proposed, including *dropout*, which randomly deactivates a fraction of neurons during training to encourage more robust representations.

Early stopping halts the training process once the validation loss no longer improves. Typically, both training and validation losses decrease during the initial epochs, indicating that the model is effectively learning from the data. After a certain point, however, the validation loss may plateau or even increase, signaling the onset of overfitting. To avoid this, early stopping terminates training at the point of minimum validation loss, thereby helping to prevent overfitting and promote better generalization. *Early stopping* does not interrupt training immediately at the first increase in the validation loss. Instead, it waits for a predefined number of epochs without improvement, known as *patience*, after which training is stopped and the best-performing set of parameters is restored. The typical behavior of training and validation losses—together with the indications of early stopping and patience—is illustrated in Figure 2.4.

2.4 Convolutional Neural Networks

Fully connected layers are not well suited for large inputs, as the number of parameters grows rapidly with the size of the network. Moreover, they discard spatial information, which can be crucial for many tasks. Indeed, in MLPs, neurons are activated only if a specific pattern appears in a particular region. However, translation invariance is often desired — that is, the ability to recognize a given pattern independently of its spatial location. For instance, in image recognition, detecting a specific pattern or object anywhere in the pixel matrix is essential. Similarly, in flood mapping, the spatial characteristics of the terrain—such as topographical patterns, slope, and elevation—determine the extent and behavior of flooding.

Convolutional layers address these limitations by applying the same trainable kernel across different regions of the input to detect local patterns. The kernel slides over the input (Figure 2.5), capturing relevant spatial structures while keeping the

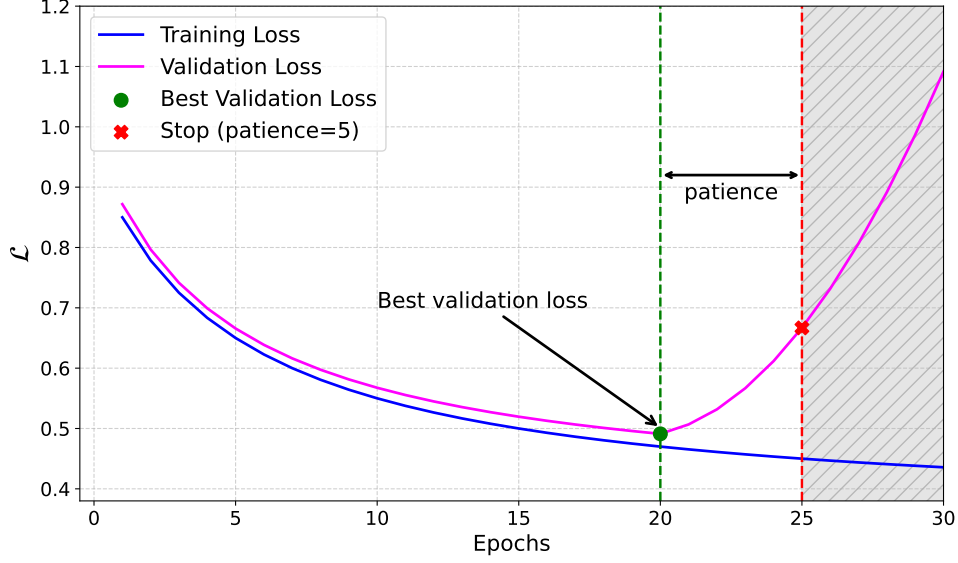


Figure 2.4: Illustration of training and validation losses over epochs. The validation loss initially decreases along with the training loss but eventually starts increasing, indicating potential overfitting. Early stopping halts training after a specified patience and the best performing parameters are recovered.

number of parameters manageable. The reuse of filters across the input dramatically reduces the number of trainable parameters and ensures translational symmetry, allowing the network to recognize features in different positions.

In a convolutional layer, a filter is formed by stacking as many kernels as the input depth (i.e. channels), and the number of filters determines the depth of the output. The set of activations produced by a single filter forms a feature map, and multiple filters in the same layer produce distinct feature maps.

Mathematically, let $X \in \mathbb{R}^{H \times W \times C}$ denote an input with height H , width W , and C channels, and let $W^{(f)} \in \mathbb{R}^{k_H \times k_W \times C}$ be the weights of filter f , where k_H and k_W denote the height and width of the filter kernel, with bias $c^{(f)} \in \mathbb{R}$. The discrete convolution is defined as:

$$Y^{(f)}(i, j) = \sigma \left(\sum_{c=1}^C \sum_{m=1}^{k_H} \sum_{n=1}^{k_W} W_{m,n,c}^{(f)} X_{i+m-1, j+n-1, c} + c^{(f)} \right), \quad (2.16)$$

where $Y^{(f)}$ is the feature map produced by filter f .

A key concept is the receptive field, which defines the region of the input that contributes to a neuron's activation in the subsequent layer. Shallow convolutional

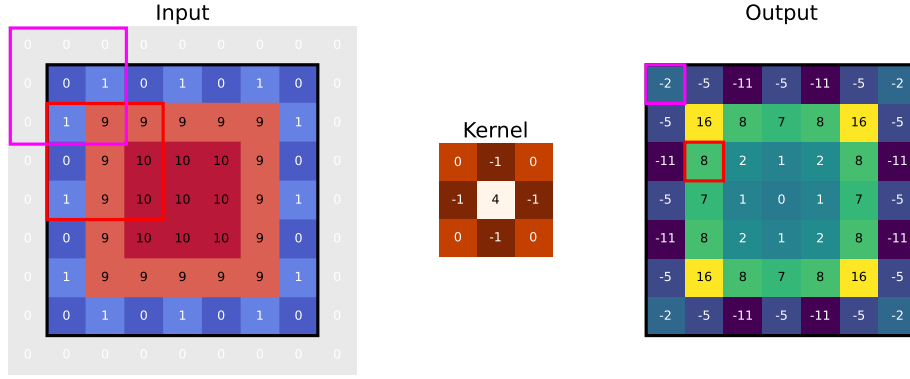


Figure 2.5: Illustration of a convolution operation. The left panel shows the input matrix with two selected 3×3 patches highlighted in red and magenta. The surrounding grey pixels represents the zeros added for padding. The center panel displays the 3×3 kernel used for the convolution (stride=1). The right panel shows the output of the convolution, with the pixels corresponding to the red and magenta input patches highlighted. The kernel is able to show in the output feature map the presence of edges (regions where pixel values change sharply) in the input.

layers typically capture simple features, such as edges or contours, following the principle of locality. During training, backpropagation adapts each filter to detect specific salient features. In contrast, a fully connected layer could only recognize a feature in a fixed location and would require distinct parameters to detect it elsewhere. Deeper layers progressively capture more complex patterns, as the receptive field grows, ultimately allowing neurons to respond to information from the entire input.

Several hyperparameters control convolutional layers:

- the kernel size ($k_H \times k_W$) defines the spatial extent of the filter;
- the stride determines how far the kernel moves at each step;
- padding can be added to control output dimensions and allow the receptive field to slide over the entire input.

Instead of tuning these parameters manually, it is usual to follow successful designs reported in the literature.

Convolutional networks typically alternate convolutional and pooling layers. Pooling operations, such as max pooling or average pooling (Figure 2.6), down-sample feature maps, retain the most relevant information, and speed up computation.

Pooling also reduces the sensitivity of convolutional layers to the exact positions of features. The main hyperparameters are pooling size, pooling stride, and the type of aggregation.

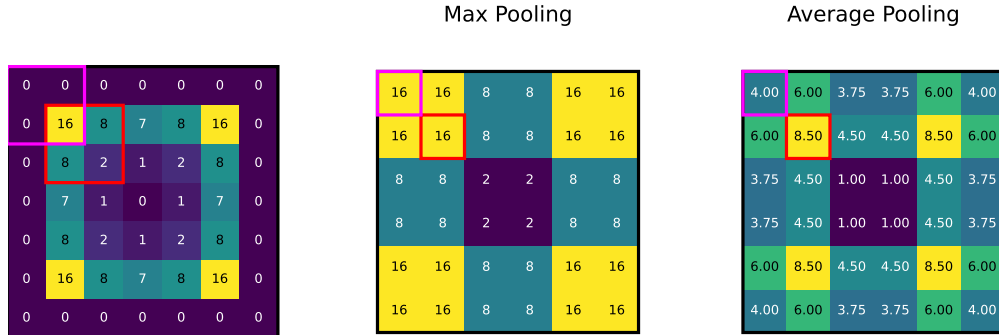


Figure 2.6: Illustration of the pooling operation. The left panel shows the feature map obtained after the convolution in Figure 2.5, followed by a ReLU activation. Two selected 2×2 regions are highlighted in red and magenta. The center panel shows the result of Max Pooling with a pooling window of size 2 and stride 1, while the right panel shows the result of Average Pooling with the same parameters.

Over the years, several CNN architectures have been proposed (Figure 2.7). The first ones were intended as classification-oriented networks for image recognition tasks.

LeNet [LeCun et al., 1989] was one of the first, featuring a shallow structure composed of two convolutional layers, two pooling layers, and three fully connected layers.

The following developments deepened the architectures, exploiting the increasing computational resources and parallel computing, introducing some modifications.

AlexNet [Krizhevsky et al., 2012] introduced a deeper network with ReLU activations, dropout, and max pooling.

VGG [Simonyan and Zisserman, 2014] employed small kernels in very deep, regular structures. Its fundamental block consists of a convolutional layer with padding, followed by a nonlinear activation such as ReLU, and a pooling layer for downsampling. These blocks are stacked multiple times and end with fully connected layers.

ResNet [He et al., 2016] introduced skip connections (Figure 2.8), which perform

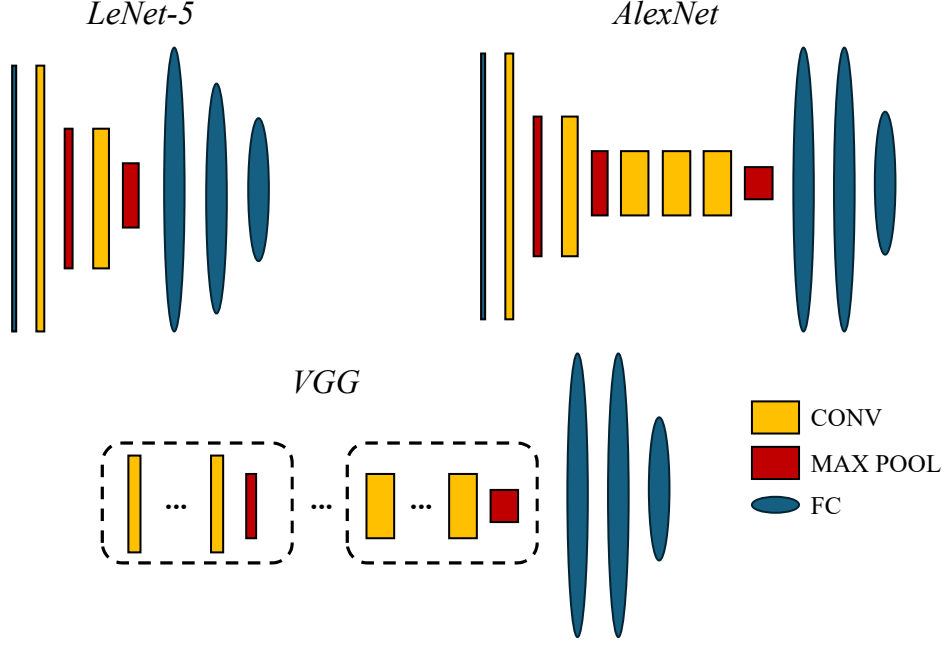


Figure 2.7: Schematic of convolutional architectures: *LeNet-5*, *AlexNet*, and *VGG*. Convolutional operations are shown in yellow, max-pooling operations in red, and fully connected layers are represented by blue ellipses.

identity mappings. In this formulation, the output y of a given layer (or block) is given by:

$$y = R(x) + x \quad (2.17)$$

that is, the sum of the input x and the residual function $R(x)$ to be learned. These connections propagate information directly to deeper layers, mitigating vanishing gradient issues in very deep networks. In fact, as the depth increases (e.g., ResNet has 152 layers), gradients tend to vanish during backpropagation, making optimization difficult.

To address this, ResNet reformulates the learning problem by allowing the layers to approximate the residual function

$$R(x) = F(x) - x,$$

where $F(x)$ denotes the desired underlying mapping. Such residual learning simplifies optimization, as it is often easier to fit the residual function $R(x)$ than the direct mapping $F(x)$. Indeed, the network only needs to make small corrections instead of

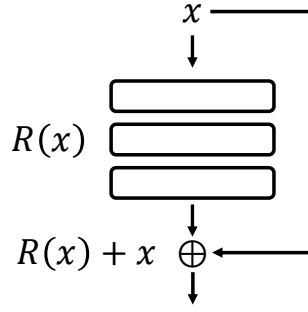


Figure 2.8: Illustration of a residual block with a three-layer skip connection

reconstructing the entire transformation, which also allows layers that could degrade performance to be effectively bypassed. Moreover, the identity connection prevents the degradation of the gradient signal without introducing additional computational complexity.

2.5 Recurrent Neural Networks and LSTM

Whereas MLPs are well suited for handling tabular data (i.e., instances $\times$ attributes), and CNNs have been specifically designed to exploit spatial invariance in image processing, sequential data requires a different modeling strategy. Recurrent Neural Networks (RNNs) address this need by processing input data in sequential form [Zhang and Xiao, 2000], such as measurements indexed in temporal order, which are common in hydrological time series.

Unlike FFNNs, RNNs have closed loops, allowing information to persist: at each step t , a simple recurrent node (commonly referred to as a cell) receives the input vector x_t and the output from the previous time step a_{t-1} , producing the output a_t . Thus, the output vector of a single cell is given by the equation:

$$a_t = \sigma(w_x^T x_t + w_a^T a_{t-1} + c) \quad (2.18)$$

Here, w_x and w_a are weight vectors, c is the bias, and the same weights are shared across all time steps. A layer is composed of multiple recurrent cells, and training is performed using the Backpropagation Through Time algorithm [Werbos, 2002]. By stacking multiple recurrent layers on top of each other, deep RNNs can be constructed, enabling the network to learn more complex temporal patterns.

The output a_t in Equation (2.18) can be generalized as a hidden state h_t , which depends on previous states (Figure 2.9). Simple RNNs often face the issue of vanishing gradients, where information from previous states diminishes rapidly after a few time steps.

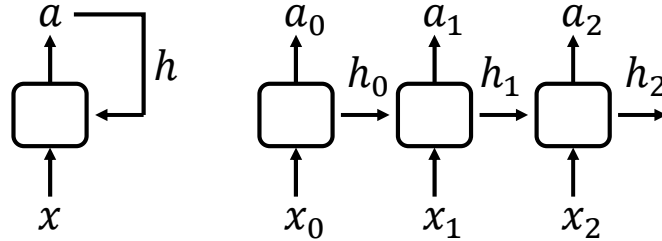


Figure 2.9: On the left, a compressed representation with a single recurrent cell; on the right, the unfolded representation across (time) steps.

To enhance cells with long-term memory for recognizing broader signal patterns, LSTM cells were introduced [Hochreiter and Schmidhuber, 1997]. The key innovation is the cell state, which acts as a memory carrying information across time steps. Within these cells, there are gate controllers that decide which inputs affect the state, whether it should reset (or forget), and how it impacts the node’s output. The *forget gate* utilizes a sigmoid function to keep or discard the information from the cell state; the *input gate* decides what new information to add to the cell state. The *cell state* is updated by combining the contributions of the forget and input gates; finally, the *output gate* determines what information to output as the hidden state. By carefully controlling these gates, whose parameters are learned during training, LSTMs can capture long-term dependencies and effectively mitigate the vanishing gradient problem of traditional RNNs. The equations governing an LSTM cell are summarized in Table 2.4 and the structure of a cell is depicted in Figure 2.10

Gated Recurrent Units [Chung et al., 2014] are a simplified formulation of LSTMs, designed to retain their effectiveness with a more compact architecture.

2.6 Autoencoders

Autoencoders are NNs designed to learn a compressed representation of input data. They are characterized by an output layer that has the same dimensionality as the

Table 2.4: Fundamental equations of an LSTM cell. Here, x_t is the input at time t ; h_{t-1} the previous hidden state; c_{t-1} the previous cell state; Φ_t, Ψ_t, O_t the forget, input, and output gates, respectively; G_t the cell input; c_t the updated cell state; and h_t the updated hidden state. Sig denotes the sigmoid activation, $\tanh(\cdot)$ the hyperbolic tangent, and $\odot$ element-wise multiplication.

LSTM Gate/State	Equation
Forget gate	$\Phi_t = Sig(W_\Phi x_t + U_\Phi h_{t-1} + b_\Phi)$
Input gate	$\Psi_t = Sig(W_\Psi x_t + U_\Psi h_{t-1} + b_\Psi)$
Cell input	$G_t = \tanh(W_G x_t + U_G h_{t-1} + b_G)$
Output gate	$O_t = Sig(W_o x_t + U_o h_{t-1} + b_o)$
Cell state update	$c_t = \Phi_t \odot c_{t-1} + \Psi_t \odot G_t$
Hidden state update	$h_t = O_t \odot \tanh(c_t)$

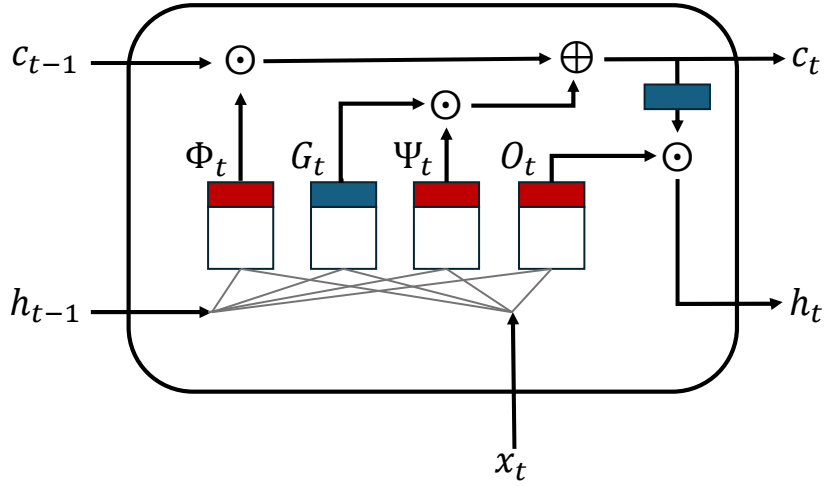


Figure 2.10: Illustration of an LSTM cell. The symbols can be referenced in Table 2.4. The boxes represent the gates, the sigmoid activations are shown in red and the hyperbolic tangent activations in blue.

input layer. The network is trained to minimize the reconstruction error between input and output, forcing the output to closely match the original input.

Effectively, the autoencoder learns which latent variables can best represent the essential information in the input, discarding redundant details.

The architecture consists of two main components (Figure 2.11):

- The *encoder* transforms the input x into a lower-dimensional latent representation $\tilde{x}$, capturing hierarchical and salient features.
- The *decoder* reconstructs the original input from the latent representation, yielding x' .

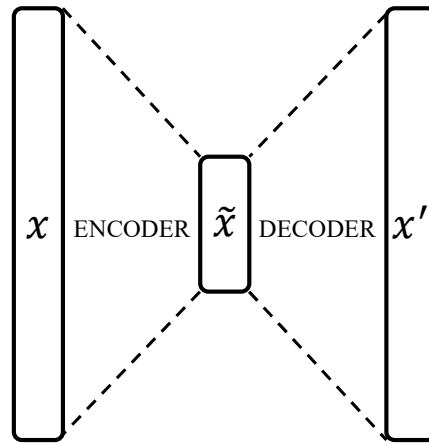


Figure 2.11: Schematic of an Autoencoder

Autoencoders can be constructed using different types of layers, from simple fully connected (dense) layers to convolutional layers, depending on the nature of the data. They illustrate the principle of learning concise and informative feature representations, which serve as the foundation for many prediction models.

2.7 CNN encoder-decoder

Unlike autoencoders, generalized encoder-decoder models are designed to produce outputs that differ from their inputs. These architectures are widely employed in

tasks such as image segmentation, where the output must preserve spatial resolution and semantic structure.

The encoder progressively extracts hierarchical features by alternating convolutional and pooling layers, thereby reducing the spatial dimensions of the input. The decoder, in contrast, replaces pooling with up-convolutional (or transpose convolution) layers, which up-sample the feature maps and reconstruct the spatial resolution. In this way, the compressed latent representation (bottleneck) is transformed back into an output of the desired size.

2.7.1 U-Net architecture

A particular focus is placed on U-Net, as it will be employed as the backbone of the two-dimensional flood hazard prediction model in Chapter 5.

The U-Net architecture [Ronneberger et al., 2015], originally proposed for biomedical image segmentation [Du et al., 2020], represents a canonical example of the convolutional encoder-decoder paradigm. It extends the basic encoder-decoder design by introducing symmetric *skip connections* between corresponding stages of the contracting (encoder) and expanding (decoder) paths. These connections transfer high-resolution feature maps directly to deeper layers, enabling the model to recover fine-grained spatial details while retaining contextual information. This mechanism also enhances data efficiency, allowing U-Net to achieve competitive performance even with relatively small datasets.

A schematic of the architecture is reported in Figure 2.12. The encoder is composed of convolutional blocks, each containing two convolutional layers with ReLU activations, doubling the number of feature channels, followed by a max-pooling operation that halves the spatial resolution. The decoder mirrors this structure: each block applies two convolutions and a transposed convolution (up-convolution) to progressively restore spatial resolution while reducing feature dimensionality. At every stage of the decoder, the output of the up-convolution is concatenated with the corresponding encoder features through skip connections, as introduced in ResNet.

Thanks to this design, U-Net has become a widely adopted architecture in a broad range of applications beyond medical imaging, including environmental monitoring

[Xu et al., 2021, Han et al., 2021], remote sensing [Shamsolmoali et al., 2019, Wang and Miao, 2022], and flood inundation mapping [Zhao et al., 2022].

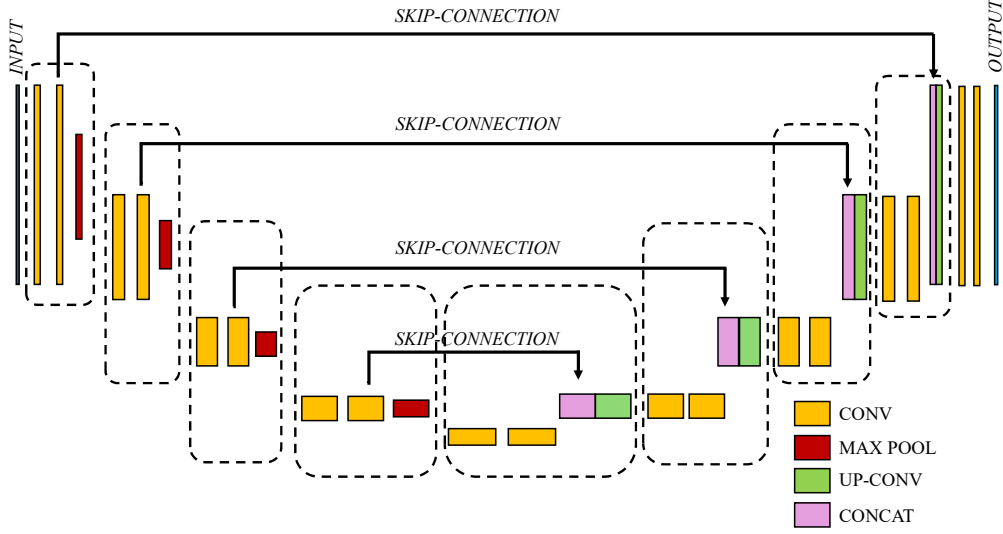


Figure 2.12: U-Net architecture. The network exhibits a symmetric encoder–decoder design organized into blocks, highlighted by dashed rectangles. Each encoder block applies two successive convolutions (yellow) followed by a max-pooling operation (red). Each decoder block mirrors this structure, performing two convolutions and an up-convolution (green). The output of the up-convolution is concatenated with the corresponding encoder feature maps (pink) via skip connections. At the end of the decoder, two additional convolutions are applied, followed by a final convolution that generates the output image with the desired number of channels.

2.8 Transformers

Transformers are a class of DL models designed to handle sequential data, such as text or time series, without relying on recurrent or convolutional structures. Their core mechanism is self-attention [Vaswani et al., 2017], which allows the model to weigh the importance of different elements in a sequence relative to each other.

The self-attention operation is defined as:

$$\text{Attention}(Q, K, V) = \text{Softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right)V, \quad (2.19)$$

where $Q \in \mathbb{R}^{N \times d_q}$ are the queries, $K \in \mathbb{R}^{N \times d_k}$ are the keys, and $V \in \mathbb{R}^{N \times d_v}$ are the

values, obtained by multiplying the input sequence X with learnable weight matrices. Here, d_q , d_k , and d_v denote the dimensionalities of the queries, keys, and values, respectively, while N represents the number of elements in the input sequence.

The Softmax function normalizes each row of the matrix of scaled dot products into a probability distribution, ensuring that attention weights are positive and sum to one:

$$\text{Softmax}(x_i) = \frac{\exp(x_i)}{\sum_{j=1}^n \exp(x_j)}, \quad i = 1, \dots, n, \quad (2.20)$$

where x_i is the i -th element of an input vector $\mathbf{x}$. This operation enables the network to assign adaptive importance to different positions in the sequence, effectively capturing long-range dependencies.

To capture multiple types of relationships simultaneously, Transformers use Multi-Head Self-Attention (MHSA):

$$\text{MHSA}(Q, K, V) = \text{Concat}(\text{head}_1, \dots, \text{head}_h)W^O, \quad (2.21)$$

with each head computed as

$$\text{head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V), \quad (2.22)$$

where W_i^Q , W_i^K , W_i^V are learnable projection matrices for the i -th head, and W^O is a final weight matrix.

In its original formulation, the Transformer architecture consists of an encoder and a decoder, each built from multiple stacked layers. Each encoder layer includes an MHSA mechanism followed by an FFNN, both equipped with residual connections and layer normalization. The decoder layers, in turn, contain three sub-layers: (i) a masked MHSA, where an upper-triangular attention mask is applied within the Softmax of Equation 2.19 to prevent access to future information; (ii) a Multi-Head Cross-Attention (MHCA), which allows the decoder queries to attend to the encoder’s key–value representations; and (iii) an FFNN, each followed by residual and normalization operations. This structure enables the decoder to combine autoregressive information from the masked self-attention with the global context extracted by the encoder through cross-attention.

Transformers have achieved remarkable success in modeling long-range dependencies in sequential data, forming the foundation for modern language models and

beyond.

2.9 Physics exploitation

In this section, two ways of incorporating physical information into DL models will be discussed extensively. The first concerns PINNs, for which an original application will be presented. The second is the methodology proposed in the present work, which takes the form of a Physically Augmented Training (PAT) approach.

2.9.1 Physics-Informed Neural Networks

PINNs represent a recent paradigm in ML that combines the flexibility of NNs with principles from physics-based modeling, which has received great interest in the field of fluid dynamics [Cai et al., 2021, Cuomo et al., 2022].

The key idea is to constrain the training process of a deep NN with physical laws that govern the observed data, thereby providing a framework for tackling problems governed by physical principles while integrating observational data with mathematical models. In this manner, physics acts as a regularization mechanism, which embeds prior knowledge of governing laws during the training phase and restricts the range of admissible solutions.

PINNs are typically employed to solve supervised learning tasks [Goswami et al., 2020] involving physical systems governed by PDEs, addressing both forward problems (solving the equations) and inverse problems (inferring unknown parameters), effectively acting as neural solvers. Indeed, PINNs act as universal function approximators and are particularly valuable in cases where the problem is mathematically ill-posed, making traditional numerical approaches unsuitable. Therefore, PINNs can be considered a new class of numerical solvers for non-linear PDEs [Raissi et al., 2019].

The general form of a PDE with unknown solution $u(z)$ can be expressed in residual form as:

$$\begin{aligned}\mathcal{N}(u(z); \boldsymbol{\nu}) - f(z) &= 0, \quad z \in \Omega \subset \mathbb{R}^d, \\ \mathcal{B}(u(z)) - g(z) &= 0, \quad z \in \partial\Omega,\end{aligned}\tag{2.23}$$

where $z := [x_1, x_2, \dots, x_{d-1}, t]$ denotes the space–time coordinate vector of dimension d , and $\boldsymbol{\nu}$ are the parameters associated with the physics of the problem. Here, the term

$$\mathcal{R}(z) = \mathcal{N}(u; \boldsymbol{\nu}) - f(z) \quad (2.24)$$

is the PDE residual, where $\mathcal{N}$ is a (possibly nonlinear) differential operator and $f(z)$ represents the forcing term of the problem. Similarly,

$$\mathcal{B}(u(z)) - g(z) \quad (2.25)$$

defines the boundary residual, where $\mathcal{B}(u(z))$ is the boundary operator and $g(z)$ specifies the prescribed boundary data. When u satisfies the governing PDE and the boundary conditions, both residuals vanish throughout the domain Ω and its boundary $\partial\Omega$, respectively. In forward problems, the function $u(z)$ satisfying Equation (2.23) with specified parameters $\boldsymbol{\nu}$ must be found by the PINN, while in inverse problems, $\boldsymbol{\nu}$ must be determined from the data.

The NN approximates the $u(z)$ with the function $F_{\Xi}(z)$ using a set of trainable parameters Ξ , obtained by minimizing a modified loss function (Figure 2.13), which has the following basic structure:

$$\mathcal{L} = w_{\text{data}}\mathcal{L}_{\text{data}} + w_{\text{PDE}}\mathcal{L}_{\text{PDE}} + w_{\text{BC}}\mathcal{L}_{\text{BC}}. \quad (2.26)$$

$\mathcal{L}_{\text{data}}$ is the Data Fitting Term, which ensures that the NN predictions align with the observed data. It is computed as an error metric between the predicted and true values; the MSE, as in Equation (2.13), is often adopted.

$\mathcal{L}_{\text{PDE}}$ seamlessly integrates the physical consistency of the model into the loss function by incorporating the residual of the PDE, as in Equation (2.24).

$\mathcal{L}_{\text{BC}}$ encodes boundary conditions as soft constraints, either using Equation (2.25) or encoding known boundary data, as in the data fitting term.

The terms w_{data} , w_{PDE} , and w_{BC} are hyperparameters used to balance the relative contributions of these terms.

Automatic differentiation [Baydin et al., 2018], which generalizes the backpropagation algorithm to compute exact derivatives of the network outputs with respect to the inputs, is used in PINNs to evaluate the spatial and temporal derivatives required in the PDE residual formulation.

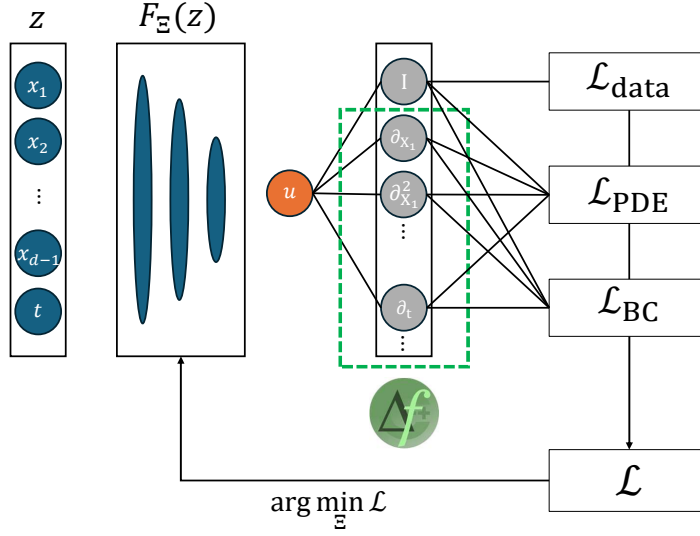


Figure 2.13: Schematic representation of a PINN. The NN approximates the unknown function $u(z)$ through $F_{\Xi}(z)$. Using Automatic Differentiation (highlighted in green), the derivatives of the predictions with respect to the inputs are computed. The total loss function consists of three components: the data-fitting term ($\mathcal{L}_{data}$), the PDE residual term ($\mathcal{L}_{PDE}$), and the boundary condition term ($\mathcal{L}_{BC}$).

The PDE residual is evaluated in a set of Collocation Points in the space-temporal domain: $\mathcal{L}_{PDE}$ is evaluated as

$$\mathcal{L}_{PDE} = \frac{\sum_{i=1}^{N_{CP}} |R_{PDE}(z_{CP,i})|^2}{N_{CP}} \quad (2.27)$$

where N_{CP} is the number of collocation points z_{CP} . These points are typically sampled within Ω using uniform, random, or more advanced sampling strategies. For instance, adaptive resampling techniques are often employed to concentrate collocation points in regions characterized by high gradients or complex dynamics.

In this sense, the nature of PINNs is inherently meshless: rather than solving the PDE on a mesh, the network ensures that the solution satisfies the physics at strategically chosen locations across the domain.

In the case of inverse problems, the NN simultaneously learns the solution field and estimates the unknown parameters by treating them as additional trainable variables. The PINN framework can ensure physically consistent estimates even when observational data are scarce or noisy.

PINNs are particularly advantageous in seamlessly integrating empirical data into models Raissi et al. [2019], a capability that traditional numerical methods lack. Nonetheless, PINNs remain inherently agnostic to boundary conditions; consequently, achieving good generalization under new boundary conditions typically requires training a new model [Donnelly et al., 2024].

Benchmark problems

An example of a PINN application is given by Jagtap et al. [2020], which proposed a PINN for various test cases, e.g., the lid-driven cavity test.

Rarefied Gas Dynamics in Cylinder Arrays

The work by Tucny et al. [2025] provides an illustrative example of the application of PINNs to complex fluid simulations. In this study, a PINN was trained using a limited number of high-fidelity numerical simulations of rarefied gas microflows to predict macroscopic flow variables such as velocity, pressure, and stress tensor in steady-state, laminar, and weakly compressible flow through a two-dimensional array of cylinders (Figure 2.14).

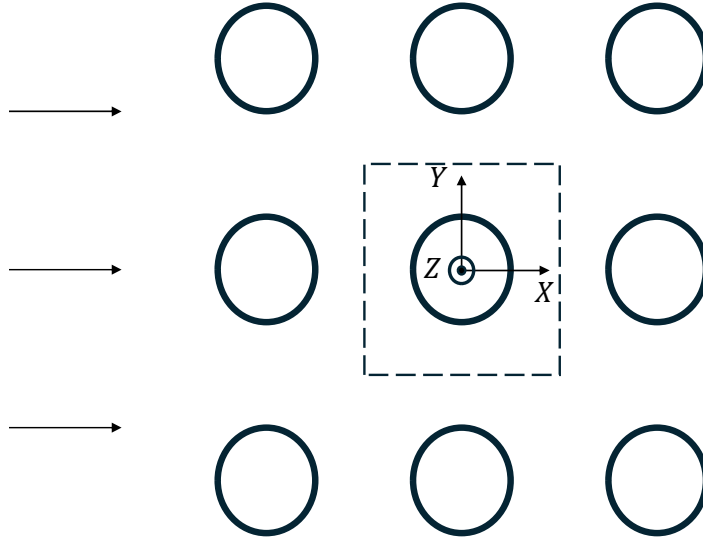


Figure 2.14: Schematic representation of the domain of the rarefied gas flow through a periodic cylinder array

Rarefied gas flows occur when the molecular mean free path, λ_m , is not negligible

compared to the characteristic length of the flow, L_c . The degree of rarefaction is characterized by the Knudsen number, defined as $\text{Kn} = \lambda_m/L_c$. The simulations used for the training were conducted in the transition regime ($0.1 < \text{Kn} < 3$).

A particularly interesting feature of this study is the structure of the loss function, which combines prediction errors of the velocity field $\mathbf{v}$, pressure p , and deviatoric stress tensor $\boldsymbol{\tau}$, together with PDE residuals enforcing the continuity equation,

$$\nabla \cdot \mathbf{v} = 0, \quad (2.28)$$

and the Cauchy momentum equation under the steady-state assumption

$$-\nabla p + \nabla \cdot \boldsymbol{\tau} - \nabla \cdot (\rho \mathbf{v} \otimes \mathbf{v}) + \rho \mathbf{f} = 0, \quad (2.29)$$

where ρ is the fluid density and $\rho \mathbf{f}$ is the external driving force.

Two additional terms enforce the no-penetration boundary condition on the cylinder surfaces,

$$\mathbf{v} \cdot \mathbf{n} = 0, \quad (2.30)$$

and the periodic boundary conditions, ensuring correct flow periodicity. The latter can be imposed through an additional residual term in the loss function, as demonstrated for the incompressible NS equations in Jin et al. [2021]. However, it can also be shown that periodic boundary conditions may equivalently be enforced by introducing a truncated Fourier series layer prior to the NN [Lu et al., 2021b, Dong and Ni, 2021].

To improve the robustness of the NN and mitigate overfitting when using a limited dataset, an additional regularization term was included in the loss function. A schematic representation of the proposed architecture is shown in Figure 2.15.

To evaluate results, the error at a position $\mathbf{x}$ between PINN prediction y of a generic variable and the numerical (target) data $\hat{y}$ can be calculated as a local distance:

$$E(\mathbf{x}, Kn) = \frac{y - \hat{y}}{\max_{\mathbf{x}} \hat{y}}$$

and the Global spatial mean distance is:

$$\bar{E}(\text{Kn}) = \frac{\iint_{\mathbf{x}} E(\mathbf{x}, Kn)}{\iint_{\mathbf{x}} \mathbf{x}}$$

The PINN, with hyperparameters set as in Table 2.5, achieved an average error $\bar{E}$ below 2% for all outputs (Figure 2.16) and effectively filtered the intrinsic

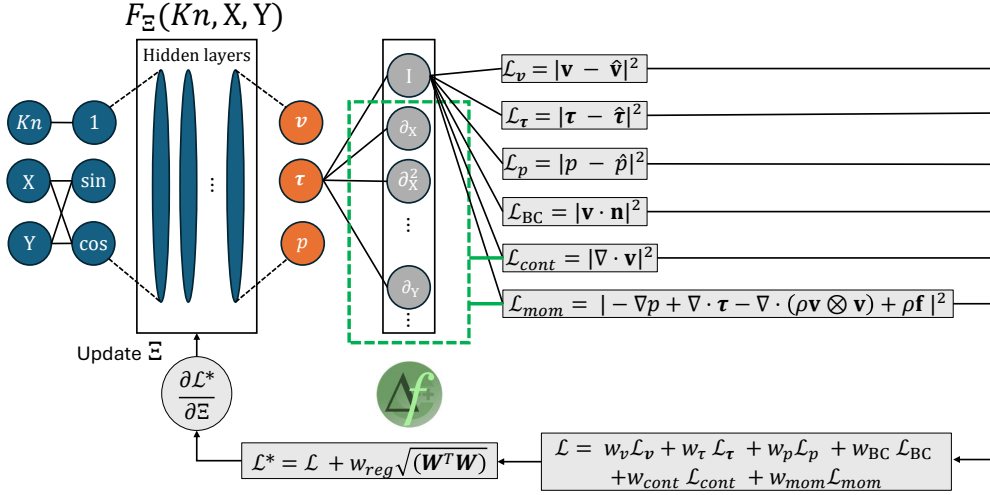


Figure 2.15: Schematic representation of the PINN architecture used for the approximation of rarefied gas flow in a cylinder array. A truncated Fourier layer (illustrated as the circles labeled $\sin$ and $\cos$) is introduced to impose hard periodic boundary conditions on the flow field. The network takes three inputs (the Knudsen number Kn and positions X , Y) and produces six outputs: the velocity components $\mathbf{v} = (v_X, v_Y)$, the deviatoric stress tensor components $\boldsymbol{\tau} = (\tau_{XX}, \tau_{YY}, \tau_{XY})$ and the pressure p . Automatic differentiation (green) is used to compute the spatial derivatives of the outputs required for evaluating the residuals of the continuity ($\mathcal{L}_{cont}$, Equation (2.28)) and momentum ($\mathcal{L}_{mom}$, Equation (2.29)) equations. $\mathcal{L}_{BC}$ enforces the non-penetration boundary condition (Equation (2.30)). The error terms $\mathcal{L}_v$, $\mathcal{L}_\tau$, and $\mathcal{L}_p$, computed using the available data $\hat{\mathbf{v}}$, $\hat{\boldsymbol{\tau}}$, and $\hat{p}$, are included in the total loss $\mathcal{L}$, each weighted by corresponding hyperparameters. The final loss function, $\mathcal{L}^*$, includes a regularization term weighted by the matrix $\mathbf{W}$ and the hyperparameter w_{reg} . Training minimizes $\mathcal{L}^*$ to update the NN parameters Ξ .

statistical noise present in the numerical simulations. The model demonstrated strong predictive capability for a flow field with $Kn = 0.7$, while its extrapolation performance deteriorated for $Kn = 5$, yet it still maintained physical consistency. Notably, each numerical simulation using Direct Simulation Monte Carlo required intensive memory and computation, taking approximately 20 hours on four GPUs, whereas training the PINN took less than two hours on a single GPU, with inference times under 2 seconds.

Table 2.5: Hyperparameters used for the PINN setup

Hyperparameter	Value
Activation function	Hyperbolic tangent
Weight initialization	Glorot normal [Glorot and Bengio, 2010]
Optimizer	Adam [Kingma and Ba, 2014]
Regularization hyperparameter	$w_{\text{reg}} = 0.001$
Error terms	MSE (Equation (2.13))
Weights of PDE residuals	$w_{\text{cont}} = w_{\text{mom}} = 0.1$
Weights on data	$w_v = w_\tau = w_p = 1$
Weights on BC	$w_{BC} = 0.1$
Number of hidden layers	10
Number of neurons per layer	100
Number of Fourier nodes	3

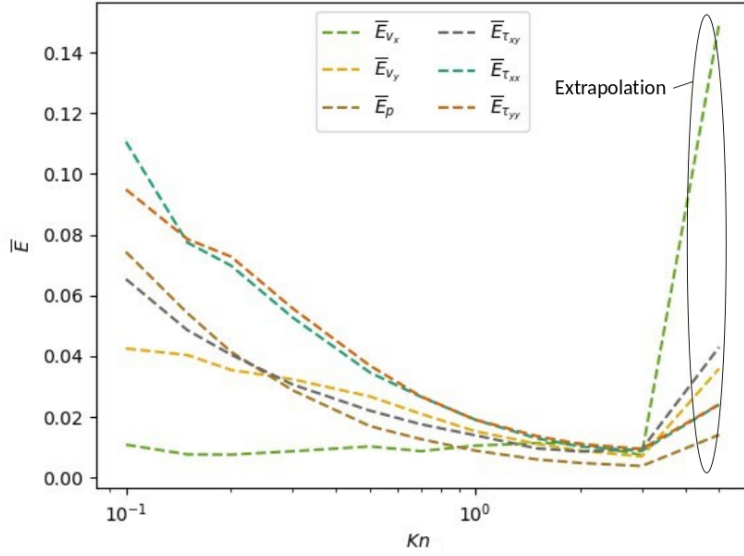


Figure 2.16: Global spatial mean distance as a function of the Knudsen number ([Tucny et al., 2025])

2.9.2 Proposed Approach: Physically Augmented Training

The previous overview in Chapter 1 highlighted a growing research interest in incorporating physical knowledge into learning frameworks. In the current chapter, the most common DL architectures have been presented, together with the paradigm

of PINNs.

Here, a novel methodology—referred to as Physically Augmented Training (PAT)—is proposed. Its objective is to integrate physical information, not through the explicit solution of PDEs, nor as a substitute for numerical models, but within existing data-driven models by modifying their training procedures.

Drawing inspiration from the PINN framework, the proposed approach enriches the loss function with an additional term that encodes relevant physical knowledge. Specifically, the total loss function is defined as a weighted combination of a data-driven loss ($\mathcal{L}_{DD}$) and a physics-based regularization term ($\mathcal{L}_P$):

$$\mathcal{L} = \lambda \cdot \mathcal{L}_{DD}(\mathbf{y}, \hat{\mathbf{y}}) + (1 - \lambda) \cdot \mathcal{L}_P(\mathbf{x}, \mathbf{y}, \hat{\mathbf{y}}), \quad (2.31)$$

where λ is a weighting factor that balances the data-driven component and the physics-based regularization term in the total loss.

In ML, regularization techniques are commonly used to limit the growth of model weights during training, mitigating potential overfitting effects [Ng, 2004]. Whereas the term $\mathcal{L}_{DD}$ still depends solely on the predicted outputs $\mathbf{y}$ and the true outputs $\hat{\mathbf{y}}$, the physical term $\mathcal{L}_P$ is now a metric that can also involve physical inputs $\mathbf{x}$. Various physical principles can thus be encoded into this term, enabling the model to capture a broader spectrum of physical behaviors and relationships hidden in the data, without relying on the explicit residuals of differential equations. This feature can be a significant advantage in applications where the conceptual model of the phenomenon is partially or entirely unknown [Qian et al., 2019], thereby bypassing limitations related to epistemic uncertainty.

From a broader perspective, this methodology shares similarities with data augmentation techniques [Shorten and Khoshgoftaar, 2019, Maharana et al., 2022, CM, 2023, Baydaroglu and Demir, 2024]. Data augmentation involves artificially expanding the training dataset by applying various transformations to the input data. The objective is to improve the model’s performance by exposing it to a more diverse set of examples, thereby enhancing generalization and robustness. Analogously, this approach allows for an enrichment of the informational content of the dataset by combining both inputs and outputs to formulate new physically-based quantities.

In this sense, it can be viewed as a form of physically-informed data augmentation, which aims to improve both prediction accuracy and generalization by embedding physical knowledge directly in the training process of data-driven applications.

An example of this concept was given by [Zhang et al., 2022], which introduced the divergence-free condition of incompressible flow in the loss function, showing that the use of this constraint could improve the accuracy of the CNN predictions for the turbulence statistics of the flood flow with wall-mounted bridge piers. A further example is offered by [Donnelly et al., 2024], who introduced a penalty term to discourage predictions that violate mass conservation.

Chapter 3

The Motion of a Pendulum as predicted by DL approaches

The physical system of a simple pendulum is used here to illustrate two different approaches that incorporate physical information into Deep Learning models to improve their application to physical systems.

3.1 Pendulum physics

The intersection between computational methods and DL for modeling complex physical systems represents a cutting-edge research field [Raymond and Camarillo, 2021]. However, simply fitting data is often insufficient to achieve accurate predictions in physical systems [Greydanus et al., 2019].

A simple yet illustrative problem in physics is predicting the motion of a pendulum in the regime of large oscillations. Its dynamics are described by the classical equation of motion:

$$\frac{d^2\theta}{dt^2} + \frac{g}{L} \sin(\theta) = 0 \quad (3.1)$$

where $\theta(t)$ is the angular displacement, g is the gravitational acceleration, and L is the length of the pendulum.

The total mechanical energy of the pendulum is given by

$$E_n = \frac{1}{2}mL^2\omega^2 + mgL(1 - \cos \theta), \quad (3.2)$$

where the first term represents the kinetic energy ($\omega = \frac{d\theta}{dt}$ is the angular velocity) and the second term corresponds to the gravitational potential energy (m is the mass of the material point), which vanishes at the equilibrium position ($\theta = 0$).

According to the principle of conservation of energy, the quantity En remains constant if no external force acts on the system:

$$\frac{dEn}{dt} = 0 \quad (3.3)$$

3.2 Dataset Creation

The dataset is synthetically generated by numerically solving equation (3.1), yielding the time evolution of the angle θ and the angular velocity ω .

The size and quality of the dataset influence the learning performance of a data-driven model that aims to represent the system. To enhance dataset quality, the numerical solution uses a very fine temporal discretization for integration, up to a specified maximum time ($t_{\max}$), ensuring the generation of multiple oscillation periods. Given that the system operates outside the small oscillation regime, the oscillation period depends on the initial conditions.

The sampling interval Δt has a further impact on dataset quality and, consequently, on model performance. Given the pendulum's periodic motion, an inappropriate choice of Δt may lead to aliasing, causing a loss of frequency information. Furthermore, if the oscillation period T is close to an integer multiple of the sampling interval Δt , the acquired data in the phase space may become sparse, requiring an increase in $t_{\max}$ to ensure a more comprehensive exploration of the system's dynamics. As a limiting case, a sampling interval exactly commensurate with the oscillation period would cause the dataset to capture identical states across periods, reducing the available information to that of a single period.

3.3 Physics exploitation

In this section, two distinct approaches to the pendulum problem are presented—one based on PINNs and the other on PAT—highlighting their fundamentally different methodologies and areas of application.

3.3.1 PINN approach

This subsection presents an illustrative application of a PINN to the pendulum’s equation of motion.

The objective of the model is to predict the angular position θ of a pendulum given the time t , using only a small set of observed (t, θ) pairs within the first period of motion (Figure 3.1). This presents a challenge for traditional data-driven NNs, as they should infer the system’s behavior with limited data.

To establish a baseline, an MLP is trained on the given dataset without incorporating any prior knowledge of the underlying physics. The MLP consists of three hidden layers, each containing 30 neurons, and is trained using an MSE loss function (Equation 2.13) based solely on the observed data points.

However, since the MLP lacks any information about the governing equations of motion, it merely fits the training data, approximating it with a polynomial-like function. As a result, the model struggles to generalize beyond the training range, failing to capture the periodic nature of the pendulum’s motion (Figure 3.2).

In contrast, the PINN approach explicitly incorporates the underlying physics of the pendulum into the learning process by introducing a physical loss term with the residuals of the differential equations governing the system:

$$\frac{d\theta}{dt} - \omega, \quad \frac{d\omega}{dt} + \frac{g}{L} \sin(\theta), \quad (3.4)$$

ensuring that the predicted outputs respect the dynamics of the system.

The PINN consists of the same MLP architecture with three hidden layers, each containing 30 neurons, and it is implemented using the *DeepXDE* library [Lu et al., 2021a]

By leveraging the underlying physics, the PINN correctly extrapolates the motion beyond the temporal range of the available dataset, recovering the expected oscillatory pattern (Figure 3.3).

This experiment underscores the key advantage of PINNs: they effectively combine data and physical laws, enabling them to model physical systems even when observational data are limited. Unlike the data-driven MLP, which relied solely on statistical patterns in the data, the PINN approach produced physically consistent

predictions.

By acting as neural solvers, PINNs offer an alternative to traditional numerical methods. It is important to note, however, that the PINN reproduces the specific physical trajectory associated with the initial condition contained in the training dataset. In this setting, the training process essentially corresponds to solving the governing differential equation for that particular set of initial conditions, as well as for the specific pendulum length. As a result, while the PINN successfully reconstructs the full periodic motion from the limited portion of the trajectory provided as data, it would not generalize to different initial conditions without being retrained accordingly.

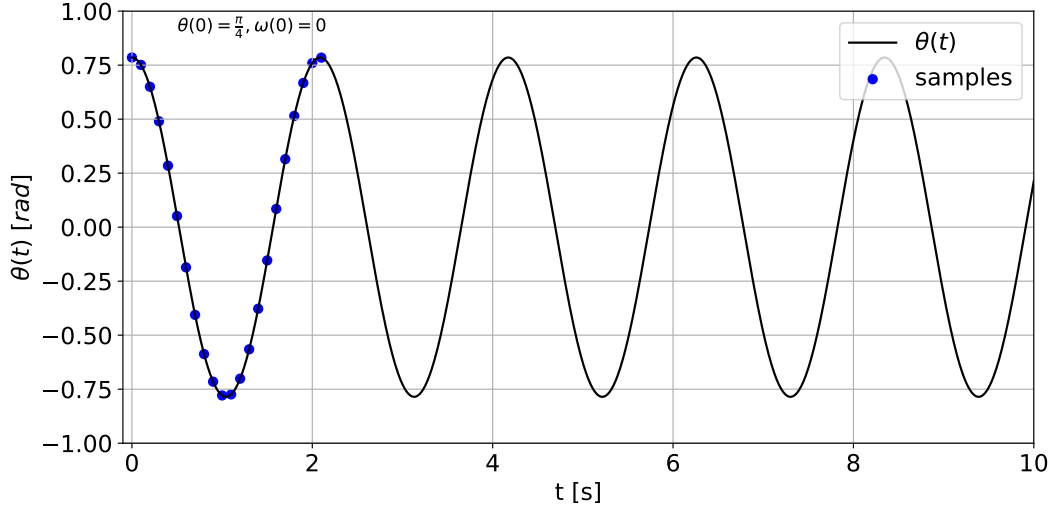


Figure 3.1: Reference solution of the motion of a pendulum from the starting position $\theta(0) = \pi/4, \omega(0) = 0$; the blue points represent the sampled dataset available for the training.

3.3.2 PAT approach

A different problem is considered to demonstrate the PAT approach, involving multiple datasets in which the MLP model learns to predict the system's future states. Specifically, the MLP is designed to reconstruct the pendulum's motion in an auto-regressive manner: it takes as input the physical state (i.e., angular position and velocity) at a given time t and predicts the physical state at the next time step $t + \Delta t$. In this way, the network acts as a time integrator, where the recursive application of

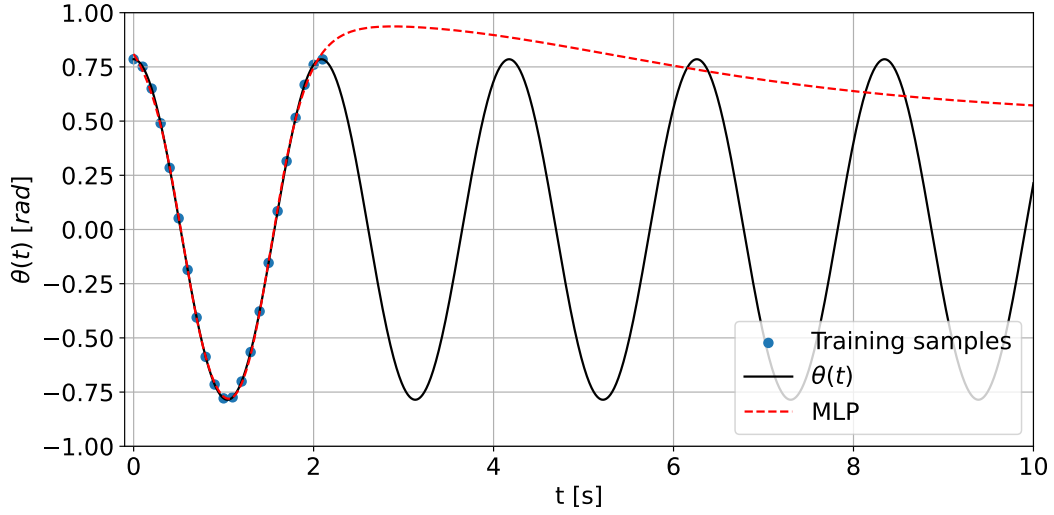


Figure 3.2: MLP prediction of the motion of a pendulum from the starting position $\theta(0) = \pi/4, \omega(0) = 0$, using the sampled dataset available for the training (blue points).

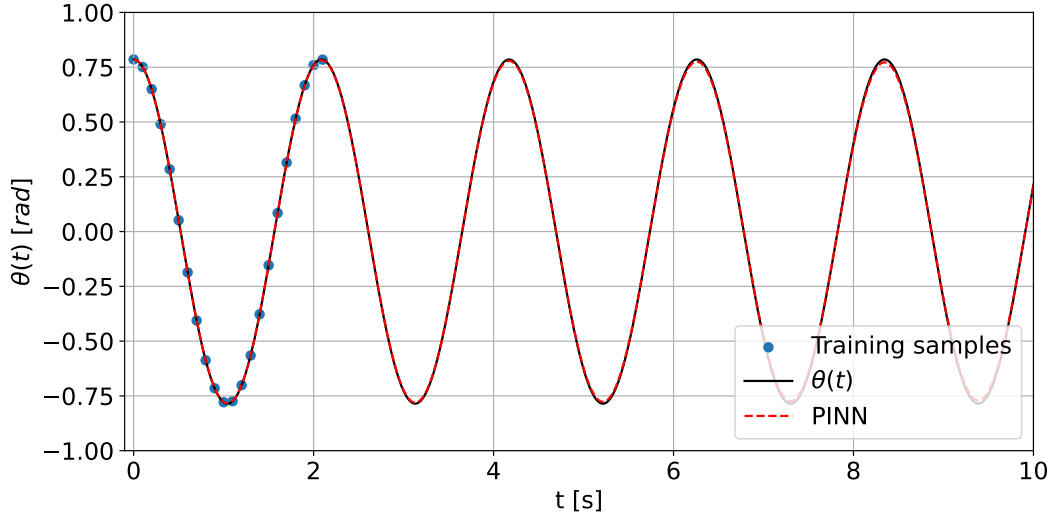


Figure 3.3: PINN prediction of the motion of a pendulum from the starting position $\theta(0) = \pi/4, \omega(0) = 0$, using the sampled dataset available for the training (blue points).

the model allows the reconstruction of the pendulum's motion and the corresponding phase space trajectory from a given initial state. It is important to note that, at each time step, the prediction error accumulates due to the recursive formulation, making the results dependent on both the accuracy of previous predictions and the model's stability, considered as its robustness to error propagation.

The benchmark purely data-driven training approach consists in minimizing a

loss function $\mathcal{L}$ between predicted (θ, ω) and target data $(\hat{\theta}, \hat{\omega})$, as in Equation (2.12). In contrast, the PAT approach incorporates physical information by modifying the loss function, as shown in Equation (2.31), thereby enforcing a priori knowledge derived from the known physics of the system.

For the pendulum, the conservation of energy serves as a fundamental guiding principle. Accordingly, an energy-related term is added to the loss function, where the predicted values of velocity and angular velocity are used to compute the predicted energy value (Equation 3.2), and any discrepancy from the true energy is penalized:

$$\mathcal{L} = \lambda \cdot MSE([\theta, \omega], [\hat{\theta}, \hat{\omega}]) + (1 - \lambda) \cdot MSE(En, \hat{En}). \quad (3.5)$$

Here, λ is a weighting factor that balances the relative contribution of the data-driven term and the energy term. It is treated as a hyperparameter and is not learned during training.

The following analysis considers three scenarios for predicting pendulum dynamics using different datasets: a single-energy dataset, a balanced three-energy-level dataset, and an unbalanced three-energy-level dataset. A comparative evaluation is performed to assess the improved predictive capabilities of the PAT approach.

Single-energy dataset

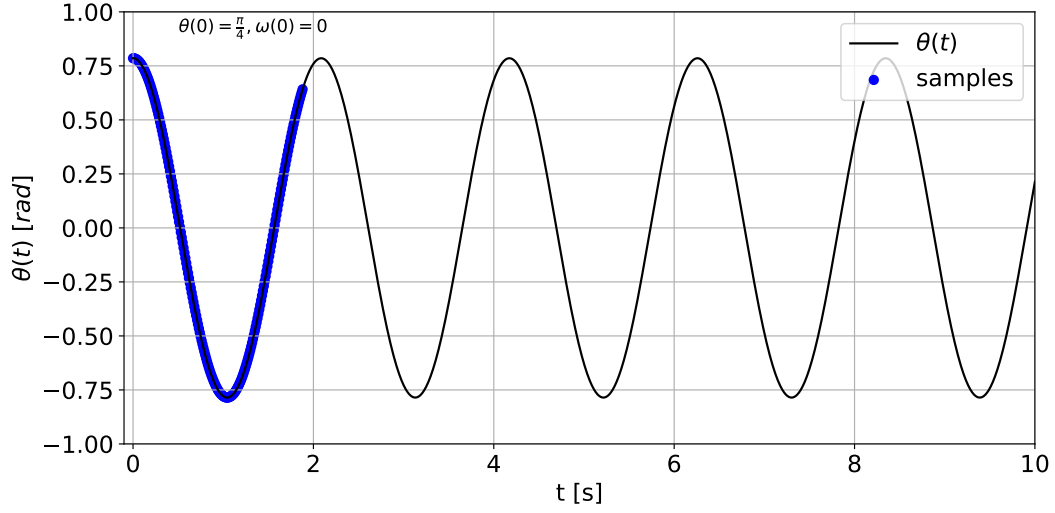
In this scenario, the dataset is formed by a single phase space trajectory, with the initial state of the system set to an angle of $\theta = \pi/4$ rad and angular velocity $\omega = 0$ rad/s. Starting from the initial conditions, a fine time series of $\theta(t)$ and $\omega(t)$ for almost a period is sampled to form the dataset at intervals of $\Delta t = 0.01$ s (Figure 3.4a)

As shown in Figure 3.4b, the dataset does not fully explore the phase space.

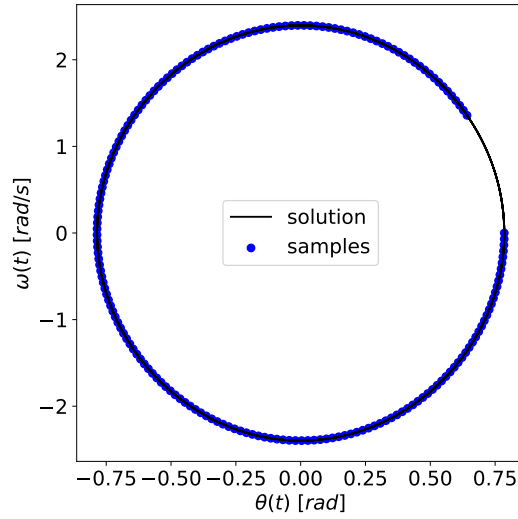
The dataset is divided into training, validation, and test sets, with shuffling, as illustrated in Figure 3.5.

An MLP with three hidden layers of 20 neurons each is trained on the selected training points. The trained model is then applied recursively to predict the system's motion over a 10-second duration, extending beyond the training data and covering previously unexplored regions of the phase space.

As shown in Figure 3.6, the PAT model accurately predicts the system's behavior, whereas the purely data-driven model, despite correctly identifying the motion's amplitude, fails to capture the exact period of oscillation, leading to larger errors in the phase trajectory sections that were not included in the training dataset.



(a)



(b)

Figure 3.4: Reference solution of the motion of a pendulum from the starting position $\theta(0) = \pi/4$, $\omega(0) = 0$; the blue points represent the sampled dataset available. Panel (a) shows the time evolution of θ , while panel (b) illustrates the phase space.

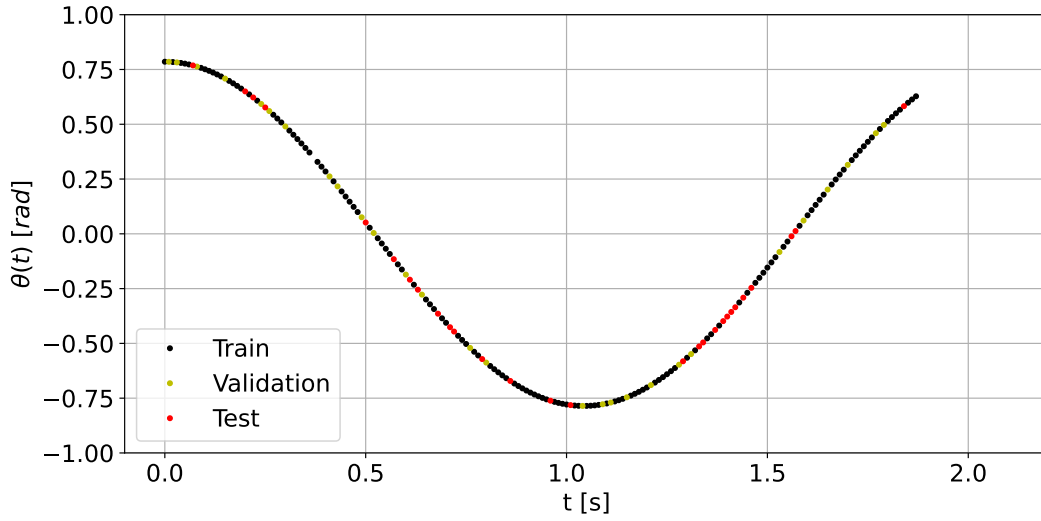


Figure 3.5: Splitting of the samples into training, validation, and test sets

Three-energy levels dataset

In this experiment, the dataset consists of three distinct phase space trajectories obtained by releasing the pendulum from initial positions of $\theta = \pi/2, \pi/3, \pi/6$ rad with an angular velocity of $\omega = 0$ rad/s.

Starting from these initial conditions, the time series of $\theta(t)$ and $\omega(t)$ are generated over a 10-second duration, with a sampling interval of $\Delta t = 0.05$ s (Figure 3.7a).

As shown in Figure 3.7b, the dataset fully explores the phase space for $\theta = \pi/2$ and $\theta = \pi/6$. However, for $\theta = \pi/3$, the samples are unevenly distributed along the trajectory due to the choice of the sampling interval Δt .

As in the previous experiment, the dataset is divided into training, validation, and test sets, as illustrated in Figure 3.8. The MLP architecture is the same as that of the previous experiment.

In Figure 3.9, the results show that the purely data-driven model remains accurate only for the first period. However, its predictions progressively deteriorate in both amplitude and period across all three trajectories.

On the other hand, the MLP with PAT demonstrates a significant improvement in long-term stability, achieving an almost perfect prediction for the lowest energy-level trajectory and better predictions for the others.

These results are further highlighted in Figure 3.10, where the predicted points

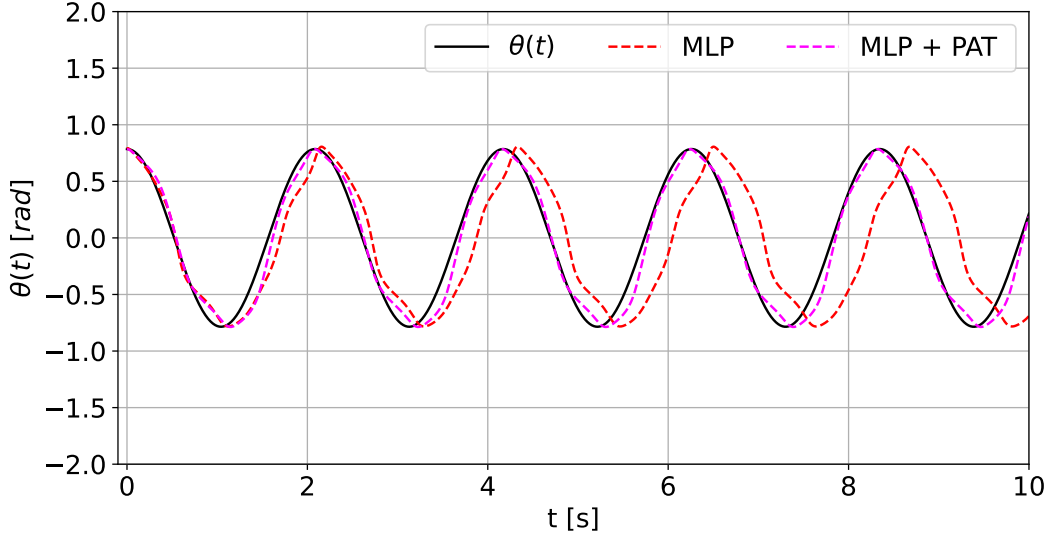


Figure 3.6: Prediction of the pendulum motion by the purely data-driven MLP in red and the MLP with the PAT strategy in magenta

align more closely with the target phase space trajectory when using the modified loss function.

Unbalanced, Three-energy levels dataset

In this scenario, the previous setup is slightly modified to introduce an imbalance in the dataset across different energy levels. Specifically, the time series is generated with varying durations: $\theta(t) = \pi/6$ is recorded for 50 seconds, $\theta(t) = \pi/3$ for 20 seconds, and $\theta(t) = \pi/2$ for 10 seconds.

As a result, the lower-energy trajectory ($\theta = \pi/6$) is overrepresented in the dataset, while the higher-energy trajectories ($\theta = \pi/3$ and $\theta = \pi/2$) have significantly fewer samples. This imbalance is deliberately introduced to bias the model’s training toward lower-energy trajectories.

The dataset is then divided into training, validation, and test sets without shuffling (Figure 3.11). As observed in the previous experiment, for $\theta = \pi/3$ the samples remain unevenly distributed along the phase space trajectory due to the chosen sampling interval $\Delta t = 0.05$ s.

This experiment provides insight into how an imbalanced dataset affects the model’s predictive capabilities and whether the PAT approach can help mitigate

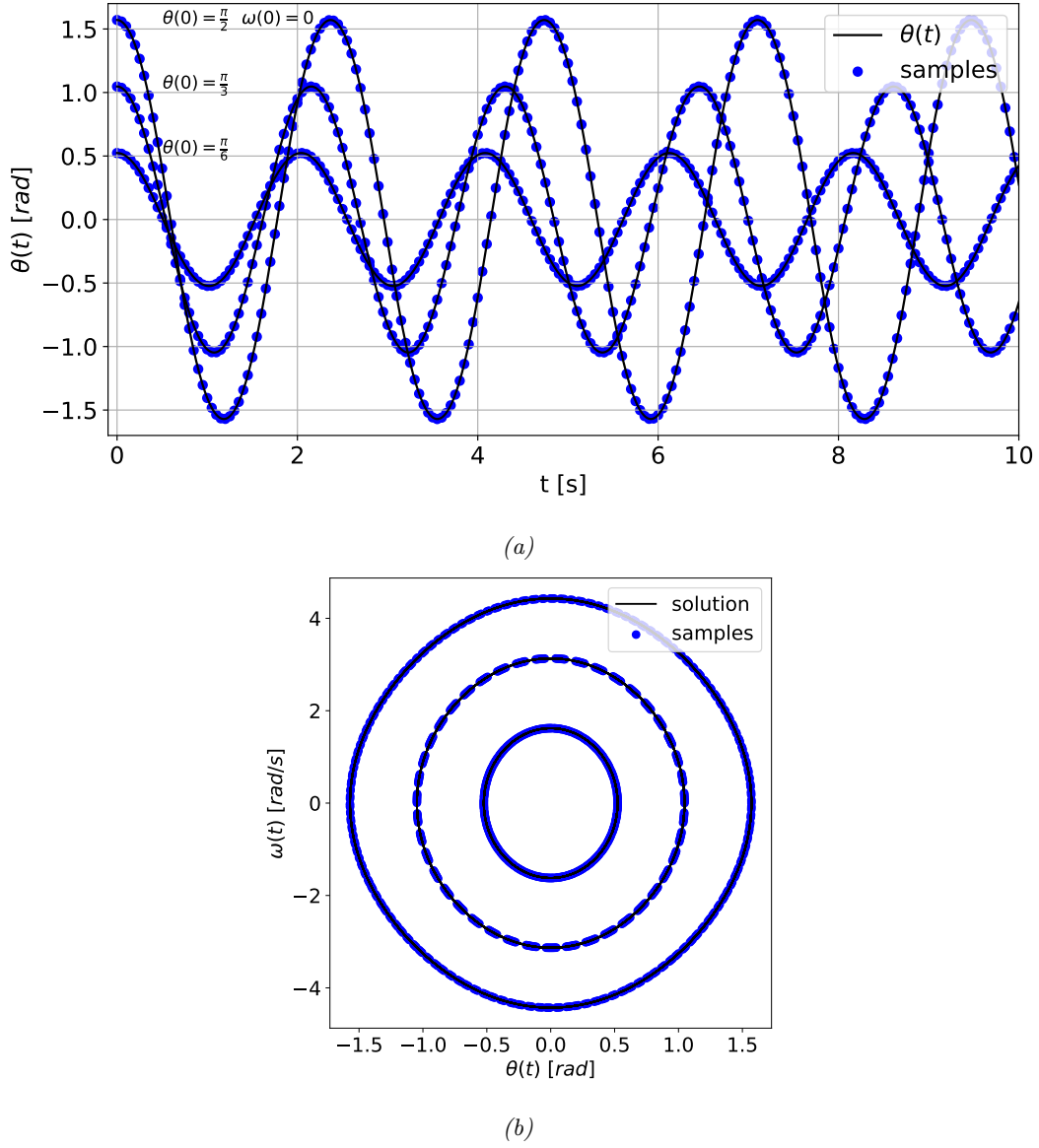


Figure 3.7: Reference solution of the motion of a pendulum from three different starting positions; the blue points represent the sampled dataset available for training. Panel (a) shows the time evolution of θ , while panel (b) illustrates the phase space.

potential biases introduced by the over-representation of lower-energy states. This test is particularly relevant, as imbalanced datasets are a common occurrence in hydrological and environmental modeling, where certain conditions are naturally over-represented while others are scarcely observed [Lu et al., 2025].

The network complexity is increased by expanding the architecture to three hidden layers, each containing 50 neurons.

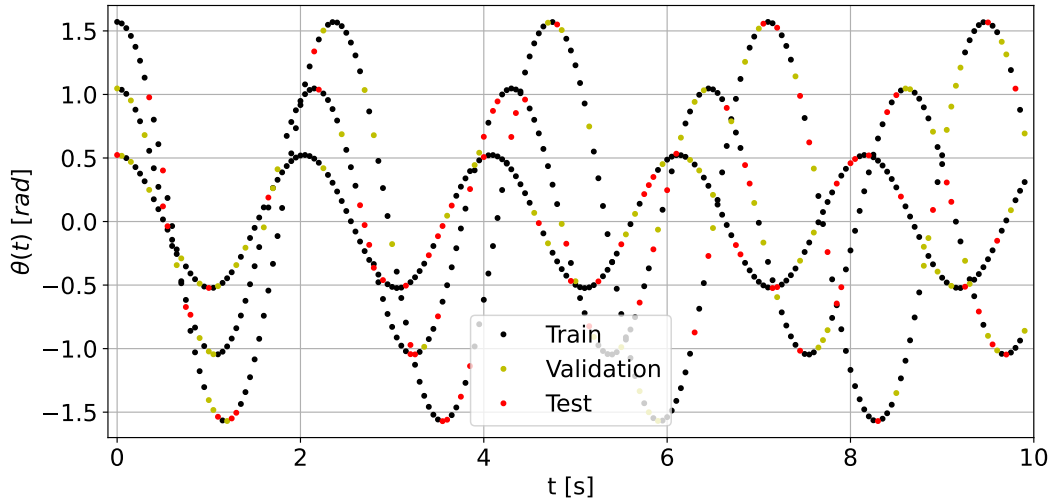


Figure 3.8: Splitting of the samples into training, validation, and test sets

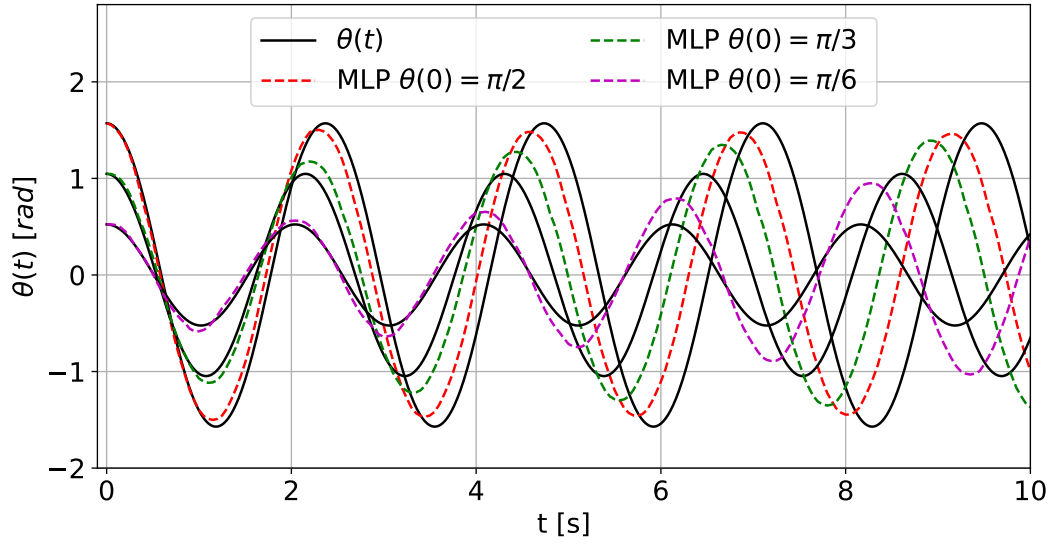
Figures 3.12 and 3.13 show the results for a 10-second prediction. With the conventional loss function, the model exhibits a gradual decay toward lower energy states over time. In contrast, the physically-modified model demonstrates remarkable stability across all three energy levels, maintaining consistency without energy dissipation, as is also evident in the phase space trajectories. However, the prediction for the intermediate energy state shows some timing errors, likely influenced by the uneven distribution of samples along this phase space trajectory.

3.4 Conclusions

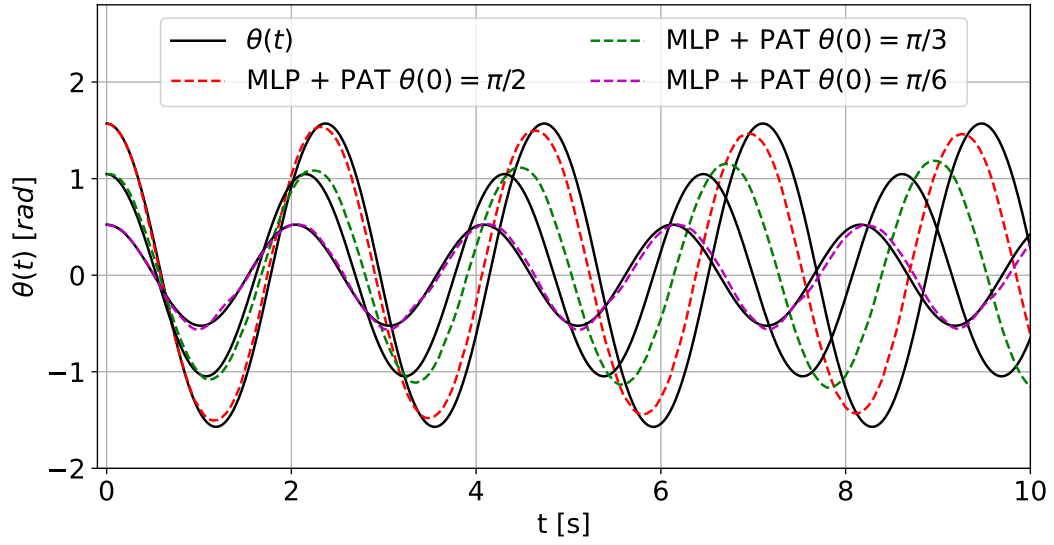
The pendulum was employed to showcase the potential of integrating physical knowledge into DL models for the simulation of physical systems.

It is important to highlight a key distinction between the two types of experiments and approaches described in Section 3.3.1 and Section 3.3.2. Both strategies share the conceptual idea of integrating physical principles into the learning process via a modification of the loss function; however, they differ significantly in their objectives and applications.

In the first part, it was demonstrated that PINNs are designed as direct alternatives to classical numerical solvers. Training a PINN is effectively equivalent to solving the differential equations for a specific set of initial or boundary conditions,



(a)



(b)

Figure 3.9: Reference solution of the motion of a pendulum from three different starting positions; the dashed lines represent the predictions obtained by recursively applying the same MLP from the three initial states. Panel (a) shows the predictions of the purely data-driven MLP, while panel (b) illustrates the predictions of the MLP trained with the PAT strategy.

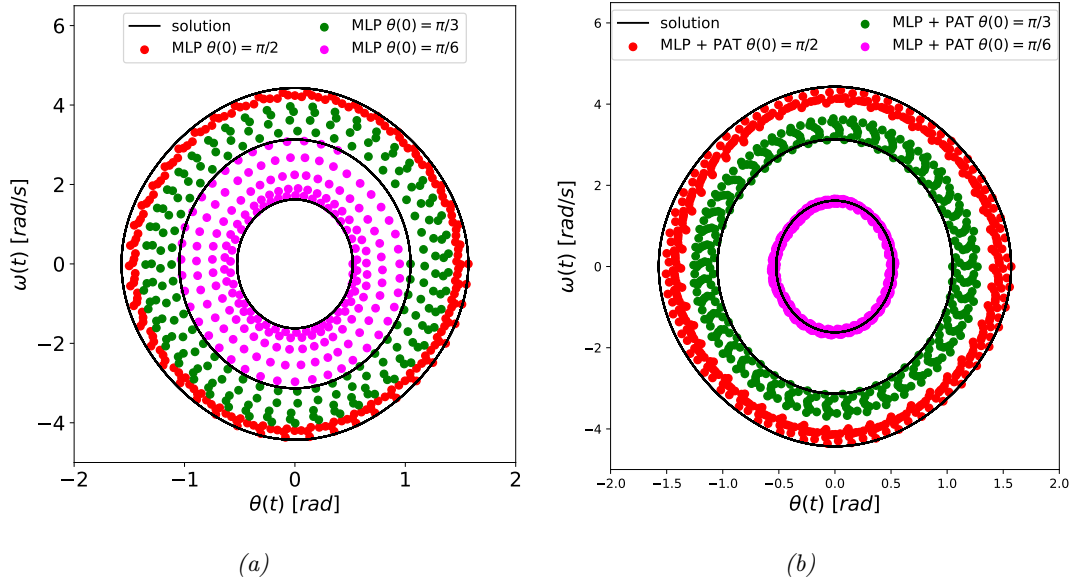


Figure 3.10: Phase diagrams of the pendulum motion predicted by the purely data-driven MLP (a) and the MLP with the PAT strategy (b), along with the reference solution.

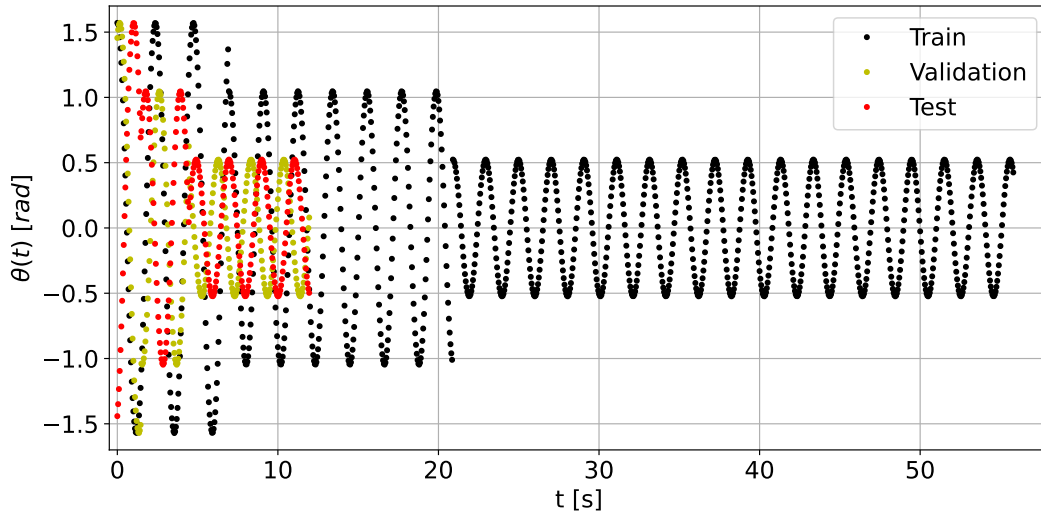
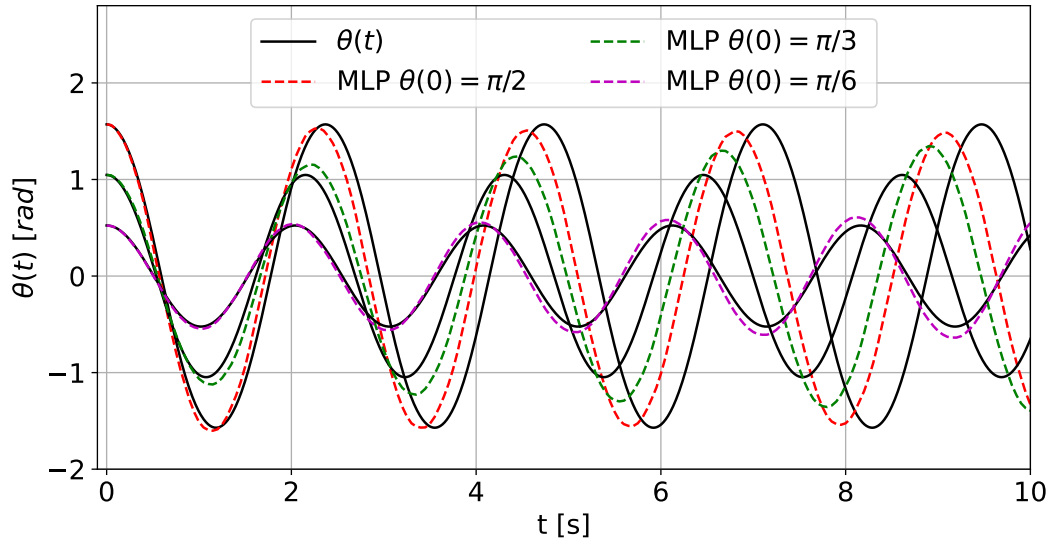
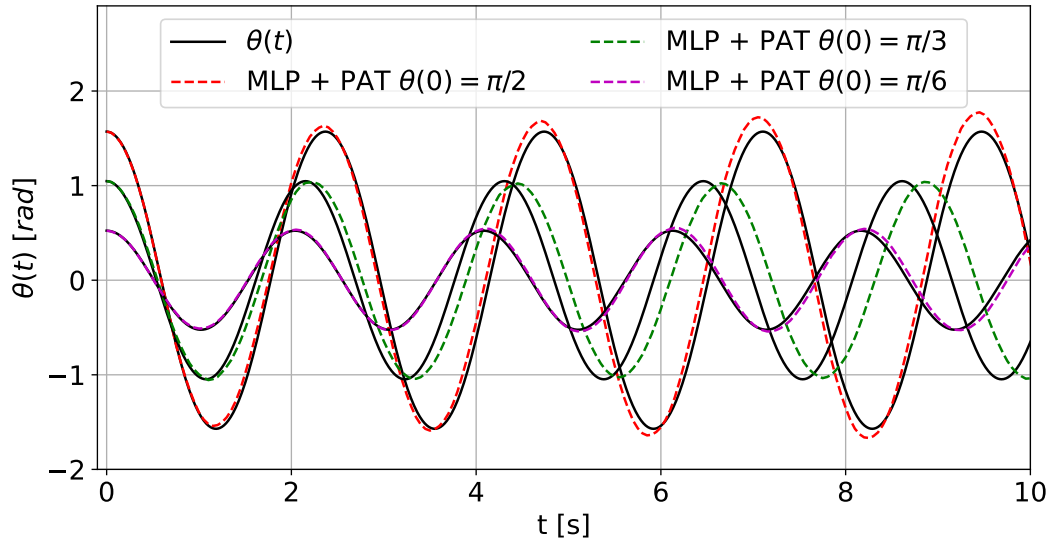


Figure 3.11: Splitting of the samples into training, validation, and test sets. The data for the lowest-energy state are preponderant.



(a)



(b)

Figure 3.12: Reference solution of the motion of a pendulum from three different starting positions; the dashed lines represent the predictions obtained by recursively applying the same MLP from the three initial states. Panel (a) shows the predictions of the purely data-driven MLP, while panel (b) illustrates the predictions of the MLP trained with the PAT strategy, in the case of unbalanced dataset.

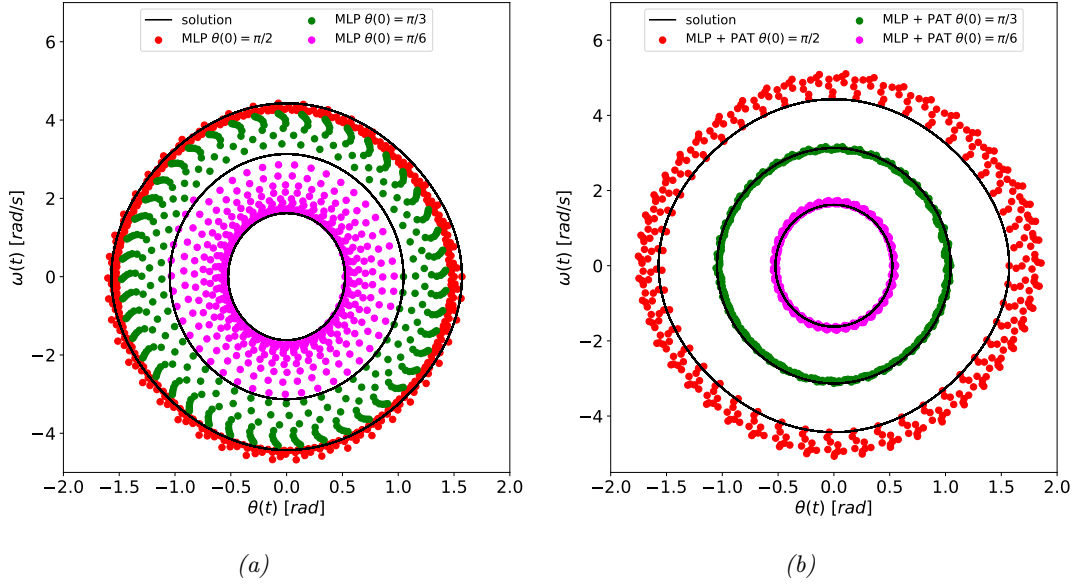


Figure 3.13: Phase diagrams of the pendulum motion predicted by the purely data-driven MLP (a) and the MLP with the PAT strategy (b), along with the reference solution.

and any change in these conditions necessitates constructing and training a new PINN from scratch.

In contrast, the PAT approach was applied in common data-driven scenarios, aiming to model cases where data—sparse or well-distributed—are available across multiple conditions, and where known physical properties (e.g., the system’s energy) can be leveraged to improve predictions. Here, physical information does not require explicit knowledge of the governing differential equations. In other words, this approach leverages expert knowledge to guide model training toward physically consistent solutions rather than directly solving the system’s PDEs. Its aim is not to serve as an alternative numerical modeling approach, but to enhance data-driven models by steering their training toward physically consistent solutions while exploiting physical properties and quantities.

Moreover, as shown in Section 3.3.2, the same data-driven model can be applied to different initial conditions without the need for retraining, highlighting a major difference from PINNs.

For the remainder of this work, reference will be made to the PAT approach, as it leverages expert physical knowledge to improve generalization while retaining the

flexibility and efficiency of classic data-driven learning.

Chapter 4

A 1D hydraulic application: varied flows in river

This chapter aims to demonstrate that the proposed PAT approach for incorporating physical concepts into existing data-driven models can be beneficial in hydraulic applications. The analysis investigates whether physically-trained NNs can generalize better than purely data-driven ones in a well-controlled experiment in environmental hydraulics.

4.1 Introduction

The PAT strategy of embedding physical knowledge—without relying on PDEs—into data-driven models is targeted at scenarios where the underlying mathematical formulation is not fully known, but expert knowledge allows the introduction of physics into the loss function. This is particularly relevant in river hydraulics, where epistemic uncertainty introduces additional challenges.

Following the approach of Cedillo et al. [2022], a simple yet non-trivial physical problem — the reconstruction of a steady-state, one-dimensional water surface profile in a rectangular channel — is employed to isolate the effects of the PAT strategy. In this work, the configuration includes the presence of an obstacle within the channel.

This problem is particularly interesting because the profiles can be reconstructed using different MLP approaches. The goal is not to evaluate the best model to

solve the problem at hand, but rather to assess the effects of the a priori physical information.

The highly informative content of the analyzed problem arises mainly from the hidden complexity of the underlying physics, associated with the possible occurrence of a hydraulic jump in mixed-flow conditions (supercritical and subcritical), which is not captured by the differential equation used to determine the water profile (i.e., the Specific Energy Equation 4.2).

The simulations span a wide range of input variations, testing the efficacy of the PAT strategy even in data-scarce scenarios. Extrapolation tests were also conducted, an aspect rarely addressed in ML studies, to mirror the challenges typically encountered in flood mapping applications. This kind of assessment is of great relevance to the application of data-driven methods to flood mapping, where cases featuring values of the observed quantities falling outside the range of the recorded series need to be predicted.

The material presented in this chapter has been published in Guglielmo et al. [2025].

4.2 Problem definition and physics

The problem at hand deals with the reconstruction of the water surface profile along a rectangular channel induced by the presence of a dam placed at the outlet cross section. This problem mimics a commonly occurring scenario of determining the area affected by the presence of a weir in a river.

The solution of the physical problem, assuming a prismatic rectangular channel and steady flow conditions, is an unknown mapping function $\mathcal{F}$:

$$h = \mathcal{F}(x; s, b, n, z_d, Q) \quad (4.1)$$

where h represents the flow depth at the longitudinal position x , measured as the distance upstream from the dam.

The mapping $\mathcal{F}$ is fully determined by the physical parameters of the system:

- s , the channel slope;

-
- b , the channel width;
 - n , the Manning coefficient (related to the channel's roughness);
 - z_d , the height of the dam;
 - Q , the water discharge.

The dataset was generated by numerically solving the specific energy equation, expressed as [Cengel and Cimbala, 2013]

$$\frac{d\mathcal{E}}{dx} = s - J. \quad (4.2)$$

Here, $\mathcal{E}$ represents the specific energy relative to the channel bottom, which, for a rectangular channel, can be calculated as

$$\mathcal{E} = h + \frac{Q^2}{2gb^2h^2} \quad (4.3)$$

where g is the gravitational constant, whereas J is the energy grade slope, calculated using the Chezy relation:

$$J = \frac{n^2 Q^2}{\Omega^2 R^{4/3}} \quad (4.4)$$

where $\Omega = bh$ is the cross-sectional flow area and $R = \Omega/(b + 2h)$ is the hydraulic radius.

The downstream boundary condition is represented by the broad-crested weir equation:

$$h = z_d + \left(\frac{3\sqrt{3}Q}{2\sqrt{2}gb} \right)^{2/3} \quad (4.5)$$

where z_d is the weir height.

The differential Equation (4.2) generating the gradually-varied water profile is not valid at the location of the hydraulic jump. Indeed, the hydraulic jump represents an internal boundary whose location (and strength of the local discontinuity) needs to be solved for through additional information. In a hydraulic jump, the transition from supercritical ($Fr > 1$) to subcritical ($Fr < 1$) flow occurs, where

$$Fr = \frac{Q}{\Omega\sqrt{gh}} \quad (4.6)$$

is the Froude number, the ratio between inertial forces and gravitational forces in the flow.

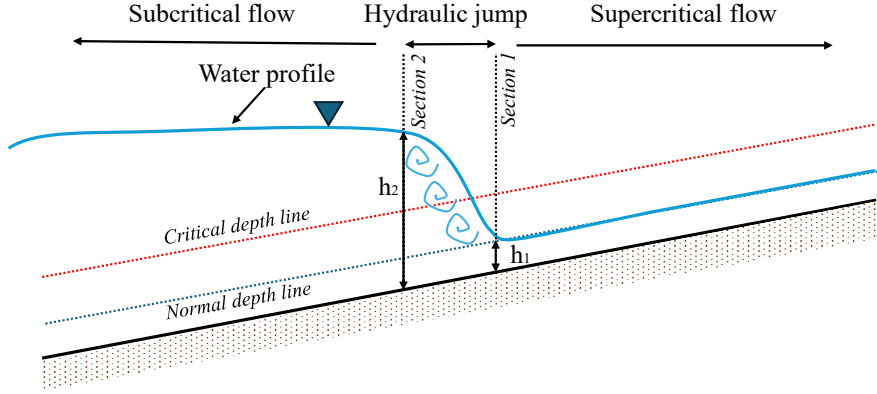


Figure 4.1: Schematic of a hydraulic jump, showing the two control sections (section 1, upstream, and section 2, downstream) with their conjugate depths h_1 and h_2 .

Examining the conservation of momentum within the fluid volume encompassing the hydraulic jump, two cross-sections denoted as 1 (upstream, supercritical flow) and 2 (downstream, subcritical flow), as shown in Figure 4.1, can be defined. The conservation of momentum can be expressed as:

$$\Pi_{p1} + M_1 = \Pi_{p2} + M_2 \quad (4.7)$$

Here, Π_{p1} and Π_{p2} represent the upstream and downstream values of the hydrostatic force, whereas M_1 and M_2 denote the upstream and downstream values of the momentum flux, respectively. For a rectangular prismatic channel, the Equation (4.7) becomes:

$$\frac{h_1^2}{2} + \frac{Q^2}{gb^2h_1} = \frac{h_2^2}{2} + \frac{Q^2}{gb^2h_2} \quad (4.8)$$

which can be arranged in the following expression, establishing a mathematical relation between h_1 and h_2 , that are called conjugate depths:

$$h_1 = \frac{h_2}{2}(-1 + \sqrt{1 + 8Fr_2^2}) \quad (4.9)$$

Fr_2 is the Froude number (Equation (4.6)) at cross-section 2.

4.3 Dataset generation

A synthetic dataset of water profiles was generated by uniformly varying s , b , n , z_d , and Q , spanning broad ranges of input variations (e.g., 100% for inputs n and z_d ,

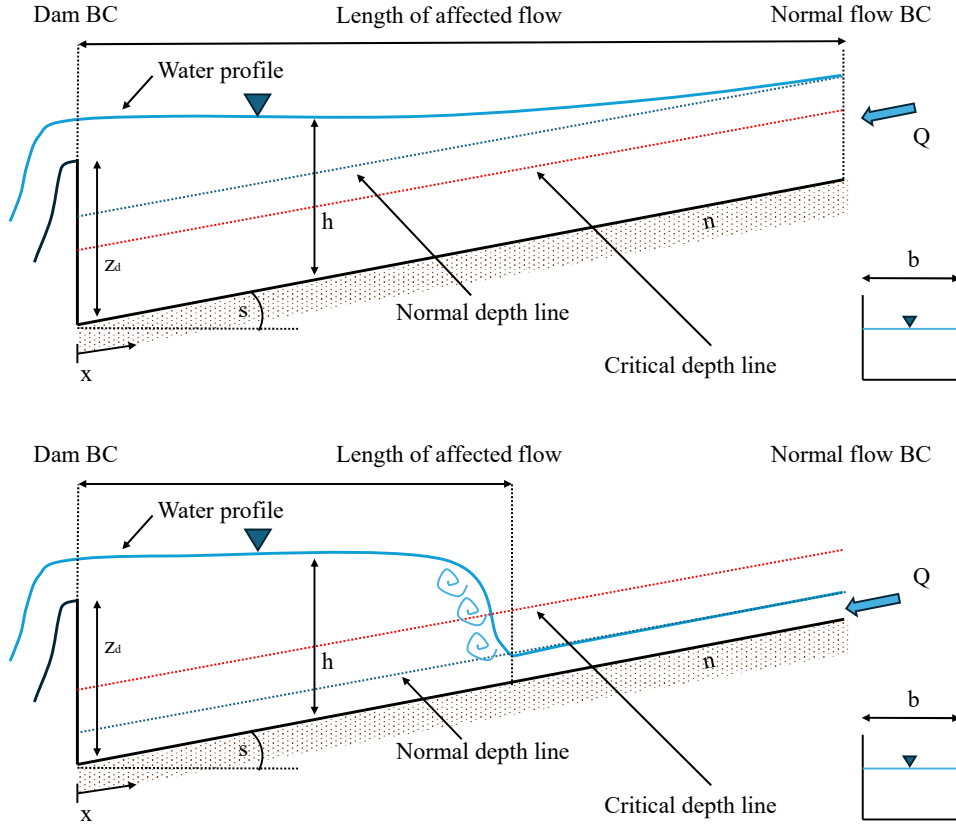


Figure 4.2: Sketch of the analyzed physical problem, i.e. the steady state water profile in a rectangular prismatic channel. The upper panel describes the subcritical case; the lower one depicts the mixed regime case. For the meaning of the symbols, the reader is referred to the text.

400% for Q , 500% for b , and 900% for s). The profiles are sampled with a fixed spatial discretization of Δx set to 10 meters, covering a total length of 5000 meters resulting in profiles of $N = 501$ points. This generation method results in a uniform sampling of the 6-dimensional domain of the $\mathcal{F}$ mapping in Equation (4.1). (see Figure 4.3)

The water profile is obtained by solving Equation (4.2) using a first-order finite difference scheme with a constant spatial discretization, Δx . An upstream marching algorithm is employed, encompassing the solution of the hydraulic jump if necessary.

The obtained profiles were randomly divided, with 70% allocated to the training set and the remaining 30% split between the validation and test sets. As is common

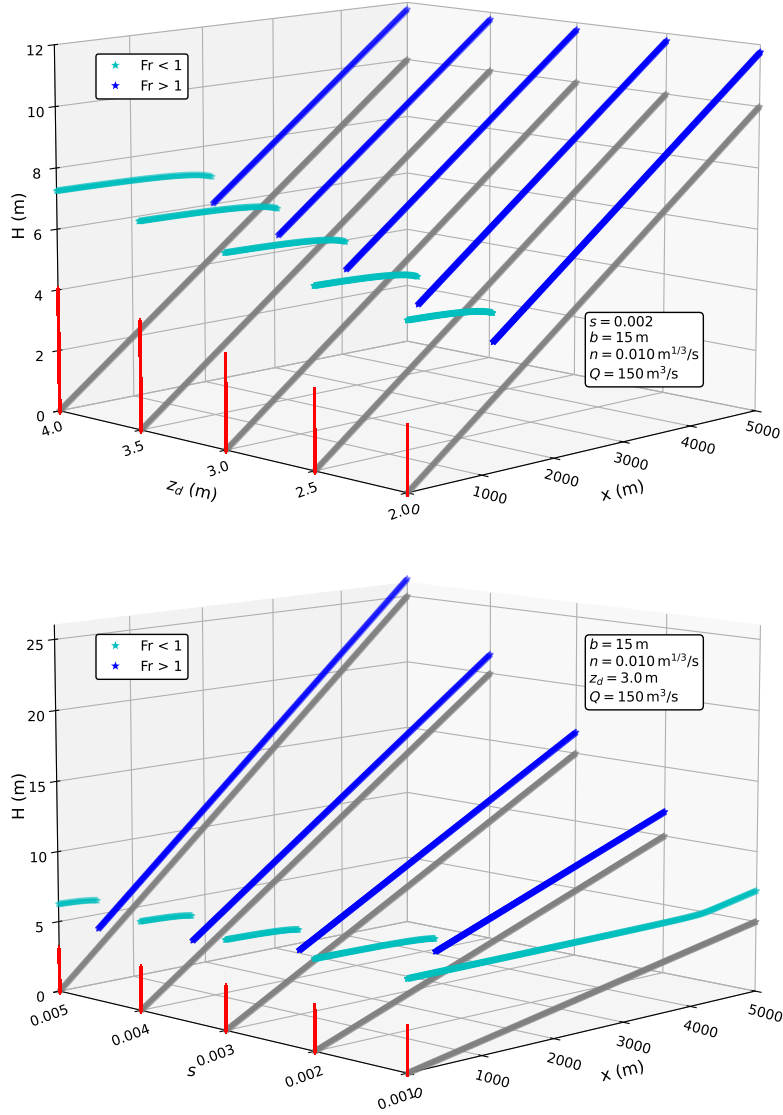


Figure 4.3: The plot shows water surface profiles from two subsets of the whole synthetic dataset. The simultaneous effects of dam height (z_d , red pillars) and channel slope (s) are illustrated in the upper and lower panel respectively, whereas fixing the remaining parameters to the values reported in the boxes. In both panels, the water surface height (H) is plotted against the distance from the dam (x); grey lines represent the bottom of the channel. Subcritical (Froude number $Fr < 1$) and supercritical ($Fr > 1$) flow regimes are depicted in cyan and blue markers, respectively. The discontinuous transitions between the two flow regimes represent hydraulic jumps.

in the ML framework, model generalization is analyzed by evaluating performance on the test set.

The input data to the various models are always normalized using the Standard Scaler, which is a commonly used data pre-processing technique to scale the features to have a mean of zero and a standard deviation of one. This is necessary to prevent troublesome problems during training due to differences in the numerical values of the features [Ali et al., 2014].

4.4 Employed Neural Network architectures

To show the versatility of the approach, three MLP approaches were examined, referred to as the Single-Point (SP), the INTegrator (INT), and the Vector-To-Sequence (VTS). It is important to state that the aim of this work is not to select the best architecture for solving the problem at hand, but to evaluate the effects of the proposed PAT methodology. Hence, there might be more effective DL approaches for solving the problem.

A sketch of the operating principles of SP, INT, and VTS approaches is illustrated in Figure 4.4, and the corresponding network topologies are depicted in Figure 4.5.

4.4.1 Single-Point

A verbatim translation of the problem formulated in Equation (4.1) is to approximate the mapping $\mathcal{F}$ with an MLP (SP):

$$h = SP_{\Xi}(x, s, b, n, z_d, Q;). \quad (4.10)$$

with Ξ the trainable parameters.

In this approach, the MLP takes the six quantities governing the phenomenon as inputs and predicts a single value for the water depth h at the stationing x as an output.

To reconstruct the whole profile $\mathbf{h}$, SP needs to be run with all the desired stationing values.

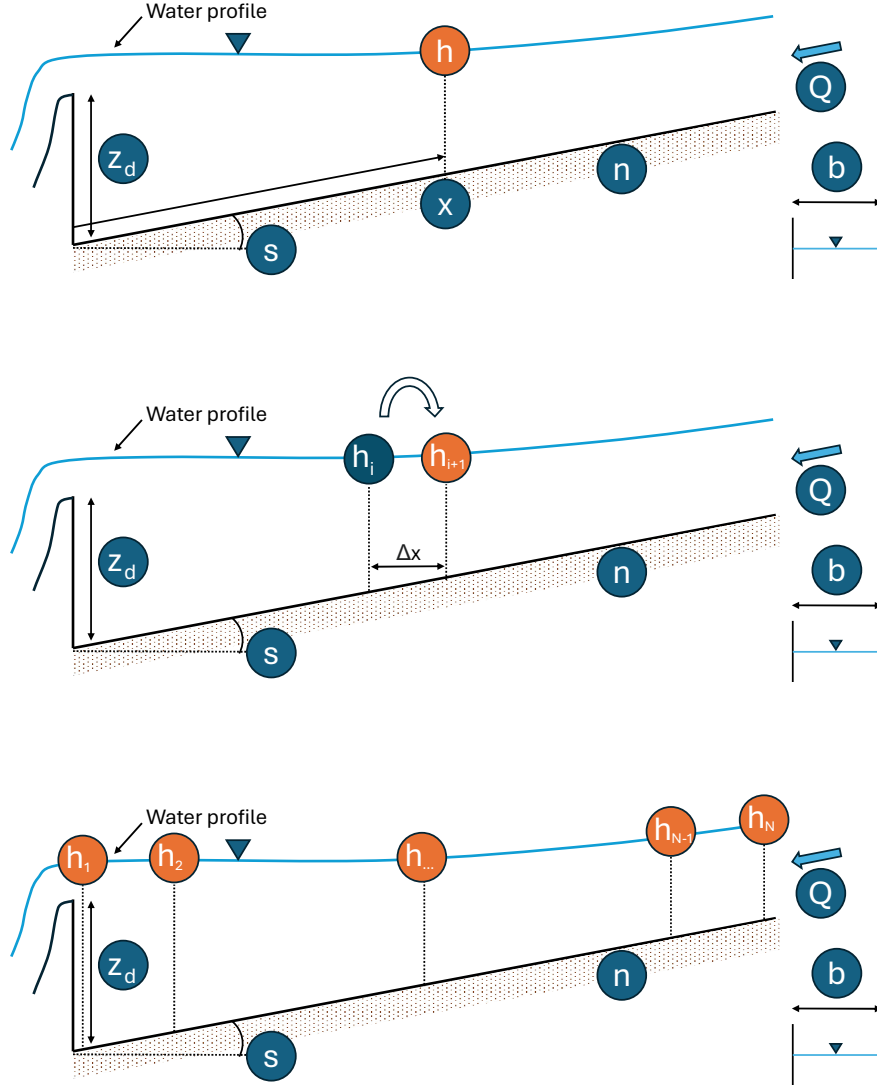


Figure 4.4: Profile reconstruction using the three different approaches; inputs and outputs are depicted as blue and orange circles, respectively; Single-Point (upper panel) outputs the flow depth at a specific stationing; Integrator (middle panel) requires the neighboring downstream value, regardless of the stationing; Vector-To-Sequence (lower panel) outputs the entire vector of flow depth (i.e. the whole water profile) at once.

4.4.2 Integrator

In this method, an MLP is utilized as a numerical integrator, i.e., designed to determine the local water depth based on its value at the adjacent station. This

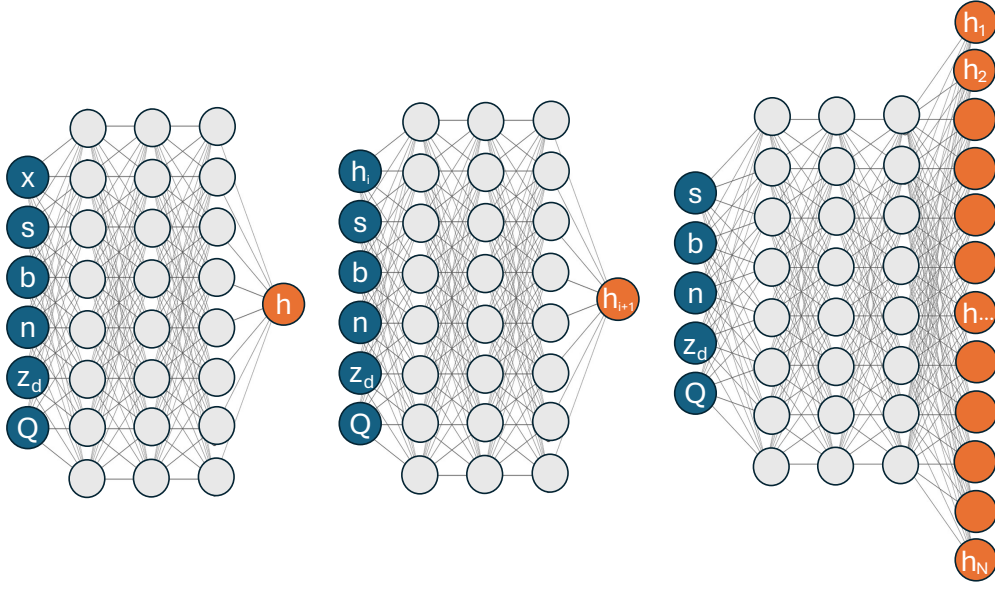


Figure 4.5: The employed MLP architectures. Left to right: Single-Point, Integrator, Vector-To-Sequence.

approach is analogous to that adopted in Chapter 3 for the pendulum experiments.

An eigenanalysis of the differential problem underpinning a steady-state 1D free surface flow, like the one chosen in this study, would require adopting either a downstream or upstream oriented solution direction based on the local flow regime, namely supercritical or subcritical, respectively. However, in the context of surrogate models, like the one based on ML, due to the lack of any physical support, such a requirement can be overlooked. In the following, an upstream moving algorithm has been chosen.

The Integrator approach requires a rearrangement of the dataset into the pairs $([h_i, s, b, n, z_d, Q], h_{i+1})$, where h_i and h_{i+1} must be uniformly spaced by Δx , and the counter i spans all stationing points of the profiles.

Due to the above structure, the approach misses any information regarding the distance from the dam. As a result, unlike the SP, it cannot predict the height at any distance from the dam but only at multiples of the fixed Δx set by the available dataset:

$$h_{i+1} = INT_{\Xi}(h_i, s, b, n, z_d, Q) \quad (4.11)$$

Starting from a known downstream boundary condition (h_1), the MLP (INT)

can be applied recursively. At each stationing, the output from the downstream one serves as input, eventually leading to the reconstruction of the whole profile:

$$h_i = INT_{\Xi}(h_{i-1}, \dots) \circ \dots \circ INT_{\Xi}(h_1, s, b, n, z_d, Q) \quad (4.12)$$

It is essential to note that any classical numerical integrator would require repeated checks for the occurrence of a hydraulic jump, as well as an *ad hoc* procedure for its solution. Instead, this model can be applied flawlessly across such discontinuity.

Two additional specific features of the approach are presented. Firstly, at each location, due to its recursive application (Equation 4.12), *INT* outputs a depth value whose error depends on the accuracy of the previous applications. As a consequence, the model’s accuracy is expected to decrease in the marching direction of the algorithm. A side advantage of such an upstream marching algorithm lies in the possibility of incorporating physical conservation balances between the current and previous locations. Secondly, the recursive application of *INT* introduces a subtle advantage over the other purely data-driven approaches: indeed, its initialization (occurring at the most downstream stationing in our case) represents an implicit imposition of a physical constraint.

4.4.3 Vector-To-Sequence

The Vector-To-Sequence employs an MLP (*VTS*) that receives as input the five parameters determining the profile solution and predicts as output the entire vector **h**:

$$\mathbf{h} = VTS_{\Xi}(s, b, n, z_d, Q) \quad (4.13)$$

The dataset must be rearranged in pairs $([s, b, n, z_d, Q], \mathbf{h})$. Just like for the Integrator approach (Equation 4.11), the input does not include stationing data, thus implying that the spacing of the output matches that of the training dataset.

An advantage of this approach is that it allows for the implementation of physical loss terms, whose formulation requires the knowledge of the whole profile, e.g., the volume of the water profile.

4.4.4 Hyperparameters

Hyperparameters are values that must be set before the training process and are not updated during the training phase. They encompass crucial features such as the number of hidden layers and neurons, the optimization algorithm along with its learning rate, and parameters related to early stopping [Chollet et al., 2015]. Despite the study not aiming to compare the three employed MLP architectures, the implemented network topologies reflect the output complexities (e.g., local depth versus whole profile): indeed, for both *SP* and *INT*, the MLP contains 3 layers with 30 neurons each, whereas *VTS* contains 3 layers with 40 neurons each. The number of layers is fixed for all the experiments, and the analysis of its influence is beyond the scope of this work.

The ReLU activation function (see Table 2.3) is employed. ReLU has been chosen for its simplicity and widespread utilization in DL applications; however, its selection is not expected to impact the broader objectives of this study, and other activation functions could also be used. The weights optimization phase of NNs exploits the Adam algorithm [Kingma and Ba, 2014] as a gradient descent optimizer.

A learning rate reduction technique, specifically ReduceLROnPlateau, within the Adam [Kingma and Ba, 2014] optimization algorithm, is applied here. The initial learning rate is set to 0.001 and is progressively reduced when approaching a minimum of the loss function. An early stopping criterion is also employed during the training phase, based on the MSE (Equation 2.13) calculated on the validation set.

The hyperparameter λ in Equation (2.31), which weighs the contribution of the physical regularization term, has been chosen individually for each model as the one yielding the best performance in a discrete set of values comprised in the interval $[0, 1]$, so as not to introduce any biases in the results.

All the hyperparameters were fixed within each model to achieve consistent results.

All the MLPs have been implemented using TensorFlow [Abadi et al., 2015] and Keras [Chollet et al., 2015].

4.5 PAT strategy

Each of the three MLP approaches has undergone both purely data-driven and PAT strategies, the latter consisting in exploiting the local values of the:

- Specific energy, as in Equation (4.3), EN strategy in the following;
- Froude number, as in Equation (4.6), FR strategy in the following.

Utilizing both the specific energy and the Froude number as conveyors of physical information exemplifies expert, a priori knowledge inputs in the training procedure.

The loss function for the purely data-driven training strategy is formulated in terms of the MSE between the target ($\hat{h}$) and predicted (h) water depths. Whereas the PAT loss terms (refer to Equation (2.31)) adopted for SP and INT read:

$$\mathcal{L}_P^{EN} = \frac{1}{K} \sum_{k=1}^K \left(\mathcal{E}(h_k, Q_k, b_k) - \mathcal{E}(\hat{h}_k, Q_k, b_k) \right)^2 \quad (4.14)$$

$$\mathcal{L}_P^{FR} = \frac{1}{K} \sum_{k=1}^K \left(Fr(h_k, Q_k, b_k) - Fr(\hat{h}_k, Q_k, b_k) \right)^2 \quad (4.15)$$

where K is the batch size, that is the number of training samples utilized at each training iteration [Abadi et al., 2015]. The specific energy value, $\mathcal{E}$, and the Froude number value, Fr , depend on both the output h and the inputs Q and b , as in (4.3) and (4.6).

The loss function for the VTS approach, the latter outputting the N -length profile, needs to be averaged over space as well, that is:

$$\mathcal{L}_P^{EN} = \frac{1}{K} \sum_{k=1}^K \left[\frac{1}{N} \sum_{i=1}^N \left(\mathcal{E}(h_i, Q_i, b_i) - \mathcal{E}(\hat{h}_i, Q_i, b_i) \right)^2 \right]_k \quad (4.16)$$

$$\mathcal{L}_P^{FR} = \frac{1}{K} \sum_{k=1}^K \left[\frac{1}{N} \sum_{i=1}^N \left(Fr(h_i, Q_i, b_i) - Fr(\hat{h}_i, Q_i, b_i) \right)^2 \right]_k \quad (4.17)$$

Since the VTS approach outputs the entire profile at once, it is also possible to test two additional training strategies using:

- The volume of water flowing in the river (namely, the area under the water profile), VOL strategy in the following;

-
- The downstream boundary condition, BC strategy in the following;

The corresponding loss terms read:

$$\mathcal{L}_P^{VOL} = \frac{1}{KN} \sum_{k=1}^K \left(\sum_{i=1}^N h_i - \sum_{i=1}^N \hat{h}_i \right)_k \quad (4.18)$$

$$\mathcal{L}_P^{BC} = \frac{1}{K} \sum_{k=1}^K \left(h_1 - \hat{h}_1 \right)_k^2 \quad (4.19)$$

Whereas the rationale behind the VOL strategy stems from an integral equivalence between predicted and true profiles, the BC strategy leverages the physical dependence of the backwater subcritical profile on the weir height.

Hereinafter in this chapter, the term "model" refers to the combination of an MLP approach and a PAT strategy.

4.6 Discussion of Results

This section presents the results obtained from the application of the three different MLP approaches, considering both the purely data-driven (DD) and the PAT strategies (EN, FR).

Solely to familiarize the reader with the physical problem being discussed, Figure 4.6 depicts a typical water level profile encompassing a hydraulic jump, as predicted by the SP, the latter having been trained with and without the inclusion of physical information. The reference solution, namely the profile resulting from the Finite Difference integration, is depicted as well.

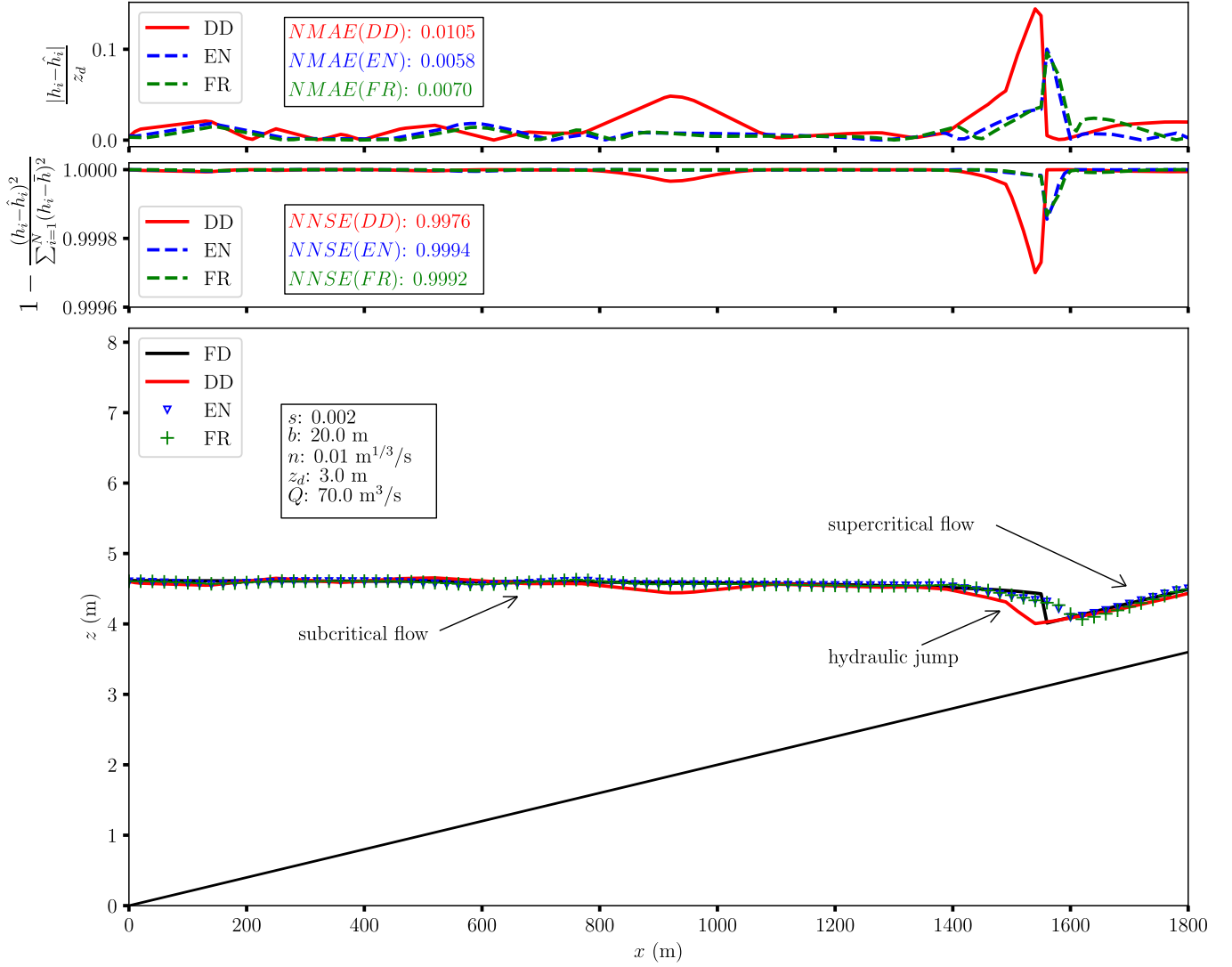


Figure 4.6: Typical outcome of the comparison between the employed training strategies and the reference solution (FD): the purely data-driven baseline (DD) and the PAT-based strategies (EN and FR). The lower panel depicts the water surface profile, as predicted by the models, and the reference solution, highlighting the hydraulic jump separating subcritical and supercritical flow regimes. The two upper panels illustrate the spatial distribution of the Normalized Absolute Error and the Nash–Sutcliffe Efficiency along the channel. The values of the corresponding aggregated metrics for the entire profile (NMAE and NNSE, as in equations (4.20) and (4.21)) and for each model are reported in the insets. Only the case of the SP approach is shown for the sake of clarity.

In paragraph 4.6.1, MLPs are trained on the complete dataset whereas in paragraph 4.6.2 several stress tests are carried out by:

-
- applying models for extrapolation, i.e., seeking predictions beyond the range of values covered during training, a common scenario for technical applications;
 - exploring both overfitting and underfitting conditions by varying either the training dataset size or the MLP complexity.

In all the above tests, the effects of the PAT strategy were evaluated.

Two key metrics are used to evaluate the performance of water profile reconstruction, namely the Normalized Mean Absolute Error (NMAE) and the Normalized Nash-Sutcliffe Efficiency (NNSE). The NMAE, assuming the dam's height z_d as a representative length scale for the flow depth, is defined as

$$\text{NMAE} = \frac{\sum_{i=1}^N |h_i - \hat{h}_i|}{N z_d}. \quad (4.20)$$

The NNSE is formulated as:

$$\text{NNSE} = \frac{1}{(2 - \text{NSE})} \quad (4.21)$$

where

$$\text{NSE} = 1 - \frac{\sum_{i=1}^N (h_i - \hat{h}_i)^2}{\sum_{i=1}^N (h_i - \bar{h})^2} \quad (4.22)$$

and $\bar{h}$ is the average depth of the profile. The rationale behind the scaling in Equation (4.21) is to bound the NSE value between 0 and 1, thus avoiding asymptotic tendency towards negative infinite values. A better-performing model is characterized by a lower NMAE and a higher NNSE.

NNSE weighs large and small errors differently, as clearly depicted in the middle panel of Figure 4.6, and thus provides a more significant measure of model performance. In this case, the NNSE is employed in addition to the NMAE due to its ability to amplify errors made close to the location of the hydraulic jump. Indeed, Equation (4.21) penalizes errors in areas where the flow depth is close to the mean value, and the hydraulic jump is bounded by the two conjugate depths, which are the closest to the mean value within each profile.

4.6.1 Complete training dataset

The cumulative frequency distributions of NMAE and NNSE are here exploited to provide a comprehensive picture of the model performance over the entire test set.

In Figure 4.7 and in Figure 4.8, a boxplot representation is employed to depict the cumulative frequency distribution of NMAE and NNSE for each model over the test set.

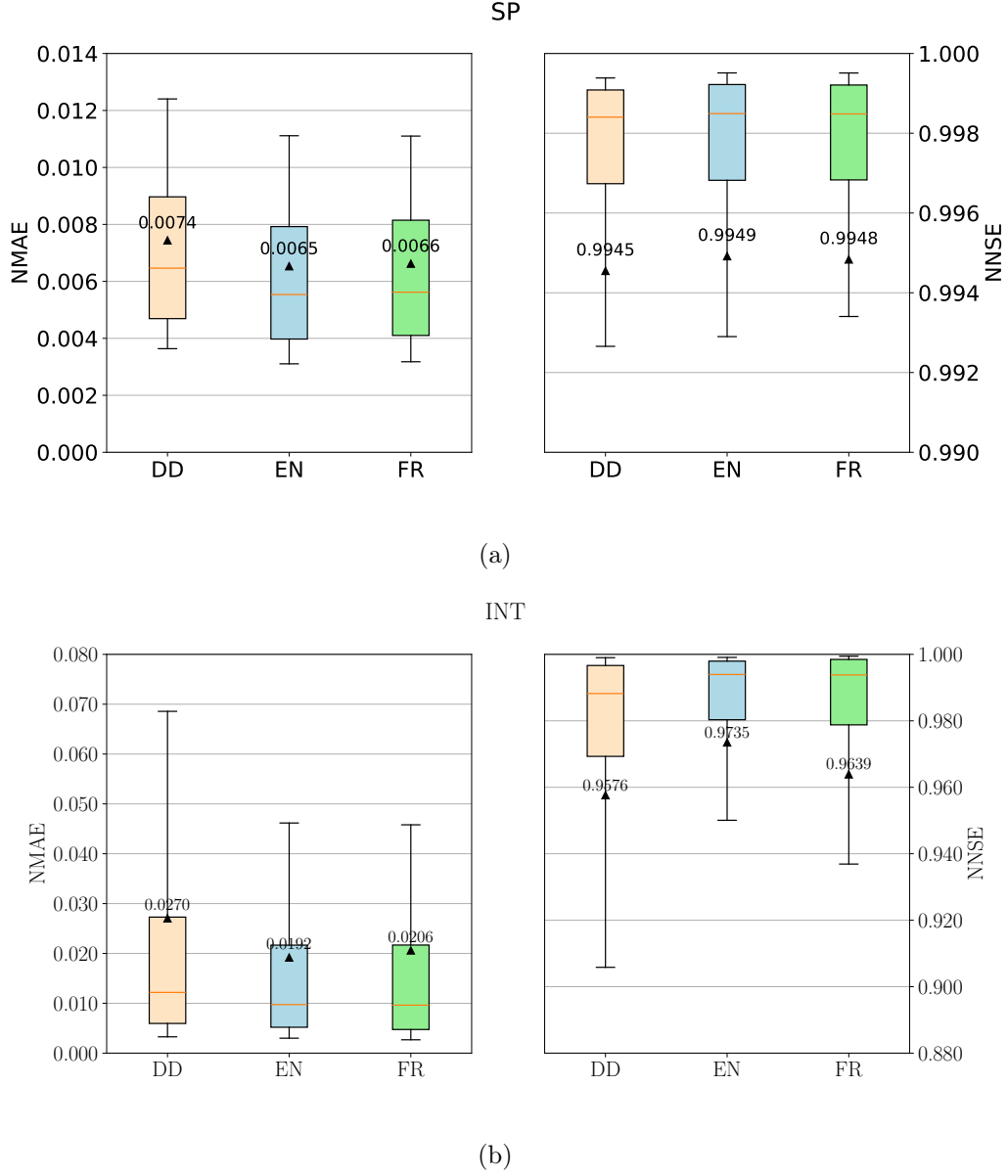


Figure 4.7: Complete training dataset: box plot of the NMAE and NNSE distributions for SP (a) and INT (b) MLP approaches; boxes extend from the 25th to 75th percentile; whiskers are placed at 10th and 90th percentiles; mean values and median values are respectively shown as triangles (with the respective numerical value) and orange segments.

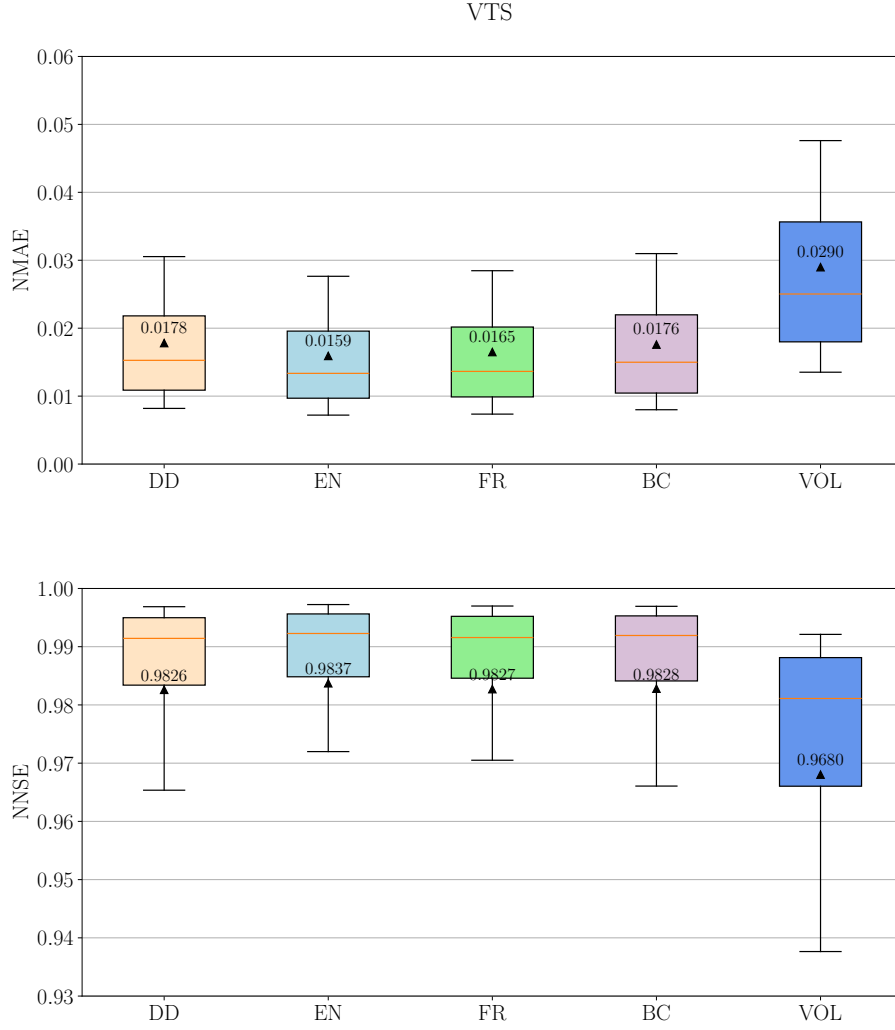


Figure 4.8: Complete training dataset: box plot of the NMAE and NNSE distributions for the VTS approach. Box plot features are the same as in Figure 4.7. VTS also yields better predictions when trained with all types of the proposed physical loss terms, but the volume (VOL) one.

An overall improvement of predictive capabilities is observed employing physical training strategies, along with a reduction in the distribution tails. Notably, these improvements do not entail any significant additional computational burden. Looking at the results, the performance obtained by embedding the Froude number training strategy looks very similar to the one given by the energy training strategy. The similarity of results stems from the fact that both the Froude number and the energy depend on the same variables, namely local depth and velocity.

In the context of VTS, all but the VOL training strategies induce a similar improvement in the model’s performance (Figure 4.8). Constraining the volume of the water profile seems to consistently provide misleading information content which detrimentally impacts the model’s performance. The reason is that the same volume is shared by a large number of possible output profiles. This example unveils that, despite apparently providing additional and physically sound information (i.e. the volume of water in the channel), the training phase may be diverted towards non-physical solutions.

4.6.2 Stress tests: extrapolation, training dataset size, and model complexity

The extrapolation capabilities of the models were examined—an aspect seldom addressed in ML frameworks but essential for ensuring applicability to flood mapping. New water profiles were generated by considering at least one of the values of the five inputs s, b, n, z_d, Q extending by 10% above or below the range covered by the training set. A set of profiles of the same size as the interpolation test set was randomly selected, resulting in a modified test set (referred to as *EX*).

The curves of average values of the cumulative frequency distributions, for both interpolation and extrapolation (*EX*) sets in small data regimes (decreasing the size of the training set by up to two orders of magnitude), are illustrated in Figure 4.9. Whereas, as expected, the performances progressively worsen by reducing the training set, the physical training strategy can almost always achieve better predictions (e.g., the improvement in NMAE is typically between 10% and 15%), thereby effectively enhancing generalization capabilities. In some isolated cases for the INT approach, performance was not improved with the physical training strategy, most likely due to a particularly unfavorable initialization of the weights. The consistent drop in overall performance for extrapolation cases, compared to interpolation cases, is expected. This performance deterioration is more pronounced with the full dataset and less significant under small data regime conditions. However, models trained with physical information still show improved performance in extrapolation scenarios, with the NMAE improvement ranging in most cases from 5% to 20%.

These results are of crucial importance for the perspective application to flood mapping. Indeed, in such field, predictions are often sought not only for scenarios falling within the quantitative range of available observations, though previously unexplored, but also for cases featuring values of the observed quantities falling out of the range of the recorded series.

An additional analysis was conducted to assess the impact of model complexity on the training approach. The total number of trainable parameters was varied while preserving the three, equally sized layer architecture, and in both a data-rich (complete dataset) and a data-scarce (5% of the complete dataset) regime. Results are summarized in Figure 4.10.

When dealing with the complete training dataset, an expected trend emerges as the number of trainable parameters decreases: performance consistently deteriorates across all three architectures. This discernible pattern clearly unveils the occurrence of underfitting. Oppositely to overfitting, underfitting occurs when the model lacks complexity compared to the information-rich training data. The lower complexity architectures now strive to capture the variability spanned by the complete dataset. In that underfitting regime, it is noteworthy that the introduction of a PAT strategy does not yield a consistent improvement in generalization results. This behavior is the consequence of the requirement for the low-complexity model, which is already struggling to discern patterns in the data, to fulfill additional constraints, such as the physical ones.

In the context of a small data regime, however, different behaviors emerge. Model performance can lose dependence on the number of parameters (as for INT) and, in some cases, it even deteriorates as the complexity of the models increases (as for VTS). Even in this clear overfitting scenario, models enhanced with the PAT strategy consistently ensure an improvement in performance. Indeed, the physical term, involving both input and output data, akin to a form of data augmentation, provides the model with further insights into the nature of the system to be interpreted.

4.7 Conclusions

The main findings of this chapter, which presents a synthetic one-dimensional free-surface hydraulic modeling case, can be summarized as follows:

- The proposed PAT methodology, similarly to data augmentation, enriches the informational content of the dataset.
- PAT systematically improves model generalization across the three examined MLP architectures, demonstrating the flexibility of the approach. Both the Froude-based and energy-based losses yield comparable improvements.
- The benefits of PAT extend to extrapolation, with the largest gains observed in data-scarce scenarios.
- Training strategies based on global constraints must be used cautiously: the volume-based loss provides misleading gradients, indicating that not all physically inspired terms are equally informative for the learning task.
- In underfitting regimes, PAT may offer little or no benefit, as the model lacks sufficient capacity to jointly satisfy both data and physical relationships.

These findings provide a strong motivation for applying the same approach to more complex hydraulic and environmental modeling tasks.

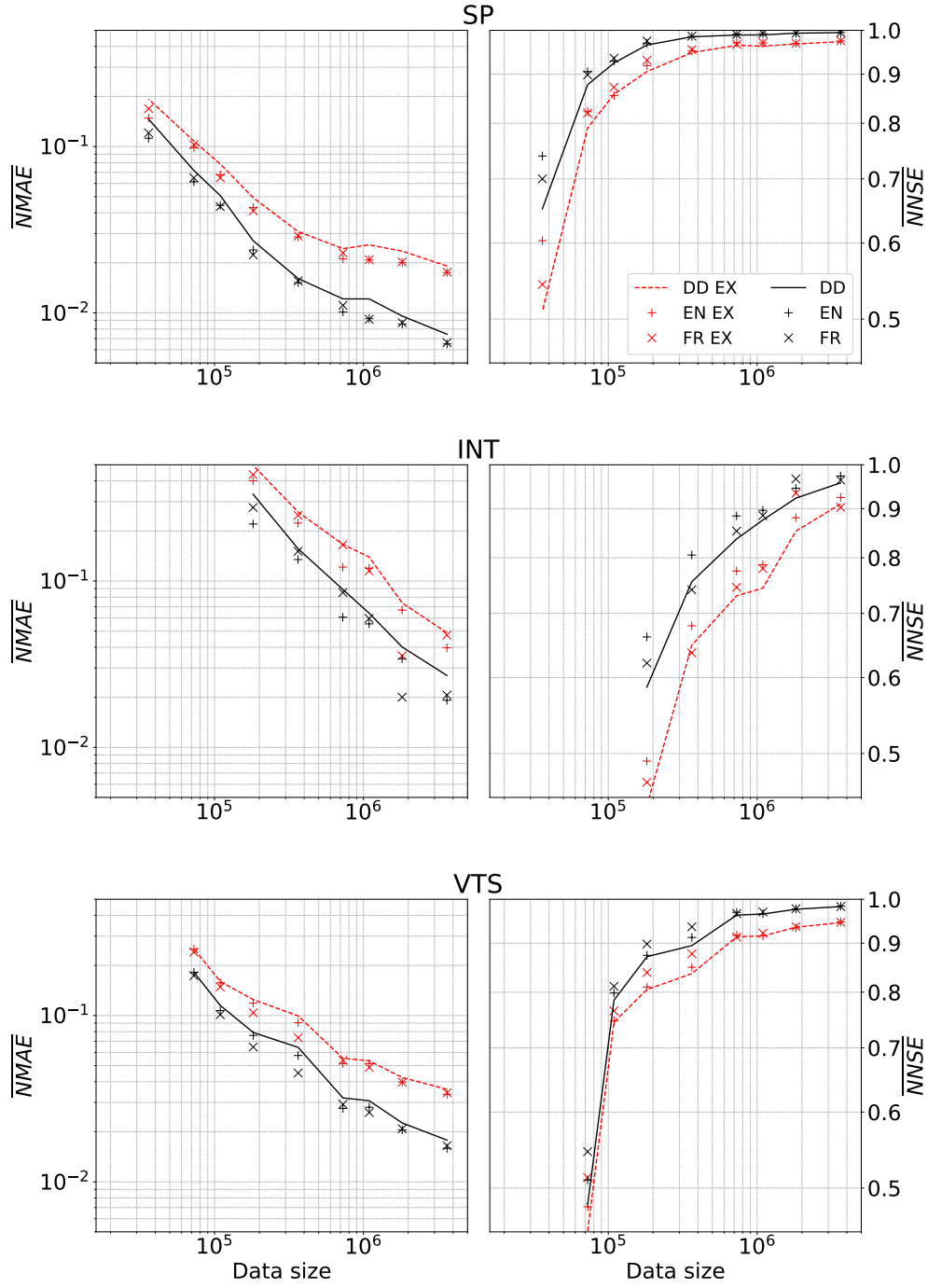


Figure 4.9: Stress tests: mean values of NMAE and NNSE on the test set against the size of the training dataset for each architecture. Black traces and markers represent models' results on the test dataset fully contained in the range of the training one; red ones regard the test dataset containing values out of the range of the training dataset (extrapolation, EX series). Although performance is lower for extrapolated scenarios, the beneficial effect of physical information remains evident.

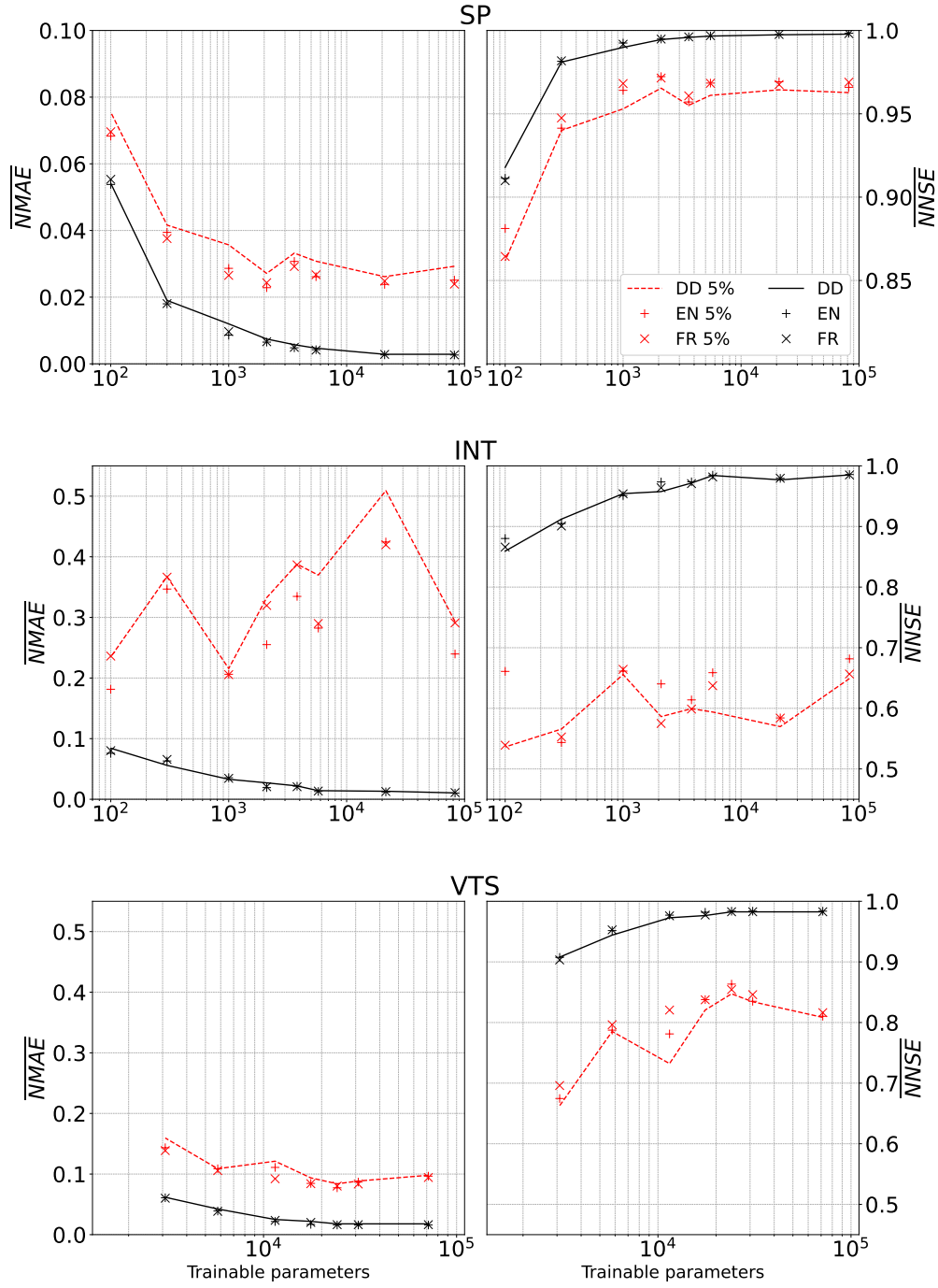


Figure 4.10: Stress tests: mean values of NMAE and NNSE on the test set against the number of trainable parameters. Black traces and markers represent models' results after training on the complete dataset; red ones show models' results after training on a reduced dataset (referred to as 5% in the legend).

Chapter 5

Flood mapping with catchment generalizability

The generalization performance of a CNN-based flood hazard prediction model is evaluated on an unseen catchment without retraining. The model is trained using data from a single catchment, reflecting typical scenarios of limited data availability. The impact of the proposed PAT strategy is assessed.

5.1 Introduction

Floods are among the most devastating natural events, and reliable hazard assessment is a primary goal for effective risk reduction (EU Floods Directive 2007/60/EC). River floods cause significant damage worldwide each year. Accurate identification of flood-prone areas is essential for risk management and land-use planning. In recent years, DL models have been increasingly applied to river flood prediction for the reasons and with the limitations discussed in Chapter 1.

Particular attention is given here to the fact that most DL models, when applied to new domains (i.e., different catchments), still require complete retraining, revealing a broader limitation in their ability to generalize beyond the data on which they were trained [Manchingal and Cuzzolin, 2025]. Developing models capable of maintaining predictive accuracy across multiple basins represents a major step toward the goal of rapid and transferable flood mapping, paving the way for more effective DL

applications in hydrology and for practical implementations such as flood hazard estimation in large ungauged areas.

Among the most relevant developments in this direction are U-Net architectures and patch-based approaches designed to enhance geometric generalization, as demonstrated by Guo et al. [2021] and later refined in Guo et al. [2022]. Building upon these studies, this chapter presents a DL framework specifically designed to achieve cross-basin transferability, with the goal of predicting river flood hazard maps in entirely unseen catchments.

The challenge of limited training data is particularly emphasized and taken to the extreme: the convolutional encoder–decoder network is trained on data from a single basin under multiple synthetic rainfall events and evaluated on a different, unseen basin using a one-to-one basin transfer setup.

Moreover, the analysis aims to assess the improvement in generalization provided by the PAT strategy (Section 2.9.2), which seeks to exploit the physical content implicitly contained in the data. The approach, which has already been tested in Chapter 4 on a simple one-dimensional hydraulic problem, is here extended to a complex two-dimensional case. Its applicability is enabled by the method’s versatility and its minimal impact on the overall computational burden.

In addition, the joint prediction of water depth and flow velocity marks a further advancement over previously reported approaches.

5.2 Methods

The aim of the employed methodology is to forecast water depth and flow velocity magnitude fields, given topographic data and rainfall forcing. With reference to Figure 5.1, the model is trained on data from a single basin and multiple rain events, and is then tested on a different, unseen basin using a one-to-one basin transfer setup. Following the patch-based approach originally proposed by Guo et al. [2022] to handle different catchment geometries, it is recognized that this strategy is also essential for enabling a CNN-based model to learn local characteristics independently of their position within the catchment. Within this framework, the PAT strategy is employed.

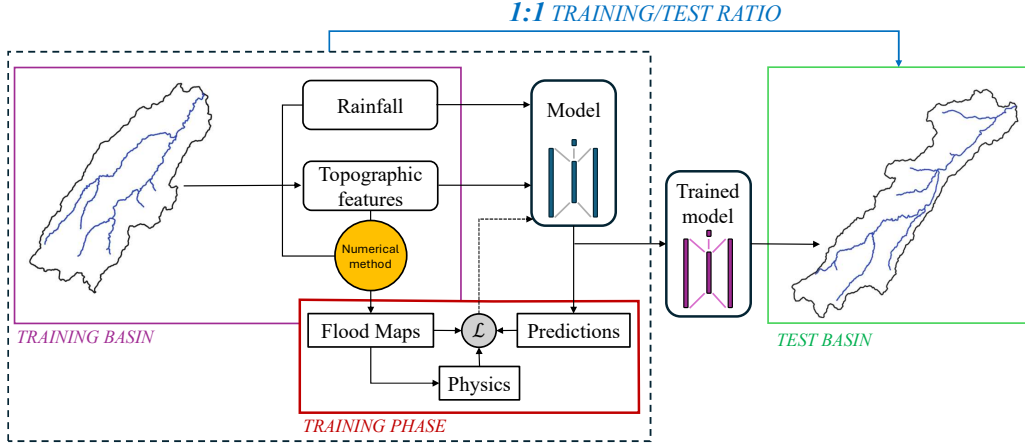


Figure 5.1: Overview of the 1:1 training/test ratio workflow: the topography of the training basin is combined with rainfall to generate reference flood maps (purple box) by means of a numerical model. The training phase (red box) employs a loss function $\mathcal{L}$ encompassing terms which encode the PAT strategy. Once trained, the model is applied, without further training, to an unseen test basin (green box).

5.2.1 Topographic data

The input features include four topographic variables that are assumed to influence flood dynamics, namely elevation, slope, aspect (i.e. direction of the steepest slope) and Topographic Wetness Index (TWI), which is defined as:

$$\text{TWI} = \ln(A / \tan \beta) \quad (5.1)$$

where A is the up-slope contributing area and β is the local slope in radians.

Elevation, slope, and aspect were selected based on the works of Guo et al. [2021] and Guo et al. [2022], while TWI was included due to its effectiveness in identifying areas of high water depth [Löwe et al., 2021] and its use in flood susceptibility studies [Wang et al., 2020]. Among the four selected inputs, TWI is the only feature that provides a basin-scale hydrological perspective, offering contextual information about a patch location within the overall catchment.

Similar to the DL architecture, adopting the same input quantities as previous studies facilitates the evaluation of the modified training procedure. Other geomorphic indices, such as the Geomorphic Flood Index [Manfreda et al., 2025], have also been shown to be highly informative for flood-proneness assessment and could be considered

in future developments of the framework.

These features are encoded as separate channels in the model’s input and normalized using the Standard Scaler [Chollet et al., 2015] before training. Normalization is essential to ensure stable and efficient model convergence, avoid feature dominance due to differing scales [Ali et al., 2014], and improve generalization. Moreover, normalizing input data guarantees model invariance to geometric conformal linear transformations.

5.2.2 Patch-based approach and reconstruction methods

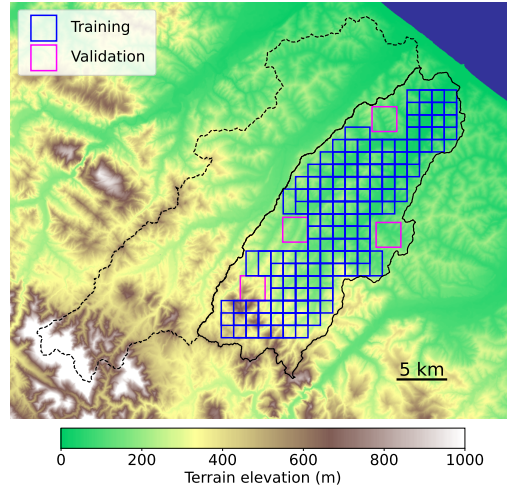
Patches are extracted from the spatial domain of the input topographic maps, along with the corresponding flood maps (i.e., water depth and velocity).

Validation patches are randomly selected within the training basin. These patches are non-overlapping and strictly contained within the basin boundaries to avoid edge effects and ensure consistency. Training patches are extracted from the remaining areas of the basin, ensuring they are also entirely contained within the basin boundaries (Figure 5.2a). Unlike validation patches, training patches are allowed to overlap, increasing the diversity of spatial patterns seen during training and improving the model’s generalization capability.

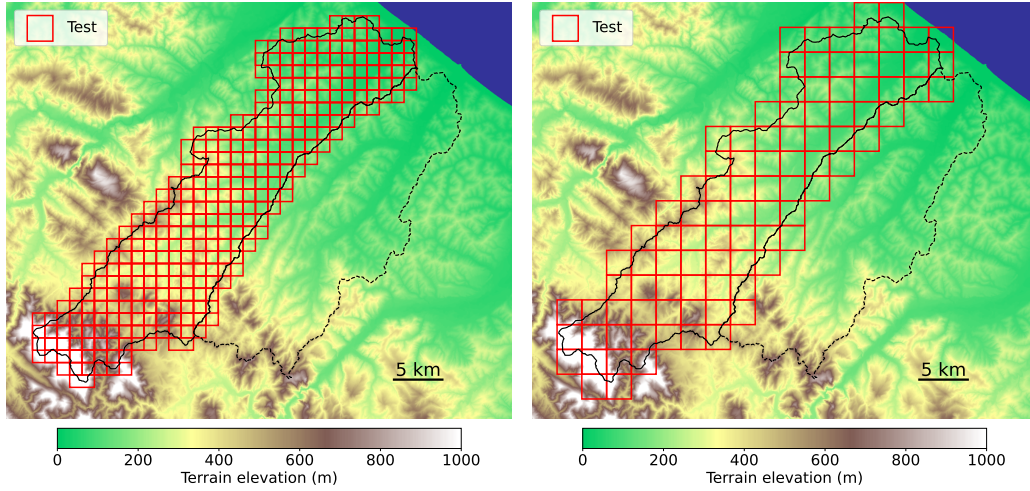
Predicting flood maps over the unseen basin (hereafter referred to as the test basin) will proceed by extracting either overlapping or non-overlapping juxtaposed patches (Figures 5.2b and 5.2c). Within the former strategy, the prediction in the overlapping regions is computed as either the mean or the maximum value among the patches. In the latter case, the final map is the simple union of the patches. It is important to note that the spatial resolution of the topographic data used for both training and testing must be consistent to ensure that the extracted patches represent comparable physical scales.

5.2.3 Loss Functions

This section outlines the loss functions used to train the models. A distinction is made between purely data-driven approaches and those augmented with physical quantities.



(a) Training and validation patches are selected within the Misa basin (continuous border). Validation patches are shown in magenta, whereas overlapping training patches are shown in blue. The test basin is depicted with a dashed border.



(b) Cesano basin (continuous border) with over- (c) Cesano basin (continuous border) with non-lapping test patches. The Misa (training) basin overlapping test patches. The Misa (training) basin is depicted with a dashed border.

Figure 5.2: Distribution of the selected patches over the training and test basins

Purely Data-Driven

The most common loss function used in DL models is the MSE as in Equation 2.14. Here, the output dimensionality is $C = 2$, and N is restricted to the total number of

valid data points N_v :

$$\text{MSE} = \frac{1}{2N_v} \sum_{i=1}^{N_v} \|\mathbf{y}_i - \hat{\mathbf{y}}_i\|_2^2 \quad (5.2)$$

Valid data points are those located within the basin boundaries, extracted across all evaluated patches.

To emphasize prediction errors occurring in regions characterized by larger water depths or velocities—i.e., areas more likely associated with flooding—and mitigate the imbalance toward non-flooded regions [Guo et al., 2021], a modified MSE loss can be employed:

$$\text{MSE}_M = \frac{1}{N_v C} \sum_{i=1}^{N_v} \sum_{c=1}^C w_{i,c} (y_{i,c} - \hat{y}_{i,c})^2 \quad (5.3)$$

where the weights $w_{i,c}$ are defined as:

$$w_{i,c} = \begin{cases} 1 & \text{if } \hat{y}_{i,c} < \tau \\ 1 + (\hat{y}_{i,c} - \tau)^\alpha & \text{otherwise} \end{cases}$$

with τ and α as hyperparameters to be set.

PAT strategy

To incorporate hydrodynamic knowledge into training, the loss function is modified by adding a regularization term based on relevant physical quantities to the purely data-driven loss, as shown in Equation 2.31.

From Equation 4.3, the specific energy is locally defined, in this case, as:

$$\mathcal{E} = h + \frac{U^2}{2g} \quad (5.4)$$

where U is the depth-averaged flow velocity magnitude.

As done in Section 4.5, the energy-based loss combines the data-driven loss (MSE) with the MSE computed on the specific energy values:

$$\mathcal{L}_{\mathcal{E}} = \lambda \cdot \text{MSE}(\mathbf{y}, \hat{\mathbf{y}}) + (1 - \lambda) \cdot \text{MSE}(\mathcal{E}(\mathbf{y}), \mathcal{E}(\hat{\mathbf{y}})) \quad (5.5)$$

An alternative version of Equation 5.5 can be defined by replacing the standard MSE (Equation 5.2) in the data-driven term with the modified loss introduced in

Equation 5.3:

$$\mathcal{L}_{\mathcal{E},M} = \lambda \cdot \text{MSE}_M(Y_{\text{true}}, Y_{\text{pred}}) + (1 - \lambda) \text{MSE}(\mathcal{E}(\mathbf{y}), \mathcal{E}(\hat{\mathbf{y}})) \quad (5.6)$$

The weighting strategy is applied only to the data term, while the energy-based component is still evaluated using the standard MSE.

The Froude number, defined for the case of Chapter 4 as in Equation 4.6, is here locally defined as:

$$Fr = \frac{u}{\sqrt{gh}} \quad (5.7)$$

Analogously, a Froude-augmented loss function can be derived:

$$\mathcal{L}_{\text{Fr}} = \lambda \cdot \text{MSE}(\mathbf{y}, \hat{\mathbf{y}}) + (1 - \lambda) \cdot \text{MSE}(Fr(\mathbf{y}), Fr(\hat{\mathbf{y}})) \quad (5.8)$$

Alternatively, the loss can be formulated by replacing the standard MSE (Equation 5.2) in the data-driven term with the modified loss (Equation 5.3)

$$\mathcal{L}_{\text{Fr},M} = \lambda \cdot \text{MSE}_M(\mathbf{y}, \hat{\mathbf{y}}) + (1 - \lambda) \cdot \text{MSE}(Fr(\mathbf{y}), Fr(\hat{\mathbf{y}})) \quad (5.9)$$

As in the experiments of Chapter 4, specific energy and Froude number are introduced as physically meaningful combinations of variables, reflecting prior expert knowledge of the system. However, they do not impose strict physical constraints; instead, they act as additional regularization terms that guide the model toward physically plausible solutions by emphasizing relevant flow characteristics.

It is worth mentioning that the above physical quantities represent just two possible choices, given that alternative physically-inspired losses could be devised to reflect other aspects of the underlying physics.

5.2.4 DL Architecture

The adopted DL model (represented in Figure 5.3) is an encoder-decoder CNN structure inspired by the U-Net architecture (Section 2.7.1), tailored for the task of flood prediction using topographic and rainfall data [Guo et al., 2021]. The rationale for this choice is to enable a clear assessment of the proposed novel training procedure. The model is designed to take as input two distinct data types:

- (i) 2D features encompassing topographic characteristics of the patch;
- (ii) a rainfall input, which generates the flood.

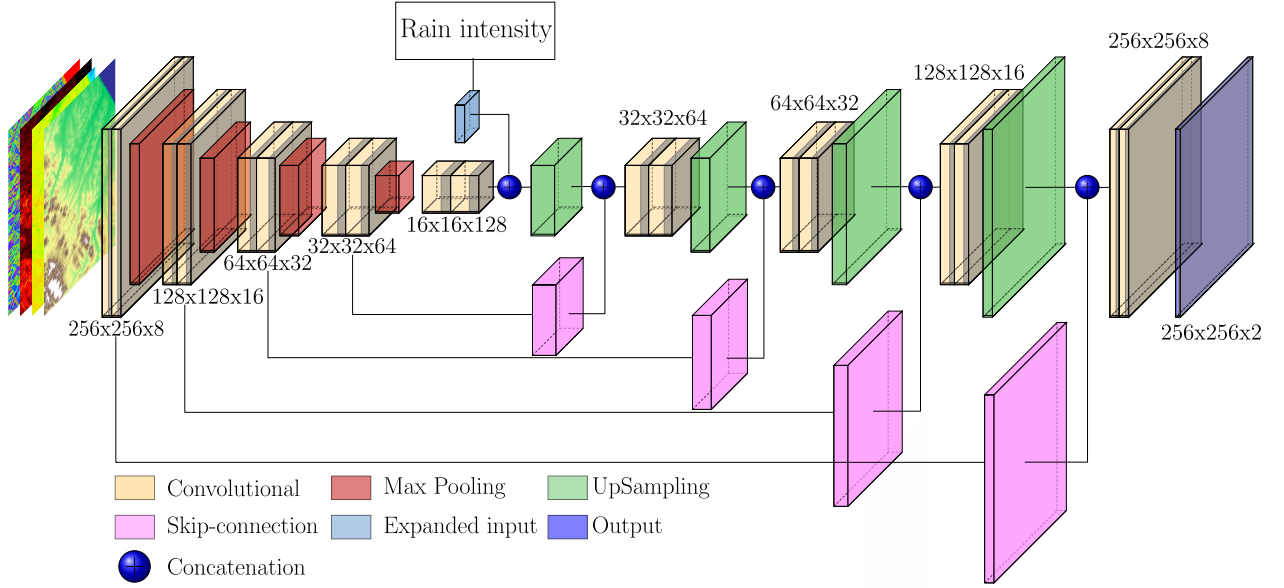


Figure 5.3: Architecture of the employed DL model. The encoder consists of four consecutive blocks, each containing two convolutional layers followed by a max pooling layer. The bottleneck includes two convolutional layers, to which the expanded rainfall input is concatenated. In the decoder, each transposed convolution is followed by the concatenation of the corresponding skip connection output from the encoder, and two additional convolutional layers. Eventually, convolutional layers provide the prediction for the two outputs, namely depth and flow velocity magnitude. The '+' symbols represent concatenation between features. The dimensions reported within each block represent the output size of the convolutional layers.

Encoder

The encoder progressively reduces the spatial dimensions while increasing the feature representation through a series of four blocks, each consisting of two convolutional layers followed by a max-pooling layer. Each encoder block doubles the number of filters compared to the previous block, starting from a base number of eight filters.

Bottleneck

Two additional convolutional layers form the encoder bottleneck. To condition the output on rainfall intensity, the rainfall scalar value is reshaped, up-sampled, and concatenated with the encoder bottleneck, enabling the model to incorporate rainfall-dependent behavior into the latent representation before decoding.

Decoder

The decoder up-samples the feature maps using Transpose Convolution layers (Deconvolutions) and incorporates skip connections from the encoder. Each decoder block mirrors its corresponding encoder block and consists of a transpose convolution that doubles the dimensions of the previous output, a concatenation with the corresponding encoder feature map (via skip connection), and two convolutional layers with the same number of filters as in the corresponding encoder block. The number of filters halves at each decoder level, in reverse order relative to the encoder.

Skip-connections are employed because the use of small convolution kernels is inherently at odds with the objective of preserving a larger perspective on the input data. While small kernels are effective at extracting local features, they may fail to capture larger-scale topographic patterns that are relevant to flood dynamics. By enabling information from shallow layers to bypass intermediate processing stages and directly contribute to deeper layers, skip-connections facilitate the integration of fine-scale and broad-scale spatial information, thereby enhancing the model’s ability to represent hydrodynamic processes across multiple spatial scales.

The final output layer is a convolutional layer with two output channels and ReLU activation, producing the predicted flood depth and flow velocity magnitude maps. The use of ReLU [Nair and Hinton, 2010] ensures that the predicted values

are non-negative.

Code development

The DL architecture was developed using TensorFlow (v2.18) [Abadi et al., 2015] and Keras (v3.8) [Chollet et al., 2015], with supplementary machine learning utilities provided by PyTorch (v2.1). The experiments were conducted on a computational system equipped with an NVIDIA GeForce RTX 3090 GPU (24 GB VRAM), operating under CUDA version 12.2.

5.3 Simulations setup

5.3.1 Hydraulic model equations

The numerical dataset used to train and evaluate the models was generated using the hydraulic software HEC-RAS, which solves the two-dimensional SW equations [U.S. Army Corps of Engineers, Hydrologic Engineering Center, 2021], obtained by depth-integrating the NS equations for free-surface flow under the assumption that the horizontal length scale is much greater than the vertical length scale. This assumption implies that the vertical velocity of the fluid is small compared to its horizontal velocity components, and that vertical pressure gradients are hydrostatic. These equations express the conservation of mass:

$$\frac{\partial h}{\partial t} + \frac{\partial(hu_x)}{\partial x} + \frac{\partial(hu_y)}{\partial y} = r(x, y, t) - f(x, y, t) + s_m(x, y, t) \quad (5.10)$$

and momentum:

$$\frac{\partial(hu_x)}{\partial t} + \frac{\partial}{\partial x}(hu_x^2 + \frac{1}{2}gh^2) + \frac{\partial(hu_xu_y)}{\partial y} = -gh\frac{\partial z_b}{\partial x} + S_{fx} + S_{ext,x} \quad (5.11)$$

$$\frac{\partial(hu_y)}{\partial t} + \frac{\partial(hu_xu_y)}{\partial x} + \frac{\partial}{\partial y}(hu_y^2 + \frac{1}{2}gh^2) = -gh\frac{\partial z_b}{\partial y} + S_{fy} + S_{ext,y} \quad (5.12)$$

where:

- $h(x, y, t)$ is the water depth;
- $u_x(x, y, t)$ and $u_y(x, y, t)$ are the depth-averaged velocity components;
- $z_b(x, y)$ is the bed elevation;

-
- $r(x, y, t)$ is the rainfall intensity, which acts as a source term;
 - $f(x, y, t)$ represents infiltration losses;
 - $s_m(x, y, t)$ represents lateral inflows or point sources;
 - S_{fx} and S_{fy} are the bed friction terms, usually modeled via the Manning equation:

$$S_{fx} = -g n^2 \frac{u_x U}{h^{1/3}}, \quad S_{fy} = -g n^2 \frac{u_y U}{h^{1/3}}, \quad (5.13)$$

- n is Manning's roughness coefficient;
- $U = \sqrt{u_x^2 + u_y^2}$ is the velocity magnitude;
- $S_{ext,x}, S_{ext,y}$ represent external forces such as wind stress or Coriolis effects.

The equations are discretized on a 2D mesh using an implicit finite-volume scheme, with adaptive time-stepping to satisfy the Courant–Friedrichs–Lewy (CFL) stability criterion.

5.3.2 Dataset generation

The two chosen catchments, namely the Misa and Cesano rivers, whose morphometric description is reported in Table 5.1, are located in central Italy (Figure 5.4). The rationale behind the choice of these basins is twofold. First, these catchments have experienced several flood events in recent years, including two major disasters in May 2014 and September 2022 [Corti et al., 2024]. Secondly, although belonging to the same regional area, they retain variations in the distributions of the input quantities that quantify the effective morphological difference between the two basins (see Figure 5.5). The Misa basin was used for both training and validation, while the Cesano basin was reserved for testing. All patches (training, validation, and test) share the same size (256×256 pixels). Within the Misa basin, four validation patches were randomly selected, as described in Section 5.2.2 and illustrated in Figure 5.2. The training patches were extracted by fixing a 50% overlap, ensuring that all patches were entirely contained within the basin boundaries.

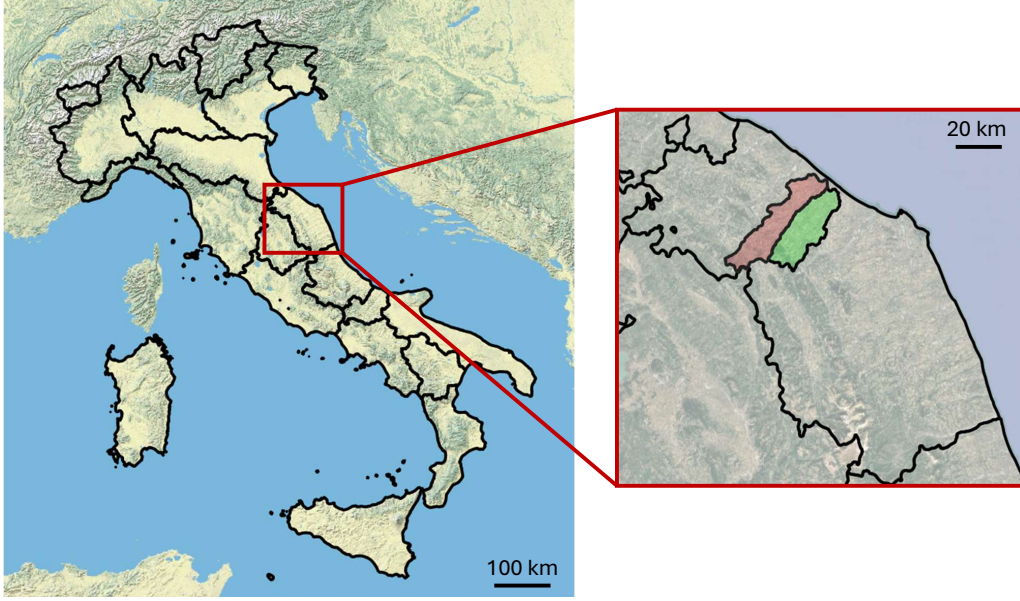


Figure 5.4: Geographical location of the Misa (green) and Cesano (red) basins in central Italy

These differences, exemplified here by variations in the input distributions but further exacerbated by the underlying spatial heterogeneity between the basins, represent a meaningful stress test for the generalization capability of the model.

The Digital Elevation Model (DEM) was obtained from Tinitaly 1.1 [Tarquini et al., 2023], which is publicly available at a 10-meter cell resolution.

Table 5.1: Morphometric characteristics of the training and test basins

Characteristic	Training Basin	Test Basin
Main river name	Misa	Cesano
Area [km ²]	375.7	412.4
Perimeter [km]	145.3	190.3
Mean elevation [m a.s.l.]	197.1	309.3
Minimum elevation [m a.s.l.]	6	9
Maximum elevation [m a.s.l.]	821	1702
Main channel length [km]	46.5	62.1

For both basins, a series of numerical simulations was performed using spatially

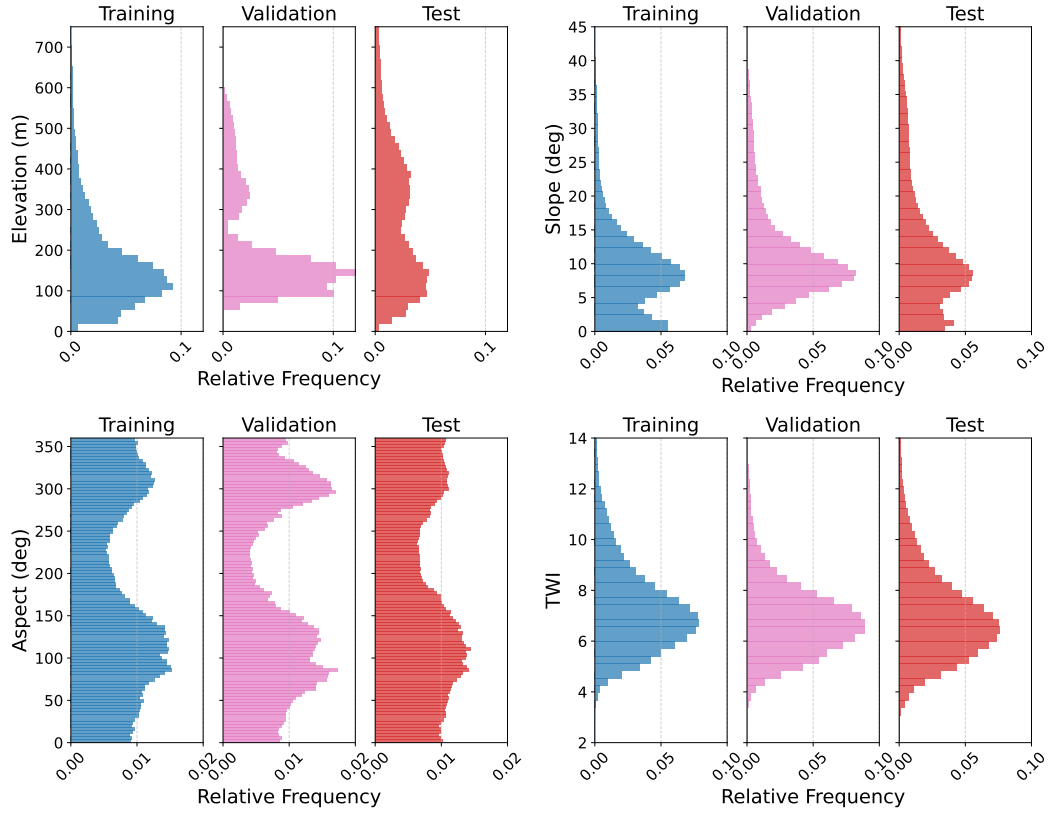


Figure 5.5: Distributions of input features (Elevation, Slope, Aspect, TWI) across training, validation, and test datasets. While TWI shows similar shape across the datasets, consistent differences can be noticed between other features, supporting the choices of the basins for the present study.

and temporally uniform rainfall inputs of different intensities. Specifically, rainfall intensities ranged from 2 mm/h to 10 mm/h in 1 mm/h increments, plus an additional simulation at 12 mm/h, resulting in a total of ten rainfall scenarios. A direct attribution of return periods to each rainfall scenario would require a dedicated frequency analysis, which is beyond the scope of this work. As detailed in Section 5.4.3, a rainfall intensity of 5.5 mm/h for the Cesano basin corresponds to the net rainfall producing a discharge associated with a 200-year return period at the basin outlet, according to previous studies [Consorzio di Bonifica delle Marche, 2019]. Therefore, the adopted range of rainfall intensities can be interpreted as spanning from moderate to extreme hydrological forcing conditions relative to this reference value.

The simulations were run until steady-state conditions were reached, with no infiltration considered. This approach was adopted to isolate the influence of topographic characteristics and to generate a physically meaningful equilibrium state for each rainfall intensity across the entire basin. This time-independent representation differs from traditional hazard mapping procedures, which typically produce spatial flood envelopes based on the maximum inundation extent or depth over time, and may therefore not correspond to a physically consistent hydrodynamic state.

Infiltration losses were neglected; therefore, the rainfall intensity should be considered as the net rainfall contributing to surface flow. Boundary conditions were defined using a stage–discharge relationship at the catchment outlet. The Manning roughness coefficient was set to a uniform value of 0.06, identical for both basins, to simplify the problem and avoid including it as an additional input to the DL model. The simulations were performed using the HEC-RAS 2D solver with the diffusive wave approximation. Since the aim of this part is to generate a controlled, physically consistent dataset, no calibration was needed, and a uniform Manning coefficient was adopted to ensure consistency across all simulations.

5.3.3 Hyperparameters

All hyperparameters were set before the training phase. Both the encoder and decoder components utilize LeakyReLU activations (Table 2.3) [Dubey and Jain,

2019] with a slope coefficient [Hochreiter and Schmidhuber, 1997] $\alpha = -0.1$. All convolutional layers employ kernels of size 6×6 . The models are compiled using the Adam optimizer [Kingma and Ba, 2014] with a learning rate of 5×10^{-5} . Model training is guided by the validation loss, and the best-performing model is retained. An Early Stopping criterion halts training when the validation loss stops improving, reducing unnecessary computation and mitigating potential overfitting. Training is performed with a batch size of 2 over a maximum of 100 epochs.

The model architecture, hyperparameters, and random seed have been fixed throughout the study to ensure fair and consistent comparisons across different models. The hyperparameter λ , which weighs the contribution of the physical information in the modified loss functions (Equations (5.5), (5.6), (5.8), and (5.9)), was selected individually for each model. Its value was determined by first identifying the appropriate order of magnitude within the interval $[0, 1]$, followed by a manual search to refine its optimal setting.

5.3.4 Model evaluation

The problem setup can be approached as either a regression task (Equation 2.2), predicting values of water depth and flow velocity, or a classification task (Equation 2.3) between flooded and non-flooded areas.

To assess the quality of the spatial reconstruction over the entire basin, the final flood maps are reconstructed from the predicted patches and evaluated using the RMSE formula (5.14):

$$\text{RMSE} = \sqrt{\frac{1}{N_d} \sum_{i=1}^N (y_i - \hat{y}_i)^2} \quad (5.14)$$

where y_i and $\hat{y}_i$ are the reference predicted and reference values, respectively, and N_d is the total number of predicted points in the domain.

In addition, the Mean Absolute Error (MAE) is also computed:

$$\text{MAE} = \frac{1}{N_d} \sum_{i=1}^N |y_i - \hat{y}_i| \quad (5.15)$$

The above metrics can bias evaluation when applied to largely dry areas; to overcome such issue, their modified counterparts, Modified MAE (MMAE) and

Modified RMSE (MRMSE), respectively, are formulated by restricting the evaluation to hydraulically relevant areas, i.e., the set F of pixels predicted as flooded by the numerical model:

$$\text{MMAE} = \frac{1}{N_f} \sum_{i=1}^{N_f} |y_i - \hat{y}_i| \quad (5.16)$$

$$\text{MRMSE} = \sqrt{\frac{1}{N_f} \sum_{i=1}^{N_f} (y_i - \hat{y}_i)^2} \quad (5.17)$$

where N_f is the size of F .

For the classification task, flooded areas were defined as regions where water depth exceeds a threshold of 0.1 m [Zhou et al., 2021], using a depth-based classification. Based on this binary classification, the following classification metrics are employed:

- **Accuracy:**

$$\text{Acc} = \frac{TP + TN}{TP + TN + FP + FN} \quad (5.18)$$

- **Precision:**

$$\text{Prec} = \frac{TP}{TP + FP} \quad (5.19)$$

- **Recall:**

$$\text{Rec} = \frac{TP}{TP + FN} \quad (5.20)$$

- **F1-score:**

$$\text{F1} = 2 \cdot \frac{\text{Prec} \cdot \text{Rec}}{\text{Prec} + \text{Rec}} \quad (5.21)$$

- **Critical Success Index:**

$$\text{CSI} = \frac{TP}{TP + FP + FN} \quad (5.22)$$

where TP , TN , FP , and FN denote the number of true positives, true negatives, false positives, and false negatives, respectively. The choice to include several metrics lies in the presence of class imbalance.

Accuracy (Equation 5.18) quantifies the overall proportion of correct predictions. Precision (Equation 5.19) measures the proportion of predicted flooded areas that are correctly identified, thus reflecting the model's reliability in positive classifications. Recall (Equation 5.20), on the other hand, assesses the model's sensitivity by

capturing its ability to identify all actual flooded regions. The F1-score (Equation 5.21) represents the harmonic mean of Precision and Recall, offering a balanced assessment of classification performance. Finally, the Critical Success Index (CSI, Equation 5.22)), which excludes true negatives from consideration, is particularly suitable for evaluating model performance in highly imbalanced contexts, such as flood extent prediction. All the above metrics range between 0 and 1, with increasing values corresponding to better performance.

5.4 Results

This section evaluates the performance of the models introduced in Section 5.2.3, trained on one basin and tested on an unseen basin, to assess the impact of PAT on generalization across topographies. The assessment is performed by comparing results both patch by patch and after patch stitching for flood map reconstruction over the entire basin.

Subsequently, a visual comparison with reference official flood hazard maps from local authorities is presented to qualitatively validate the prediction framework.

5.4.1 Comparison of Models Outcomes: patch-level

Table 5.2 presents the RMSE values for each model, limited to the overlapping dataset for the sake of brevity. Note that at patch-level, N in Equation (5.14) is replaced by N_p , which denotes the total number of points within the patches. The first column of Table 5.2 contains the names that will be employed in the following to refer to the models; the second column reports the associated loss function.

The percentage variations indicate the improvement or deterioration relative to the corresponding purely data-driven baseline model (DD1 for MSE loss or DD2 for MSE_M loss).

The results show appreciable improvements in depth prediction when the additional physical term is introduced into the loss functions, suggesting that PAT can support better generalization.

It is worth highlighting that validation errors are consistently lower than those of the test set, since validation patches, although unseen like the test ones, are extracted

Table 5.2: Validation and test (overlapping dataset) RMSEs for depth and velocity. Percentage variations refer to the corresponding purely data-driven model (DD1, DD2). Errors are averaged across all rainfall intensities.

Model		Depth RMSE (m)		Velocity RMSE (m/s)	
Name	Loss	Validation	Test	Validation	Test
DD1	Eq. (5.2)	0.2174	0.3642	0.1080	0.1806
EN1	Eq. (5.5)	0.2078 (-4.4%)	0.3534 (-2.7%)	0.1111 (+2.9%)	0.1824 (+1.0%)
FR1	Eq. (5.8)	0.2127 (-2.2%)	0.3615 (-0.7%)	0.1068 (-1.1%)	0.1804 (-0.1%)
DD2	Eq. (5.3)	0.2332	0.3597	0.1135	0.1841
EN2	Eq. (5.6)	0.2143 (-8.1%)	0.3560 (-1.0%)	0.1124 (-1.0%)	0.1856 (+0.8%)
FR2	Eq. (5.9)	0.2213 (-5.1%)	0.3488 (-3.0%)	0.1103 (-2.8%)	0.1835 (-0.3%)

from the training basin. Despite being randomly chosen, the consistently better predictions on validation patches suggest that the morphologies of the validation patches retain similarities with the training ones. This behaviour indicates that geomorphological characteristics may play a significant role in model performance, with intra-basin variability being smaller than inter-basin variability. While no explicit sensitivity analysis on geomorphological features has been carried out in this study, the observed performance gap between validation and test sets supports this interpretation. A more rigorous assessment (e.g., cross-validation or leave-one-out experiments) would be necessary to quantify the extent to which model accuracy is affected by geomorphological differences. This certainly deserves future in-depth analysis with a particular focus on the possible fruitful exploitation of basin similarities.

Regarding the prediction of flow velocities, the impact of the modified losses is less consistent. The Froude-based models yield modest improvements, while the energy-based losses occasionally lead to slight deterioration.

5.4.2 Comparison of Model Outcomes: Full basin map reconstruction

To visually assess the model's performance at the basin scale, we present full-map predictions of depth (Figure 5.6) and velocity (Figure 5.7). For selected rainfall scenarios, the reference values, model predictions, and corresponding error maps are shown. This visualization allows for assessing both the overall prediction quality and the spatial distribution of errors.

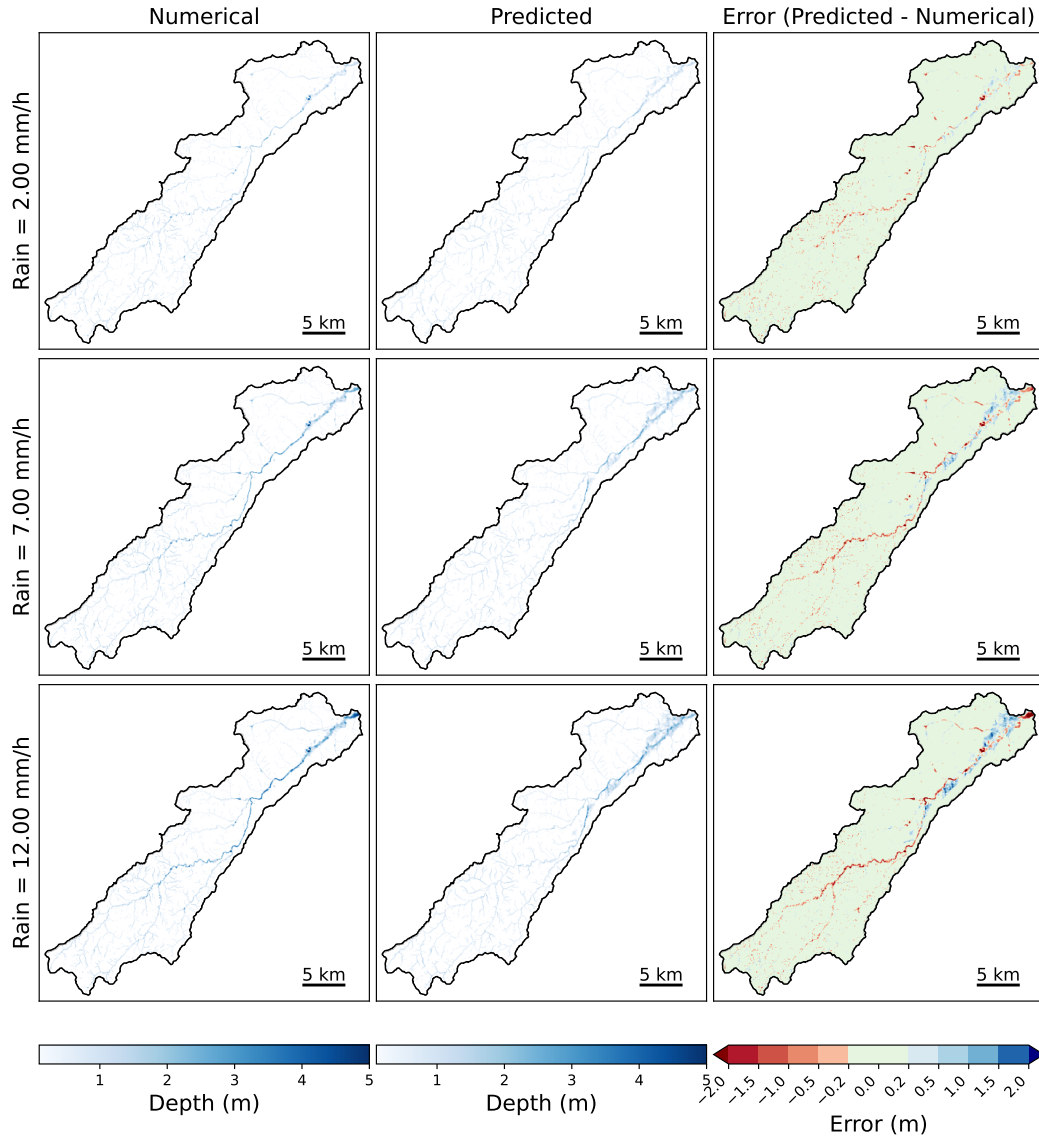


Figure 5.6: Reference (Numerical, left column) and predicted (model EN1, reconstruction method: mean, central column) depth maps, as well as error maps (right column) for selected rainfall scenarios in the test basin.

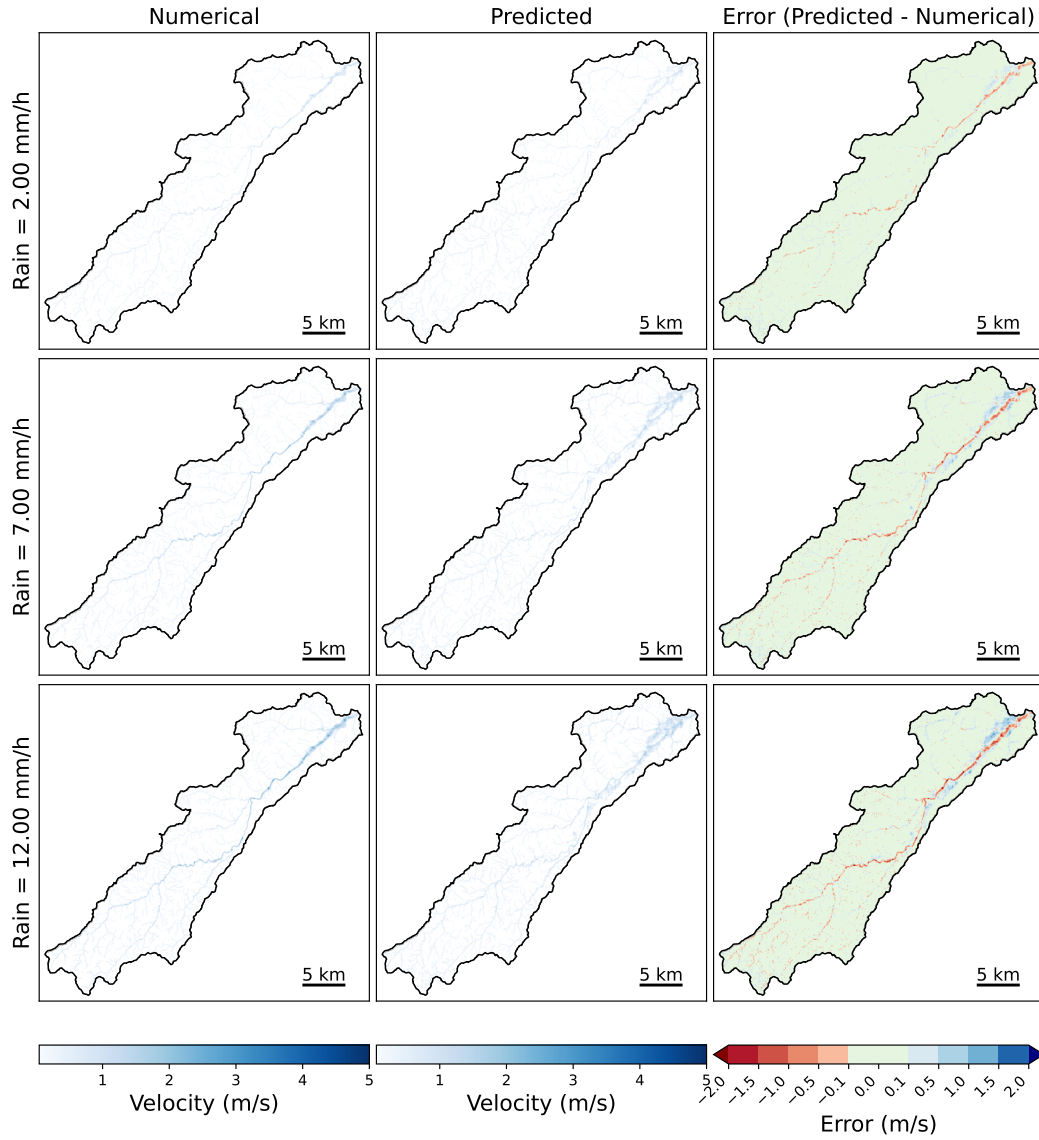


Figure 5.7: Reference (Numerical, left column) and predicted (model FR2, reconstruction method: mean, central column) velocity maps, as well as error maps (right column) for selected rainfall scenarios in the test basin.

Figure 5.8 presents a close-up of a selected region within the test basin, characterized by a complex hydrological network.

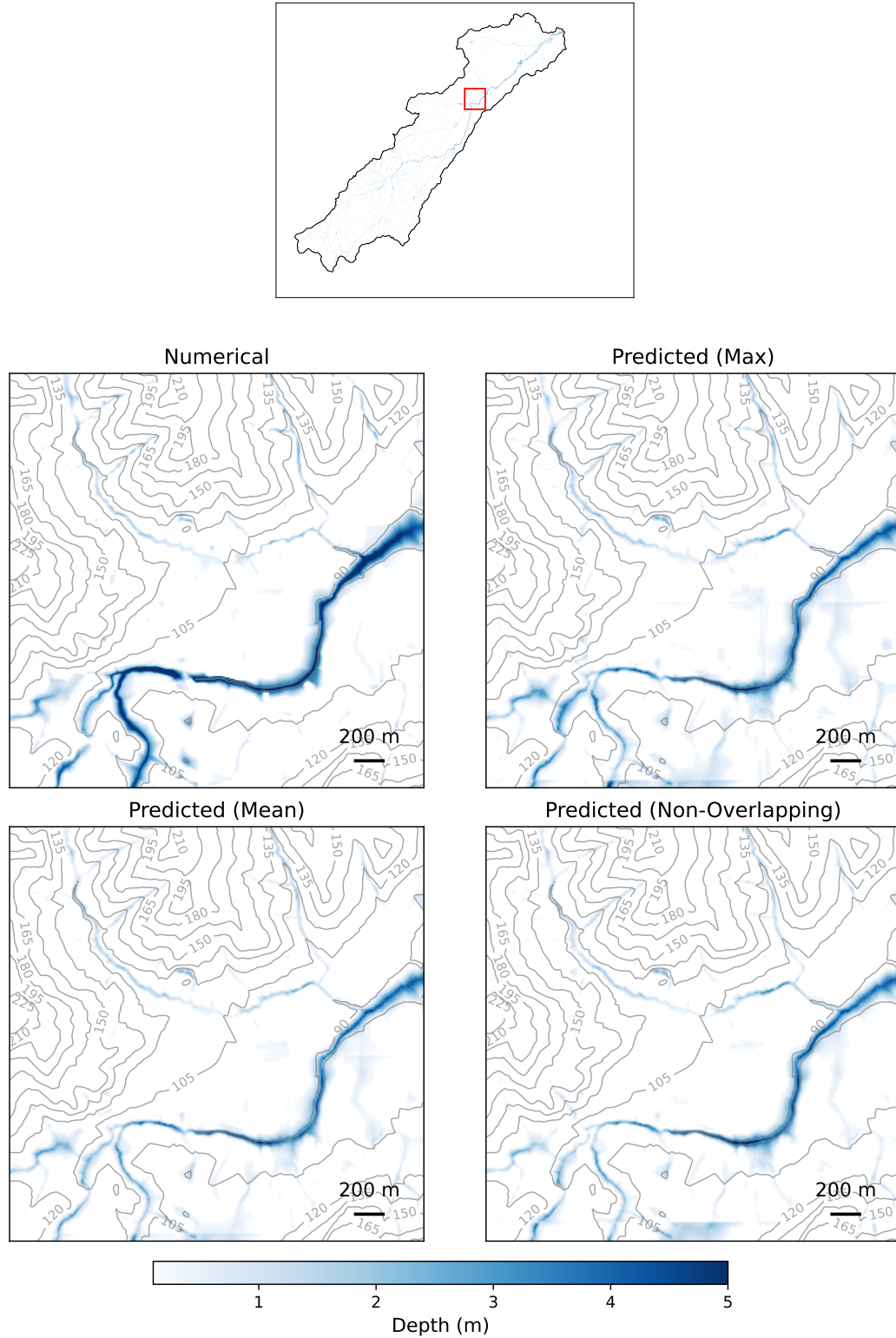


Figure 5.8: Comparison of Reference values (Numerical) and predicted depth maps (model EN1) with different reconstruction methods. The selected area (red square) is located within the test basin and its position is shown in the upper overall map.

Regression metrics

A holistic evaluation of the predictive capabilities of the models is reported in Table 5.3, through the calculation of the mean performance metrics for each model and the reconstruction method, considering both depth and velocity outputs.

Table 5.3: MAE (Equation (5.15)), MMAE (Equation (5.16)), RMSE (Equation (5.14)) and MRMSE (Equation (5.17)), for each model and reconstruction method for depth and velocity, averaged over all rainfall intensity scenarios. The minimum values in each column are highlighted in bold, while, for each model and metric, the best reconstruction method is indicated by underlined values.

Model	Reconstruction Method	Depth (m)				Velocity (m/s)			
		MAE	MMAE	RMSE	MRMSE	MAE	MMAE	RMSE	MRMSE
DD1	Max	0.085	<u>0.684</u>	<u>0.343</u>	<u>1.050</u>	0.059	<u>0.366</u>	0.178	<u>0.483</u>
	Mean	<u>0.080</u>	0.703	0.344	1.093	<u>0.053</u>	0.376	<u>0.171</u>	0.500
	Non-Overlapping	0.082	0.721	0.356	1.118	0.054	0.383	0.175	0.509
EN1	Max	0.097	0.655	0.347	<u>0.980</u>	0.063	<u>0.372</u>	0.181	<u>0.499</u>
	Mean	<u>0.082</u>	0.644	<u>0.327</u>	0.996	<u>0.054</u>	0.381	<u>0.172</u>	0.512
	Non-Overlapping	0.085	0.675	0.340	1.032	0.055	0.390	0.178	0.523
FR1	Max	0.089	<u>0.674</u>	0.345	<u>1.026</u>	0.058	<u>0.365</u>	0.178	<u>0.483</u>
	Mean	<u>0.081</u>	0.688	<u>0.341</u>	1.066	0.052	0.377	0.170	0.502
	Non-Overlapping	0.082	0.696	0.347	1.073	0.053	0.385	0.175	0.512
DD2	Max	0.098	0.663	0.358	<u>0.984</u>	0.065	0.346	0.187	0.463
	Mean	<u>0.085</u>	<u>0.655</u>	<u>0.336</u>	0.997	<u>0.056</u>	0.354	<u>0.173</u>	0.477
	Non-Overlapping	0.086	0.664	0.343	1.007	0.057	0.361	0.179	0.488
EN2	Max	0.089	<u>0.664</u>	0.344	<u>0.997</u>	0.064	<u>0.358</u>	0.187	<u>0.482</u>
	Mean	<u>0.079</u>	0.668	<u>0.331</u>	1.027	<u>0.055</u>	0.370	<u>0.174</u>	0.501
	Non-Overlapping	0.079	0.675	0.336	1.031	0.056	0.377	0.180	0.509
FR2	Max	0.089	0.649	0.340	0.967	0.064	<u>0.348</u>	0.187	<u>0.465</u>
	Mean	0.078	<u>0.648</u>	0.324	0.991	<u>0.055</u>	0.354	<u>0.172</u>	0.478
	Non-Overlapping	0.080	0.667	0.335	1.015	0.055	0.361	0.177	0.488

The results clearly show that the reconstruction method used to generate whole flood maps from single patches plays a critical role. Across most models, the Mean reconstruction method generally provides lower errors for standard metrics (MAE and RMSE). In contrast, the Max reconstruction method often yields better performance in terms of the modified metrics (MMAE and MRMSE), which are

computed exclusively over truly wet areas, focusing evaluation on regions of actual flooding. Both Mean and Max outperform the Non-overlapping approach.

Consistently with earlier analyses on the test patch dataset, the EN1 and FR2 models provide the overall best results for depth predictions. The DD2 model was specifically designed to assign greater weight to higher values, aiming to reduce errors in hydraulically relevant wet zones. While DD2 tends to improve error metrics within these wet areas, the inclusion of an additional physical term in the loss function allows EN1 and FR1 models—based on the unmodified MSE loss (Eqs. (5.5) and (5.8))—to achieve comparable or even better results in such zones. The FR2 model (Eq. (5.9)) also delivers strong performance.

Figure 5.9 compares the reconstructed depth maps produced by the DD2 and FR2 models, alongside a map showing the absolute error difference between the two models. Such visualization highlights how the FR2 model provides a more accurate representation of flow patterns in the steeper channels, which are abundant in the upstream basin area. This improvement is consistent with the training strategy of FR2, given that here Froude numbers are generally higher and therefore these zones were explicitly weighted more during training.

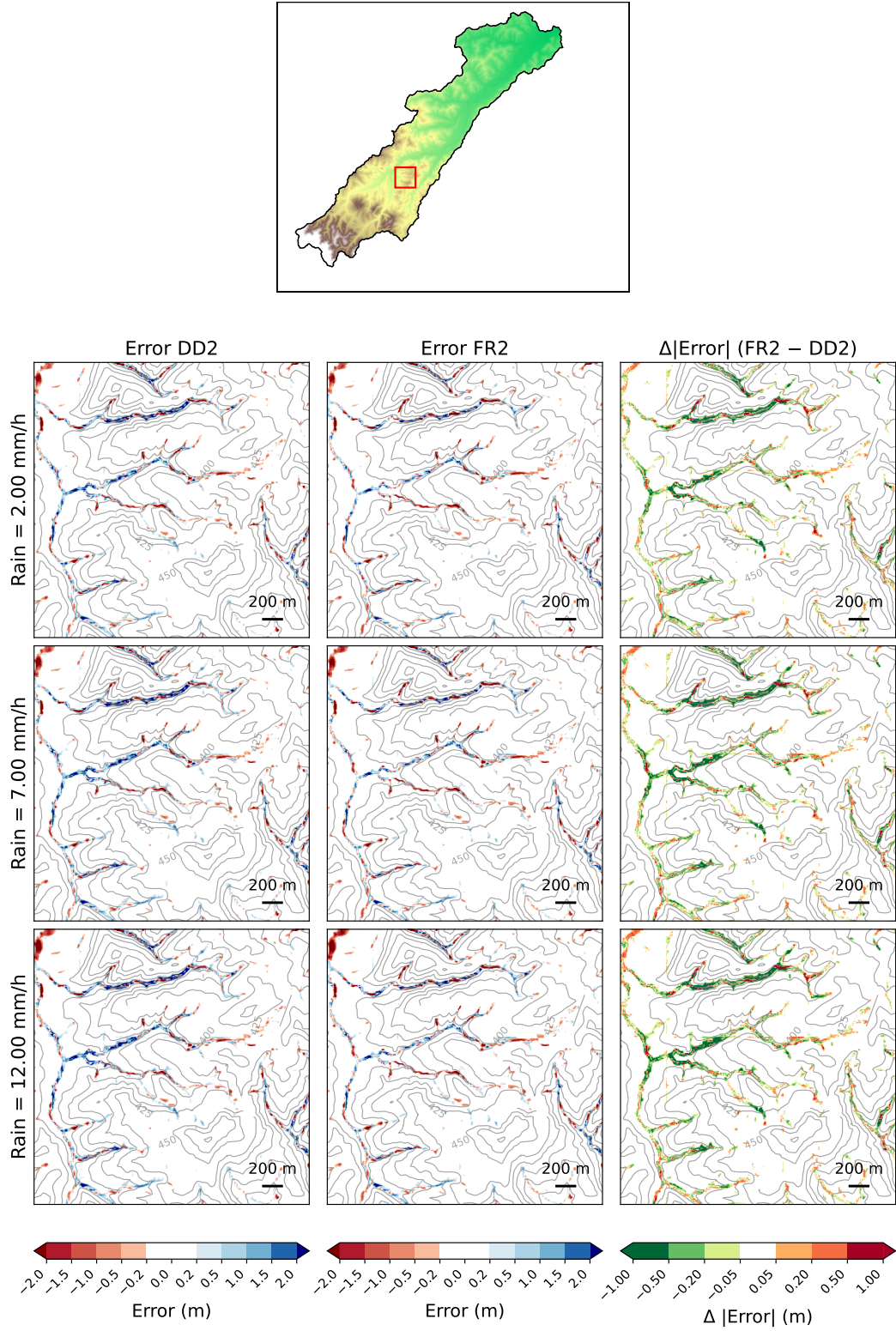


Figure 5.9: Reconstructed error depth maps (mean method) from the DD2 and FR2 models in a steep upstream area of the test basin (red square in the overview map), along with the corresponding maps of absolute error differences.

Figure 5.10 further investigates the model's depth-prediction response to variations in rainfall intensity. Regression metrics are normalized by the average depth values over flooded areas for each rainfall level, allowing for comparison across different hydrological conditions.

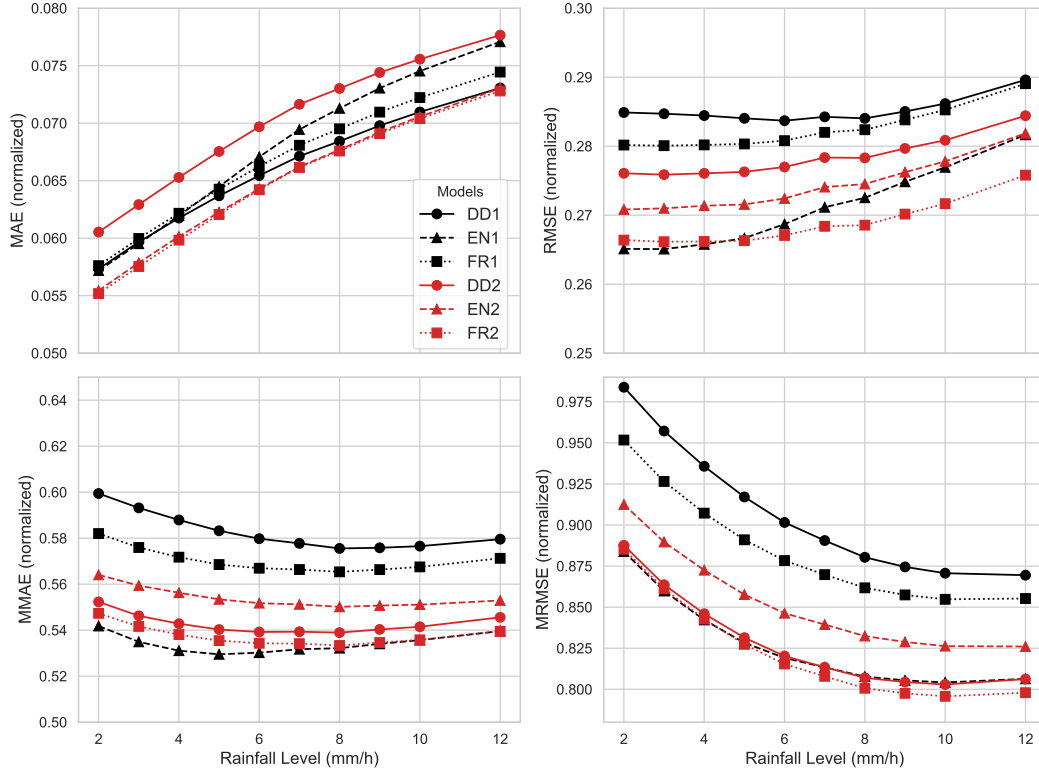


Figure 5.10: Prediction error metrics for water depth as a function of rainfall intensity across different models. Results are computed using the Mean reconstruction method.

An increasing trend in rainfall intensities is generally expected for MAE values due to the spatial expansion of flooded areas, which introduces new regions where errors are likely to accumulate. Figure 5.10 confirms such expectation for depth predictions, given that the normalized MAE computed over the entire domain increases with rainfall intensity. In contrast, the RMSE, which is more sensitive to large errors, remains stable or shows only a slight increase.

However, the metrics limited to the regions flooded in the reference (lower panels in Figure 5.10) remain stable or even decrease, indicating no degradation in model performance within wet regions as water depths increase. Similar trends are observed for velocity predictions and suggest that, while errors grow with the spatial extent of

flooding, model accuracy within inundated zones remains robust.

Classification Metrics and error distribution analysis

Table 5.4 reports classification metrics averaged across rainfall intensities for the considered models and the reconstruction methods described in Section 5.2.2.

Table 5.4: Classification metrics for depth prediction, averaged over rainfall intensities. The best values for each metric are highlighted in bold, while, for each model and metric, the best reconstruction method is indicated by underlined values.

Model	Method	Acc	Prec	Rec	F1	CSI
DD1	Max	0.930	0.539	<u>0.811</u>	0.648	0.479
	Mean	0.939	0.589	0.777	<u>0.670</u>	<u>0.504</u>
	Non-Overlapping	<u>0.940</u>	<u>0.596</u>	0.763	0.669	0.503
EN1	Max	0.915	0.480	0.886	0.623	0.452
	Mean	0.931	0.542	0.850	0.662	0.495
	Non-Overlapping	<u>0.933</u>	<u>0.552</u>	0.827	<u>0.662</u>	<u>0.495</u>
FR1	Max	0.926	0.523	<u>0.832</u>	0.642	0.473
	Mean	0.937	<u>0.576</u>	0.796	0.668	0.502
	Non-Overlapping	<u>0.939</u>	0.586	0.782	<u>0.670</u>	<u>0.504</u>
DD2	Max	0.918	0.490	<u>0.852</u>	0.622	0.452
	Mean	0.930	0.539	0.820	0.650	0.482
	Non-Overlapping	<u>0.933</u>	<u>0.554</u>	0.809	<u>0.657</u>	<u>0.489</u>
EN2	Max	0.928	0.529	<u>0.839</u>	0.649	0.480
	Mean	0.939	0.589	0.795	0.676	0.511
	Non-Overlapping	<u>0.941</u>	<u>0.601</u>	0.780	<u>0.679</u>	<u>0.513</u>
FR2	Max	0.929	0.535	<u>0.840</u>	0.654	0.486
	Mean	0.939	0.585	0.805	0.678	0.512
	Non-Overlapping	0.942	0.605	0.786	0.683	0.519

Overall, the Non-Overlapping reconstruction method generally provides the best performance across most models, achieving the highest Accuracy, Precision, F1-score, and CSI. This behavior is mainly associated with a reduction in false positives, as overlapping contributions are avoided. In contrast, the Max reconstruction method consistently yields the highest Recall values for several models, notably EN1, by

reducing false negatives.

Among the models, FR2 achieved the best overall performance, while the EN1 model achieved the highest Recall. Overall, the physically-augmented models outperformed the DD baselines, confirming their improved spatial generalization.

The classification metrics show a slight dependence on rainfall intensity, noting only a moderate reduction in Accuracy and an improvement in Recall due to the reduction of false negatives.

Spatial classification accuracy is further investigated in Figure 5.11. The model exhibits a low number of False Negatives, which are mainly concentrated in the downstream sections of the basin (Figure 5.12). The number of Positive classifications increases with rainfall intensity, reflecting the expected expansion of flooded areas. Overall, the majority of the basin is correctly classified, indicating reliable model performance across varying precipitation levels.

5.4.3 Comparison with official flood mapping

Finally, the predicted flood maps are qualitatively compared with those provided by official local authorities to demonstrate the capability of the proposed data-driven approach in delivering comprehensive hazard mapping over large areas.

The official map plan of flooded areas for the test basin, as provided by the Basin Authority [Autorità di Bacino Distrettuale dell'Appennino Centrale, 2022], includes only a 200-year return period flood extent. As is common for minor basins in Italy, official maps are obtained by modeling flow routing only in main channels, due to the economic and computational costs of detailed hydrodynamic simulations [Pavesi et al., 2022].

In the absence of specific details regarding the hydrological modeling used in the official flood hazard mapping, the 200-year return period rainfall intensity is set at 5.5 mm/h. This value corresponds to the net rainfall intensity that produces the 200-year return period discharge at the outlet section, as reported in previous hydrological studies [Consorzio di Bonifica delle Marche, 2019]. This assumption represents a strong simplification, as it implicitly assigns the same return period to rainfall and discharge, neglecting the complexity of the flood generation process.

A qualitative comparison at the basin scale (Figure 5.13) shows that the proposed DL model (here, the FR2 model with the Mean reconstruction method is adopted) is able to effectively reproduce the main inundated areas identified in the official map. In addition, the data-driven approach captures several secondary flow paths and localized inundation patterns that are not represented in the official mapping (see Figure 5.14). Most of the discrepancies are observed in the area near the outlet section, where deviations from the numerical model used as a reference have already been identified. In this region, the incised channel is not accurately represented by the employed DEM resolution; therefore, the differences with official flood mapping are exacerbated. Moreover, the influence of boundary conditions further contributes to the observed discrepancies.

Differences between the predicted and reference maps can be attributed to multiple factors, including discrepancies in DEM resolution, as well as differences in hydrological and hydraulic modeling assumptions.

Nevertheless, the ability of the proposed models—trained on a different basin—to generalize and identify both primary and secondary inundation patterns highlights their potential as a complementary tool for extensive hazard assessment.

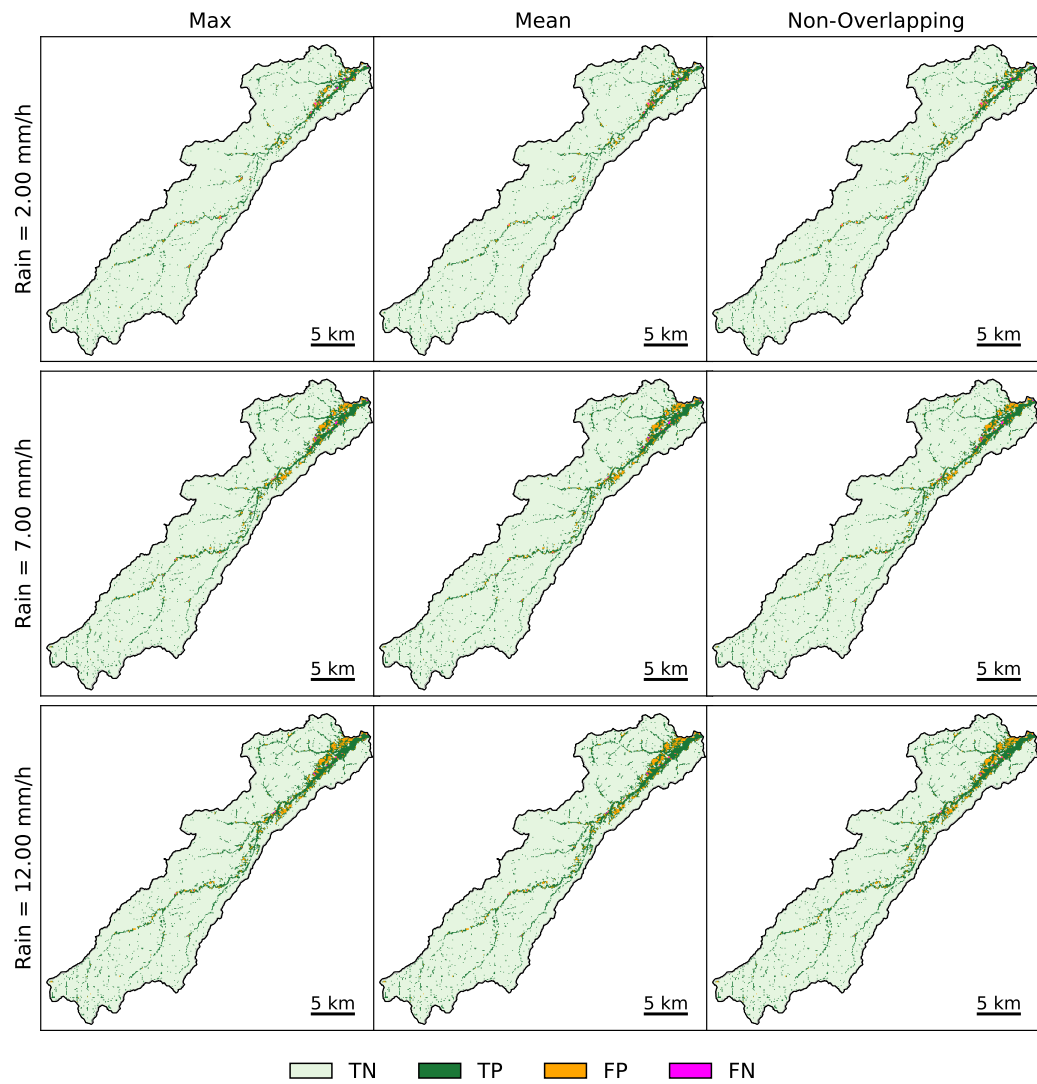


Figure 5.11: Spatial distribution of the classification results produced by the FR2 model over the test basin, for three rainfall intensities and reconstruction methods.

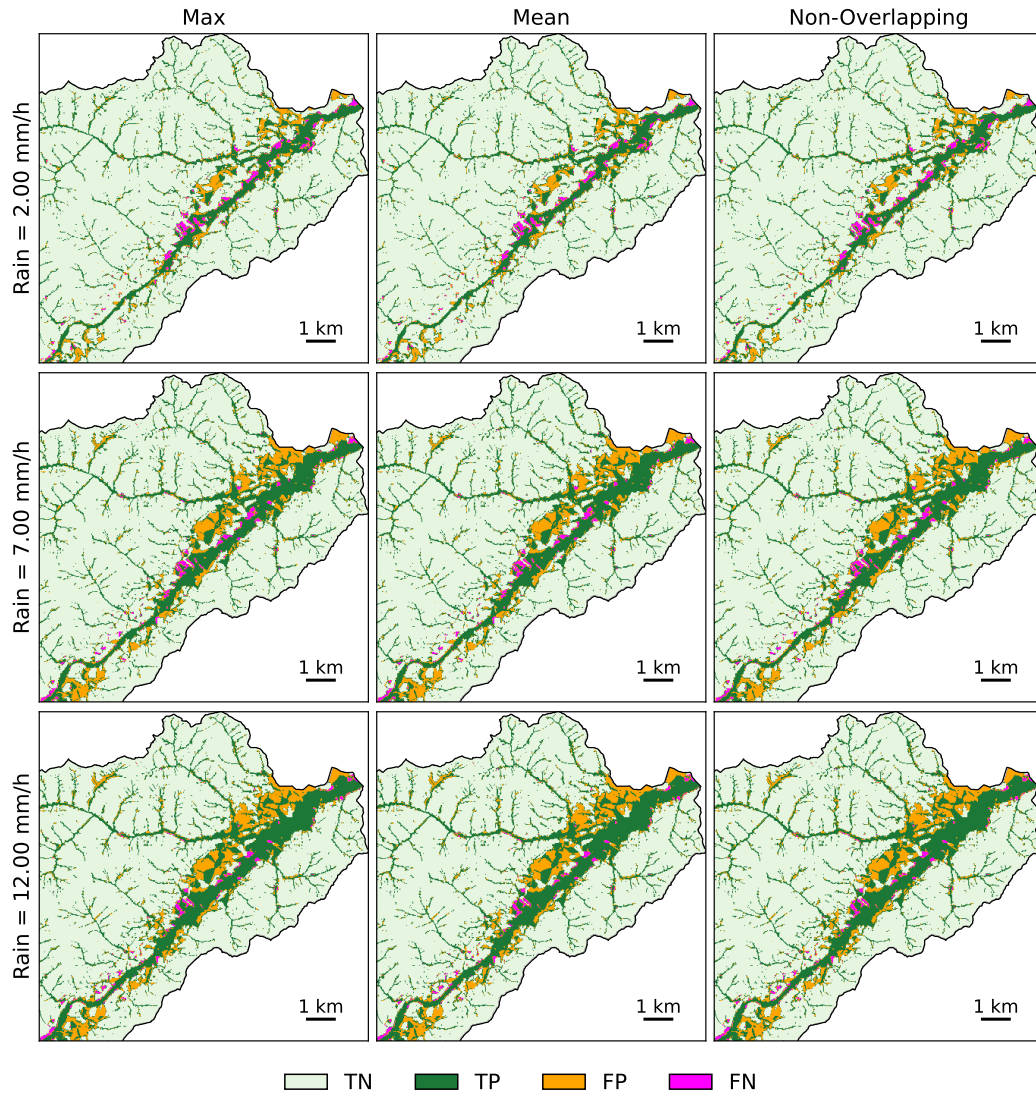


Figure 5.12: Spatial distribution of the classification results produced by the FR2 model over the downstream area of test basin, for three rainfall intensities and reconstruction methods.

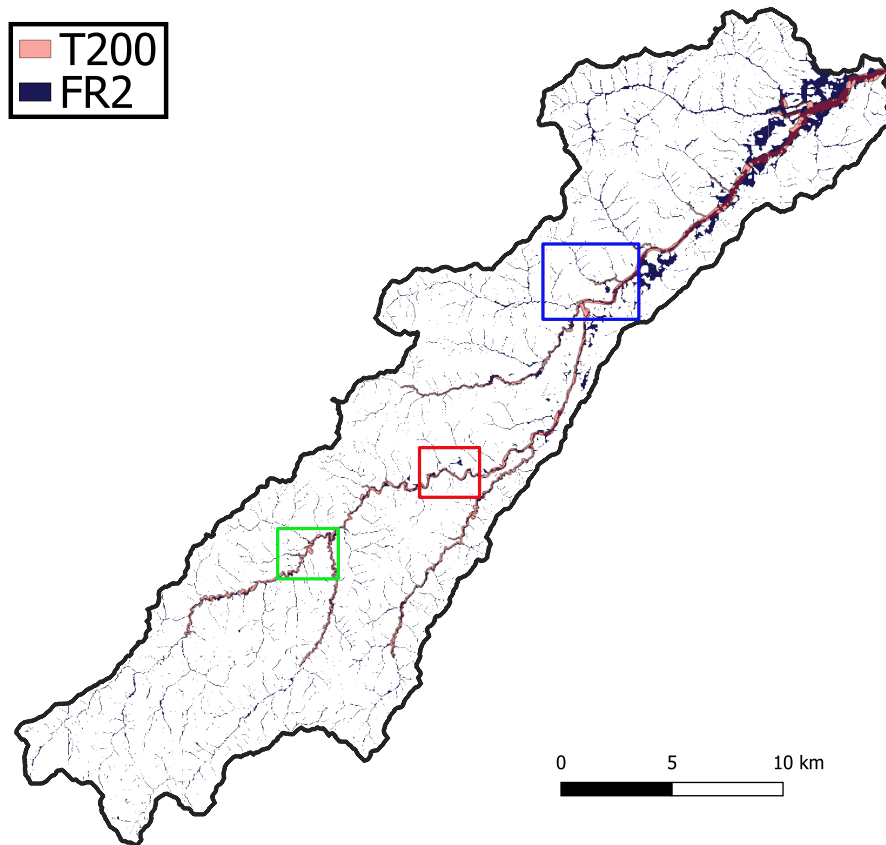
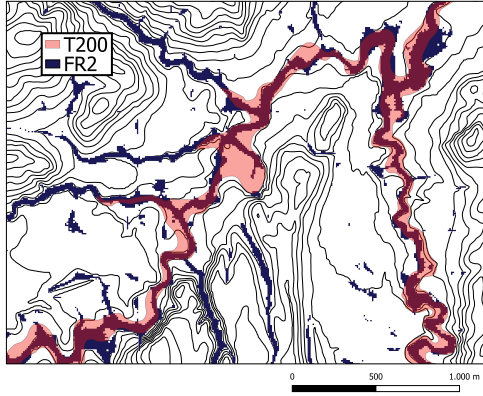
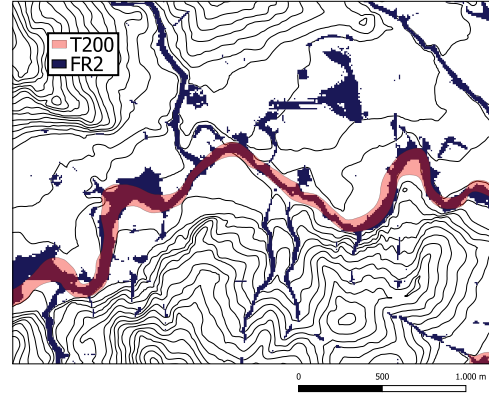


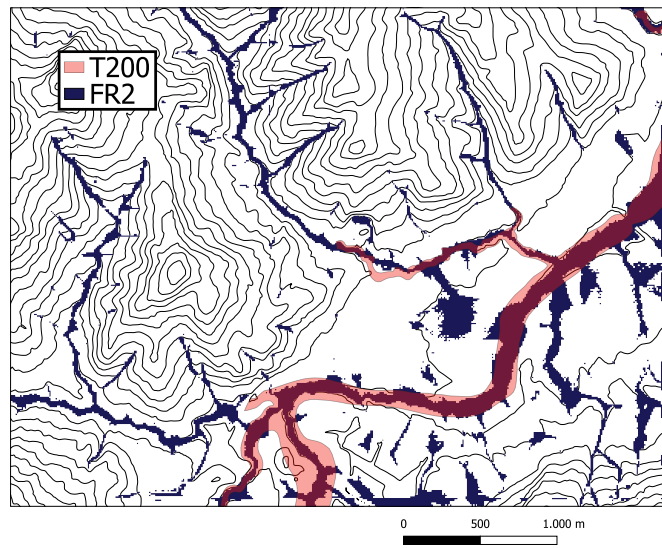
Figure 5.13: Comparison between the inundation areas predicted by the FR2 model and the official flood hazard map (T200). Overview of the entire basin with highlighted regions (green, red, and blue rectangles) selected for details.



(a) Green rectangle



(b) Red rectangle



(c) Blue rectangle

Figure 5.14: Detailed comparison in the selected areas shown in Figure 5.13.

Chapter 6

Discussion

In this chapter, the significance of the obtained results is discussed, with a focus on the proposed framework for predicting flood hazard maps across different basins.

6.1 Between Numerical and Deep Learning Approaches

In river hydraulics, physical processes are described by the NS equations, which are commonly simplified to the SW equations (5.10, 5.11, and 5.12) and solved by well-established, physically based, and widely trusted numerical methods. The strengths of these models lie in their solid physical derivation and in the interpretability of the governing equations.

However, their main drawbacks concern the high computational cost, which limits both the temporal and spatial scales that can be realistically simulated, as well as the need to introduce simplifications that may not hold under all flow conditions, together with parameter equifinality. The need to run ensembles of simulations to quantify the uncertainty associated with model parameters exacerbates the computational burden. This issue stems from the limited availability of observational data, which complicates their proper calibration.

As discussed in Chapter 1, DL models have gained attention as a potential alternative to traditional numerical approaches. These data-driven models are expected to overcome some of the critical limitations of numerical models by learning directly from data rather than explicitly solving the governing physics. In doing so,

they promise fast, large-scale, and even real-time predictions, potentially producing results that would otherwise be computationally prohibitive.

It is important to acknowledge, however, that the intended scope of data-driven models differs from that of physics-based solvers. They are better suited for producing large-scale, approximate estimates rather than detailed, site-specific analyses.

Nevertheless, in the field of flood mapping and hydraulics, the scarcity of extensive, high-quality data — and the substantial effort required to obtain it through field measurements — creates a paradox. Data-driven models are conceived as an alternative to physically based simulations; yet, in practice, they rely on the outputs of such models as reference data to obtain sufficiently large training datasets.

Moreover, the generated dataset must be sufficiently comprehensive to sample a wide range of relevant scenarios in order to achieve reliable performance during inference. If the model is required to extrapolate beyond the conditions represented in the training data, its predictions may become entirely unreliable. Therefore, the idea of DL as a possible panacea — at least in the context of flood mapping — must be significantly tempered.

Undoubtedly, DL opens the door to new possibilities that were previously unattainable. However, for a meaningful evaluation and effective application of DL, the focus should remain on its generalization capabilities. These can be pursued, or at least improved, through the incorporation of physical knowledge — the unifying element at the core of this work. A generalizable data-driven approach offers the possibility to calibrate models on data-rich catchments and subsequently apply them to ungauged ones.

6.2 Embedding physical knowledge

The limited amount of available data, compared to the larger datasets typically used in data-driven problems, nevertheless contains a much richer form of information — namely, physical knowledge — that would be lost if purely data-driven approaches were applied.

In this context, PINNs have received considerable attention and represent the state of the art for PIML. The present work, in its early stages, was inevitably influenced by

the extensive body of literature on PINNs. However, as shown in Section 2.9.1, PINNs are DL models that, in many respects, do not bypass the concept of a numerical model; rather, they offer an attractive computational alternative. The application of PINNs to real-world hydraulic problems, however, is not straightforward. For flood mapping and river hydraulics, such approaches appear to have limited practical utility. Indeed, PINNs remain subject to many of the same limitations—and, in some cases, additional challenges—inherent to numerical simulations, which restrict their feasible applications.

In this work, an alternative approach (PAT), delineated in Section 2.9.2, has been envisaged. Within the data-driven nature of the approach, a-priori physical knowledge is incorporated into the learning process through the inclusion of a regularization term in the loss function. By combining output variables in a physically meaningful way, the informational content of the training data is expected to be enhanced and enriched, guiding the model toward more physically consistent behavior.

The independence of the method from the explicit knowledge of the governing PDEs and their parameters represents a significant advantage when dealing with large epistemic uncertainty, as is commonly encountered in river hydraulics. Since the modification concerns only the training phase—by adding additional terms to the loss function—the overall computational cost is affected only marginally.

The improvement in the model’s generalization capability due to the inclusion of physically inspired loss terms was evaluated across three progressively complex systems.

In the simplest case, that of the pendulum (Chapter 3), the experiments revealed that the proposed methodology plays a decisive role in enabling the model to correctly interpret the data.

In the second case, involving one-dimensional backwater profiles (Chapter 4), the results confirmed a promising increase in generalization, expressed as improved performance on the test set. Better predictive skills were obtained in the presence of overfitting, whereas it was found that underfitting cannot be mitigated by the added physical information. It is worth highlighting that, in both stress tests—small-data regime and extrapolation—the analysis revealed that incorporating physical

information leads to a clear performance gain, which is particularly relevant for flood mapping applications.

As a complementary finding, the analysis also unveiled the potential detrimental effect of an apparently informative physical constraint, namely the volume conservation term. In this sense, the study calls for the development of a systematic framework to assess the actual informational content of physical constraints.

6.3 The problem of catchment generalization

In the third case study presented in Chapter 5, the proposed methodology was applied to a two-dimensional hydraulic problem addressing the challenge of catchment generalization. Although this case involved certain limitations and simplifications — which will be discussed — it holds great significance for the practical applicability of the proposed approach.

The aim of this experiment was to generate flood maps for a previously unseen catchment by leveraging the predictive capability of a DL model trained on a single, distinct catchment. Such a challenging scenario effectively mirrors real-world situations in flood hazard assessment for ungauged basins, where one must rely on geomorphological and hydrological similarities with data-rich regions.

Peculiarities of data-driven flood prediction

For the nature of the problem, the datasets for flood mapping are unbalanced, i.e., larger areas are either un-flooded or have low depths. Results show that the model tends to underestimate regions with higher flood depths. This limitation is partially alleviated when reconstructing the flood map using a maximum merging strategy across overlapping patches, although this approach leads to an overestimation of inundated areas, which is a predictable consequence. Conversely, the non-overlapping reconstruction method introduces more pronounced discontinuities between adjacent patches, which is also expected. Using the mean value across overlapping regions helps to balance these effects.

It is crucial to acknowledge that the utmost accuracy attainable is fundamentally restricted by the quality of the training datasets, whether they are observational or

synthetic in nature.

It is also worth noting that recent developments in simplified conceptual models for flood mapping face the crucial challenge of assigning a frequency of occurrence to their outcomes in order to employ the results in planning scenarios. The patch-based approach adopted here mitigates this issue, as the entire basin under scrutiny can be partitioned into sub-basins: for a given return period, the flood map for each partition can be derived by imposing the corresponding rainfall intensity.

Limitations of the flood mapping experiment

To maintain control over the experimental conditions, the simulations were simplified by assuming spatially and temporally uniform rainfall inputs and neglecting infiltration, effectively treating rainfall as net precipitation. Additionally, the modeling framework assumed steady-state conditions, thereby disregarding the temporal dynamics typically associated with flood wave propagation.

These simplifications were introduced to isolate the influence of topographic features and to ensure the attainment of a consistent and physically meaningful flood state. However, they represent substantial idealizations that should be carefully considered when extending the approach to real-world hazard assessments, where unsteady hydraulic processes can play a decisive role in determining flood hazard.

Furthermore, the outflow boundary conditions at the downstream section were not included in the DL model, which may have influenced the resulting flood patterns in downstream areas.

While rainfall-on-grid simulations are generally subject to higher uncertainties compared to traditional channel-based modeling approaches, the objective here is not predictive accuracy but the generation of internally consistent hydrodynamic responses across scenarios. Therefore, these modeling choices affect the absolute accuracy of target fields but do not compromise the validity of the comparative and learning-oriented framework proposed in this work.

Finally, it is important to clarify that the physical quantities incorporated in this work represent only a subset of possible choices. Other physically meaningful terms could be considered. The selected quantities rely solely on local information and

do not require upstream data, making them fully compatible with the patch-based modeling framework — a characteristic that is essential for the approach adopted here.

Discussion of the results and computational times

Results indicate that the proposed PAT approach improves generalization even in this complex experiment. The incorporation of physical constraints led to the largest performance gains in flood depth prediction. This is a particularly valuable outcome given that the accurate estimation of depth is generally more critical for risk assessment and decision-making than precise velocity predictions, which, on the other hand, are subject to considerable uncertainty even in physics-based numerical models. The modified loss functions seem to emphasize different flood zones: energy loss emphasizes high-energy areas, while Froude loss focuses on regions with higher Froude numbers.

Even within this simplified experimental framework, the comparison between predicted flood extents associated with a specific return period in the test basin and official flood hazard maps provided by public authorities was made. This allows for a preliminary validation of the procedure and a qualitative assessment of its potential for practical applications and highlights the capability of the proposed framework to support large-scale flood hazard assessment in data-scarce regions.

Notwithstanding the above limitations, the framework is suitable for direct hazard assessment: indeed, each rain intensity could be associated with a corresponding return period, adding value to the risk assessment by providing a direct link between predicted flood impacts and the probability of occurrence.

The geomorphological and hydrological differences between the two basins, such as drainage structure, slope, and topographic complexity, underscore the difficulty of spatial generalization. Nonetheless, the models demonstrated promising performance in transferring knowledge from the training basin to the test basin, indicating an encouraging robustness to terrain variability.

Regarding runtime performance, the DL approach demonstrates remarkably fast execution times. The prediction of the entire catchment — covering an area on

the order of a few square kilometers with a spatial resolution of 10 m — can be completed in just a few seconds once the model is trained, without the need for high-performance computing. Furthermore, due to the relatively limited amount of training data, the model training itself requires approximately 20 minutes on the employed GPU. Additional time must be considered for hyperparameter tuning, which can be particularly demanding.

For comparison, the generation of the training dataset for the two basins using numerical simulations (without parallelization) required execution times on the order of one day.

Despite the above considerations, it is worth highlighting that, when comparing the computational burden of a data-driven approach to the effort and time required to carry out a traditional study for flood hazard assessment, only the inference time needs to be accounted for: indeed, the rationale behind that comparison lies on the availability of a pre-trained, generalizable model.

Chapter 7

Conclusions

Current hydraulic and hydrology studies employing data-driven techniques are often limited by small training datasets and fail to generalize beyond the specific conditions represented in the data. While these approaches have demonstrated proof-of-concept capabilities, their lack of generalization implies that, for every new domain or set of boundary conditions, a dedicated dataset must be available and the model retrained. This issue is particularly critical in flood hazard mapping, where generating high-resolution hydraulic simulations is numerically demanding and often computationally expensive. Such a requirement substantially reduces the practical advantages of data-driven models over traditional numerical approaches, leaving fast inference as their only clear benefit. Therefore, achieving adequate generalization to unseen locations and/or boundary conditions appears to be crucial.

At a deeper level, purely data-driven models fail to fully exploit the rich physical information inherently present in environmental hydraulic datasets. Unlike typical ML datasets, such as images or text, these data encode underlying physical laws and dependencies that, if properly leveraged, could significantly enhance model performance.

Drawing inspiration from the concept of Physics-Informed ML, this study investigated the potential benefits of modifying the training process by including a priori physical information within the loss function. The PAT methodology was first tested in the controlled physical framework of a simple pendulum, which was chosen as a benchmark problem. The results were encouraging, showing that the physically

constrained model captured the underlying dynamics more effectively and improved predictions under varying boundary conditions.

The modified training approach was then evaluated on the prediction of the backwater profile in a prismatic channel, which—although representing a simple and controlled case—includes the complex effect of the possible presence of a hydraulic jump. This experiment, characterized by a controlled setting yet nonlinear behavior, enabled the assessment of the method’s effectiveness under data-scarce and overfitting-prone scenarios. The promising results obtained in extrapolation tasks support the feasibility of the method for hydraulics and hydrology problems, where such challenges are particularly common.

Building on these encouraging findings, the research extended the investigation to a more complex two-dimensional river flood problem at the catchment scale, focusing on improving the catchment generalization of data-driven flood mapping models. Catchment generalization represents a pivotal challenge for the practical application of data-driven flood models: the need for retraining in each new basin geometry severely undermines the computational advantages that such models offer over traditional numerical simulations. Moreover, assessing river flood hazards in ungauged catchments remains one of the most pressing challenges in hydrology and risk management.

To address this issue, the study developed a DL framework designed to achieve cross-basin transferability. Specifically, a convolutional encoder–decoder architecture was trained on a single basin under multiple synthetic rainfall events and tested on a distinct, unseen basin without retraining. This setup mimicked realistic data-scarce conditions commonly encountered in hydrological applications and represented a considerably challenging task for a purely data-driven model. The experiments were conducted on two river basins in central Italy—the Misa and Cesano Rivers—which are adjacent and share similar hydrological characteristics, yet exhibit markedly different geomorphological and channel network configurations. The proposed framework aims to bridge the gap between data-driven modeling, physical consistency, epistemic uncertainty, and generalization.

To augment the informational content of the data for the training process,

two physically meaningful quantities—specific energy and Froude number—were introduced as additional terms in the loss function. Their inclusion aimed to guide the network toward physically consistent predictions, encouraging the preservation of hydraulic relationships.

Results showed that encoding physical information into the loss function enhances the model’s generalization capability in both regression (continuous prediction of depth and velocity) and classification (flooded vs. non-flooded areas) tasks in the test basin.

Comparisons with official flood hazard maps for a selected return period in the test basin confirmed that the method holds promise for practical applications, particularly in data-scarce regions or ungauged basins where traditional hydrodynamic modeling is not feasible. Although the framework presented here represents a simplified formulation of the broader flood hazard estimation problem and leaves room for several important developments, the approach has demonstrated its potential to support extensive flood hazard mapping in ungauged basins.

7.1 Future developments

This study proposed and validated a methodology to introduce physical information into data-driven models for environmental applications, demonstrating that generalization capabilities can be enhanced under conditions of data scarcity, overfitting, and extrapolation. Beyond the specific case study, the proposed methodology can be applied flexibly to other DL architectures and extended to additional environmental-simulation contexts. The approach contributes to bridging the gap between numerical modeling and data-driven applications, paving the way for environmental simulation frameworks that integrate the interpretability of physics with the efficiency of ML.

This direction is especially relevant to flood-mapping applications, where reducing the black-box nature of NNs remains a central challenge. Enhancing uncertainty quantification and model transparency will be essential to foster trust and enable integration into decision-making workflows for operational use. In this regard, developing tools to assess prediction confidence and to constrain the network to physical processes would represent an important step forward.

More specifically, for the flood-hazard mapping framework presented in this work, several promising research avenues emerge to overcome current limitations and move toward more accurate hazard estimation. Further advances should consider unsteady flow conditions, such as time-dependent rainfall–runoff events. Moreover, the physical quantities used in this study—specific energy and Froude number—represent only a subset of the possible forms of a priori physical information that could be embedded to improve physical consistency. A systematic exploration of architectural choices, training strategies, and the use of larger and more diverse datasets could further enhance model stability, efficiency, and adaptability across catchments. The capability of the proposed models to extrapolate beyond the range of rainfall intensities included in the training dataset deserves further investigation.

Finally, a fruitful development would be to quantify the geometric “distance” between catchments and link it with hydrological similarity metrics within a comprehensive framework, allowing for the assessment of how generalization performance and prediction uncertainty change with increasing dissimilarity from the training domain.

Bibliography

- M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems. 2015. URL <https://www.tensorflow.org/>. Software available from tensorflow.org.
- R. J. Abraham, L. M. See, and D. P. Solomatine. *Practical hydroinformatics: computational intelligence and technological developments in water applications*, volume 68. Springer Science & Business Media, 2008.
- P. Ahmadi, M. J. B. Valadan Zoej, M. Mokhtarzade, N. Kardan Halvaie, and E. Ghaderpour. A capsule network framework for flood mapping integrating remote sensing fusion techniques. *Environmental Research Communications*, 2025.
- P. J. M. Ali, R. H. Faraj, E. Koya, P. J. M. Ali, and R. H. Faraj. Data normalization and standardization: a technical report. *Mach Learn Tech Rep*, 1(1):1–6, 2014.
- D. Amitrano, G. Di Martino, A. Di Simone, and P. Imperatore. Flood detection with sar: A review of techniques and datasets. *Remote Sensing*, 16(4):656, 2024.
- A. Apicella, F. Donnarumma, F. Isgrò, and R. Prevete. A survey on modern trainable activation functions. *Neural Networks*, 138:14–32, 2021.
- Autorità di Bacino Distrettuale dell’Appennino Centrale. Piano Stralcio di Bacino per

-
- l'Assetto Idrogeologico (PAI) – Regione Marche: aggiornamento 2016. Technical report, Autorità di Bacino Distrettuale dell'Appennino Centrale, 2022.
- A. Y. Barrera-Animas, L. O. Oyedele, M. Bilal, T. D. Akinosho, J. M. D. Delgado, and L. A. Akanbi. Rainfall prediction: A comparative analysis of modern machine learning algorithms for time-series forecasting. *Machine Learning with Applications*, 7:100204, 2022.
- M. S. Bartlett, J. Van Blitterswyk, M. Farella, J. Li, C. Smith, A. J. Parolari, L. Krishnamoorthy, and A. Mrad. Physically based dimensionless features for pluvial flood mapping with machine learning. *Water Resources Research*, 61(4): e2024WR039086, 2025.
- Ö. Baydaroglu and I. Demir. Temporal and spatial satellite data augmentation for deep learning-based rainfall nowcasting. *Journal of Hydroinformatics*, 26(3): 589–607, 2024.
- A. G. Baydin, B. A. Pearlmutter, A. A. Radul, and J. M. Siskind. Automatic differentiation in machine learning: a survey. *Journal of Machine Learning Research*, 18:1–43, 2018.
- R. Bellman. *Adaptive Control Processes: A Guided Tour*. Princeton University Press, Princeton, NJ, 1961.
- R. Bentivoglio, E. Isufi, S. N. Jonkman, and R. Taormina. Deep learning methods for flood mapping: a review of existing applications and future research directions. *Hydrology and Earth System Sciences*, 26(16):4345–4378, 2022.
- R. Bentivoglio, E. Isufi, S. N. Jonkman, and R. Taormina. Rapid spatio-temporal flood modelling via hydraulics-based graph neural networks. *Hydrology and Earth System Sciences*, 27(23):4227–4246, 2023.
- S. Berkhahn, L. Fuchs, and I. Neuweiler. An ensemble neural network model for real-time prediction of urban floods. *Journal of Hydrology*, 575:743–754, 2019.
- K. Beven. A manifesto for the equifinality thesis. *Journal of hydrology*, 320(1-2): 18–36, 2006.

-
- T. Bindas, W.-P. Tsai, J. Liu, F. Rahmani, D. Feng, Y. Bian, K. Lawson, and C. Shen. Improving large-basin river routing using a differentiable muskingum-cunge model and physics-informed machine learning. *Authorea Preprints*, 2023.
- G. Blöschl. *Runoff prediction in ungauged basins: synthesis across processes, places and scales*. Cambridge University Press, 2013.
- A. Bomers and S. J. Hulscher. Neural networks for fast fluvial flood predictions: Too good to be true? *River Research and Applications*, 39(8):1652–1658, 2023.
- L. Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.
- L. Breiman, J. Friedman, R. A. Olshen, and C. J. Stone. *Classification and regression trees*. Chapman and Hall/CRC, 2017.
- M. M. Bronstein, J. Bruna, Y. LeCun, A. Szlam, and P. Vandergheynst. Geometric deep learning: going beyond euclidean data. *IEEE Signal Processing Magazine*, 34(4):18–42, 2017.
- S. L. Brunton, B. R. Noack, and P. Koumoutsakos. Machine learning for fluid mechanics. *Annual review of fluid mechanics*, 52(1):477–508, 2020.
- T. Cache, M. S. Gomez, T. Beucler, J. Blagojevic, J. P. Leita, and N. Peleg. Enhancing generalizability of data-driven urban flood models by incorporating contextual information. *Hydrology and Earth System Sciences Discussions*, 2024: 1–23, 2024.
- S. Cai, Z. Mao, Z. Wang, M. Yin, and G. E. Karniadakis. Physics-informed neural networks (pinns) for fluid mechanics: A review. *Acta Mechanica Sinica*, 37(12): 1727–1738, 2021.
- S. Cedillo, A.-G. Núñez, E. Sánchez-Cordero, L. Timbe, E. Samaniego, and A. Alvarado. Physics-informed neural network water surface predictability for 1d steady-state open channel cases with different flow types and complex bed profile shapes. *Advanced Modeling and Simulation in Engineering Sciences*, 9(1):1–23, 2022.

-
- Y. Cengel and J. Cimbala. *Ebook: Fluid mechanics fundamentals and applications (si units)*. McGraw Hill, 2013.
- D. Chaulagain, P. R. Rimal, S. N. Ngando, B. E. K. Nsafon, D. Suh, and J.-S. Huh. Flood susceptibility mapping of kathmandu metropolitan city using gis-based multi-criteria decision analysis. *Ecological Indicators*, 154:110653, 2023.
- A. M. Chen, H.-m. Lu, and R. Hecht-Nielsen. On the geometry of feedforward neural network error surfaces. *Neural computation*, 5(6):910–927, 1993.
- F. Chollet et al. Keras. 2015.
- V. Chow. *Applied hydrology*. McGraw-hill, 1971.
- J. Chung, C. Gulcehre, K. Cho, and Y. Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.
- B. CM. Robust human detection system in flood related images with data augmentation. *Multimedia Tools and Applications*, 82(7):10661–10679, 2023.
- T. Cohen, M. Weiler, B. Kicanaoglu, and M. Welling. Gauge equivariant convolutional networks and the icosahedral cnn. In *International conference on Machine learning*, pages 1321–1330. PMLR, 2019.
- S. Cole, R. Moore, V. Bell, and D. Jones. Issues in flood forecasting: ungauged basins, extreme floods and uncertainty. In *Frontiers in Flood Forecasting, 8th Kovacs Colloquium, UNESCO, Paris*, volume 305, pages 103–122, 2006.
- Consorzio di Bonifica delle Marche. Studio idrogeomorfologico e della qualità ambientale dei bacini idrografici della regione marche. Technical report, 2019.
- S. Conti. Artificial intelligence for weather forecasting. *Nature Reviews Electrical Engineering*, 1(1):8–8, 2024.
- C. Cortes and V. Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

-
- M. Corti, M. Francioni, A. Abbate, M. Papini, L. Longoni, et al. Analysis and modelling of the september 2022 flooding event in the misa basin. *Italian journal of engineering geology and environment*, 1(Special Issue 1):69–76, 2024.
- S. Cuomo, V. S. Di Cola, F. Giampaolo, G. Rozza, M. Raissi, and F. Piccialli. Scientific machine learning through physics-informed neural networks: Where we are and what’s next. *Journal of Scientific Computing*, 92(3):88, 2022.
- S. Das. Flood susceptibility mapping of the western ghat coastal belt using multi-source geospatial data and analytical hierarchy process (ahp). *Remote Sensing Applications: Society and Environment*, 20:100379, 2020.
- R. Dasgupta, S. Das, G. Banerjee, and A. Mazumdar. Revisit hydrological modeling in ungauged catchments comparing regionalization, satellite observations, and machine learning approaches. *HydroResearch*, 2023.
- C. Dawson and R. Wilby. Hydrological modelling using artificial neural networks. *Progress in physical Geography*, 25(1):80–108, 2001.
- S. Dazzi, R. Vacondio, and P. Mignosa. Flood stage forecasting using machine-learning methods: A case study on the parma river (italy). *Water*, 13(12):1612, 2021.
- C. A. do Lago, M. H. Giacomoni, R. Bentivoglio, R. Taormina, M. N. G. Junior, and E. M. Mendiondo. Generalizing rapid flood predictions to unseen urban catchments with conditional generative adversarial networks. *Journal of Hydrology*, 618:129276, 2023.
- A. Domeneghetti, S. Vorogushyn, A. Castellarin, B. Merz, and A. Brath. Probabilistic flood hazard mapping: effects of uncertain boundary conditions. *Hydrology and Earth System Sciences*, 17(8):3127–3140, 2013.
- S. Dong and N. Ni. A method for representing periodic functions and enforcing exactly periodic boundary conditions with deep neural networks. *Journal of Computational Physics*, 435:110242, 2021.

-
- J. Donnelly, S. Abolfathi, J. Pearson, O. Chatrabgoun, and A. Daneshkhah. Gaussian process emulation of spatio-temporal outputs of a 2d inland flood model. *Water Research*, 225:119100, 2022.
- J. Donnelly, A. Daneshkhah, and S. Abolfathi. Physics-informed neural networks as surrogate models of hydrodynamic simulators. *Science of the Total Environment*, 912:168814, 2024.
- G. Du, X. Cao, J. Liang, X. Chen, and Y. Zhan. Medical image segmentation based on u-net: A review. *Journal of Imaging Science & Technology*, 64(2), 2020.
- A. K. Dubey and V. Jain. Comparative study of convolution neural network’s relu and leaky-relu activation functions. In *Applications of Computing, Automation and Wireless Systems in Electrical Engineering: Proceedings of MARC 2018*, pages 873–880. Springer, 2019.
- K. Duraisamy, G. Iaccarino, and H. Xiao. Turbulence modeling in the age of data. *Annual review of fluid mechanics*, 51(1):357–377, 2019.
- J. Eichelsdörfer, S. Kaltenbach, and P.-S. Koutsourelakis. Physics-enhanced neural networks in the small data regime. *arXiv preprint arXiv:2111.10329*, 2021.
- EU. Floods Directive 2007/60/EC: Directive of the European Parliament and of the Council of 23 October 2007 on the assessment and management of flood risks. Official Journal of the European Union, L 288, 27–34, 2007.
- D. Feng, Z. Tan, and Q. He. Physics-informed neural networks of the saint-venant equations for downscaling a large-scale river model. *Water Resources Research*, 59(2):e2022WR033168, 2023.
- N. Fraehr, Q. J. Wang, W. Wu, and R. Nathan. Assessment of surrogate models for flood inundation: The physics-guided lsg model vs. state-of-the-art machine learning models. *Water Research*, 252:121202, 2024.
- R. Gao and R. K. Jaiman. Predicting fluid–structure interaction with graph neural networks. *Physics of Fluids*, 36(1), 2024.
-

-
- S. Gao, Y. Huang, S. Zhang, J. Han, G. Wang, M. Zhang, and Q. Lin. Short-term runoff prediction with gru and lstm networks without requiring time step optimization during sample generation. *Journal of Hydrology*, 589:125188, 2020.
- A. Garzón, Z. Kapelan, J. Langeveld, and R. Taormina. Machine learning-based surrogate modeling for urban water networks: review and future research directions. *Water Resources Research*, 58(5):e2021WR031808, 2022.
- J. Gawlikowski, C. R. N. Tassi, M. Ali, J. Lee, M. Humt, J. Feng, A. Kruspe, R. Triebel, P. Jung, R. Roscher, et al. A survey of uncertainty in deep neural networks. *Artificial Intelligence Review*, 56(Suppl 1):1513–1589, 2023.
- A. Gebrehiwot, L. Hashemi-Beni, G. Thompson, P. Kordjamshidi, and T. E. Langan. Deep convolutional neural network for flood extent mapping using unmanned aerial vehicles data. *Sensors*, 19(7):1486, 2019.
- K. Ghahraman, B. Nagy, and F. Nooshin Nokhandan. Flood-prone zones of meandering rivers: Machine learning approach and considering the role of morphology (kashkan river, western iran). *Geosciences*, 13(9):267, 2023.
- X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256. JMLR Workshop and Conference Proceedings, 2010.
- I. Goodfellow, Y. Bengio, A. Courville, and Y. Bengio. *Deep learning*, volume 1. MIT press Cambridge, 2016.
- S. Goswami, C. Anitescu, S. Chakraborty, and T. Rabczuk. Transfer learning enhanced physics informed neural network for phase-field modeling of fracture. *Theoretical and Applied Fracture Mechanics*, 106:102447, 2020.
- S. Greydanus, M. Dzamba, and J. Yosinski. Hamiltonian neural networks. *Advances in neural information processing systems*, 32, 2019.
- G. Guglielmo, A. Montessori, J.-M. Tucny, M. La Rocca, and P. Prestininzi. A
-

-
- priori physical information to aid generalization capabilities of neural networks for hydraulic modeling. *Frontiers in Complex Systems*, 2:1508091, 2025.
- Z. Guo, J. P. Leitaó, N. E. Simões, and V. Moosavi. Data-driven flood emulation: Speeding up urban flood predictions by deep convolutional neural networks. *Journal of Flood Risk Management*, 14(1):e12684, 2021.
- Z. Guo, V. Moosavi, and J. P. Leitaó. Data-driven rapid flood prediction mapping with catchment generalizability. *Journal of Hydrology*, 609:127726, 2022.
- D. Gupta, B. B. Hazarika, M. Berlin, U. M. Sharma, and K. Mishra. Artificial intelligence for suspended sediment load prediction: a review. *Environmental earth sciences*, 80(9):346, 2021.
- L. Han, H. Liang, H. Chen, W. Zhang, and Y. Ge. Convective precipitation nowcasting using u-net model. *IEEE Transactions on Geoscience and Remote Sensing*, 60:1–8, 2021.
- Z. Hao, S. Liu, Y. Zhang, C. Ying, Y. Feng, H. Su, and J. Zhu. Physics-informed machine learning: A survey on problems, methods and applications. *arXiv preprint arXiv:2211.08064*, 2022.
- M. A. Hariri-Ardebili, G. Mahdavi, L. K. Nuss, and U. Lall. The role of artificial intelligence and digital technologies in dam engineering: Narrative review and outlook. *Engineering Applications of Artificial Intelligence*, 126:106813, 2023.
- K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.
- S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.
- K. Hornik, M. Stinchcombe, and H. White. Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5):359–366, 1989.
- M. Hrachowitz, H. Savenije, G. Blöschl, J. J. McDonnell, M. Sivapalan, J. Pomeroy, B. Arheimer, T. Blume, M. Clark, U. Ehret, et al. A decade of predictions in
-

-
- ungauged basins (pub)—a review. *Hydrological Sciences Journal*, 58(6):1198–1255, 2013.
- K.-I. Hsu, H. V. Gupta, and S. Sorooshian. Artificial neural network modeling of the rainfall-runoff process. *Water resources research*, 31(10):2517–2530, 1995.
- A. D. Jagtap, E. Kharazmi, and G. E. Karniadakis. Conservative physics-informed neural networks on discrete domains for conservation laws: Applications to forward and inverse problems. *Computer Methods in Applied Mechanics and Engineering*, 365:113028, 2020.
- B. Jamali, E. Haghighat, A. Ignjatovic, J. P. Leitão, and A. Deletic. Machine learning for accelerating 2d flood models: Potential and challenges. *Hydrological Processes*, 35(4):e14064, 2021.
- G. James, D. Witten, T. Hastie, and R. Tibshirani. *An introduction to statistical learning: with applications in R*, volume 103. Springer, 2013.
- X. Jin, S. Cai, H. Li, and G. E. Karniadakis. Nsfnets (navier-stokes flow nets): Physics-informed neural networks for the incompressible navier-stokes equations. *Journal of Computational Physics*, 426:109951, 2021.
- S. Kabir, S. Patidar, X. Xia, Q. Liang, J. Neal, and G. Pender. A deep convolutional neural network model for rapid prediction of fluvial flood inundation. *Journal of Hydrology*, 590:125481, 2020.
- H. Kamali and M.-R. Erfanian. Characterization of ventilated supercavitation regimes using bayesian optimized random forest models. *Physics of Fluids*, 37(1), 2025.
- I.-F. Kao, J.-Y. Liou, M.-H. Lee, and F.-J. Chang. Fusing stacked autoencoder and long short-term memory for regional multistep-ahead flood inundation forecasts. *Journal of Hydrology*, 598:126371, 2021.
- F. Karim, M. A. Armin, D. Ahmedt-Aristizabal, L. Tychsen-Smith, and L. Petersson. A review of hydrodynamic and machine learning approaches for flood inundation modeling. *Water*, 15(3):566, 2023.
-

-
- G. E. Karniadakis, I. G. Kevrekidis, L. Lu, P. Perdikaris, S. Wang, and L. Yang. Physics-informed machine learning. *Nature Reviews Physics*, 3(6):422–440, 2021.
- A. Kashefi, D. Rempe, and L. J. Guibas. A point-cloud deep learning framework for prediction of fluid flow fields on irregular geometries. *Physics of Fluids*, 33(2), 2021.
- K. Khosravi, H. Shahabi, B. T. Pham, J. Adamowski, A. Shirzadi, B. Pradhan, J. Dou, H.-B. Ly, G. Gróf, H. L. Ho, et al. A comparative assessment of flood susceptibility modeling using multi-criteria decision-making analysis and machine learning methods. *Journal of Hydrology*, 573:311–323, 2019.
- K. Khosravi, M. Panahi, A. Golkarian, S. D. Keesstra, P. M. Saco, D. T. Bui, and S. Lee. Convolutional neural network approach for spatial prediction of flood hazard at national scale of iran. *Journal of Hydrology*, 591:125552, 2020.
- D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
- A. H. Kohanpur, S. Saksena, S. Dey, J. M. Johnson, M. S. Riasi, L. Yeghiazarian, and A. M. Tartakovsky. Urban flood modeling: Uncertainty quantification and physics-informed gaussian processes regression forecasting. *Water Resources Research*, 59(3):e2022WR033939, 2023.
- F. Kratzert, D. Klotz, C. Brenner, K. Schulz, and M. Herrnegger. Rainfall–runoff modelling using long short-term memory (lstm) networks. *Hydrology and Earth System Sciences*, 22(11):6005–6022, 2018.
- F. Kratzert, D. Klotz, M. Herrnegger, A. K. Sampson, S. Hochreiter, and G. S. Nearing. Toward improved predictions in ungauged basins: Exploiting the power of machine learning. *Water Resources Research*, 55(12):11344–11354, 2019.
- A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.
-

-
- D. Kumar, A. Singh, P. Samui, and R. K. Jha. Forecasting monthly precipitation using sequential modelling. *Hydrological sciences journal*, 64(6):690–700, 2019.
- V. Kumar, K. V. Sharma, T. Caloiero, D. J. Mehta, and K. Singh. Comprehensive overview of flood modeling approaches: A review of recent advances. *Hydrology*, 10(7):141, 2023.
- J. N. Kutz. Deep learning in fluid dynamics. *Journal of Fluid Mechanics*, 814:1–4, 2017.
- I. E. Lagaris, A. Likas, and D. I. Fotiadis. Artificial neural networks for solving ordinary and partial differential equations. *IEEE transactions on neural networks*, 9(5):987–1000, 1998.
- Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989.
- Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.
- Y. Lee, H. Yang, and Z. Yin. Piv-dcnn: cascaded deep convolutional neural networks for particle image velocimetry. *Experiments in Fluids*, 58(12):171, 2017.
- T. Li, L. Biferale, F. Bonaccorso, M. A. Scarpolini, and M. Buzzicotti. Synthetic lagrangian turbulence by generative diffusion models. *Nature Machine Intelligence*, 6(4):393–403, 2024a.
- W. Li, C. Liu, C. Hu, C. Niu, R. Li, M. Li, Y. Xu, and L. Tian. Application of a hybrid algorithm of lstm and transformer based on random search optimization for improving rainfall-runoff simulation. *Scientific Reports*, 14(1):11184, 2024b.
- Z. Li and I. Demir. U-net-based semantic classification for flood extent extraction using sar imagery and gee platform: A case study for 2019 central us flooding. *Science of The Total Environment*, 869:161757, 2023.
-

-
- Q. Lin, J. Leandro, S. Gerber, and M. Disse. Multistep flood inundation forecasts with resilient backpropagation neural networks: Kulmbach case study. *Water*, 12(12):3568, 2020.
- B. Liu, Y. Li, M. Ma, and B. Mao. A comprehensive review of machine learning approaches for flood depth estimation. *International Journal of Disaster Risk Science*, pages 1–13, 2025.
- W. Lou, X. Gao, J. H. W. Lee, J. Liu, J. Peng, L. Dong, and K. Gao. A highly generalizable data-driven model for spatiotemporal urban flood dynamics real-time forecasting based on coupled cnn and convlstm. *EGUsphere*, 2025:1–24, 2025.
- R. Löwe, J. Böhm, D. G. Jensen, J. Leandro, and S. H. Rasmussen. U-flood—topographic deep learning for predicting urban pluvial flood water depth. *Journal of Hydrology*, 603:126898, 2021.
- L. Lu, X. Meng, Z. Mao, and G. E. Karniadakis. Deepxde: A deep learning library for solving differential equations. *SIAM review*, 63(1):208–228, 2021a.
- L. Lu, R. Pestourie, W. Yao, Z. Wang, F. Verdugo, and S. G. Johnson. Physics-informed neural networks with hard constraints for inverse design. *SIAM Journal on Scientific Computing*, 43(6):B1105–B1132, 2021b.
- L. Lu, T. Yan, Y. Chen, H. Wang, T. Yang, and Z. Liu. A lake water level prediction method based on data augmentation and physics-informed neural networks with imbalanced data. *Journal of Hydrology*, page 134660, 2025.
- K. Maharana, S. Mondal, and B. Nemade. A review: Data pre-processing and data augmentation techniques. *Global Transitions Proceedings*, 3(1):91–99, 2022.
- R. B. Mahesh, J. Leandro, and Q. Lin. Physics informed neural network for spatial-temporal flood forecasting. In *Climate Change and Water Security: Select Proceedings of VCDRR 2021*, pages 77–91. Springer, 2022.
- H. R. Maier, A. Jain, G. C. Dandy, and K. P. Sudheer. Methods used for the development of neural networks for the prediction of water resource variables in

-
- river systems: Current status and future directions. *Environmental modelling & software*, 25(8):891–909, 2010.
- S. K. Manchingal and F. Cuzzolin. Position: Epistemic artificial intelligence is essential for machine learning models to know when they do not know. *arXiv preprint arXiv:2505.04950*, 2025.
- S. Manfreda, J. A. Saavedra Navarro, C. Albertini, R. Zhuang, F. D. Pacia, S. Chaturvedi, and C. Samela. Geomorphic flood index 2.0: Enhanced tools for delineating flood-prone areas in data-scarce regions. *Available at SSRN 5461780*, 2025.
- D. Maruyama, P. Bekemeyer, S. Görtz, S. Coggon, and S. Sharma. Data-driven bayesian inference of turbulence model closure coefficients incorporating epistemic uncertainty. *Acta Mechanica Sinica*, 37(12):1812–1838, 2021.
- W. S. McCulloch and W. Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4):115–133, 1943.
- T. M. Mitchell. *Machine Learning*. McGraw-Hill, New York, 1997. ISBN 978-0070428072.
- A. Mosavi, P. Ozturk, and K.-w. Chau. Flood prediction using machine learning models: Literature review. *Water*, 10(11):1536, 2018.
- V. Nair and G. E. Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 807–814, 2010.
- P. C. Nayak, K. Sudheer, D. Rangan, and K. Ramasastri. A neuro-fuzzy computing technique for modeling hydrological time series. *Journal of Hydrology*, 291(1-2): 52–66, 2004.
- G. Nearing, D. Cohen, V. Dube, M. Gauch, O. Gilon, S. Harrigan, A. Hassidim, D. Klotz, F. Kratzert, A. Metzger, et al. Global prediction of extreme floods in ungauged watersheds. *Nature*, 627(8004):559–563, 2024.
-

-
- G. S. Nearing, F. Kratzert, A. K. Sampson, C. S. Pelissier, D. Klotz, J. M. Frame, C. Prieto, and H. V. Gupta. What role does hydrological science play in the age of machine learning? *Water Resources Research*, 57(3):e2020WR028091, 2021.
- E. Nemni, J. Bullock, S. Belabbes, and L. Bromley. Fully convolutional neural network for rapid flood segmentation in synthetic aperture radar imagery. *Remote Sensing*, 12(16):2532, 2020.
- S. Nevo, E. Morin, A. Gerzi Rosenthal, A. Metzger, C. Barshai, D. Weitzner, D. Voloshin, F. Kratzert, G. Elidan, G. Dror, et al. Flood forecasting with machine learning models in an operational framework. *Hydrology and Earth System Sciences*, 26(15):4013–4032, 2022.
- A. Y. Ng. Feature selection, l_1 vs. l_2 regularization, and rotational invariance. In *Proceedings of the twenty-first international conference on Machine learning*, page 78, 2004.
- H. D. Nguyen, D. K. Dang, Y. N. Nguyen, C. Pham Van, T. T. Van Nguyen, Q.-H. Nguyen, X. L. Nguyen, L. T. Pham, V. T. Pham, and Q.-T. Bui. Integration of machine learning and hydrodynamic modeling to solve the extrapolation problem in flood depth estimation. *Journal of Water and Climate Change*, page jwc2023573, 2023.
- M. Panahi, A. Jaafari, A. Shirzadi, H. Shahabi, O. Rahmati, E. Omidvar, S. Lee, and D. T. Bui. Deep learning neural networks for spatially explicit prediction of flash flood probability. *Geoscience Frontiers*, 12(3):101076, 2021.
- L. Pavesi, C. D’Angelo, E. Volpi, and A. Fiori. Rescue: A geomorphology-based, hydrologic-hydraulic model for large-scale inundation mapping. *Journal of Flood Risk Management*, 15(4):e12841, 2022.
- B. Peng, Z. Meng, Q. Huang, and C. Wang. Patch similarity convolutional neural network for urban flood extent mapping using bi-temporal satellite multispectral imagery. *Remote Sensing*, 11(21):2492, 2019.
- M. Pianforini, S. Dazzi, A. Pilzer, and R. Vacondio. Real-time flood maps forecasting
-

-
- for dam-break scenarios with a transformer-based deep learning model. *Journal of Hydrology*, 635:131169, 2024.
- B. Pradhan, S. Lee, A. Dikshit, and H. Kim. Spatial flood susceptibility mapping using an explainable artificial intelligence (xai) model. *Geoscience Frontiers*, 14(6):101625, 2023.
- P. Prestininzi, G. Di Baldassarre, G. Schumann, and P. Bates. Selecting the appropriate hydraulic model structure using low-resolution satellite imagery. *Advances in Water Resources*, 34(1):38–46, 2011.
- X. Qi, G. A. de Almeida, and S. Maldonado. Physics-informed neural networks for solving flow problems modeled by the 2d shallow water equations without labeled data. *Journal of Hydrology*, 636:131263, 2024.
- K. Qian, A. Mohamed, and C. Claudel. Physics informed data driven model for flood prediction: application of deep learning in prediction of urban flood development. *arXiv preprint arXiv:1908.10312*, 2019.
- J. Qin, J. Liang, T. Chen, X. Lei, and A. Kang. Simulating and predicting of hydrological time series based on tensorflow deep learning. *Polish Journal of Environmental Studies*, 28(2), 2019.
- M. Raissi, P. Perdikaris, and G. E. Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, 2019.
- S. J. Raymond and D. B. Camarillo. Applying physics-based loss functions to neural networks for improved generalizability in mechanics problems. *arXiv preprint arXiv:2105.00075*, 2021.
- L. Rokach and O. Maimon. Clustering methods. In *Data mining and knowledge discovery handbook*, pages 321–352. Springer, 2005.
- O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical image computing and computer-assisted*

-
- intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18*, pages 234–241. Springer, 2015.
- F. Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386, 1958.
- D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, 1986.
- L. B. Santos, E. V. Escobar-Silva, L. F. Satolo, R. S. Oyarzabal, M. M. Diniz, R. G. Negri, G. R. Lima, S. Stephany, J. A. Soares, J. S. Duque, et al. Machine learning-based hydrological models for flash floods: a systematic literature review. *Smart Construction and Sustainable Cities*, 3(1):21, 2025.
- C. Sarker, L. Mejias, F. Maire, and A. Woodley. Flood mapping with convolutional neural networks using spatio-contextual pixel information. *Remote Sensing*, 11(19):2331, 2019.
- S. Sellars. “grand challenges” in big data and the earth sciences. *Bulletin of the American Meteorological Society*, 99(6):ES95–ES98, 2018.
- P. Shamsolmoali, M. Zareapoor, R. Wang, H. Zhou, and J. Yang. A novel deep structure u-net for sea-land segmentation in remote sensing images. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 12(9):3219–3232, 2019.
- C. Shen. A transdisciplinary review of deep learning research and its relevance for water resources scientists. *Water Resources Research*, 54(11):8558–8593, 2018.
- C. Shorten and T. M. Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of Big Data*, 6(1):1–48, 2019.
- K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.
- M. Sit, B. Z. Demiray, Z. Xiang, G. J. Ewing, Y. Sermet, and I. Demir. A comprehensive review of deep learning applications in hydrology and water resources. *Water Science and Technology*, 82(12):2635–2670, 2020.
-

-
- F. Sofos, C. Stavrogiannis, K. K. Exarchou-Kouveli, D. Akabua, G. Charilas, and T. E. Karakasidis. Current trends in fluid research in the era of artificial intelligence: A review. *Fluids*, 7(3):116, 2022.
- D. P. Solomatine and A. Ostfeld. Data-driven modelling: some past experiences and new approaches. *Journal of hydroinformatics*, 10(1):3–22, 2008.
- M. Taghizadeh, Z. Zandsalimi, M. Shafiee-Jood, and N. Alemazkoo. Multi-fidelity graph neural networks for efficient and accurate flood hazard mapping. *Environmental Modelling & Software*, page 106654, 2025.
- S. Tarquini, I. Isola, M. Favalli, A. Battistini, and G. Dotta. Tinitaly, a digital elevation model of italy with a 10 meters cell size (version 1.1). *Istituto Nazionale di Geofisica e Vulcanologia (INGV)*, 10, 2023.
- J.-M. Tucny, M. Lauricella, M. Durve, G. Guglielmo, A. Montessori, and S. Succi. Physics-informed neural networks for microflows: Rarefied gas dynamics in cylinder arrays. *Journal of Computational Science*, 87:102575, 2025.
- U.S. Army Corps of Engineers, Hydrologic Engineering Center. *HEC-RAS River Analysis System: Hydraulic Reference Manual*. US Army Corps of Engineers, Davis, California, version 6.0 edition, 2021. URL <https://www.hec.usace.army.mil/software/hec-ras/documentation.aspx>. HEC-RAS Hydraulic Reference Manual.
- S. P. Van, H. M. Le, D. V. Thanh, T. D. Dang, H. H. Loc, and D. T. Anh. Deep learning convolutional neural network in rainfall–runoff modelling. *Journal of Hydroinformatics*, 22(3):541–561, 2020.
- A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.
- H. Wang and F. Miao. Building extraction from remote sensing images using deep residual u-net. *European Journal of Remote Sensing*, 55(1):71–85, 2022.

-
- H. Wang, H. Qin, G. Liu, S. Huang, Y. Qu, X. Qi, and Y. Zhang. Hierarchical attention network for short-term runoff forecasting. *Journal of Hydrology*, 638:131549, 2024.
- Y. Wang, Z. Fang, H. Hong, and L. Peng. Flood susceptibility mapping using convolutional neural network frameworks. *Journal of hydrology*, 582:124482, 2020.
- M. Waqas and U. W. Humphries. A critical review of rnn and lstm variants in hydrological time series predictions. *MethodsX*, 13:102946, 2024.
- P. J. Werbos. Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*, 78(10):1550–1560, 2002.
- Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and P. S. Yu. A comprehensive survey on graph neural networks. *IEEE transactions on neural networks and learning systems*, 32(1):4–24, 2020.
- T. Xu and F. Liang. Machine learning for hydrologic sciences: An introductory overview. *Wiley Interdisciplinary Reviews: Water*, 8(5):e1533, 2021.
- Z. Xu, S. Wang, L. V. Stanislawski, Z. Jiang, N. Jaroenchai, A. M. Sainju, E. Shavers, E. L. Usery, L. Chen, Z. Li, et al. An attention u-net model for detection of fine-scale hydrologic streamlines. *Environmental Modelling & Software*, 140:104992, 2021.
- Z. Xu, Z. Liu, and Y. Peng. Performance comparison of prediction of hydraulic jump length under multiple neural network models. *IEEE Access*, 2024.
- F. Yang, W. Ding, J. Zhao, L. Song, D. Yang, and X. Li. Rapid urban flood inundation forecasting using a physics-informed deep learning approach. *Journal of Hydrology*, 643:131998, 2024.
- H. Yin, L. Zhao, M. Zhu, and Y. Zhang. Runoff prediction in gauged and ungauged basins using transformer-xaj model. *Journal of Hydrology*, page 133954, 2025.
- M. G. Zanoni, B. Majone, and A. Bellin. A catchment-scale model of river water quality by machine learning. *Science of the Total Environment*, 838:156377, 2022.
-

-
- B. Zeng, G. Huang, and W. Chen. Research progress and prospects of urban flooding simulation: From traditional numerical models to deep learning approaches. *Environmental Modelling & Software*, 183:106213, 2025.
- J.-S. Zhang and X.-C. Xiao. Predicting chaotic time series using recurrent neural network. *Chinese Physics Letters*, 17(2):88, 2000.
- Z. Zhang, K. Flora, S. Kang, A. B. Limaye, and A. Khosronejad. Data-driven prediction of turbulent flow statistics past bridge piers in large-scale rivers using convolutional neural networks. *Water Resources Research*, 58(1):e2021WR030163, 2022.
- J. Zhao, Y. Li, P. Matgen, R. Pelich, R. Hostache, W. Wagner, and M. Chini. Urban-aware u-net for large-scale urban flood mapping using multitemporal sentinel-1 intensity and interferometric coherence. *IEEE Transactions on Geoscience and Remote Sensing*, 60:1–21, 2022.
- Y. Zhou, W. Wu, R. Nathan, and Q. J. Wang. A rapid flood inundation modelling framework using deep learning with spatial reduction and reconstruction. *Environmental Modelling & Software*, 143:105112, 2021.
- Y. Zhu, N. Zabaras, P.-S. Koutsourelakis, and P. Perdikaris. Physics-constrained deep learning for high-dimensional surrogate modeling and uncertainty quantification without labeled data. *Journal of Computational Physics*, 394:56–81, 2019.