PHYSICS-GUIDED STRATEGIES FOR ENHANCING NEURAL NETWORKS TRAINED WITH LIMITED DATA

Jose Guadalupe Perez Zamora

Doctoral Program in Computer Science

APPROVED:

Olac Fuentes, Chair, Ph.D.

Vinod Kumar, Ph.D.

Aritran Piplai, Ph.D.

Stephen Crites, Ph.D.
Dean of the Graduate School

PHYSICS-GUIDED STRATEGIES FOR ENHANCING NEURAL NETWORKS TRAINED WITH LIMITED DATA

by

Jose Guadalupe Perez Zamora

DISSERTATION

Presented to the Faculty of the Graduate School of

The University of Texas at El Paso

in Partial Fulfillment

of the Requirements

for the Degree of

DOCTOR OF PHILOSOPHY

Department of Computer Science

THE UNIVERSITY OF TEXAS AT EL PASO

December 2025

Acknowledgments

I am deeply grateful to Dr. Olac Fuentes for his mentorship and guidance throughout my academic career, from Data Structures all the way to the completion of this dissertation. Thank you for your constant support, encouragement, and belief in me. It was an honor to be part of your Vision and Learning Laboratory at UTEP, meeting and collaborating with so many talented people, and having had the chance to lead the group for a time.

I also thank my research collaborators, Dr. Arshad M. Khan, Dr. Vinod Kumar, and Dr. Christopher Kiekintveld, as well as my friends and colleagues from their labs: Dr. Bibek Aryal, Dr. Arturo Rodriguez, Dr. Moinul Morshed Porag Chowdhury, Dr. Anthony Ortiz, Dr. Hassan Mahmudulla, Dr. Alexandro Arnal, Harshavardhini Bagavathyraj, Jose A. Terrazas, Rafael Baez Ramirez, and Monjur Bin Shams. Thank you for your insight, friendship, and support.

I would also like to acknowledge my mentors and colleagues from my internship at the Johns Hopkins Applied Physics Laboratory: Dr. Luis Puche, Dr. Pedro Rodriguez, Beatrice Garcia, Kevin Chiou, and Malik Jackson. Thank you for teaching me so much, for making me feel welcome and confident in my skills, and for appreciating my work.

I am also thankful to the programs that made my education possible: Mission Early College High School for funding my core undergraduate courses. The BUILDing Scholars program at UTEP and COURI for connecting me with Dr. Fuentes, supporting my studies and conference travel, and preparing me for graduate research. CAHSI for awarding me the Google–CAHSI Dissertation Award. And the Computer Science Department for continued research and teaching assistant opportunities.

Finally, I thank my family and friends for their unwavering support and belief in me. To my parents, my siblings Alex and Danny, my niece Xime, and my beautiful wife Reanne. All your love and support made all this possible.

Abstract

Deep neural networks excel at a wide range of processing tasks across various disciplines. However, the quantity and quality of data significantly impact network performance. In specialized domains, high-quality datasets are often difficult to gather, interpret, and curate for effective learning. Few-shot learning techniques, including transfer learning, data augmentation, and meta-learning, have emerged to address these constraints.

We propose three **physics-guided strategies** for enhancing neural networks trained with limited data: (1) combining existing models like LSTMs with Physics-Informed Neural Networks through two-branch architectures that merge their outputs, (2) deriving custom physics-informed data augmentation algorithms to expand limited datasets, and (3) adding physics-informed terms to the loss functions of existing models like U-Nets. These strategies are applied to two challenging problems: Fluid Flow Velocity Prediction in mechanical engineering and Glacier Segmentation in geology.

For fluid flow prediction, we developed a novel **Physics-Informed LSTM** architecture that achieved **73.7%** improvement in U-velocity and **85.3%** improvement in V-velocity prediction compared to LSTM-only approaches. For glacier segmentation, our **physics-informed data augmentation** improved clean-ice IoU from 68.17 to 71.22 (a **4.5%** gain) and debris-covered ice IoU from 35.94 to 45.92 (a **27.8%** gain) over the baseline. We also developed a **physics-informed velocity loss** that fuses glacier velocity fields to promote physically consistent segmentations. The combination of all our glacier segmentation strategies achieved a state-of-the-art Debris-Covered Ice IoU of **46.07%**, representing a **28.2%** relative improvement over prior benchmarks. To enable our velocity physics experiments and open the door for future higher-resolution studies, we also built a production-ready, generalizable glacier-velocity dataset creation pipeline that fuses ITS_LIVE dynamics with Landsat/ICIMOD data.

This research demonstrates that integrating domain knowledge of physics into neural

networks provides a viable pathway for improving performance on data-limited problems where well-understood physical laws with mathematical representations can guide the learning process.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 The Data-Intensive Nature of Neural Networks

Deep neural networks have become the state-of-the-art in machine learning, achieving leading performance in image [1–3], text [4, 5], and video processing tasks across numerous disciplines. This success is driven by advances in optimization and training methods [6]. From large language models like GPT-3 [7] to object detection in self-driving cars [8], the impact of deep learning is widespread. However, the quantity and quality of data significantly impact network performance.

1.2 The Challenge of Data Acquisition

Data acquisition is a cornerstone of machine learning. Without sufficient high-quality data, models cannot be effectively trained. The scale of data required by modern architectures is immense. For example, GPT-3 was trained on 45 terabytes of text [7], while Google's JFT-300M dataset contains 300 million images [9]. While large datasets for common tasks like RGB image classification are sometimes publicly available [10, 11], specialized fields such as neuroscience, geology, and medicine often lack such resources. In these domains, high-quality datasets are often difficult to gather, interpret, and curate for effective learning, and data collection may require expert knowledge, specialized equipment, and significant time and financial investment, creating a bottleneck that limits the application of deep learning to new scientific frontiers.

1.3 Few-Shot Learning: Maximizing Performance With Limited Data

To address the challenges of data scarcity, the field of "Few-Shot Learning" has emerged [12]. The goal of few-shot learning is to train effective deep neural networks using a minimal amount of labeled data, thereby making deep learning more accessible. These approaches typically adapt a component of the traditional training pipeline to perform well with limited samples. The three main strategies are **transfer learning** [13], **data augmentation** [14], and **meta-learning** [15].

Although we present these approaches as separate fields in the area of "few-shot learning", it is very common to combine multiple approaches as part of your final model to maximize performance.

Transfer Learning Transfer learning involves adapting a pre-trained model to a new, related problem. This is typically done by fine-tuning a model that has been trained on a large dataset, such as ImageNet. While effective, the success of transfer learning depends on the similarity between the source and target domains. For specialized datasets, such as those in geology or medicine, finding a suitable pre-trained model can be challenging.

Data Augmentation Data augmentation techniques generate new training samples from an existing dataset. For images, common methods include rotation, cropping, and color adjustments. However, these transformations must be carefully chosen to be relevant to the task. For example, flipping a "6" in the MNIST dataset might turn it into a "9," leading to incorrect labels [16]. The effectiveness of data augmentation depends on designing transformations that preserve the data's original labels and features.

Meta-Learning Meta-learning, or "learning to learn," aims to train a model on a variety of learning tasks to enable rapid adaptation to new tasks. One approach, Model-Agnostic Meta-Learning (MAML), learns an optimal initialization for a model's weights, allowing it to be fine-tuned with only a few examples [17]. Another popular strategy is metric learning, which focuses on learning a distance function to compare data points. Siamese Networks [18], Prototypical Networks [19], and Relation Networks [20] are all examples of metric learning approaches that have been successfully applied to few-shot learning problems.

1.4 Few-Shot Learning By Integrating Physics Into Neural Networks

1.4.1 Reasoning Behind Integrating Physics

Humans excel at learning new concepts by leveraging extensive prior understanding, a capability often lacking in purely data-driven neural networks. This dissertation focuses on integrating physical principles into neural networks to enhance their performance and physical consistency. Physics, as a well-established scientific discipline, offers a wealth of mathematical formulations. Since real-world problems are governed by fundamental physical laws, incorporating these principles provides a complementary approach to purely data-driven methods. By leveraging these governing equations, we can improve model performance without needing additional data, leading the network towards physically consistent and robust solutions.

Scope of Physics Integration in Few-Shot Learning In this dissertation, physics integration is applied exclusively to traditional supervised few-shot learning approaches. While transfer learning and meta-learning are commonly employed strategies for addressing data scarcity, combining them with physics-guided methods was not pursued here. Meta-learning focuses on learning from distributions of tasks to enable rapid adaptation to new problems, whereas transfer learning leverages knowledge from pre-trained models to improve performance on related tasks. Integrating physics into these strategies would

require additional considerations, such as designing task- or domain-specific constraints compatible with the meta-learning framework or the source-target domain relationships in transfer learning. Evaluating these combinations was beyond the scope of this work. By focusing on traditional supervised few-shot learning, this thesis emphasizes clear and direct application of physics-guided strategies, allowing for rigorous analysis of their effectiveness across distinct real-world problems. Exploration of physics integration with meta-learning and transfer learning is left as a direction for future research.

1.4.2 Physics-Informed Neural Networks (PINNs)

There exists an area of study called Physics-Informed Neural Networks (PINNs) [21] where the aim is to integrate physics that can be described by systems of partial differential equations (PDEs) directly into the loss function of a network. Unless working with simulation data, it will be nearly impossible to collect all the variables and data needed to describe how a specific problem behaves exactly at all times. There is just too much data at too many different scales, from quantum effects all the way to space-time gravitational influences. Due to this, scales and times are often discretized when working with these PDEs like when using the Finite Element Method (FEM) to compute solutions for these PDEs. PINNs allow modeling of some of these intrinsic variables in the data using the physical laws (PDEs) governing their interactions to increase model performance.

However, it is essential to acknowledge the boundaries of this methodology. Physics-informed approaches thrive when the underlying dynamics are governed by well-defined, continuous partial differential equations. Their efficacy is challenged by phenomena exhibiting discontinuities, sharp gradients, or stochastic behaviors which are difficult to capture via standard PDE constraints. Furthermore, the reliance on known governing equations limits their applicability in scenarios where the physics are not fully understood or are computationally prohibitive to model explicitly.

1.4.3 Contribution: Physics-Guided Strategies

This thesis introduces and evaluates three distinct physics-guided strategies, applying them to datasets from different disciplines to assess their generalizability.

- Developing hybrid physics-informed network architectures that combine existing models (e.g., LSTMs) with Physics-Informed Neural Networks (PINNs) through two-branch models.
- Designing custom physics-informed data augmentation algorithms to expand limited datasets.
- Integrating physics-informed terms directly into the loss functions of existing models (e.g., U-Nets).

An overview of these strategies is presented in Figure 1.1.

These strategies were applied and evaluated with two specific problems from two different disciplines where physical laws play a crucial role.

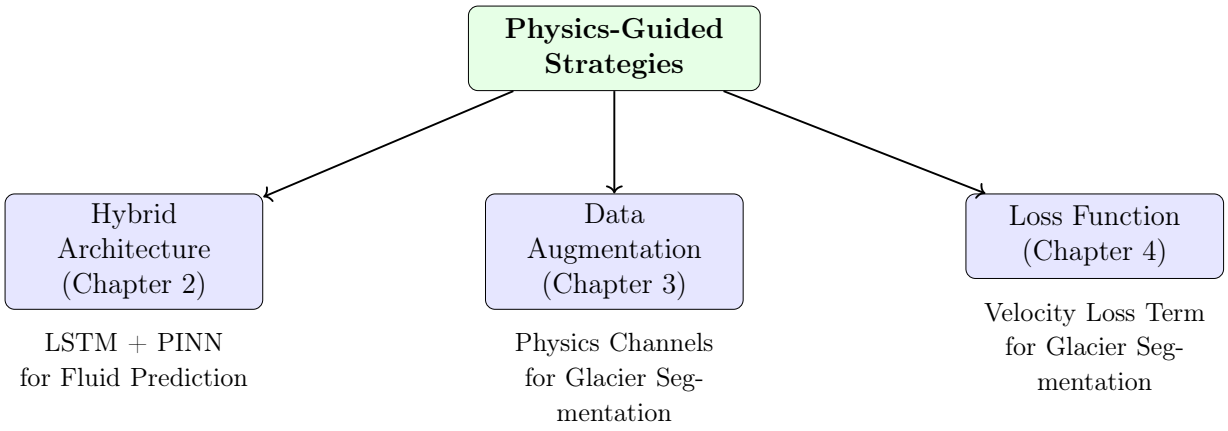


Figure 1.1: Overview of the three physics-guided strategies explored in this thesis.

The next section will describe those problems, and the last section will describe how the thesis is structured and what the thesis contributions are in more detail.

1.5 Specific Problems from Other Disciplines Guided by Physical Laws

Our research focused on applying physics to neural networks to improve model performance on problems where underlying physical laws provide strong guidance. We selected problems from two disciplines-Fluid Flow Velocity Prediction in mechanical engineering and Glacier Segmentation in geology-whose data inherently follows the laws of physics.

1.5.1 Fluid Flow Velocity Prediction

Understanding how fluids flow is critical for the study and the development of airplanes, cars, boats, rockets, and many other mechanical systems. One of the widely researched approaches used to study fluid flow is setting up Computational Fluid Dynamics (CFD) simulations using software developed specifically for that purpose like Ansys Fluent [22]. It is unfeasible to model every particle of every fluid we are interested in modeling at every scale and every point in space due to computational and time limitations, so it is necessary to define a discretized mesh of finite elements of specific size as shown in **Figure 1.2** below.

After deciding the scale which we are interested in modeling and studying, the next step is to define the relevant geometry, fluids, and boundary conditions for the simulation as shown in **Figures 1.3 and 1.4** below. Finally, a solver is selected, and the simulation is run for a specific period of time.

One drawback of high-fidelity CFD simulations is their computational expense. While fluid flow datasets exist, many researchers require predictions for specific problems with unique geometries and boundary conditions. Given that fluid flows are well-described by the Navier-Stokes equations-a system of partial differential equations (PDEs)-Physics-Informed

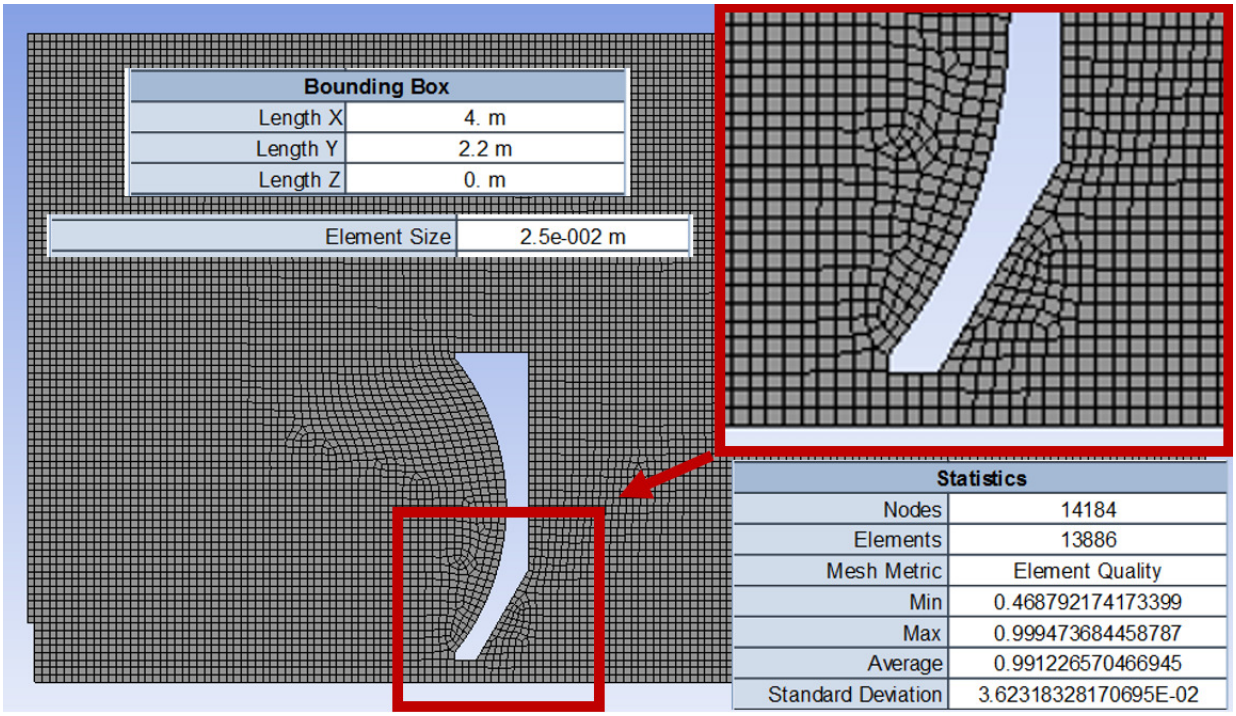


Figure 1.2: Example of a discretized mesh of finite elements used for a CFD fluid flow simulation. Source: [23]

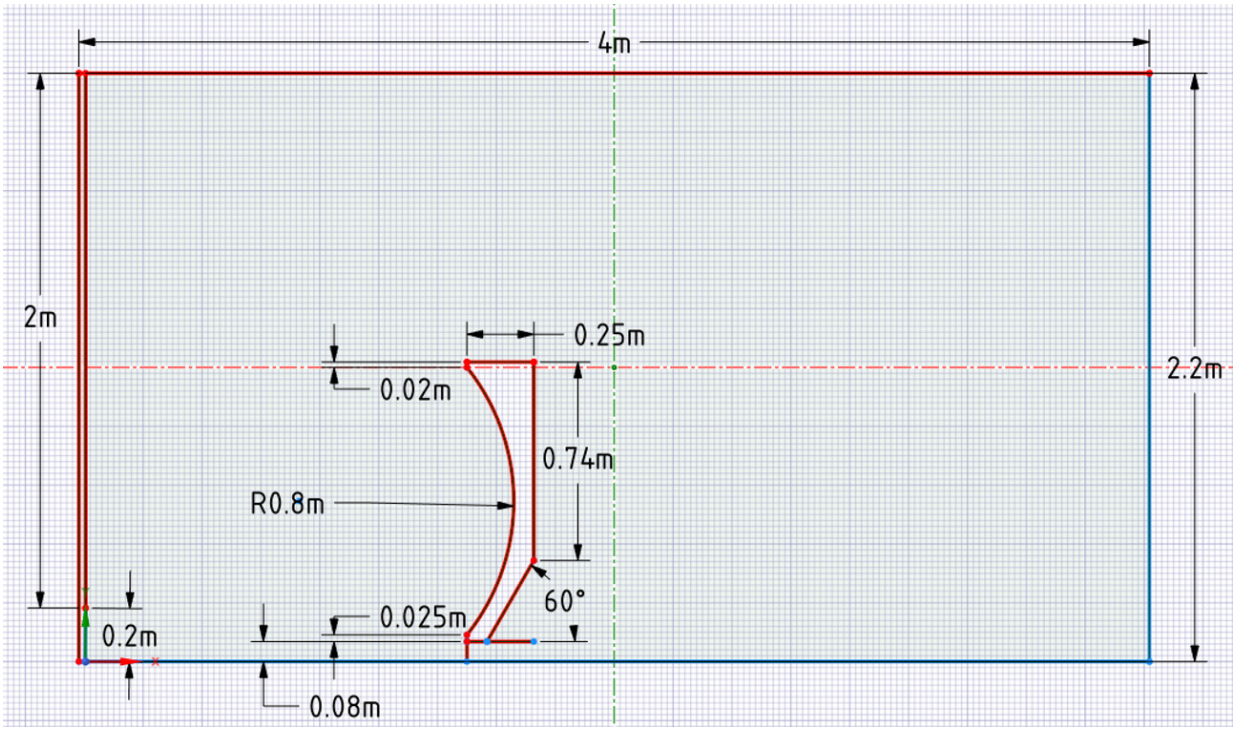


Figure 1.3: Example of geometry used for a CFD fluid flow simulation. This modeled geometry is for a proposed water-braking mechanism for a pusher sled system used in the Holloman Air Force Base for experiments. Source: [23]

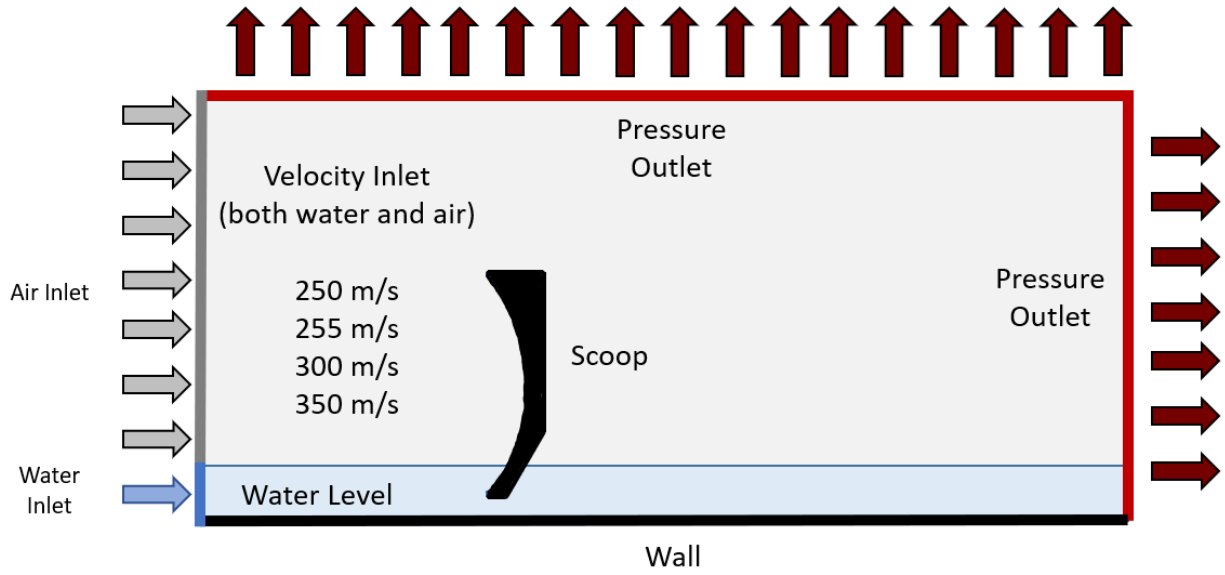


Figure 1.4: Example of boundary conditions used for a CFD fluid flow simulation. These boundary conditions model how the geometry (water-braking scoop) will interact with the water as the pusher sled mechanism pushes the scoop at specific initial velocities. Source: [23]

Neural Networks [21] offer a powerful approach to leverage this foundational understanding, enhancing model performance and physical consistency. The problem presented in this thesis is a well-posed, laminar flow problem. While simplified compared to turbulent regimes, it serves as an essential proof-of-concept (a toy problem in the academic sense) to validate the efficacy of our proposed hybrid architecture before applying it to complex, chaotic systems.

1.5.2 Glacier Segmentation

Glaciers are a vital source of water for people and wildlife across many different regions of the world, as not only do they provide a source of drinking water but also water for watering crops and generating hydroelectric power [24]. As these regions rely heavily on glacial melt as a water source, it is important to monitor and keep track of changes that happen to these glaciers over time.

Multiple satellites such as NASA's Landsat-7, Landsat-8, and Sentinel-2 have captured multispectral images of these glaciers over time. Glaciologists use these images to perform glacier mapping - identifying clean ice, debris-covered ice, and regular rocks through image segmentation. An example is shown in **Figure 1.5** below.

However, manually labeling multispectral satellite images to identify clean ice and debris-covered ice glaciers is a very time-consuming task. This is due to two main reasons:

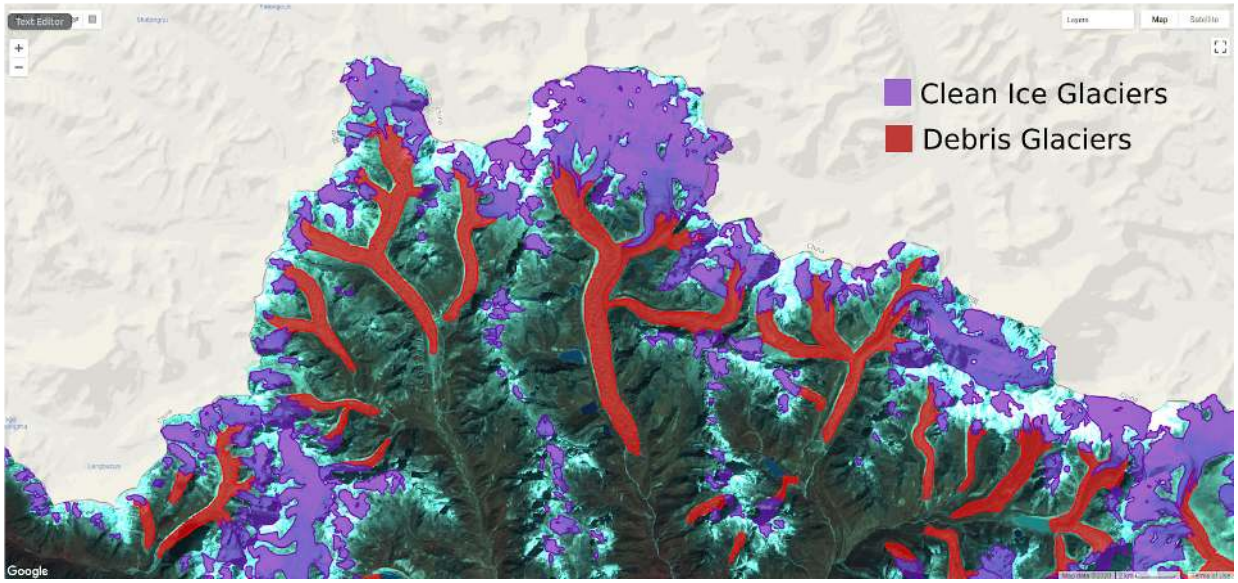


Figure 1.5: Example of glacier mapping or segmentation of a satellite image into clean ice and debris-covered ice glaciers. Source: [25]

- Satellite images have more channels than the standard three-channel (RGB) in digital cameras, so they are multispectral images and require the usage of specialized tools (like Quantum Geographic Information System (QGIS)) and knowledge about what channels capture which bands of the electromagnetic spectrum to visualize and analyze them properly.

- High-resolution images are difficult to segment as there are too many individual pixels, and although glaciologists use tools such as QGIS to label the images using geometric shapes instead of pixel-by-pixel to speed up the process, it remains time-consuming and difficult to accurately label the borders or boundaries between the clean ice and debris-covered ice specifically due to the amount of fine detail and care needed.

Labeling a single patch of a glacier using Sentinel-2 satellite imagery can take an expert hours, and the Hindu-Kush Himalayas (HKH) glaciers are made up of at least 256 of such patches.[1]

A labeled glacier dataset exists for the Hindu-Kush Himalayas (HKH) region, created by the International Centre for Integrated Mountain Development (ICIMOD) using NASA's Landsat-7 satellite imagery [26].

There also exists a dataset of glacier ice velocities created by the National Snow and Ice Data Center containing the surface velocities of major glacier-covered regions in the world including the Hindu-Kush Himalayas (HKH) spanning from 1985 to 2018 and compiled from multiple of NASA's Landsat satellites called the "Making Earth System Data Records

[1]Estimated effort and patch count from personal communication with a university geology professor; no published source was provided.

for Use in Research Environments (MEaSUREs) Inter-mission Time Series of Land Ice Velocity and Elevation (ITS_LIVE) Regional Glacier and Ice Sheet Surface Velocities" dataset [27]. We will use these datasets to evaluate our proposed contributions.

1.6 Thesis Contributions

This dissertation introduces and evaluates three distinct physics-guided strategies to enhance the performance and physical consistency of neural networks. Each strategy is applied to a challenging real-world problem, demonstrating the power of integrating physical principles into deep learning. The primary contributions are:

Hybrid Physics-Informed Model for Fluid Flow Prediction (Chapter 2): We propose a novel two-branch neural network combining a Long Short-Term Memory (LSTM) network with a Physics-Informed Neural Network (PINN). This hybrid model is applied to fluid flow velocity prediction, where the LSTM branch learns from sequential simulation data and the PINN branch enforces the Navier-Stokes equations. We demonstrated that this approach achieved **73.7%** improvement in U-velocity and **85.3%** improvement in V-velocity prediction compared to LSTM-only approaches.

Physics-Informed Data Augmentation for Glacier Segmentation (Chapter 3): To address limited labeled satellite imagery for glacier mapping, we developed a custom data augmentation algorithm grounded in ice flow physics. By generating new training samples that are both realistic and physically plausible, our method improves the performance of a state-of-the-art segmentation model [25] without requiring architectural changes. This strategy improved clean-ice IoU from 68.17 to 71.22 (a **4.5%** gain) and debris-covered ice IoU from 35.94 to 45.92 (a **27.8%** gain) over the baseline.

Physics-Informed Loss Function for Glacier Segmentation (Chapter 4): We integrated temporal velocity data and a physics-informed loss function into the training of a U-Net segmentation model. A foundational contribution of this chapter is the creation of a robust, temporally-aligned glacier velocity dataset from ITS_LIVE observations.

By incorporating these glacier dynamics and penalizing predictions that violate observed flow consistency (e.g., classifying moving terrain as background), the model produces more physically consistent segmentations. This strategy improved Debris-Covered Ice segmentation performance by **10.13 percentage points (28.2% relative improvement)** compared to the state-of-the-art baseline.

These contributions showcase a flexible framework for integrating physics into deep learning, offering complementary strategies to maximize model performance without additional data. Furthermore, our best performing model, which combines the physics-informed data augmentation from Chapter 3 with the physics-informed loss function from Chapter 4, achieved a Debris-Covered Ice IoU of **46.07%**, representing a **10.13 percentage point (28.2% relative)** improvement over the state-of-the-art. The final chapter of this thesis

discusses the broader implications of this work and outlines promising directions for future research.

Chapter 2

Physics-Informed LSTM Network For Velocity Prediction of Fluid Flow Simulations

2.1 The Importance of Fluid Flow Velocity Prediction

Predicting fluid flow is a critical task in mechanical engineering, with applications ranging from the design of aircraft wings to the optimization of turbines. By modeling fluid-geometry interactions, we can predict fluid velocities throughout a system over time. While high-fidelity simulations provide accurate predictions, they are computationally expensive. This chapter introduces a novel approach that combines deep learning with physics to achieve accurate and efficient fluid flow prediction.

As described in Chapter 1, Computational Fluid Dynamics (CFD) simulations using software such as Ansys Fluent [22] are widely used for fluid flow velocity prediction, as shown in Figure 2.1. The simulation algorithms differ in their approaches, with higher-fidelity algorithms using more resources to produce higher quality results more closely matching reality. Other algorithms sacrifice some fidelity for lower computational costs by making certain assumptions or simplifications such as considering flow equations in two dimensions instead of three, simulating single-phase flow instead of multi-phase, assuming the flow is incompressible, or considering fluids to be non-reactive to avoid computing additional chemical reaction effects.

2.2 Background: Key Neural Network Architectures

2.2.1 Long Short-Term Memory Networks for Time-Series Data

Long Short-Term Memory (LSTM) networks are a type of Recurrent Neural Network (RNN) designed to capture long-term dependencies in sequential data [5]. LSTMs use a unique cell structure, illustrated in Figure 2.2, with input, forget, and output gates to selectively store and retrieve information over long sequences. This makes them well-suited for time-series prediction tasks, such as fluid flow simulation, where the state at a given time depends on a history of previous states.

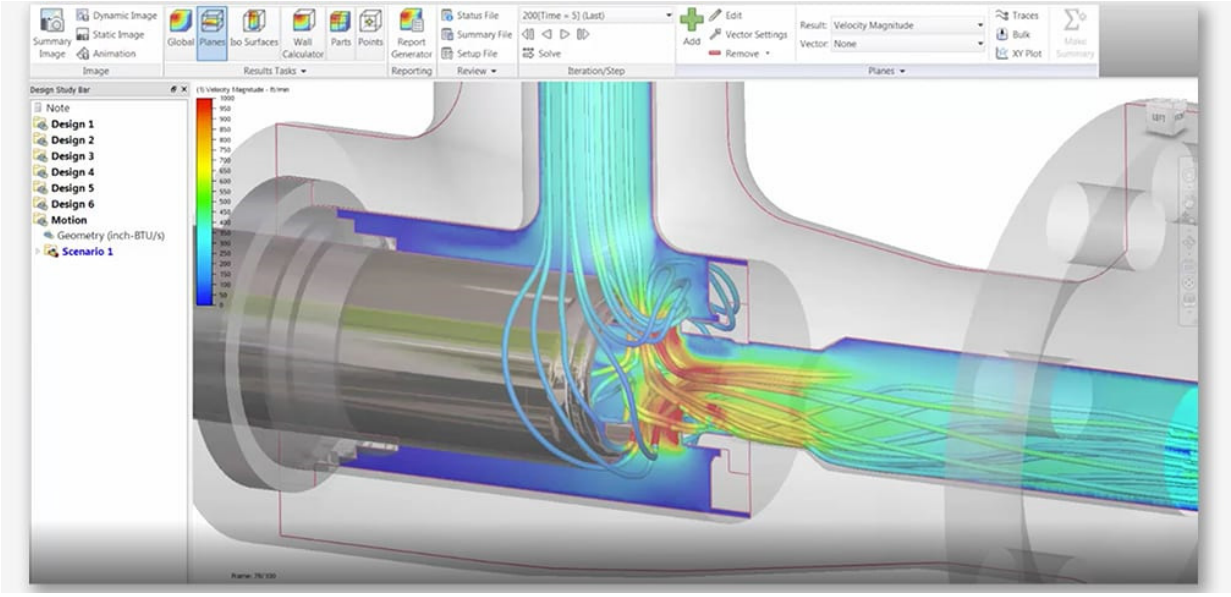


Figure 2.1: An example of a Computational Fluid Dynamics (CFD) simulation predicting the fluid flow velocities in an engine.

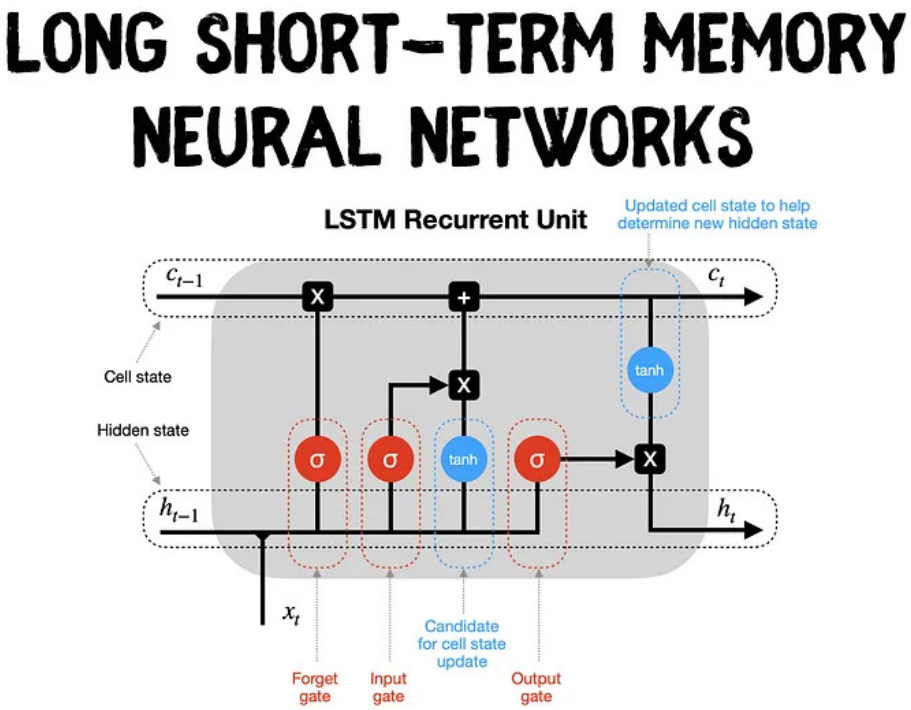


Figure 2.2: An LSTM cell or unit showing all the gates and operations.

LSTM Architecture

The core of the LSTM architecture is the cell state, which acts as a memory channel, allowing information to flow through the network. Gates, implemented as neural network layers with sigmoid activations, control the flow of information into and out of the cell state. A value of 0 closes the gate, while a value of 1 opens it, allowing the network to selectively remember or forget information at each time step.

LSTM Gates As an example, consider the prediction of the next word in a sentence given all the previous words. Consider the sentence:

"To pass this test you need to study".

Since this is a sequence, we index the words in order:

$$x_0 = \text{“To”}, \quad x_1 = \text{“pass”}, \quad x_2 = \text{“this”}, \quad \dots$$

The LSTM can be trained by feeding the input sequence

$$x = [x_0, x_1, x_2, \dots, x_T]$$

and asking the network to predict the sequence shifted one position to the left:

$$y = [y_0, y_1, y_2, \dots, y_T] \quad \text{where} \quad y_t = x_{t+1}.$$

Thus, at each time step the model receives a word and must predict the next one.

At the beginning of training and at the start of every new sequence, the hidden state and cell state are initialized to zero:

$$h_{-1} = 0, \qquad c_{-1} = 0.$$

LSTM Computation at Time Step t At time step t, the LSTM receives the current input word embedding x_t, the previous hidden state h_{t-1}, and the previous cell state c_{t-1}. The gates operate as follows:

Forget Gate Decides how much of the previous cell state to keep:

$$f_t = \sigma(W_f x_t + U_f h_{t-1} + b_f).$$

Input Gate Determines what new information to store in the cell state:

$$i_t = \sigma(W_i x_t + U_i h_{t-1} + b_i),$$

$$\tilde{c}_t = \tanh(W_c x_t + U_c h_{t-1} + b_c),$$

where $\tilde{c}_t$ is the candidate cell content.

Cell State Update The forget and input gates combine to update memory:

$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t.$$

Output Gate Determines what part of the updated cell state becomes the hidden output:

$$o_t = \sigma(W_o x_t + U_o h_{t-1} + b_o),$$

$$h_t = o_t \odot \tanh(c_t).$$

Here h_t is the output of the LSTM at time step t, and is used to predict the next word via a softmax layer:

$$\hat{y}_t = \text{softmax}(W_y h_t + b_y).$$

Bidirectional LSTMs

While standard LSTMs process sequences in the forward direction only, bidirectional LSTMs enhance temporal understanding by processing the sequence in both forward and backward directions simultaneously [28]. This architecture maintains two separate hidden states: one processing the sequence from start to end ($\overrightarrow{h_t}$), and another from end to start ($\overleftarrow{h_t}$). The final output at each timestep combines information from both directions, allowing the model to capture both past context and future context within the sequence. This capability is particularly valuable in fluid flow prediction where the state at a given time point may depend on both historical and upcoming flow patterns. The combined hidden state h_t is typically formed by concatenating the forward and backward hidden states:

$$h_t = [\overrightarrow{h_t}; \overleftarrow{h_t}] \tag{2.1}$$

Training LSTMs with Backpropagation Through Time

LSTMs are trained using Backpropagation Through Time (BPTT) [29], an extension of standard backpropagation that unfolds the recurrent network through multiple time steps. The total loss $\mathcal{L}$ is the sum of the losses at each time step, and the gradient is computed by summing the contributions from each time step in the past:

$$\frac{\partial \mathcal{L}}{\partial W} = \sum_{t=1}^{T} \frac{\partial \mathcal{L}_t}{\partial W} \tag{2.2}$$

The primary challenge that LSTMs address through their gated architecture is the vanishing and exploding gradient problem that plagues basic recurrent neural networks [5]. In traditional RNNs, gradients can either shrink exponentially (vanishing) or grow exponentially (exploding) as they propagate through many time steps. The LSTM's gate mechanisms with sigmoid activations help maintain gradient flow over extended sequences by providing more stable gradient pathways, enabling effective learning of temporal patterns that span hundreds or thousands of time steps.

2.2.2 Physics-Informed Neural Networks (PINNs) for Fluid Flow Prediction

Physics-Informed Neural Networks (PINNs) integrate physical laws, such as the Navier-Stokes equations, directly into the neural network's training process [21]. Unlike purely data-driven models, PINNs are constrained to produce solutions that are consistent with fundamental physical principles. This is achieved by adding a physics-based loss term to the standard data-driven loss function. This term penalizes the network for violating the governing physical equations, guiding the model toward physically plausible solutions, as illustrated in Figure 2.3.

Mathematical Foundation of PINNs

The core innovation of PINNs lies in their loss function formulation, which combines traditional data fidelity with physics-based constraints. For a neural network f_θ with parameters θ that approximates solution to partial differential equation $\mathcal{N}[u] = 0$, the total loss function takes the form:

$$\mathcal{L}_{total} = \mathcal{L}_{data} + \lambda \mathcal{L}_{physics}$$

where $\mathcal{L}_{data}$ represents the mean squared error between network predictions and available training data, $\mathcal{L}_{physics}$ enforces satisfaction of governing equations, and λ balances the relative importance of data fidelity versus physics constraints.

The physics loss component is computed by evaluating the residual of the governing equations at collocation points throughout the domain:

$$\mathcal{L}_{physics} = \frac{1}{N_c} \sum_{i=1}^{N_c} |\mathcal{N}[f_\theta(x_i, t_i)]|^2$$

where N_c is the number of collocation points and (x_i, t_i) represent spatial-temporal coordinates where physics constraints are enforced.

Automatic Differentiation for PDE Constraints

A critical enabling technology for PINNs is automatic differentiation, which allows exact computation of partial derivatives required for evaluating PDE residuals. For fluid flow problems, we are specifically interested in the 2D Navier-Stokes equations for incompressible flow, which describe conservation of momentum and mass:

$$\begin{aligned} \frac{\partial u}{\partial t} + \frac{\partial p}{\partial x} + \lambda_1 \left(u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y} \right) - \lambda_2 \left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \right) &= 0 \\ \frac{\partial v}{\partial t} + \frac{\partial p}{\partial y} + \lambda_1 \left(u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y} \right) - \lambda_2 \left(\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2} \right) &= 0 \\ \frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} &= 0 \end{aligned} \tag{2.3}$$

where u and v are velocity components, p is pressure.

To enforce these equations, PINNs require computation of first and second-order spatial derivatives as well as temporal derivatives. Using automatic differentiation frameworks like PyTorch, these derivatives are computed analytically through the computational graph, avoiding numerical approximation errors associated with finite difference methods.

For a neural network output $u(x, t)$, automatic differentiation provides:

$$\frac{\partial u}{\partial t}, \quad \frac{\partial u}{\partial x}, \quad \frac{\partial^2 u}{\partial x^2}, \quad \frac{\partial^2 u}{\partial y^2}$$

These derivatives are then substituted directly into the Navier-Stokes equations to form the physics loss component. This residual term acts as an explicit regularizer: by penalizing violations of the PDEs, the network is nudged toward physically plausible solutions even when temporal samples are sparse, improving generalization beyond the observed timesteps.

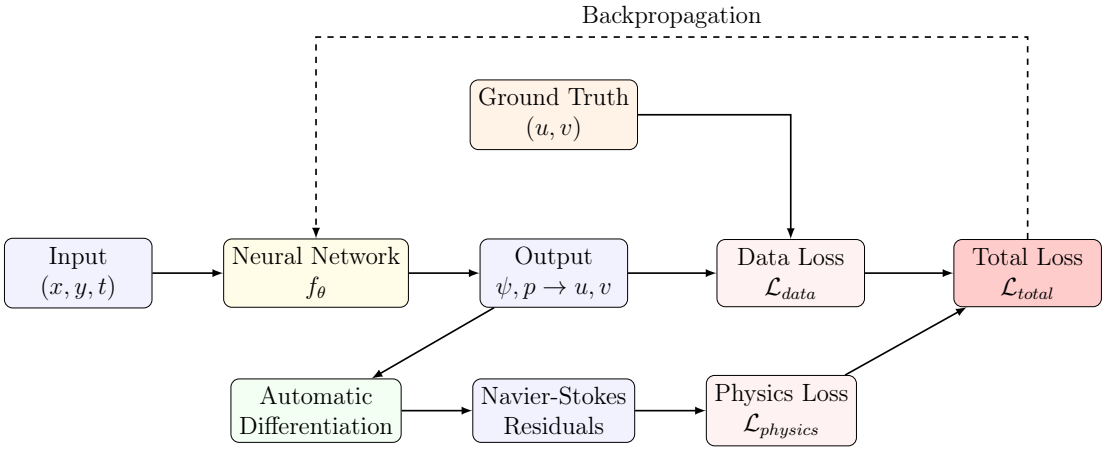


Figure 2.3: Workflow of a Physics-Informed Neural Network (PINN). The network outputs are differentiated with respect to the inputs to to compute the PDE residuals, which are then minimized as part of the loss function.

2.3 Problem Formulation and Evaluation

2.3.1 Fluid Flow Velocity Prediction as a Learning Problem

We formulate the fluid flow velocity prediction as a **regression problem**. The goal is to use a history of observations to predict future values.

Formally, the input X_t is a sequence of observation vectors from previous timesteps, represented as:

$$X_t = [x_{t-k}, \ldots, x_{t-1}] \tag{2.4}$$

where k is the lookback window (number of previous steps). Each x_i in the sequence is a single observation vector at timestep t_i. The output Y_t is the predicted velocity vector at the target time t.

Example: Using observations from times $t = 1.0$ s, 1.1 s, 1.2 s, we predict the velocity at $t = 1.3$ s. The time interval (e.g., 0.1 s) is determined by the simulation settings.

Table 2.1 details the specific physical measurements contained in each single observation vector x_i and the target output Y_t.

Table 2.1: Regression problem formulation for fluid flow velocity prediction.

Role	**Component**	**Unit**	**Description**
Input (x_i)	Spatial Coordinates	meters	Position (x, y) in the domain.
	Timestep	seconds	Time t_i of the observation.
	Pressure	atm	Fluid pressure p.
	Velocity (Horizontal)	m/s	Velocity u.
	Velocity (Vertical)	m/s	Velocity v.
Target Output (Y_t)	Velocity (Horizontal)	m/s	Predicted velocity u at time t.
	Velocity (Vertical)	m/s	Predicted velocity v at time t.

2.3.2 Measuring Network Performance with the Mean Squared Error

To measure the performance of our neural network, the Mean Squared Error (MSE) metric was used which computes how much our predictions deviated from actual values on average. The metric is defined in Equation 2.5 below.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_{pred} - y_{true})^2 \tag{2.5}$$

where y_{true} was the actual value, y_{pred} was our predicted value, and n was the number of values or samples being compared. For perfect predictions, the value of $MSE = 0$. Therefore, the closer to 0, the better the model's performance.

2.3.3 The Fluid Flow Velocity Dataset

The data was generated using the Ansys application, which can be used to run CFD simulations. A simulation environment in which we modeled a water-braking scoop mechanism in 2D space, constrained the domain of the model to a 2.2 m by 4 m box, and ran the simulations with different inlet velocity profiles. The simulation geometry, mesh, and boundary conditions can be seen in **Figures 1.3, 1.2, and 1.4** in the Introduction section.

The following figures, including Figures 2.4, 2.5, and 2.6, demonstrate the simulation results for different boundary conditions at specific times in the simulation. In figures containing two images, the left image represents the pressure and velocity contours, while the right image represents the velocity and water volume fraction.

The seven measurements stored for each data sample in the simulation are detailed in Table 2.2.

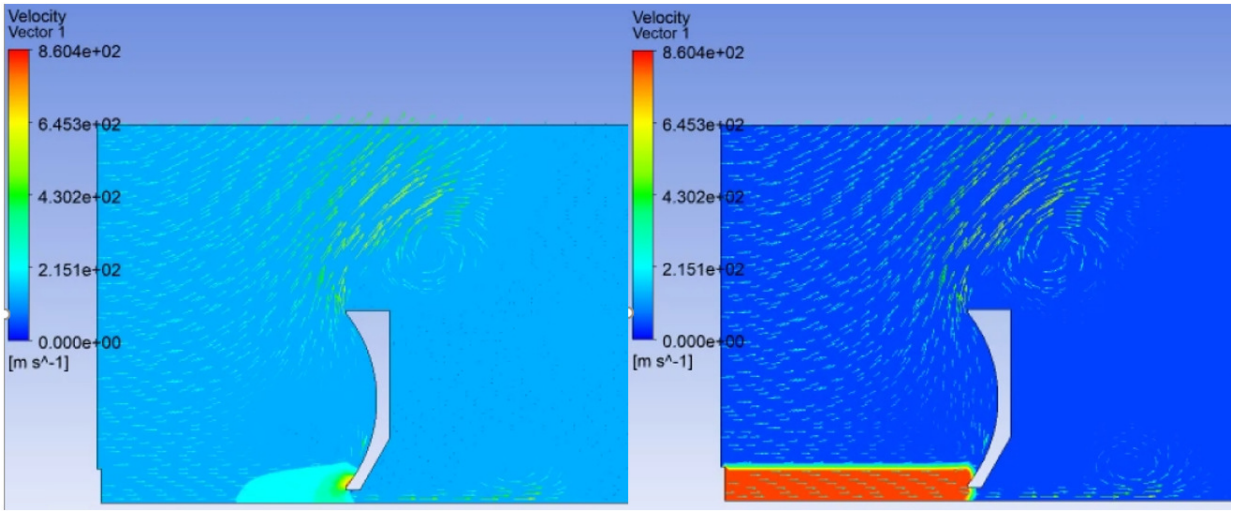


Figure 2.4: Velocity Inlet = 250 m/s, Time = 0.011 s

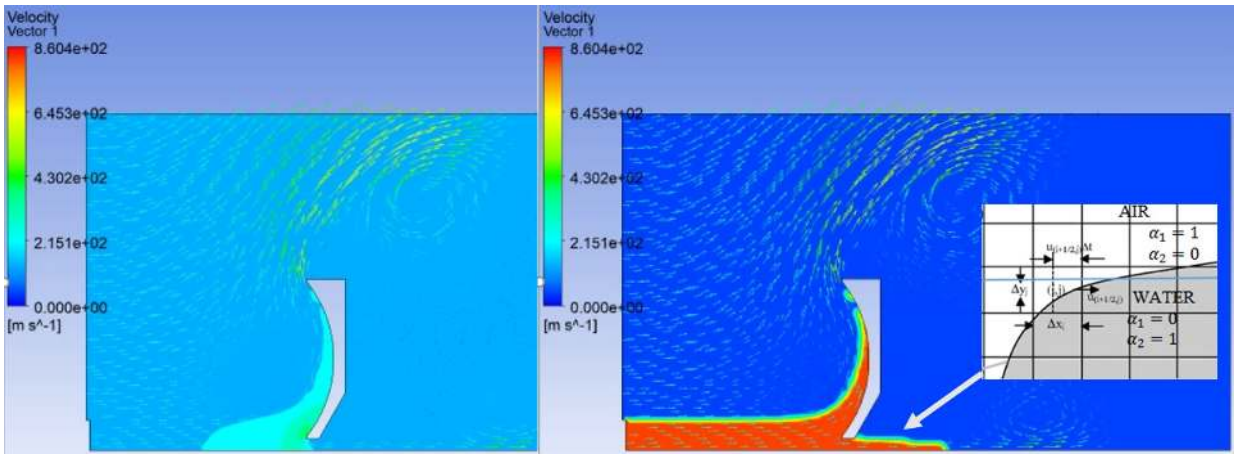


Figure 2.5: Velocity Inlet = 250 m/s, Time = 0.016 s.

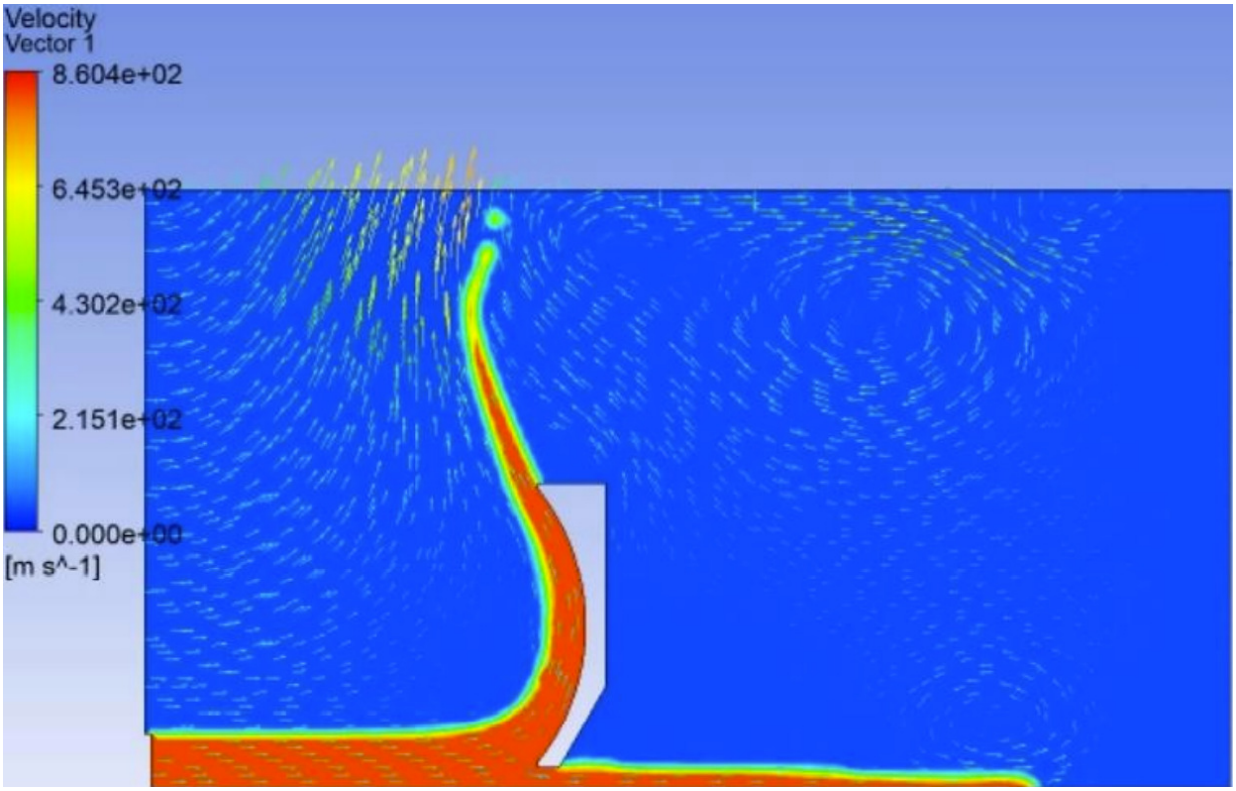


Figure 2.6: Velocity Inlet = 250 m/s, Time = 0.025 s

Table 2.2: Description of the seven measurements stored for each data sample in the CFD simulation dataset.

Symbol	Name	Unit/Range	Short Description
x	X-Coordinate	meters (m)	Spatial position in the x-axis
y	Y-Coordinate	meters (m)	Spatial position in the y-axis
t	Timestep	seconds (s)	The specific time point of the sample
p	Pressure	atm	Fluid pressure at that spatial position
u	u-velocity	m/s	Velocity component in the x-direction
v	v-velocity	m/s	Velocity component in the y-direction
$w.vf$	Water Vol. Fraction	0 - 1	Proportion of water in the volume of that point

Hereafter, data samples are referred to with the letter x, so for example x^p is the pressure of the sample x and x_t^p is the pressure of the sample x at a specific point in time t.

2.4 Proposed Hybrid Physics-Informed LSTM Model

We addressed fluid flow prediction with a new neural network architecture combining Physics-Informed Neural Networks (PINNs) and Long Short-Term Memory Networks (LSTMs) [23]. Specifically, we created a two-branch network architecture that takes as inputs the measurements taken from a Computational Fluid Dynamics (CFD) simulation and predicts what the velocity components (in 2D) will be for a given timestep.

2.4.1 Proposed Network Architecture

The proposed two-branch neural network used a range of time measurements as input, called the lookback, and output the predicted velocity components u and v for the next timestep at the same spatial position. The LSTM branch incorporated knowledge about the sequence and the Physics-Informed branch incorporated knowledge about the Partial Differential Equations (PDEs) describing incompressible 2D discrete Navier-Stokes fluid flows. The architecture is as follows, and is illustrated in Figure 2.7:

LSTM Branch

The LSTM branch had two stacked bi-directional LSTM layers with 32 activations each, followed by a TimeDistributed layer that used a fully connected layer of 32 activations with a linear activation function. The TimeDistributed layer applied the same weights to the LSTM outputs one timestep at a time. The LSTM layers were followed by a fully connected

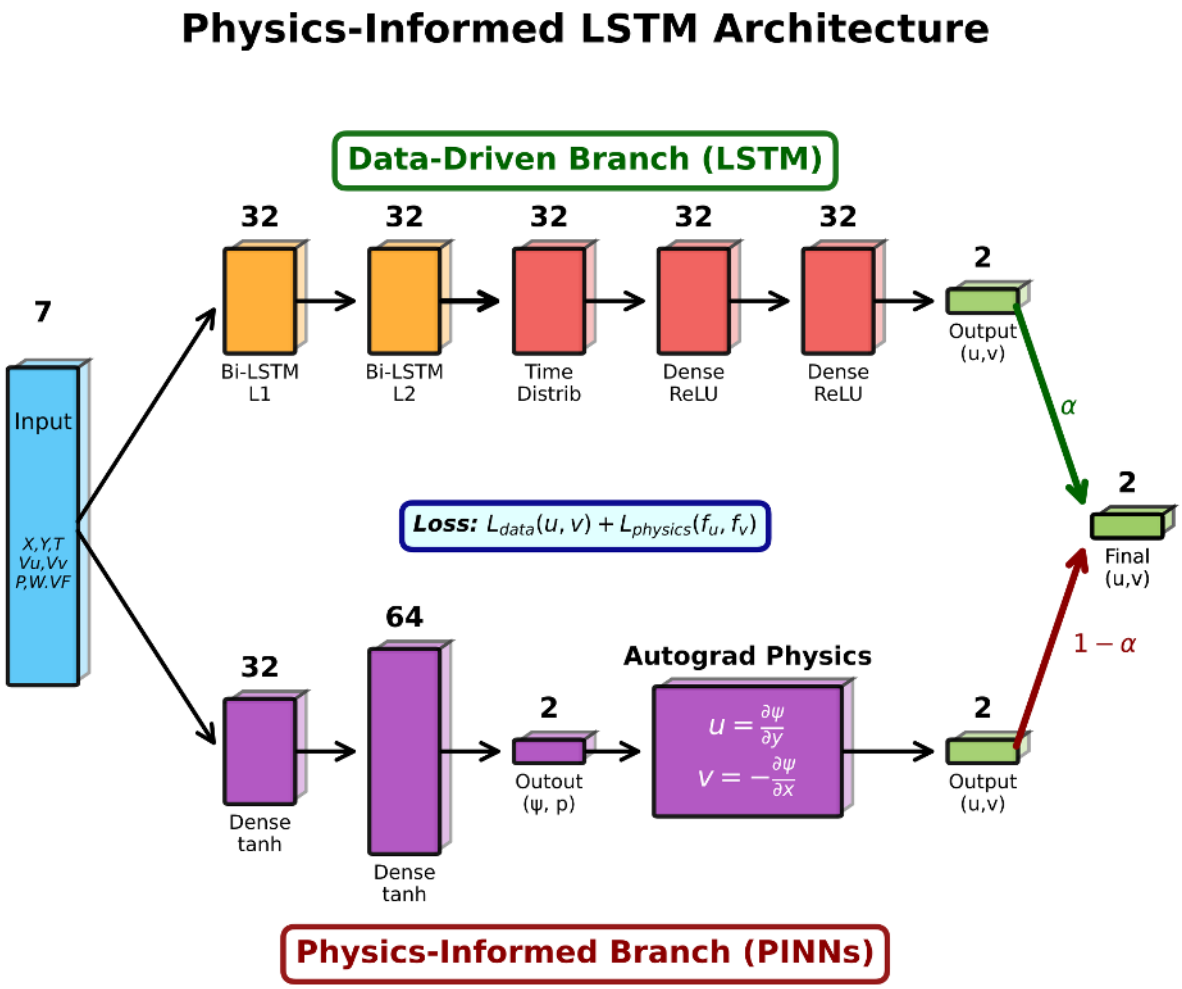


Figure 2.7: Physics-Informed LSTM Architecture showing both the LSTM and Physics-Informed branches and their connection to the final output prediction.

layer with 32 activations using the Rectified Linear Unit (ReLU) [30] non-linear activation function, defined as:

$$\text{ReLU}(x) = \max(0, x) \tag{2.6}$$

followed by a fully connected output layer. The output layer consisted of two outputs, the x-component of the velocity field, denoted as u_{lstm} and the y-component of the velocity field, denoted as v_{lstm}. The LSTM layers were connected to the fully connected part of the network by concatenating the bidirectional hidden states output by the final LSTM layer.

Physics-Informed Branch

As described in the Background section, the Physics-Informed branch enforces the 2D Navier-Stokes equations. We assumed that for some latent function $\psi(x, y, t)$

$$\begin{aligned} u &= \psi_y \\ v &= -\psi_x \end{aligned} \tag{2.7}$$

Then, we approximated the stream function $\psi(x, y, t)$ and pressure $p(x, y, t)$ using a dense neural network. This network takes three inputs (x, y, t) and has two outputs representing ψ and p. The velocity components u and v are subsequently computed from the analytical derivatives of ψ. The physics-informed loss also uses two parameters, λ_1 and λ_2, where λ_1 was fixed and λ_2 was made learnable.

$$\begin{aligned} f_u(x, y, t) &:= \frac{\partial u}{\partial t} + \frac{\partial p}{\partial x} + \lambda_1 \left(u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y} \right) - \lambda_2 \left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \right) \\ f_v(x, y, t) &:= \frac{\partial v}{\partial t} + \frac{\partial p}{\partial y} + \lambda_1 \left(u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y} \right) - \lambda_2 \left(\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2} \right) \end{aligned} \tag{2.8}$$

The combined physics loss we needed to minimize was then computed from the following four losses:

$$\begin{aligned} MSE_u &= \frac{1}{n}\sum_{i=1}^{n}(\psi_y - u^i)^2 \\ MSE_v &= \frac{1}{n}\sum_{i=1}^{n}(-\psi_x - v^i)^2 \\ MSE_{f_u} &= \frac{1}{n}\sum_{i=1}^{n}(f_u(x^i, y^i, t^i))^2 \\ MSE_{f_v} &= \frac{1}{n}\sum_{i=1}^{n}(f_v(x^i, y^i, t^i))^2 \end{aligned} \tag{2.9}$$

The components of the total physics loss are defined as follows:

MSE_u, MSE_v Consistency terms ensuring the predicted velocities (u, v) match the derivatives of the latent stream function (ψ).

MSE_{f_u}, MSE_{f_v} Physics residual terms penalizing violations of the Navier-Stokes momentum equations in the x and y directions.

The total loss was therefore:

$$MSE_{total} = MSE_u + MSE_v + MSE_{f_u} + MSE_{f_v} \tag{2.10}$$

The architecture consisted of a simple dense network with 3 layers. The first layer had 32 activations and used the hyperbolic tangent non-linear activation function (tanh), followed by another fully connected layer with 64 activations using tanh once again, followed by the output layer with two outputs. The automatic differentiation system in PyTorch, called `autograd`, was used to compute the partial derivatives required for the computation of the final physics loss.

Two-Branch Combined Model

The two outputs of each branch were combined by using a weight α that was set between 0 and 1. For our experiments, $\alpha = 0.5$ (but in the future it could be made a learnable parameter)

$$\begin{aligned} u_{pred} &= \alpha * u_{lstm} + (1 - \alpha) * u_{pinns} \\ v_{pred} &= \alpha * v_{lstm} + (1 - \alpha) * v_{pinns} \end{aligned} \tag{2.11}$$

where (u_{pred}, v_{pred}) were the final predictions, (u_{lstm}, v_{lstm}) were the two outputs of the LSTM branch, and (u_{pinns}, v_{pinns}) were the two outputs of the Physics-Informed branch.

2.4.2 Baseline Used for Comparison with Proposed Network

The baseline model assumes the velocities at the next timestep are the same as in the previous timestep.

$$\begin{aligned} prev_u &= x_t^u \\ prev_v &= x_t^v \\ y_{t+1}^{pred} &= (prev_u, prev_v) \end{aligned} \tag{2.12}$$

where t was the current timestep, x_t was the current data sample, x_t^u was the u-velocity component of x_t, x_t^v was the v-velocity component of x_t, and y_{t+1}^{pred} was our predicted velocity.

2.5 Experimental Results

Table 2.3 summarizes the results of the experiments and presents the mean squared error for the velocity components U and V for each model. All models were trained with a maximum of 200 epochs with early stopping used to find the best performing model based on validation loss.

Table 2.3: Comparison of the performance (MSE) of different models when using the 250 m/s simulation for training and the 300 m/s simulation for testing. This tests the model's ability to generalize to a different simulation condition.

Model	**U**	**V**
Baseline	5.455	2.561
PINNS Only	5,709	12,038
LSTM Only	1.7811	8.6962
PINNS+LSTM	**0.4677**	**1.2794**

2.6 Conclusions and Future Directions

2.6.1 Experimental Results Analysis

The experimental results demonstrate error reductions of **73.7%** in U-velocity and **85.3%** in V-velocity through the integration of physics constraints with temporal modeling. The Physics-Informed LSTM achieved a mean squared error of 0.4677 for the U-velocity component and 1.2794 for the V-velocity component, representing improvements of **73.7%** and **85.3%** respectively over both individual approaches. The baseline method, which assumes no change between timesteps, produced MSE values of 5.455 and 2.561 for U and V components respectively, establishing a lower bound for meaningful prediction performance.

Notably, the PINNs-only approach faced challenges capturing transient dynamics, with MSE values of 5,709 and 12,038. This indicates that physics constraints alone, without explicit temporal modeling, were insufficient for this time-dependent problem, as the dense network architecture struggled to capture the sequential evolution of flow patterns critical for accurate velocity prediction in transient CFD simulations. The LSTM-only approach showed intermediate performance with MSE values of 1.7811 and 8.6962, demonstrating that temporal modeling alone captured important sequential patterns but lacked physical consistency. The Physics-Informed LSTM successfully combined the strengths of both approaches, achieving approximately **73.7%** improvement in U-velocity prediction and **85.3%** improvement in V-velocity prediction compared to the LSTM-only approach.

2.6.2 Limitations and Future Work

Despite the promising results, several limitations of the current Physics-Informed LSTM approach warrant consideration for future research. The current implementation is restricted to two-dimensional incompressible flows, while many practical engineering applications require three-dimensional modeling of compressible fluids. Extending the architecture to 3D problems would significantly increase computational requirements and may require architectural modifications to handle the additional spatial dimensions.

The training process demonstrated sensitivity to hyperparameter selection, with several experiments showing overfitting tendencies that required careful regularization through dropout and early stopping. The fixed combination weight between LSTM and PINN branches ($\alpha = 0.5$) may not be optimal for all flow regimes or spatial regions, suggesting

the need for adaptive weighting mechanisms that can dynamically adjust based on local flow characteristics.

The current approach also faces challenges with high Reynolds number flows exhibiting turbulent behavior. The smooth nature of neural network solutions may struggle to capture the discontinuous phenomena and chaotic dynamics characteristic of turbulence, indicating a need for specialized architectures or hybrid approaches that can better handle multi-scale flow physics. This limitation is particularly pronounced in chaotic systems, where sensitive dependence on initial conditions leads to divergent trajectories. Future work should explore specialized architectures capable of capturing the long-term unpredictability and complex dynamics characteristic of turbulent flows.

Additionally, computational resource limitations prevented full convergence training for all models. The models were trained with a maximum of 200 epochs, but some configurations may have benefited from additional training epochs to achieve optimal performance. Future work with access to greater computational resources should explore extended training periods to ensure each model reaches its full convergence potential.

Future research directions should explore several promising avenues. Extension to three-dimensional problems represents a natural progression, requiring modifications to both the LSTM architecture for handling spatial sequences and the PINN formulation for 3D Navier-Stokes equations. Integration of attention mechanisms, which began gaining prominence in 2022, could enhance the model's ability to capture long-range dependencies and focus on physically relevant regions of the flow domain.

Transfer learning approaches could enable the model to adapt to different geometries and initial conditions without requiring complete retraining, addressing the current limitation of training on specific scoop configurations. Validation with experimental data from real-world water braking mechanisms would be essential to establish the practical applicability of the approach beyond simulation environments.

Finally, exploration of more advanced physics-informed architectures, including neural operators and Fourier-based methods, could provide improved computational efficiency and accuracy for large-scale problems. The integration of multiphase flow physics would expand the applicability to problems involving cavitation, boiling, or other complex fluid phenomena that were beyond the scope of the current single-phase implementation.

2.7 Chapter Contributions

This chapter implements and evaluates the first physics-guided strategy proposed in this thesis: **developing hybrid physics-informed network architectures that combine existing models (e.g., LSTMs) with Physics-Informed Neural Networks (PINNs) through two-branch models**.

This work makes several significant contributions to the field of physics-informed machine learning for fluid dynamics. First, it introduces a novel two-branch architecture that effectively combines temporal modeling capabilities of LSTMs with physical constraints from PINNs, representing one of the early demonstrations of sequential physics-informed networks for time-dependent CFD problems. Second, it provides comprehensive experimental validation on a practical engineering problem, demonstrating the feasibility of the

approach for real-world applications beyond academic benchmark problems.

Third, the systematic development methodology and extensive experimentation provide valuable insights into the design principles for physics-informed recurrent networks, including the importance of proper regularization, hyperparameter selection, and loss component balancing. Finally, this work establishes a foundation for future research in hybrid physics-informed architectures, particularly for applications where both temporal evolution and physical consistency are essential requirements.

The complete implementation and experimental configurations are publicly available at `https://github.com/DeveloperJose/Python-Physics-LSTM` to support reproducibility and enable further research development.

Chapter 3

Physics-Informed Data Augmentation for Glacier Segmentation

3.1 Introduction

Glacier segmentation is the process of identifying and delineating different types of ice and debris in satellite imagery. This task is crucial for monitoring the health of glaciers, such as those in the Hindu Kush Himalayas (HKH), which are a vital source of fresh water for a significant portion of the world's population. Accurate segmentation allows us to track changes in glacier mass and extent over time, providing critical data for climate modeling and water resource management.

The boundary-aware self-learning U-Net framework by Aryal et al. [25] established a strong baseline for glacier segmentation. Its primary strength lies in optimizing geometric boundary delineation by learning from spatial patterns in multispectral imagery. However, this reliance on spectral data presents a key limitation: when debris obscures glacial ice, the spectral signatures of ice and rock become ambiguous, making boundaries difficult to identify. Furthermore, a purely image-based approach lacks any intrinsic understanding of the physical laws, such as gravity and fluid dynamics, that govern glacier formation and shape their boundaries.

To address this, we introduce a physics-informed data augmentation strategy. Instead of relying only on spectral data, we encode physical principles from elevation and other spectral data directly into the model's input. This guides the network with physical constraints and ultimately produces segmentations that are not only geometrically precise but also physically plausible.

3.2 Background

3.2.1 Image Semantic Segmentation: Grouping Similar Pixels Together

The task of semantic segmentation refers to grouping similar or related pixels of an image together, like those of a specific object. For example, grouping all the pixels of people, roads, buildings, or trees as shown in Figure 3.1.

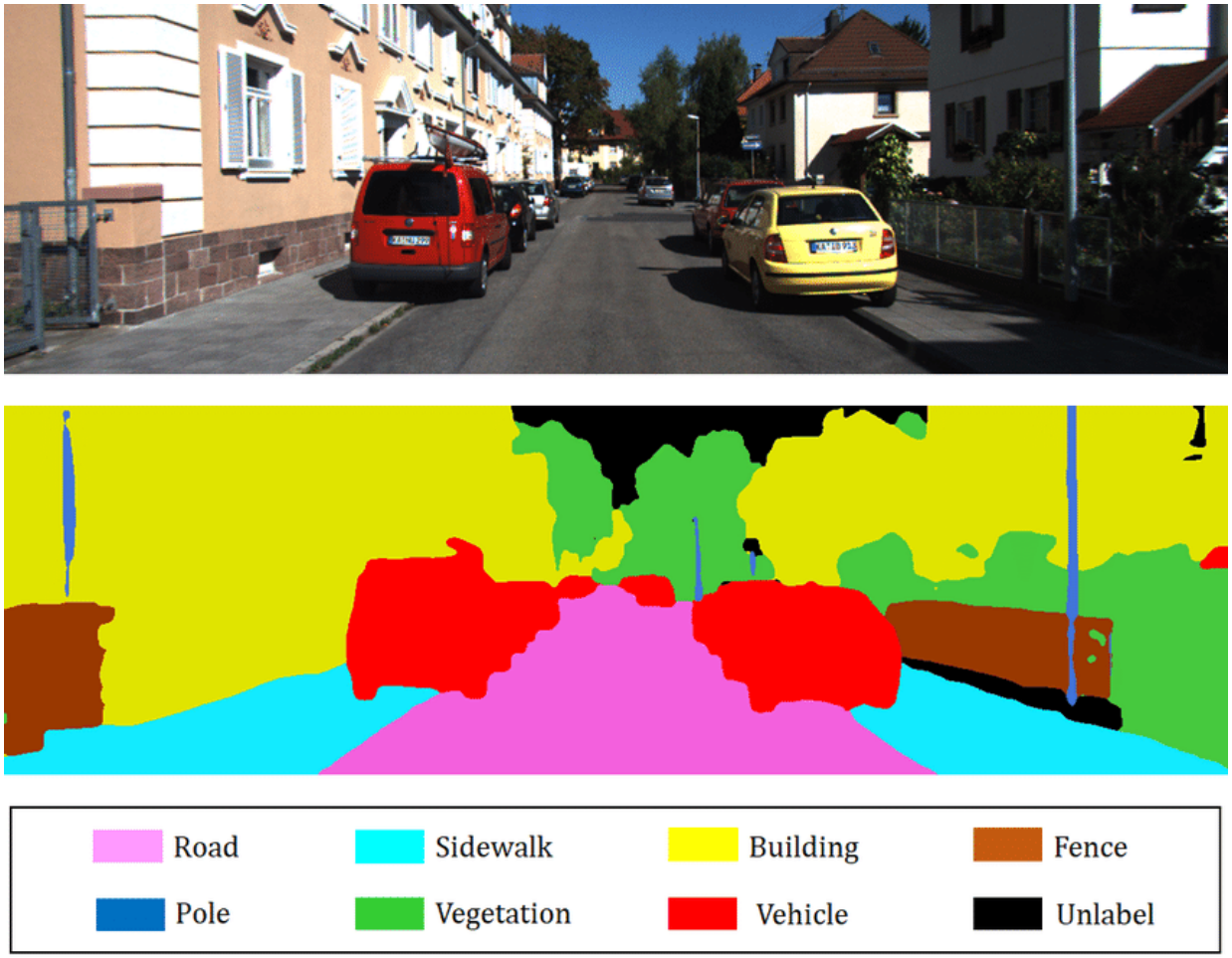


Figure 3.1: An example of semantic segmentation of an image. The pixels are grouped together by different categories. Source: `https://towardsai.net/p/l/machine-learning-7`

3.2.2 Glacier Mapping Through Segmentation of Ice in Multispectral Images

Glacial ice segmentation or glacier mapping is simply the task of semantic segmentation applied to multispectral satellite images of glaciers. The goal is to determine for each pixel in the image whether the pixel is an area of glacial ice, debris-covered ice, or background (regular rocks) as shown in the figure below:

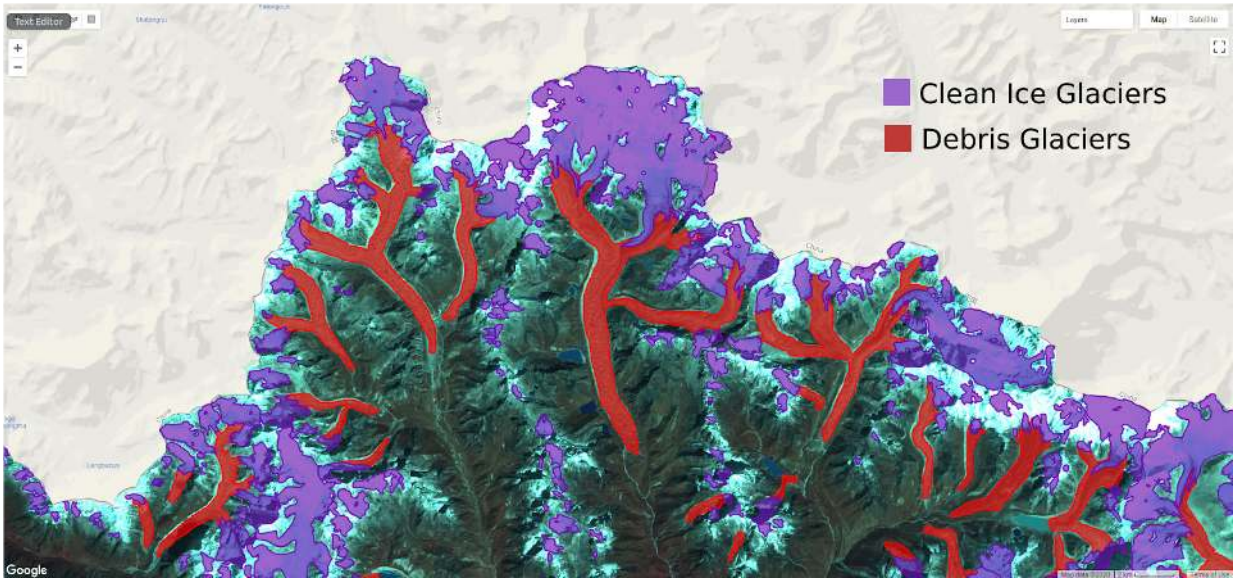


Figure 3.2: Semantic segmentation of a glacier satellite image by a human annotator. Source: [25]

3.2.3 Convolutional Neural Networks: The Foundation of Image Analysis

Convolutional Neural Networks (CNNs) are the cornerstone of modern computer vision [31]. They are designed to automatically learn spatial hierarchies of features from images. A CNN consists of several key components: convolutional layers, which apply filters (or kernels) to extract features like edges and textures. The convolution operation for a 2D image I and a kernel K is defined as:

$$(I * K)(i, j) = \sum_{m} \sum_{n} I(i - m, j - n)K(m, n) \tag{3.1}$$

Pooling layers, which downsample the feature maps to reduce computational complexity. For example, max pooling takes the maximum value in a local neighborhood:

$$\text{MaxPooling}(A)_{i,j} = \max_{(m,n) \in R_{i,j}} A_{m,n} \tag{3.2}$$

where $R_{i,j}$ is a local neighborhood around the location (i, j). And fully connected layers, which perform classification based on the learned features. This architecture, illustrated in Figure 3.3, has proven to be highly effective for a wide range of image-related tasks, from classification to segmentation.

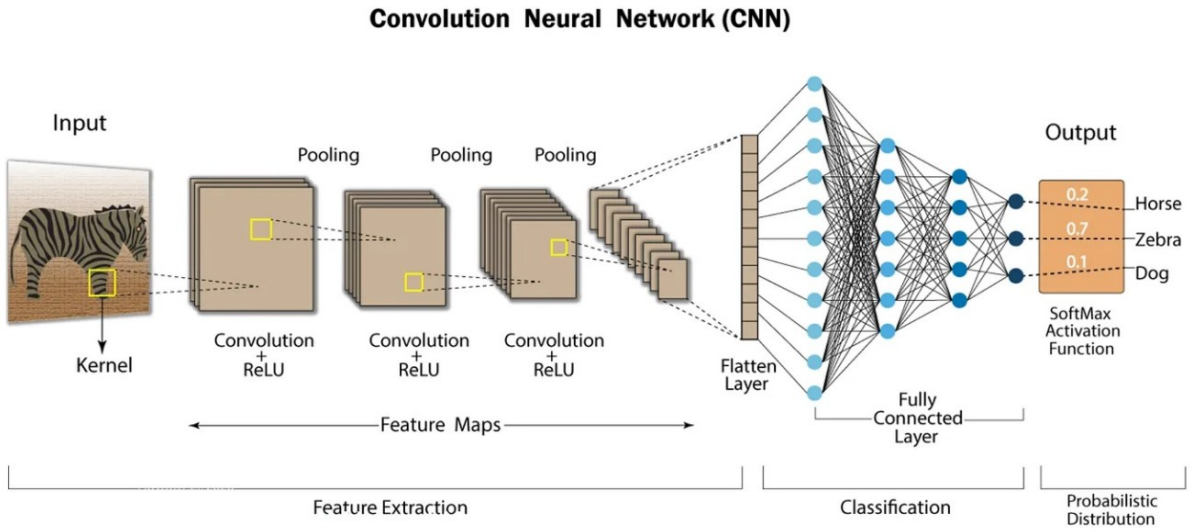


Figure 3.3: An example of a CNN model for an animal classification problem. Features are extracted with convolutions and pooling, and then used in a fully connected network for a final classification.

3.2.4 U-Net: The Standard for Image Segmentation

The U-Net architecture, shown in Figure 3.4, is a specialized CNN designed for semantic segmentation [32]. It consists of a contracting path (encoder) that captures context and a symmetric expanding path (decoder) that enables precise localization. The encoder follows a typical CNN architecture, with convolutional and pooling layers that downsample the input image and extract features. The decoder upsamples the feature maps and combines them with high-resolution features from the contracting path via skip connections. This architecture allows the network to make predictions for each pixel in the input image, making it the standard for segmentation tasks.

3.2.5 Baseline State-of-Art Network

The baseline model used to evaluate our proposed data augmentation technique was the Boundary-Aware U-Net for Glacier Segmentation model developed and published in 2023 by Aryal et al. [25]. Its architecture is shown in Figure 3.5. No modifications to the model were made except for the addition of the physics-informed data augmentation technique.

The baseline model builds upon the standard U-Net architecture with several key modifications designed specifically for glacier segmentation:

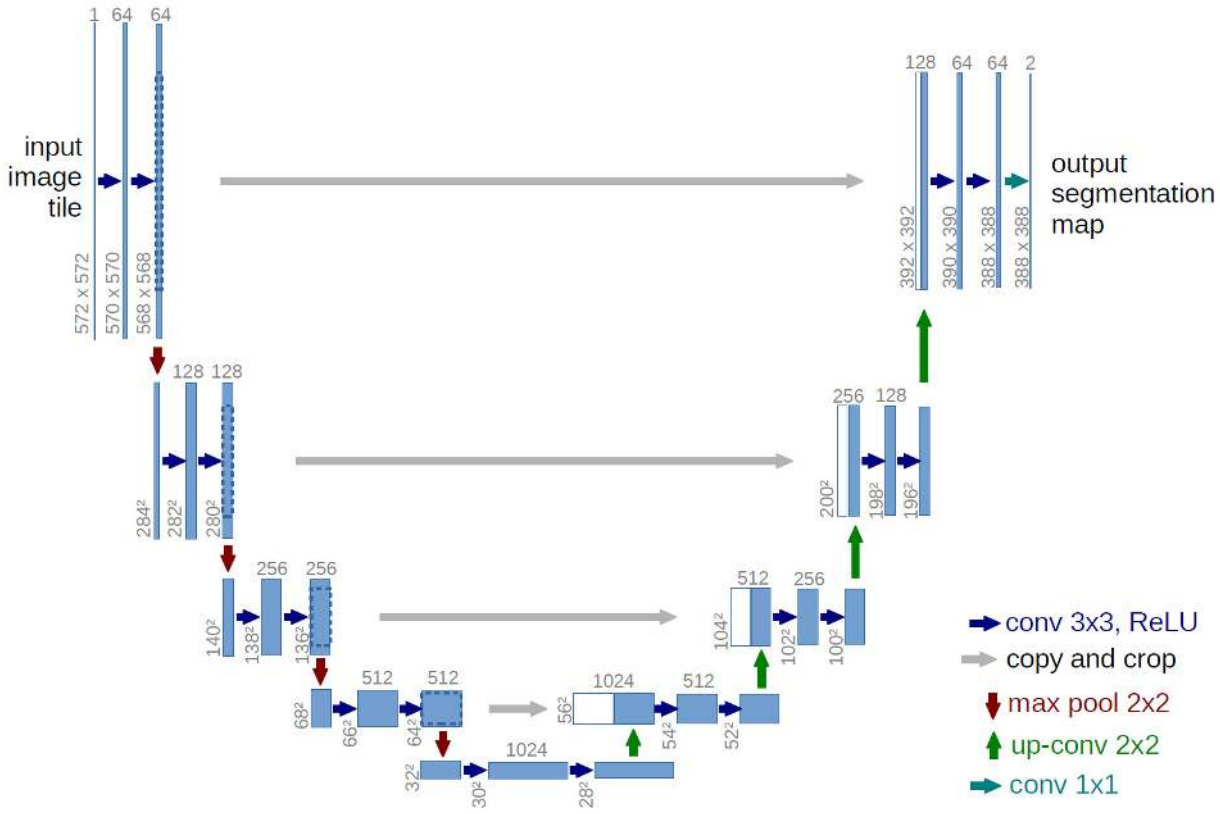


Figure 3.4: Architecture of a U-Net. Source: [25]

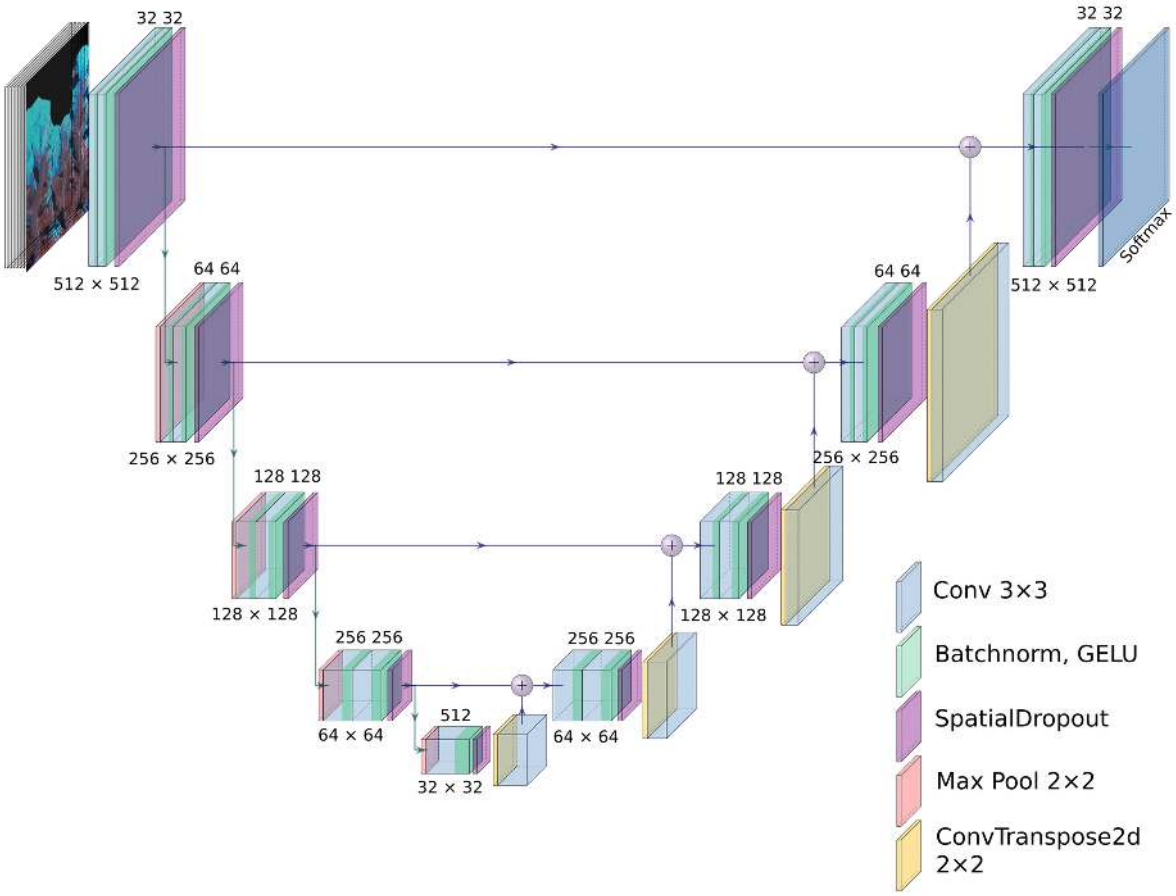


Figure 3.5: Architecture for State-of-Art Network. Source: [25]

Self-Learning Boundary Optimization

The model incorporates a specialized loss function that emphasizes accurate boundary delineation between clean ice and debris-covered ice regions. A common loss function for segmentation tasks is the Dice Loss, which is a measure of overlap between two samples [33]:

$$\text{DiceLoss} = 1 - \frac{2|A \cap B|}{|A| + |B|} \tag{3.3}$$

where A and B are the predicted and ground truth segmentation masks, respectively.

The Boundary loss is a distance metric based on the boundaries of the segmentation masks [34]. It is defined as:

$$L_{Boundary} = \int_{\partial G} \phi_G(p) dp \tag{3.4}$$

where ∂G is the boundary of the ground truth segmentation and $\phi_G(p)$ is a distance map computed from the predicted segmentation.

It attempted to balance these two components using a learnable uncertainty weighting scheme based on the following equation:

$$L_{Total} = \frac{1}{2\sigma_1^2} L_{Dice} + \frac{1}{2\sigma_2^2} L_{Boundary} + |\log(\sigma_1 \sigma_2)| \tag{3.5}$$

3.3 Methodology

3.3.1 Problem Formulation

We formulate glacier mapping as a **semantic segmentation problem**. The objective is to learn a function that assigns a class label to every pixel in a multispectral satellite image. The inputs and outputs are detailed in Table 3.2.

In simpler terms, we are training a deep learning model to look at satellite imagery and elevation maps, just as a glaciologist would, to identify which parts of the image correspond to clean ice, which parts are ice hidden underneath rocky debris, and which parts are just regular background terrain. The goal is to automate this identification process with high accuracy, even in challenging terrain.

3.3.2 Proposed Physics-Informed Data Augmentation

Our proposed method, illustrated in Figure 3.6, extracts four physics-informed features from the Digital Elevation Model (DEM) and concatenates them with the original satellite imagery. This enriches the input data with topographical information that is directly relevant to glacier formation and flow.

Topographic Flow Accumulation

Our primary physics-informed feature is the Topographic Flow Accumulation. This method encodes the physical landscape into the neural network by transforming raw elevation data

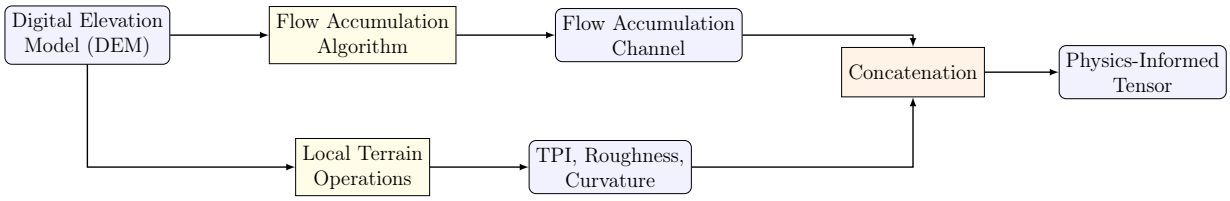


Figure 3.6: Workflow for extracting physics-informed features from the Digital Elevation Model (DEM).

into a feature that represents water and ice flow. Rather than running complex simulations, we use a graph traversal algorithm to calculate an abstract representation of potential flow accumulation. This derived channel explicitly highlights the natural downhill paths and valley structures where glaciers form, as shown in Figure 3.7.

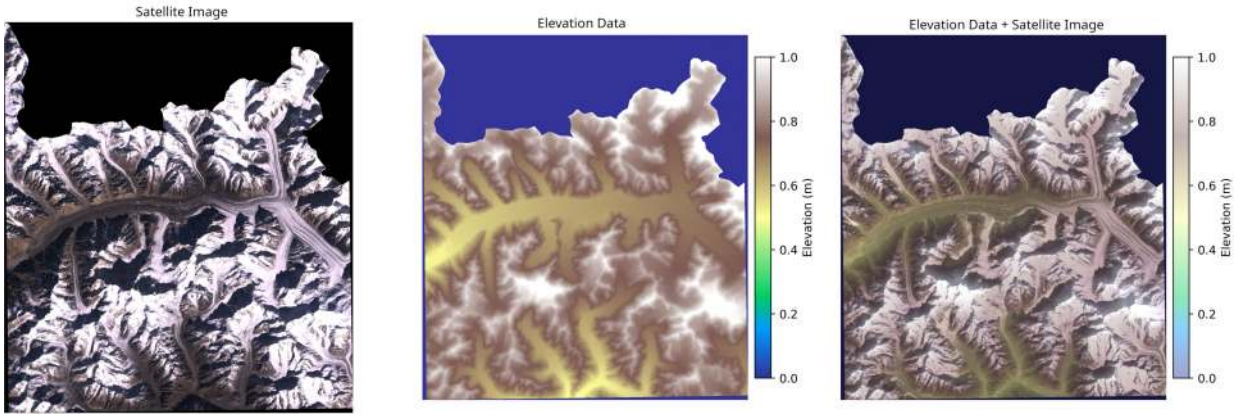


Figure 3.7: Elevation example showing the terrain relief.

Specifically, the algorithm for the flow accumulation feature extraction is as follows, and an example of its application is shown in Figure 3.8:

1. Take the elevation map as the input image *elevation*
2. Create an empty accumulation map of the same size as the input called *output*
3. For each pixel in the image, we perform topographic flow accumulation. A constraint is applied so that only unprocessed pixels at a lower elevation than the current pixel can receive flow, ensuring water and ice follow natural downhill paths.
4. Each time a pixel is visited, the *output* at that pixel's position increases by 1 as 1 unit of water/ice volume reaches that location.
5. Finally, the raw accumulation map is normalized as a separate preprocessing step before being fed into the neural network. This is a critical step for stable and effective training.

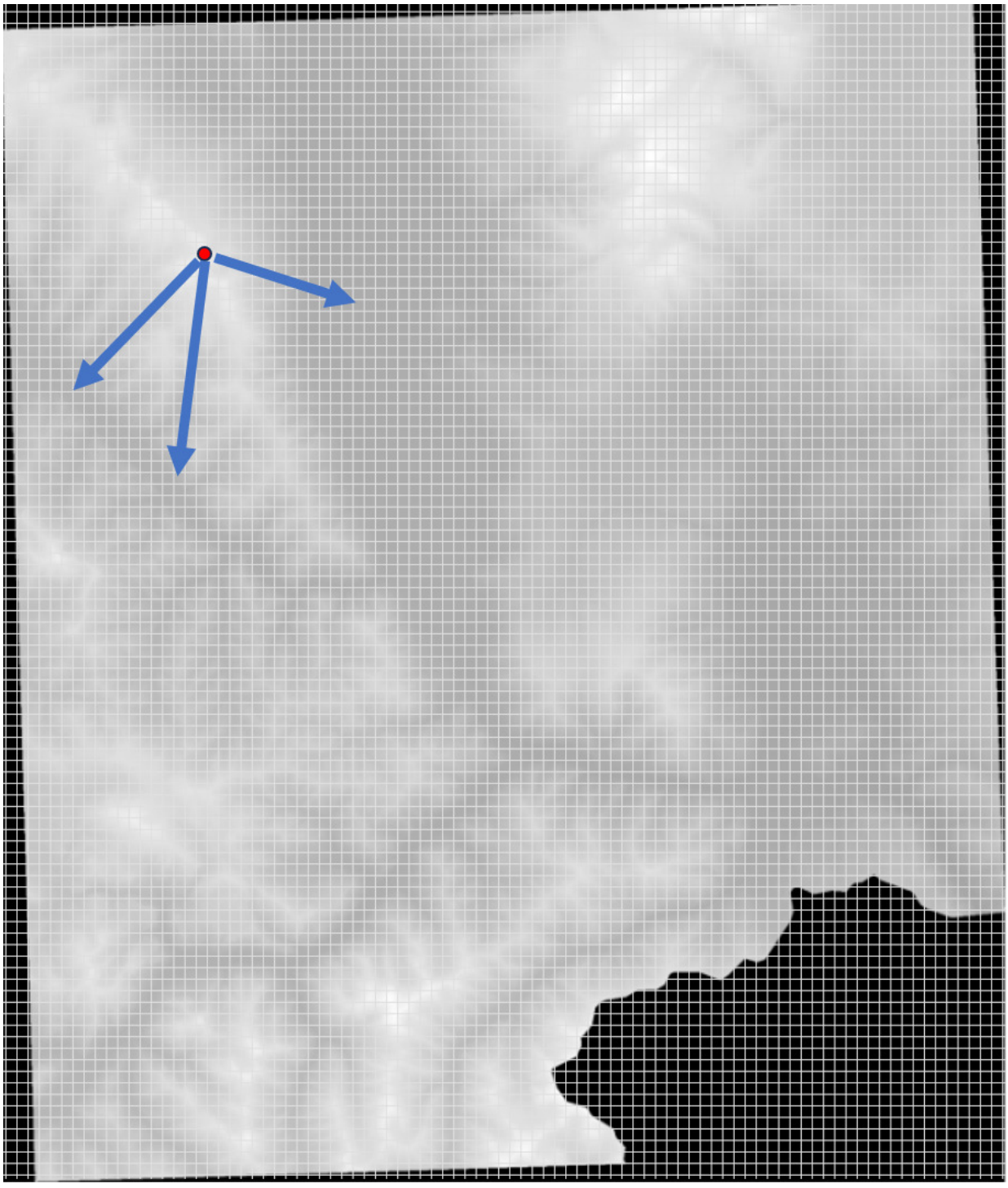


Figure 3.8: For each pixel in the image we performed topographic flow accumulation to simulate water and ice movement across the glacier terrain, then encoded such information as an additional physics-informed channel.

The detailed topographic flow accumulation algorithm:

Algorithm 1 Topographic Flow Accumulation Process

```
im ← elevation map from satellite image
output ← previous accumulated output from other pixels
u ← row of source pixel
v ← column of source pixel
source ← (u, v)
processed = {source}
Q = queue with source enqueued
while Q ≠ Empty do
    x = Q.dequeue()
    curr_elevation ← im[x]
    if x ≠ source then
        output[x] += 1
    end if
    for n in get_neighbors(im, x) do
        n_elevation ← im[n]
        if n not processed & n_elevation < curr_elevation then
            Add n to processed
            Q.enqueue(n)
        end if
    end for
end while
```

Figure 3.9 presents an example of the results after performing the augmentation.

Another example of the topographic flow accumulation result is shown in Figure 3.10.

While the high-level logic is presented here, the implementation is optimized using techniques like downscaling and subsampling to ensure computational tractability. The method simulates global flow dynamics by treating the elevation map as a directed graph where edges connect pixels to their lower-elevation neighbors.

Local Terrain Features

Complementing the global flow accumulation, we incorporate three standard local terrain features computed using sliding window operations (convolutional kernels) on the elevation map. These features capture local surface geometry and texture, and are summarized in Table 3.1.

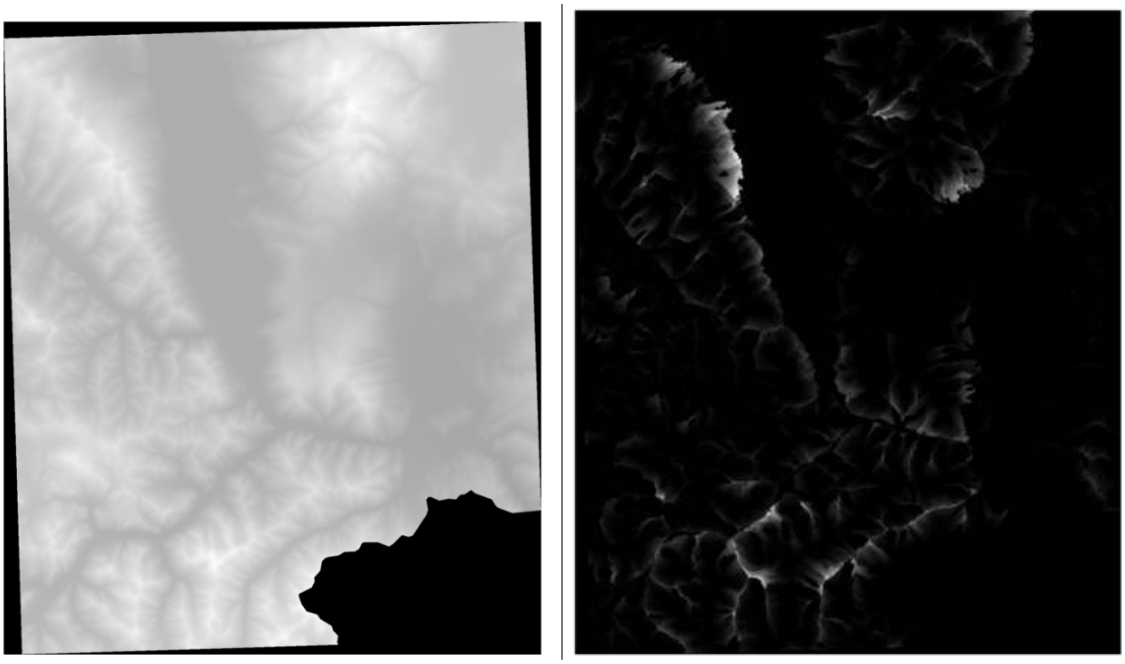


Figure 3.9: Left is the elevation map from one of the glacier satellite images where white pixels represent peaks and darker pixels represent valleys. Right is the resulting topographic flow accumulation channel.

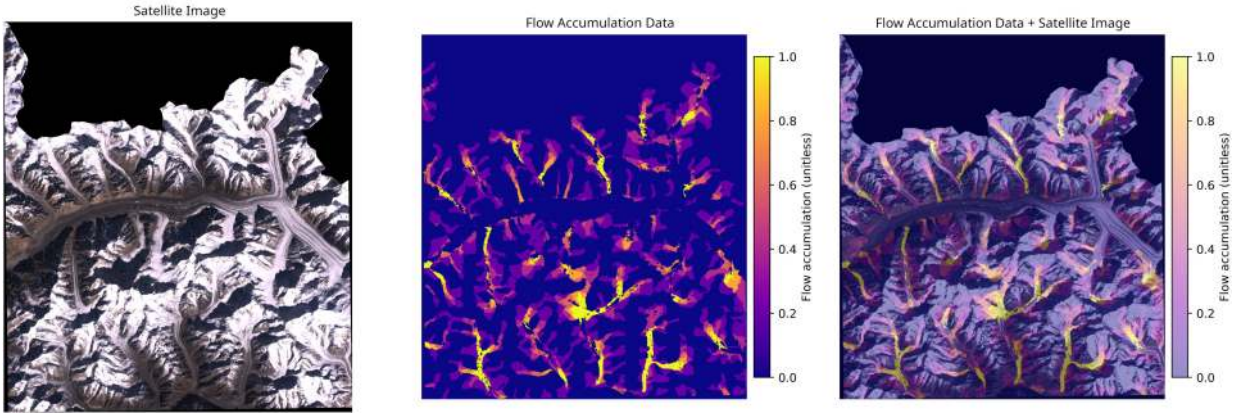


Figure 3.10: Another example of the topographic flow accumulation result.

Table 3.1: Summary of local terrain features derived from the Digital Elevation Model.

Feature	**Calculation Basis**	**Physical Significance**
TPI	Elevation diff. from local mean	Identifies valleys (neg.) and ridges (pos.)
Roughness	Std. dev. of local elevation	Quantifies terrain irregularity/texture
Plan Curvature	Second derivatives of elevation	Measures surface convexity/concavity perpendicular to slope

The mathematical formulations for these features are as follows:

- **Topographic Position Index (TPI):** Calculated as the difference between a pixel's elevation and the average elevation of its surrounding area [35]. This feature highlights valleys and ridges and is computed using a uniform box filter, as shown in Figure 3.11.

$$TPI = Z_0 - \frac{1}{N}\sum_{i \in W} Z_i \tag{3.6}$$

  where Z_0 is the elevation of the central pixel and W is a local window of N pixels.

- **Surface Roughness:** Defined as the standard deviation of elevation within a local neighborhood, quantifying the terrain's irregularity [36]. This is computed using a sliding window operation, as shown in Figure 3.12.

$$Roughness = \sqrt{\frac{1}{N-1}\sum_{i \in W}(Z_i - \bar{Z})^2} \tag{3.7}$$

  where $\bar{Z}$ is the mean elevation in the window W.

- **Plan Curvature:** Measures the curvature of the surface perpendicular to the direction of the steepest slope [37]. It is calculated from the second derivatives of the elevation map Z using finite difference kernels (e.g., Sobel or central differences), as shown in Figure 3.13.

$$Curvature = \frac{Z_{xx}Z_y^2 - 2Z_{xy}Z_xZ_y + Z_{yy}Z_x^2}{(Z_x^2 + Z_y^2)^{3/2}} \tag{3.8}$$

  where Z_x and Z_y are the first partial derivatives (gradient) and Z_{xx}, Z_{yy}, and Z_{xy} are the second partial derivatives.

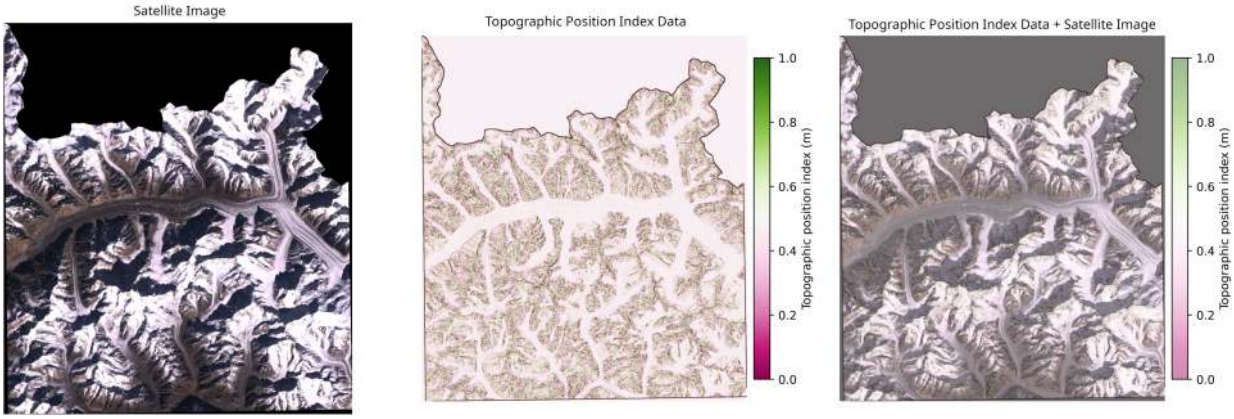


Figure 3.11: Example of Topographic Position Index (TPI) feature.

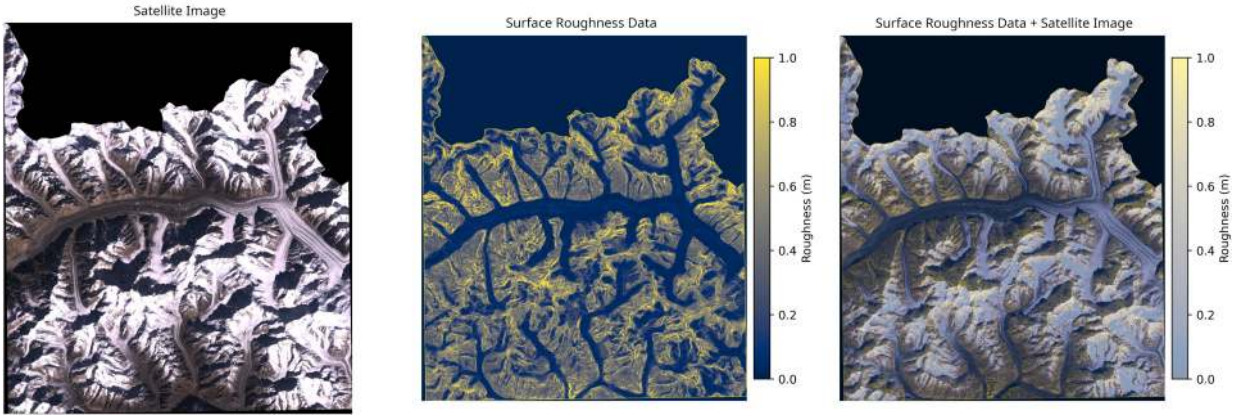


Figure 3.12: Example of Surface Roughness feature.

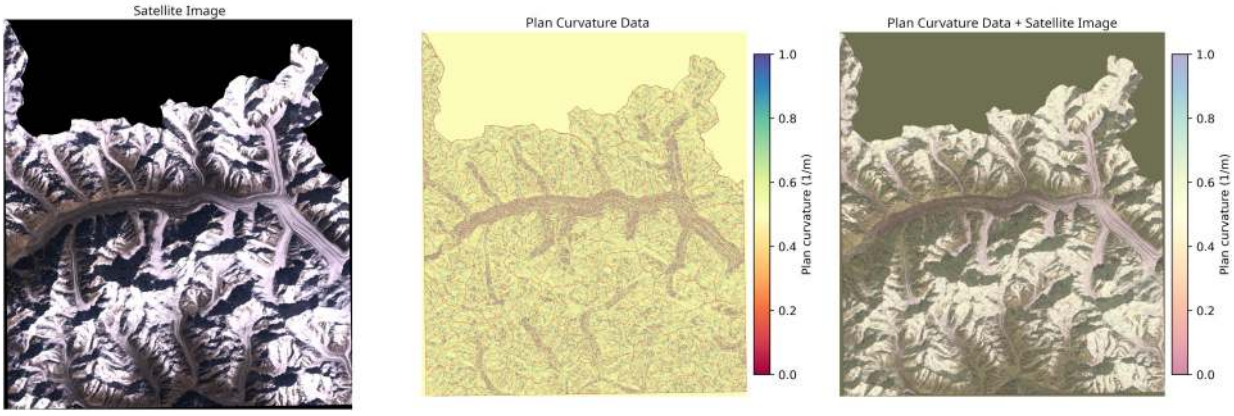


Figure 3.13: Example of Plan Curvature feature.

3.4 Experimental Design

3.4.1 The HKH Glacier Dataset

The dataset used in this study is a comprehensive collection of glaciological data covering the Hindu Kush Himalayas (HKH) region from 2002 to 2008. It integrates three primary data sources to provide a rich foundation for training and evaluating our segmentation models.

The first key component is the multispectral satellite imagery from NASA's Landsat 7 satellite, an example of which is shown in Figure 3.14. This includes 7-band images (B1-B7) captured by the Enhanced Thematic Mapper Plus (ETM+) sensor, spanning the visible, near-infrared, and shortwave infrared spectral ranges [38]. Each band provides unique information about the earth's surface properties. For instance, the visible bands (Blue, Green, Red) capture standard optical features, while the Near-Infrared (NIR) and Shortwave Infrared (SWIR) bands are particularly effective for distinguishing between snow, ice, clouds, and vegetation-a critical capability for mapping debris-covered glaciers. Additionally, the Thermal Infrared band offers insights into surface temperature, which can help differentiate between supraglacial debris and the surrounding bedrock. These images have been pre-processed to handle the Scan Line Corrector (SLC) failure, ensuring data quality and consistency. The second source is a high-resolution Digital Elevation Model (DEM), which provides detailed elevation and slope information. This topographical data is not only used as a direct input to the model but also serves as the foundation for computing our physics-informed features. The final component is the set of expert-delineated glacier boundaries from the International Centre for Integrated Mountain Development (ICIMOD) [26]. These ground truth labels, created by glaciology experts, provide the basis for our three-class classification system: **Background (0)** for non-glacier terrain, **Clean Ice (1)** for pure glacial ice, and **Debris-Covered Ice (2)** for ice mixed with rocks and sediment. The specific input features derived from these sources are summarized in Table 3.2.

Table 3.2: Segmentation problem formulation inputs and outputs.

Role	Component	Unit/Range	Description
Input (X)	Band 1	$0.45 - 0.52\mu m$	Blue (Visible). Useful for mapping water bodies and distinguishing soil from vegetation.
	Band 2	$0.52 - 0.60\mu m$	Green (Visible). Sensitive to vegetation vigor. Peak reflectance for vegetation.
	Band 3	$0.63 - 0.69\mu m$	Red (Visible). Discriminates vegetation slopes.
	Band 4	$0.77 - 0.90\mu m$	Near-Infrared (NIR). Emphasizes biomass content and shorelines.
	Band 5	$1.55 - 1.75\mu m$	Shortwave Infrared (SWIR). Discriminates moisture content of soil and vegetation. Penetrates thin clouds.
	Band 6	$10.40 - 12.50\mu m$	Thermal Infrared. Measures surface temperature. Useful for thermal mapping.
	Band 7	$2.09 - 2.35\mu m$	Shortwave Infrared (SWIR). Improves discrimination of mineral and rock types.
	Elevation	meters	Terrain Elevation from the Digital Elevation Map (DEM)
	Slope	degrees	Derived terrain steepness from DEM.
Output (Y)	Class 0	Background	Rocks, vegetation, water.
	Class 1	Clean Ice	Pure glacier ice.
	Class 2	Debris Ice	Ice mixed with debris (rocks or sediment).

To prepare this data for our models, we employ a multi-stage preprocessing pipeline. Large satellite images are first partitioned into 512×512 pixel patches with a 64-pixel overlap to ensure complete coverage. Patches with less than 10% glacier pixels are filtered out to focus computational resources on relevant areas. We then perform feature engineering to compute additional spectral indices (NDVI, NDWI, NDSI) [39–41], color space features

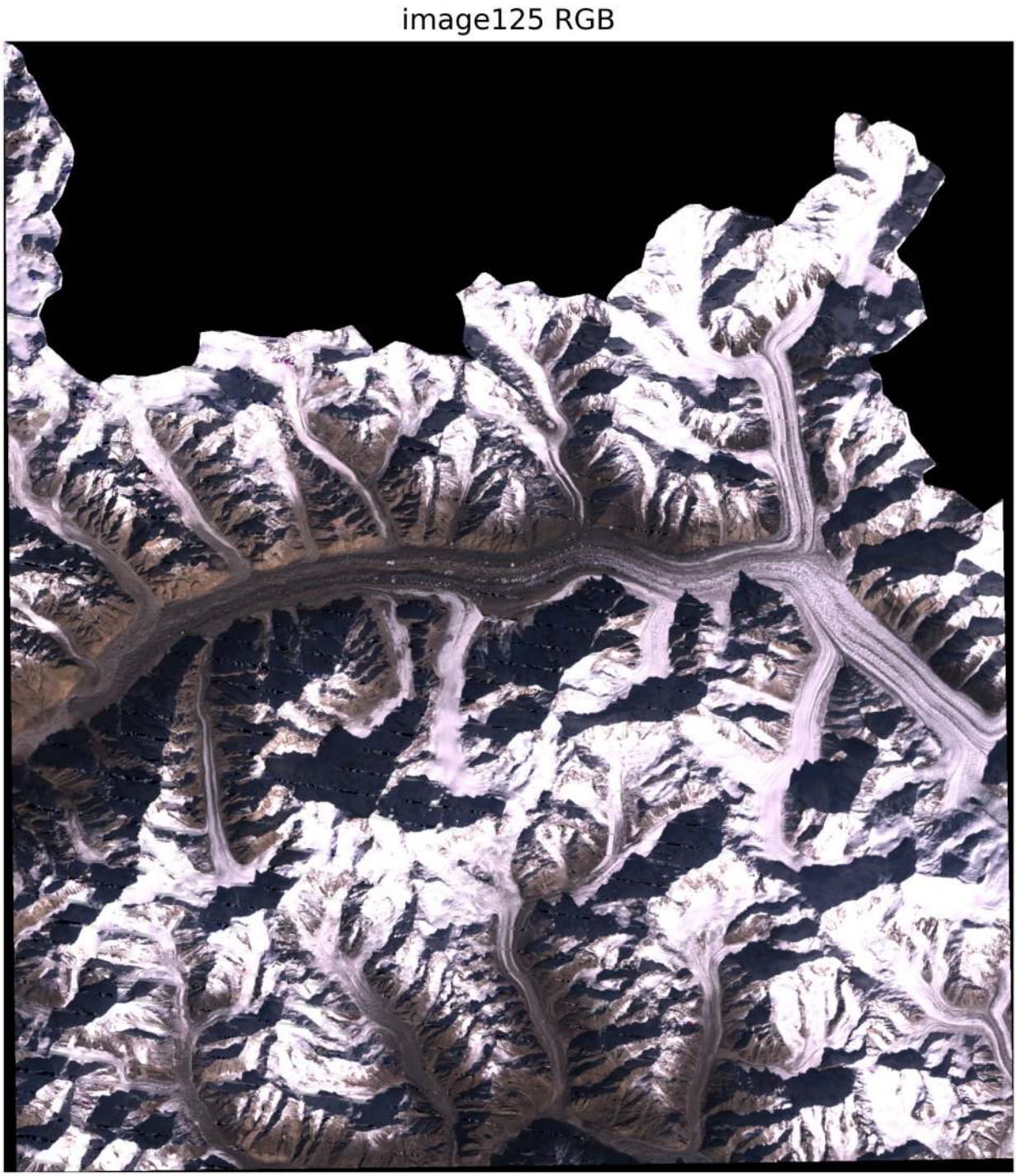


Figure 3.14: Sample RGB composite of a glacier from the HKH dataset.

(HSV), and other features derived from the DEM data. The final processed dataset, with its distribution across training, validation, and test splits, is organized using stratified random sampling to ensure a representative distribution of glacier coverage across all sets, as detailed in Table 3.3. This is the same preprocessing pipeline presented in Aryal et al. [25] but includes the new physics-informed data augmentation component.

Table 3.3: HKH glacier patch distribution after preprocessing (kept patches only).

Split	**Number of Patches**	**Percentage**
Training	529	66.8%
Validation	102	12.9%
Testing	161	20.3%
Total	**792**	**100%**

3.4.2 Dataset Statistics and Class Imbalance

The curated HKH dataset remains highly imbalanced even after patch filtering. For the retained splits, debris-covered ice constitutes only about 1.7–2.8% of valid pixels (train/val/test), while background dominates (∼71–74%) and clean ice represents ∼23–26%. Approximately 77% of candidate patches (2,741 out of 3,533 processed) were skipped during preprocessing due to low glacier content. The physics-informed inputs and losses were introduced to help counter this imbalance.

3.4.3 Measuring Network Performance

To comprehensively evaluate the performance of our segmentation models, we employ several standard metrics. While Intersection over Union (IoU) serves as our primary metric for overall segmentation accuracy, we also utilize Precision and Recall to provide a detailed understanding of model behavior.

Intersection over Union (IoU)

The Intersection over Union (IoU), also known as the Jaccard Index, measures the overlap between the predicted segmentation mask and the ground truth mask. It is defined as the ratio of the area of intersection to the area of union between the predicted and ground truth sets, as illustrated in Figure 3.15.

$$IoU = \frac{|A \cap B|}{|A \cup B|} = \frac{TP}{TP + FP + FN} \tag{3.9}$$

where A is the set of ground truth pixels, B is the set of predicted pixels, TP (True Positives) are correctly classified pixels, FP (False Positives) are background pixels incorrectly classified as the target class, and FN (False Negatives) are target pixels incorrectly classified as background. IoU ranges from 0 to 1, with 1 indicating a perfect match.

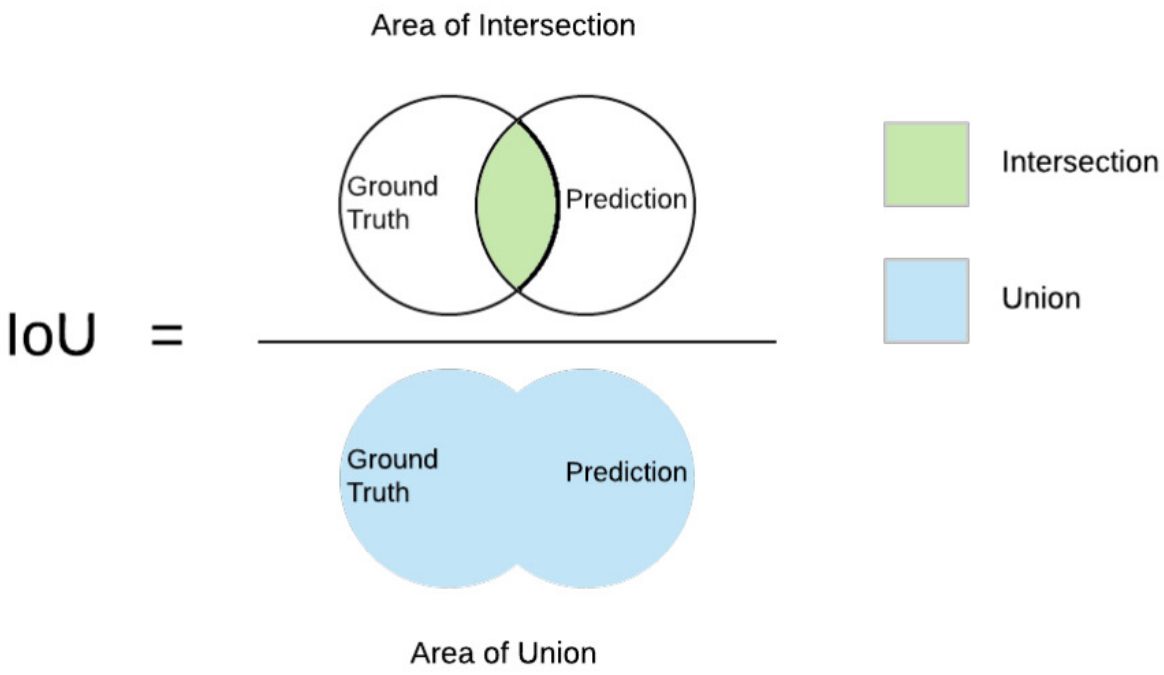


Figure 3.15: Definition of Intersection over Union.

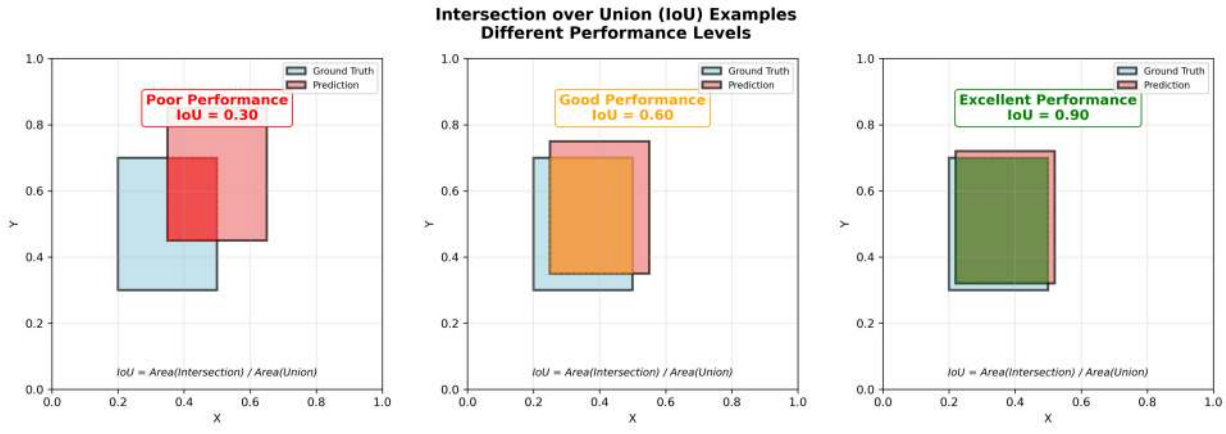


Figure 3.16: Three examples of different IoU values and what type of performance they represent.

Precision and Recall

Precision and Recall provide complementary insights into model errors.

Precision measures the accuracy of positive predictions: of all the pixels the model predicted as glacier ice, what fraction were actually glacier ice? High precision indicates a low false positive rate.

$$Precision = \frac{TP}{TP + FP} \tag{3.10}$$

Recall (or Sensitivity) measures the completeness of positive predictions: of all the actual glacier ice pixels, what fraction did the model correctly identify? High recall indicates a low false negative rate.

$$Recall = \frac{TP}{TP + FN} \tag{3.11}$$

3.5 Results and Discussion

3.5.1 Experimental Setup

We evaluate our proposed physics-informed data augmentation against two benchmarks to validate its effectiveness. The **Standard U-Net** [32] serves as a baseline, representing a traditional deep learning approach without domain-specific modifications. The **Boundary-Aware U-Net** [25] serves as the state-of-the-art (SOTA) benchmark for this dataset, incorporating a specialized loss function for boundary optimization.

Our proposed method extends the SOTA Boundary-Aware architecture by augmenting the input space with physics-derived terrain features. We evaluate two configurations:

- **Ours (Flow Only):** Adds only the Topographic Flow Accumulation channel to test the impact of global flow dynamics.
- **Ours (Full Physics):** Integrates all four physics channels (Flow Accumulation, TPI, Roughness, Curvature) to capture both global flow paths and local surface geometry.

3.5.2 Quantitative Analysis

Table 3.4 presents the comparative performance of all models. The results demonstrate that incorporating physics-informed features yields an IoU increase of **9.98 percentage points (27.8% relative improvement)** for Debris-Covered Ice. The loss plot for the best Clean Ice Physics model is shown in Figure 3.17.

Table 3.4: Comparative performance of glacier segmentation models. All values are in percentages (%). Best results are highlighted in bold.

	Debris-Covered Ice			Clean Ice		
Model	**IoU**	**Prec.**	**Rec.**	**IoU**	**Prec.**	**Rec.**
Standard U-Net	28.50	-	-	65.60	-	-
Boundary-Aware (SOTA) [25]	35.94	51.97	53.81	68.17	81.59	80.55
Ours (Flow Only)	38.50	58.90	52.60	63.50	78.90	76.40
Ours (Full Physics)	**45.92**	**71.89**	**55.96**	**71.22**	**85.39**	**81.10**

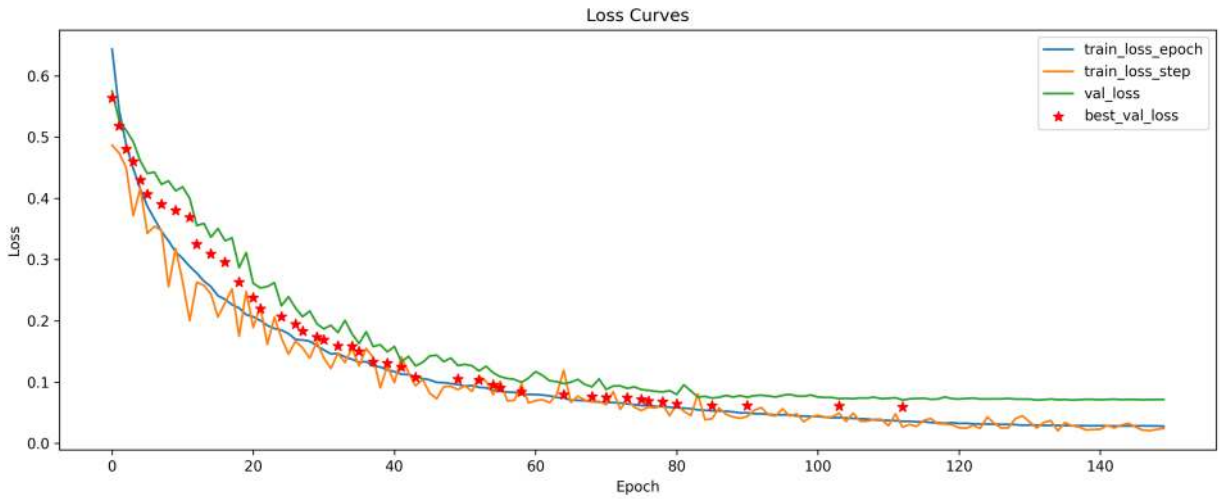


Figure 3.17: Loss plot for the best Clean Ice Physics model.

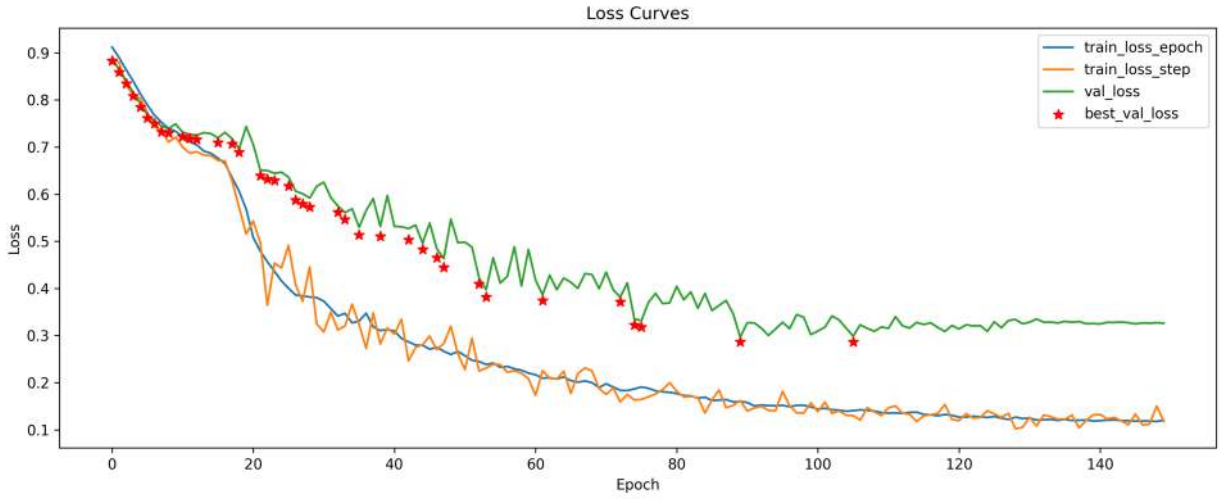
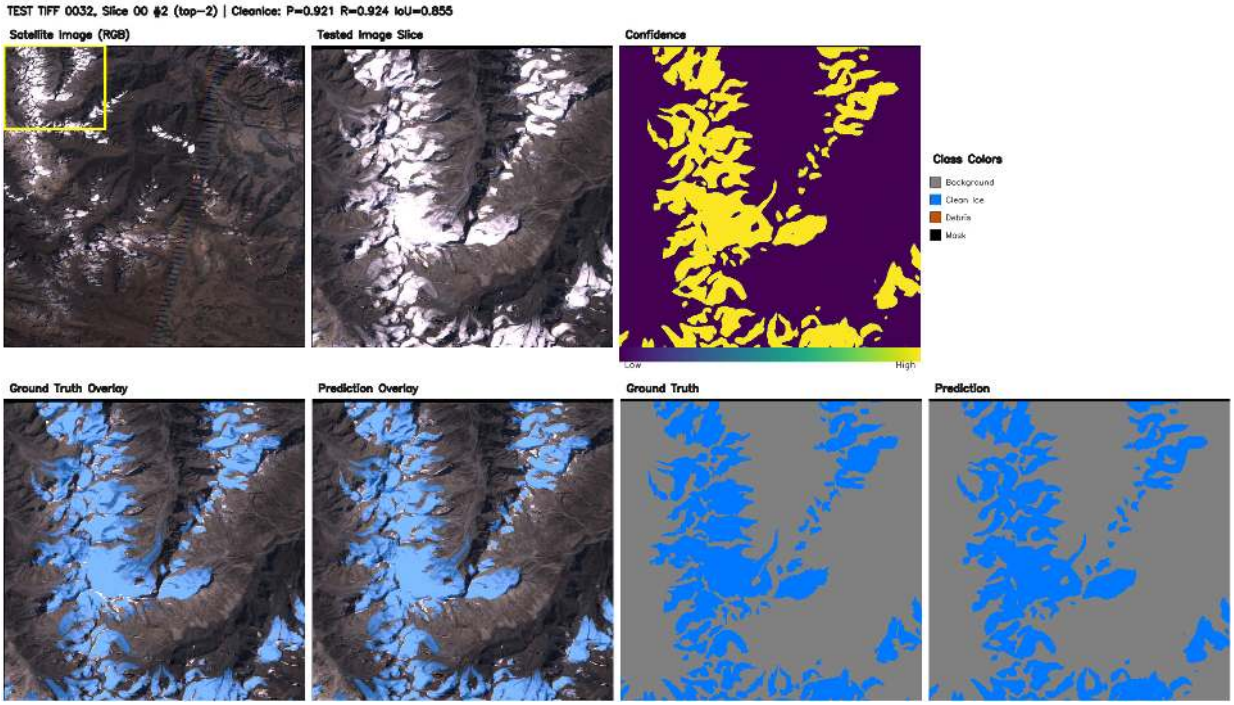
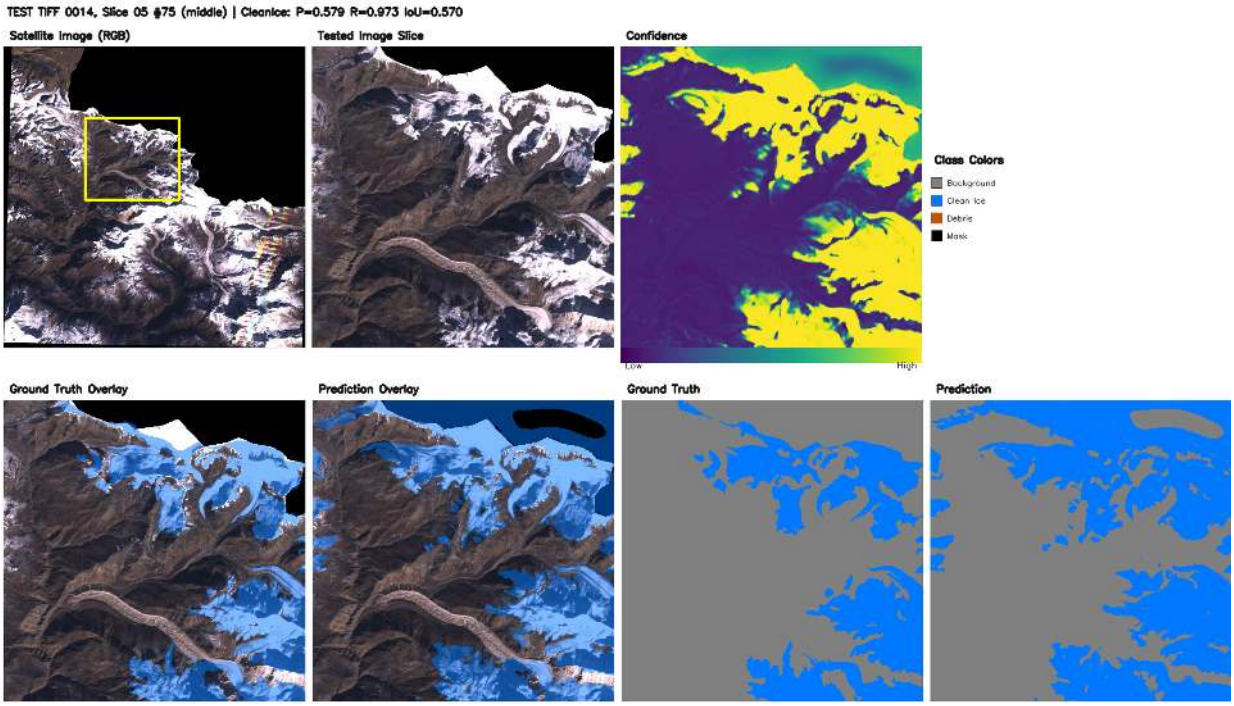


Figure 3.18: Loss plot for the best Debris Ice Physics model.

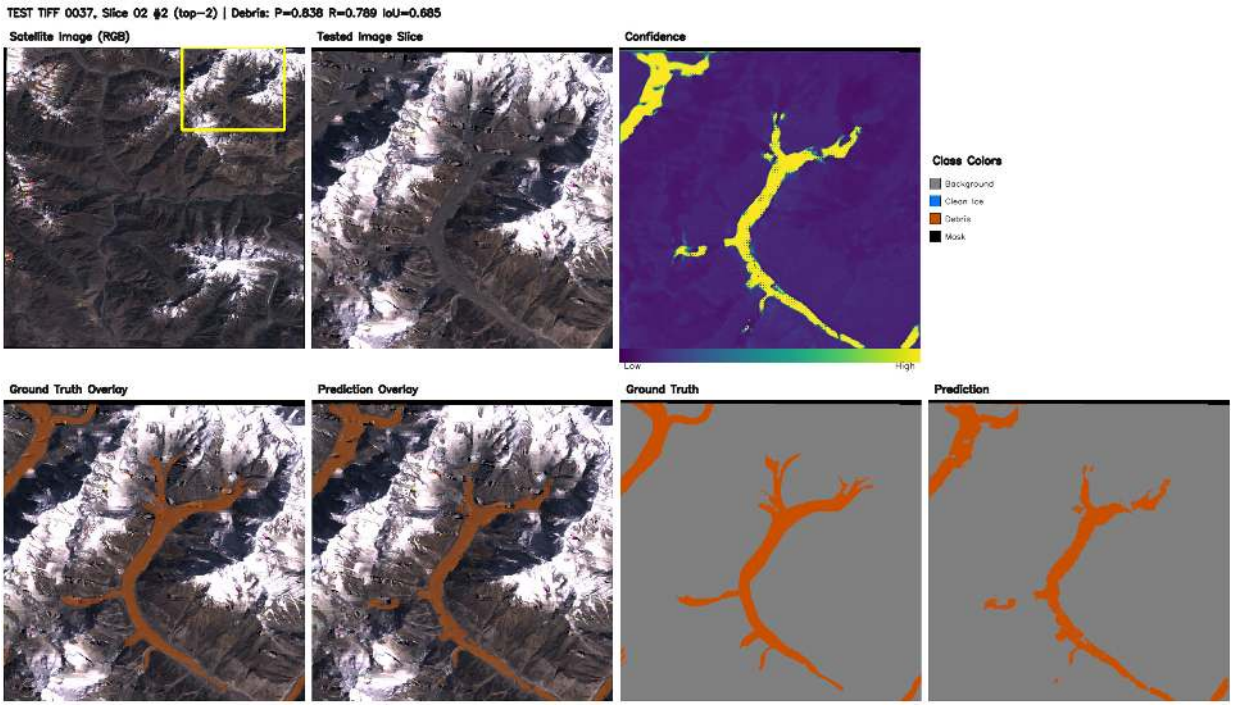


(a) Good prediction

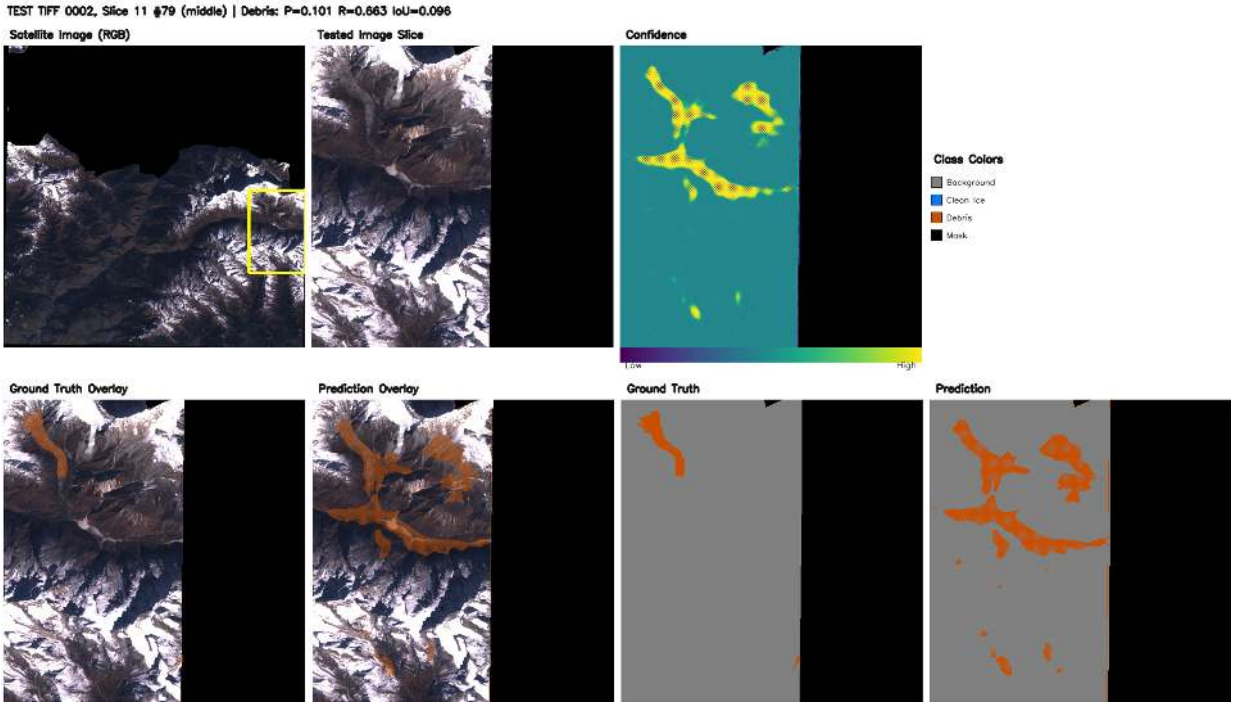


(b) Medium quality prediction

Figure 3.19: Examples of good and medium quality predictions for Clean Ice (CI).



(a) Good prediction



(b) Medium quality prediction

Figure 3.20: Examples of good and medium quality predictions for Debris-Covered Ice (DCI).

The Full Physics model achieves the highest performance across all metrics. Notably, it improves the Intersection over Union (IoU) for Debris-Covered Ice by **+9.98 percentage points** over the state-of-the-art Boundary-Aware baseline (45.92% vs. 35.94%). This confirms that the added terrain features provide critical discriminative cues that help the network distinguish debris-covered ice from the surrounding background, a distinction that is often spectrally ambiguous. The loss plot for the best Debris Ice Physics model is shown in Figure 3.18.

The "Flow Only" configuration provided valuable insights: while it improved Debris-Covered Ice segmentation (38.50% IoU) compared to the SOTA, it exhibited a slight reduction in Clean Ice performance. This suggests that while flow accumulation effectively highlights potential glacier paths (aiding debris detection), relying on it exclusively might introduce noise or over-constrain clean ice regions. However, this characteristic was effectively balanced when combined with the local geomorphometric features (TPI, Roughness, Curvature) in the "Full Physics" model, leading to superior performance in both classes. The four physics-informed channels effectively guide the model toward physically plausible glacier paths and steeper terrain, countering the severe class imbalance.

3.6 Conclusion

The experimental results demonstrate that our proposed physics-informed data augmentation leads to a **9.98 percentage point improvement (27.8% relative improvement)** in debris-covered ice segmentation performance. The use of Topographic Flow Accumulation alone provided a strong foundation by modeling global flow dynamics, while the integration of local terrain features (TPI, Roughness, Curvature) further enhanced the model's accuracy by capturing fine-grained surface geometry.

Future work will explore the application of these techniques to other segmentation architectures and the inclusion of additional physics-based features. Additionally, further investigation into the optimal weighting and normalization of these physics channels could yield even greater performance gains. As the model architecture was kept the same to make a fair comparison between our contributions and the baselines, there is also room for hyperparameter searching to further push the performance of the model predictions.

3.7 Chapter Contributions

This chapter implements and evaluates the second physics-guided strategy proposed in this thesis: **designing custom physics-informed data augmentation algorithms to enrich the feature space of limited datasets**.

To address the ambiguity of debris-covered glaciers in multispectral imagery, we developed a custom data augmentation pipeline grounded in the physics of ice flow. Rather than relying solely on spectral data, our method enhances the state-of-the-art segmentation model [25] by augmenting the input data with physically significant terrain attributes. Specifically, we derived features such as flow accumulation, TPI, roughness, and curvature from DEM data and integrated them as additional input channels. This approach success-

fully leverages physical domain knowledge to enhance the model's ability to distinguish debris-covered ice from the background, a task that has been proven to be challenging for neural network models and glaciologists alike.

The complete implementation and experimental configurations are publicly available at `https://github.com/DeveloperJose/Python-Glacier-Mapping-by-Segmentation` to support reproducibility and enable further research development.

Chapter 4

Integrating Glacier Dynamics with Physics-Informed Loss Functions for Glacier Segmentation

4.1 The Importance of Incorporating Glacier Dynamics for Segmentation

Building on the physics-informed data augmentation approach from the previous chapter, this section introduces a glacier dynamics-guided enhancement strategy. By incorporating temporal data, specifically glacier velocity, we can model glaciers as dynamic systems that evolve over time. While static terrain analysis provides valuable insights into potential flow paths, it lacks information about the actual movement of the ice. Integrating glacier dynamics allows the model to distinguish between actively flowing ice and stationary features that may have similar spectral or topographic characteristics. By integrating temporal constraints into the model training, we can further enhance segmentation performance beyond what is possible with static terrain analysis alone.

4.2 Background: Relevant Concepts

This chapter builds directly upon the U-Net architecture and the boundary-aware learning framework discussed in Chapter 3. The key conceptual shift here is moving from purely data-driven feature extraction (even with physics-informed augmentation) to a constraint-based learning approach where the model is explicitly penalized for violating physical laws governing glacier flow.

4.3 Motivation: Limitations of Data-Driven Dynamics

Direct integration of physics constraints into the loss function provides an additional mechanism for guiding network training beyond data-driven supervision. This approach implements the third strategy from the thesis framework: physics-informed loss functions. By embedding physical laws directly into the optimization objective, we can penalize predictions that are physically implausible, even if they match the labeled data in some areas.

While simply adding velocity data as input channels (as explored in Chapter 3 with static features) provides the network with more information, it does not guarantee that the

network respects the underlying physical principles. For example, a network might learn to associate high velocity with ice, but it might not strictly enforce the constraint that ice must flow downhill or that flow direction should align with the terrain gradient. By explicitly penalizing violations of these physical laws in the loss function, we can guide the network towards solutions that are not only accurate but also physically consistent.

4.4 A Foundational Contribution: A Generalizable Pipeline for Fusing Dynamic and Static Geospatial Datasets

A foundational contribution of this work is the development of a robust, generalizable pipeline for fusing dynamic and static geospatial datasets. While static terrain data (e.g., elevation) is spatially consistent, integrating temporal velocity data from different coordinate systems and resolutions presents significant engineering challenges.

To address this, we engineered a production-ready pipeline that automates the extraction, alignment, and validation of glacier velocity data. The system is designed to process arbitrary geospatial inputs, making it adaptable for future sensors or regions. The data fusion process follows the five-stage pipeline illustrated in **Figure 4.1** and detailed below:

1. **Datacube Discovery and Ranking:** For every Landsat satellite image, the system queries the ITS_LIVE datacube catalog to find all overlapping velocity products. Unlike simple point-based lookups, we perform a geometric intersection between the satellite image footprint and the datacube boundaries. Qualifying datacubes are then ranked by overlap percentage, ensuring the model prioritizes the most complete data sources for each specific glacier.

2. **Temporal Alignment (7-Year Median):** Glacier flow is dynamic and seasonal. To obtain a robust signal that aligns with the ICIMOD ground truth labels (centered around 2005), we extract velocity data from a 7-year window (2002–2008). We compute the pixel-wise temporal median for this period. This statistical aggregation filters out transient noise, seasonal fluctuations, and measurement outliers, yielding a stable representation of the glacier's dominant flow pattern.

3. **Cross-Zone Reprojection:** A major challenge in large-scale geospatial analysis is the mismatch of Coordinate Reference Systems (CRS). Landsat images and ITS_LIVE datacubes often reside in different UTM zones. Our pipeline automatically detects these mismatches and performs a mathematically rigorous reprojection, transforming the velocity vectors into the target Landsat coordinate system while preserving their magnitude and directional correctness.

4. **Spatial Resampling:** The raw velocity data is provided at a 120m resolution, which is coarser than the 30m Landsat imagery. We upsample the velocity grid using bilinear interpolation to match the spatial resolution and pixel grid of the satellite imagery.

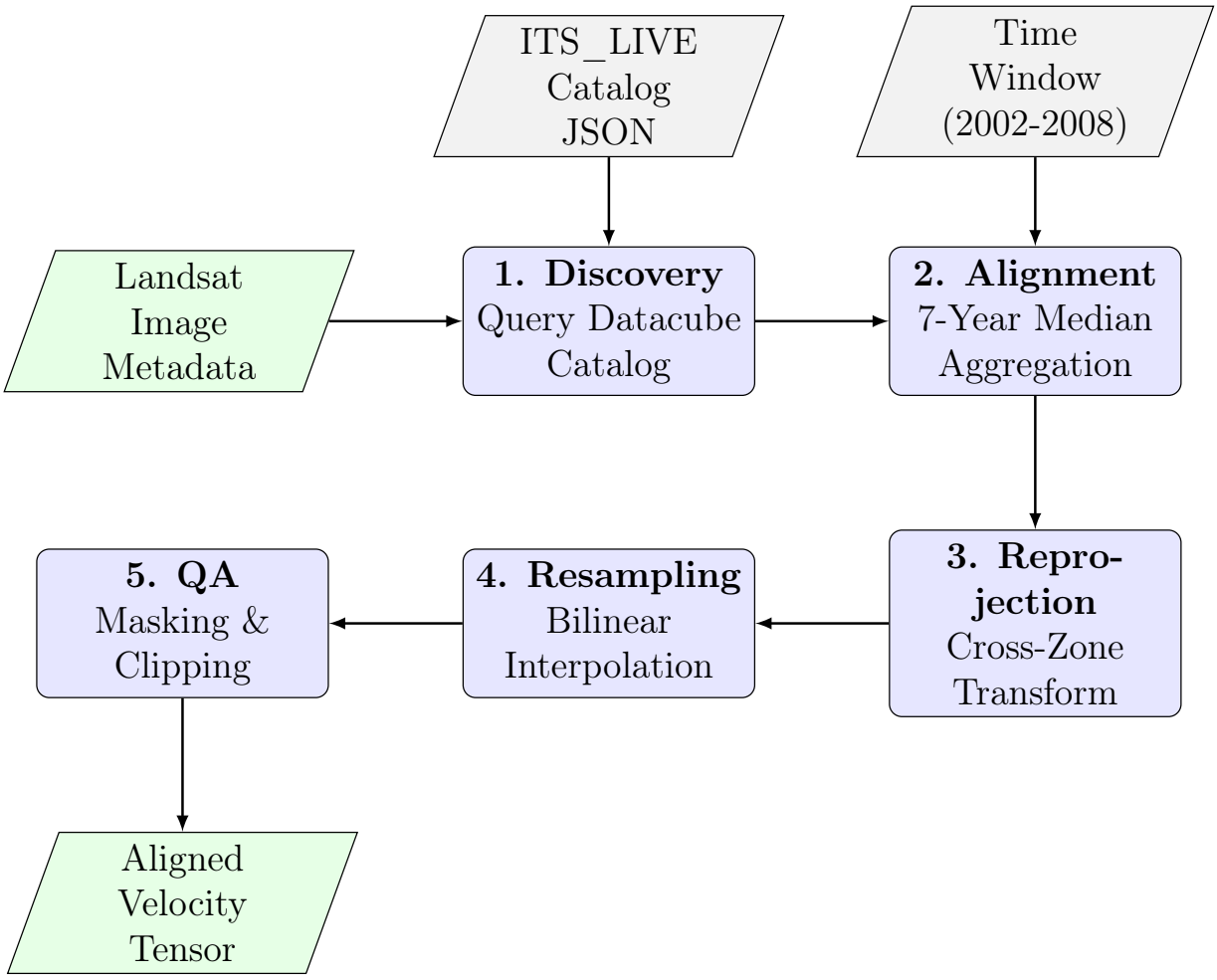


Figure 4.1: The automated data fusion pipeline for integrating ITS_LIVE velocity datacubes with Landsat imagery. The system handles spatial discovery, temporal aggregation, and cross-zone reprojection to generate a pixel-aligned velocity tensor.

This ensures that every pixel in the neural network input tensor has a corresponding, spatially aligned velocity vector.

5. **Quality Assurance and Masking:** Real-world sensor data often contains artifacts. The pipeline applies a validity mask to flag pixels where velocity data is missing or statistically unreliable. Furthermore, we enforce physical plausibility constraints, clipping extreme values (e.g., errors $> 50{,}000$ m/yr) that represent sensor noise rather than physical ice movement.

This pipeline processes 202 glacier images across the Hindu Kush Himalaya, generating a 4-channel velocity product $(v_{magnitude}, v_x, v_y, mask)$ that is perfectly aligned with our spectral and topographic data. Examples of the velocity magnitude are shown in Figure 4.2, the X-component of the velocity is shown in Figure 4.3, and the Y-component of the velocity is shown in Figure 4.4.

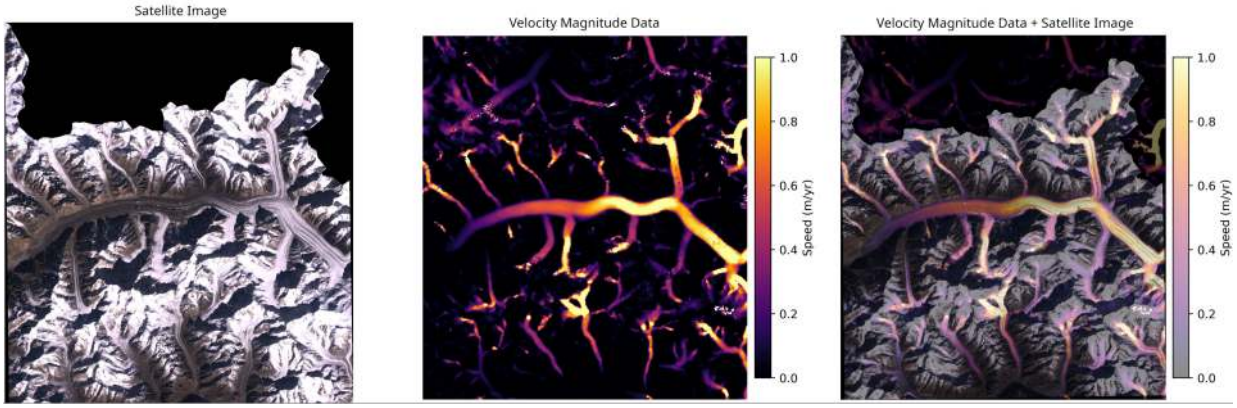


Figure 4.2: Example of the Velocity Magnitude (overall flow speed) of a satellite image. From left to right, the RGB satellite image, the velocity magnitude data, and the data overlaid on top of the satellite image.

4.5 Problem Formulation and Evaluation

4.5.1 Glacier Segmentation with Dynamic Data

We extend the problem formulation from Chapter 3 to include dynamic features. The input is a concatenation of spectral, physics-informed data augmentation, and glacier dynamics channels, while the output remains a three-class segmentation mask. The inputs and outputs are detailed in Table 4.1.

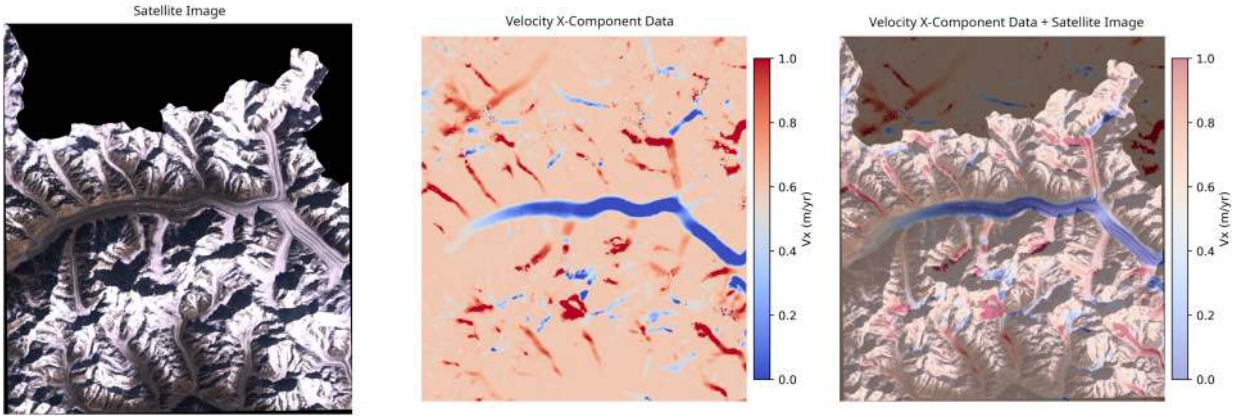


Figure 4.3: Example of the Velocity X-Component (East-West flow) of a satellite image. From left to right, the RGB satellite image, the velocity x-component data, and the data overlaid on top of the satellite image.

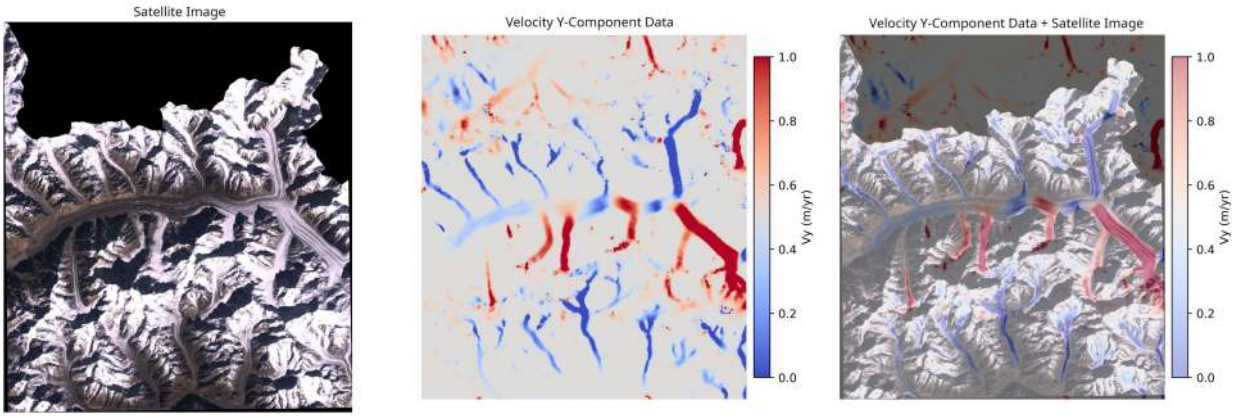


Figure 4.4: Example of the Velocity Y-Component (North-South flow) of a satellite image. From left to right, the RGB satellite image, the velocity y-component data, and the data overlaid on top of the satellite image.

Table 4.1: Extended problem formulation with static and dynamic physics inputs.

Role	Component	Unit	Description
	Channels 1-13	-	(Detailed in Table 3.2).
	Vel. Mag.	m/yr	Speed of ice flow.
	Vel. X (v_x)	m/yr	East-West flow component.
	Vel. Y (v_y)	m/yr	North-South flow component.
	Mask	0/1	Data validity flag.
Output (Y)	Class 0	Label	Background.
	Class 1	Label	Clean Ice.
	Class 2	Label	Debris-Covered Ice.

4.5.2 Measuring Network Performance

Consistent with Chapter 3, we evaluate performance using Intersection over Union (IoU), Precision, Recall, and Confusion Matrix analysis. These metrics allow us to quantify the specific impact of adding dynamic information, particularly on the reduction of false positives (e.g., stationary rock features) and the improved delineation of active ice boundaries. Please refer to Section 3.4.3 in Chapter 3 for detailed definitions of these metrics.

4.6 Proposed Approach: Integrating Glacier Dynamics with Physics-Informed Loss Functions

We integrated glacier velocity data into the glacier segmentation dataset presented previously as 4 new additional channels: Velocity Magnitude (overall glacier flow speed in m/year), Velocity X-Component (East-west flow direction and magnitude), Velocity Y-Component (North-south flow direction and magnitude), and a Velocity Mask (binary mask used to inform the model where velocity data was either not found or was invalid).

4.6.1 Proposed Physics-Informed Loss Function

Our primary contribution in this chapter is the development of a multi-component loss function that integrates glacier dynamics. This was achieved in two stages: first, by correcting and standardizing the multi-task learning framework of the baseline model, and second, by introducing a novel physics-informed velocity loss term.

Correcting the Multi-Task Loss Weighting

The baseline model [25] combined a **Dice loss** for region-based accuracy and a **Boundary loss** for edge definition. The Dice Loss is defined as [33]:

$$L_{Dice} = 1 - \frac{2|A \cap B|}{|A| + |B|} \tag{4.1}$$

where A and B are the predicted and ground truth segmentation masks, respectively. The Boundary loss is a distance metric based on the boundaries of the segmentation masks [34]. It is defined as:

$$L_{Boundary} = \int_{\partial G} \phi_G(p) dp \tag{4.2}$$

where ∂G is the boundary of the ground truth segmentation and $\phi_G(p)$ is a distance map computed from the predicted segmentation. It attempted to balance these two components using a learnable uncertainty weighting scheme based on the following equation:

$$L_{Total} = \frac{1}{2\sigma_1^2} L_{Dice} + \frac{1}{2\sigma_2^2} L_{Boundary} + |\log(\sigma_1 \sigma_2)| \tag{4.3}$$

However, the final regularization term is a non-standard formulation that is problematic because it penalizes sigmas that are either very large or very small. The term $|\log(\sigma_1\sigma_2)|$ approaches infinity as the product of the sigmas approaches both zero and infinity. This effectively forces the product $\sigma_1\sigma_2$ to remain close to their initial values, preventing the network from learning to dynamically prioritize one loss component over the other. For instance, an optimal training strategy might involve focusing on the region-based L_{Dice} in the early stages and later refining the edges using the $L_{Boundary}$. The original loss function's structure restricts this dynamic learning process by locking the weights.

Our first step was to replace this with the standard, well-established uncertainty weighting method proposed by Kendall et al. [42]. For each component of the total loss (Dice, Boundary, and our new Velocity loss), the final loss is calculated as:

$$L_{Total} = \sum_i \left(\frac{1}{2\sigma_i^2} L_i + \frac{1}{2} \log(\sigma_i^2) \right) \tag{4.4}$$

where L_i is the individual loss component (e.g., L_{Dice}) and σ_i is a learnable parameter representing the uncertainty of that task. This formulation allows the model to automatically learn the optimal balance between the different loss terms by down-weighting tasks with higher uncertainty. This correction provides a more stable and theoretically sound foundation for multi-objective optimization.

A Simplified Physics-Informed Velocity Loss

With a stable multi-task framework in place, we introduced a third loss component to incorporate glacier dynamics. Instead of relying on complex mathematical formulations, we implemented a simplified velocity-aware loss function grounded in the statistical analysis of our training data. Our analysis confirms a distinct separation in flow dynamics: background

terrain features are statistically stationary, with a median velocity of approximately 2.06 m/yr and a tight distribution (75th percentile at 3.16 m/yr). In contrast, glacier ice exhibits significantly higher motion, with Clean Ice and Debris-Covered Ice showing mean velocities of 19.87 m/yr and 21.86 m/yr, respectively.

Guided by Kolmogorov-Smirnov tests (see Table A.2 in Appendix A.1), these distinct velocity profiles support a straightforward logic: if the network's prediction contradicts these physical trends, it is likely incorrect.

The logic is grounded in a single, statistically robust check:

- If the network predicts **Background** (static terrain), but the velocity is **high**, the prediction is penalized. This is based on the principle that moving features are almost certainly ice.

We deliberately exclude a penalty for stationary ice, as our analysis showed that a significant portion of valid glacier ice (e.g., stagnant tongues) exhibits near-zero velocity. To handle the high dynamic range of glacier speeds (0 to >3000 m/yr), we employ a sigmoid-based probability function centered at the background's 75th percentile (3.16 m/yr). This calculates a "Probability of Motion" (P_{moving}) for each pixel. The loss term penalizes the network specifically when it assigns a high probability to the Background class on a pixel where $P_{moving} \approx 1$. This approach effectively teaches the network to distinguish glaciers not just by their appearance, but by their movement, without penalizing it for correctly identifying stagnant ice.

4.7 Experimental Results and Discussion

4.7.1 Experimental Setup

To isolate the impact of incorporating dynamic glacier information, we compare our proposed dynamic physics-informed model against the baselines and the best-performing static model from Chapter 3.

The evaluation includes the following configurations:

- **Standard U-Net:** The baseline deep learning model [32].
- **Boundary-Aware U-Net (SOTA):** The state-of-the-art model for this dataset [25].
- **Ours (Velocity Channels Only)**: The best performing model from Chapter 3, with only the Landsat bands and velocity information as an additional channel without the velocity loss or any physics-informed data augmentation.
- **Ours (Velocity Channels + Loss):** The proposed model in this chapter, which adds the physics-informed velocity loss alongside the velocity channel.
- **Ours (Complete Physics-Informed):** Our final proposed model, which integrates the physics-informed data augmentation from Chapter 3 with the dynamic velocity loss from this chapter.

4.7.2 Quantitative Analysis

Table 4.2 presents the comparative results.

Table 4.2: Comparative performance including velocity physics integration. All values are in percentages (%).

	Debris-Covered Ice			**Clean Ice**		
Model	**IoU**	**Prec.**	**Rec.**	**IoU**	**Prec.**	**Rec.**
Standard U-Net	28.50	-	-	65.60	-	-
Boundary-Aware (SOTA) [25]	35.94	51.97	53.81	**68.17**	81.59	80.55
Ours (Velocity Channels Only)	32.40	70.25	37.56	70.78	**82.27**	83.52
Ours (Velocity Channels + Loss)	41.91	66.40	53.20	61.83	64.90	**92.90**
Ours (Complete Physics-Informed)	**46.07**	**71.95**	**56.16**	65.85	72.36	87.98

The results in Table 4.2 demonstrate a clear progression in model performance. While adding velocity data as input channels provides a **3.90 percentage point improvement (13.7% relative gain)** over the Standard U-Net baseline, the inclusion of the physics-informed velocity loss yields a further **9.51 percentage point increase (29.4% relative gain)** over the channels-only approach, demonstrating that enforcing physical consistency is more effective than passive feature addition. Critically, our final model, which combines the physics-informed data augmentation strategies from Chapter 3 with the dynamic velocity loss function from this chapter, achieves a new state-of-the-art for Debris-Covered Ice segmentation with an IoU of 46.07%, representing a relative improvement of over **28%** compared to the previous state-of-the-art baseline. This **28.2%** relative improvement underscores the value of a holistic physics-guided approach that informs the model both through its input data and its learning objective.

While Debris-Covered Ice performance surged to state-of-the-art levels, achieving an IoU of 46.07% (a **28.2%** relative improvement), Clean Ice performance remained robust with an IoU of 65.85%. A key observation is the change in Clean Ice precision from 85.39% to 72.36%, which suggests a shift in the model's classification strategy. This outcome highlights the model's strong focus on the more challenging Debris-Covered Ice class, where the physics-informed loss provides benefits of **10.13 percentage points (28.2% relative improvement)** over the baseline.

We hypothesize that several factors may contribute to this trade-off, which require further investigation. From a **data perspective**, the resolution mismatch between the 120m velocity data and the 30m satellite imagery could introduce boundary artifacts. Additionally, as noted by Aryal et al. [25], the ground truth labels themselves have inherent inaccuracies, which could challenge a model being explicitly guided by a physics-informed loss function. In terms of **network optimization**, the physics-informed velocity loss, which penalizes background predictions on moving terrain, may encourage higher recall at the expense of precision. Confirming these hypotheses is a key area for future work.

The velocity-informed loss is designed to reduce false negatives on moving ice by penalizing background predictions proportionally to observed speed. The methodology establishes

a principled way to inject dynamics directly into training, complementing the static terrain analysis.

4.8 Conclusions and Future Directions

This chapter demonstrates that coupling segmentation decisions to observed glacier motion through a physics-informed loss function provides meaningful improvements over passive feature addition. By penalizing the model for classifying moving terrain as static background, the velocity-aware loss suppresses false negatives on active ice features, addressing a key challenge in debris-covered glacier mapping. The success of this approach highlights the value of embedding physical constraints directly into the optimization objective rather than relying solely on spectral or topographic inputs. Future research should explore tighter integration with higher-resolution velocity products and the development of more complex velocity loss functions to further improve segmentation accuracy.

4.9 Chapter Contributions

This chapter implements and evaluates the third physics-guided strategy proposed in this thesis: **integrating physics-informed terms directly into the loss functions of existing models (e.g., U-Nets)**.

This chapter makes two distinct contributions:

1. **Development of a Generalizable Geospatial Data Fusion Pipeline:** We developed a production-ready pipeline to fuse dynamic ITS_LIVE glacier velocity data with the static HKH glacier dataset. This methodology solves significant engineering challenges related to multi-source data integration, temporal alignment, and cross-UTM-zone reprojection. It is designed to be adaptable to any geospatial data in standard formats (e.g., GeoTIFF, shapefiles), providing a critical tool for future research and a pathway to integrate higher-resolution satellite imagery to overcome the limitations of the Landsat 7 data.

2. **Development of a Physics-Informed Loss Function Strategy:** We proposed and evaluated a novel loss function that penalizes predictions violating physical laws of glacier flow (specifically, classifying moving terrain as background). By integrating this loss into the U-Net training, we demonstrated that enforcing physical consistency leads to more robust segmentation performance, particularly in distinguishing active ice from static terrain features, surpassing the benefits of simply adding velocity data as input channels.

The complete implementation and experimental configurations are publicly available at `https://github.com/DeveloperJose/Python-Glacier-Mapping-by-Segmentation` to support reproducibility and enable further research development.

Chapter 5

Conclusions and Future Work

5.1 Summary of Contributions

This dissertation introduced three physics-guided strategies to enhance neural network performance and physical consistency. First, a novel Physics-Informed LSTM architecture achieved **73.7%** improvement in U-velocity and **85.3%** improvement in V-velocity prediction compared to LSTM-only approaches for fluid flow prediction. Second, our physics-informed data augmentation improved clean-ice IoU from 68.17 to 71.22 (a **4.5%** gain) and debris-covered ice IoU from 35.94 to 45.92 (a **27.8%** gain) over the baseline for glacier segmentation. Third, a dynamic physics-guided approach that incorporated glacier velocity fields via a physics-informed loss achieved a state-of-the-art Debris-Covered Ice IoU of **46.07%**, representing a **28.2%** relative improvement over prior benchmarks, and was enabled by a production-ready pipeline that fuses ITS_LIVE dynamics with Landsat/ICIMOD data, establishing a reusable foundation for multi-source geospatial learning. Furthermore, the 46.07% IoU benchmark highlights the synergy between the strategies in Chapters 3 and 4. This shows that a holistic approach that informs a model through both its input data and its learning objective is most effective. Together, these strategies demonstrate that embedding physics through architecture changes, new inputs, and new loss terms can improve model performance without needing additional data.

5.2 Limitations and Potential Future Work

Despite promising results, this work has several limitations. The fluid flow experiments in Chapter 2 were restricted to 2D incompressible flows, a simplification that neglects the effects of **turbulence** common in high-Reynolds-number industrial applications. Furthermore, compute-limited training budgets restricted the exploration of deeper temporal models that might better capture complex, long-range flow dynamics. Lastly, while LSTMs are effective, future research should explore more advanced architectures like Transformers or Neural Operators. These newer models may be better suited for capturing long-range dependencies and multi-scale physics in complex dynamic systems, offering another path for improvement over our current research.

For the glacier studies, the reliance on 30m Landsat 7 resolution introduces the **mixed-pixel problem**, where single pixels at a glacier's edge contain both ice and terrain due to the resolution of the images. This creates label ambiguity that a physics-informed loss struggles to resolve without sub-pixel information, which could be improved by using higher resolution images from a newer satellite mission like Sentinel-2. Additionally, incomplete velocity data results in sparse regions without reliable velocity measurements. This leads

the model's loss function to effectively revert to a standard data-driven objective, reducing the global consistency of the physical constraints.

5.3 Generalizability and Broader Impact

While this research focused on fluid dynamics and glaciology, the core methodology-integrating physical laws into deep learning pipelines-is broadly applicable. In medical imaging, for instance, blood flow dynamics governed by hemodynamic equations could enhance vessel segmentation in low-contrast MRI scans, guiding the model to identify physically plausible vascular structures. The true contribution of this work is a generalized blueprint for injecting domain knowledge at three distinct levels: the model **architecture** (hybrid models with PINNs), the **input data** (physics-informed augmentation), and the **loss function** (physics-informed loss functions penalizing outputs that deviate from governing physical laws).

This multi-level integration strategy offers a flexible framework for any scientific field with well-defined governing laws. Furthermore, the glaciology research has direct societal implications. More accurate and reliable glacier mapping is critical for monitoring climate change, forecasting sea-level rise, and managing freshwater resources for downstream communities, demonstrating how these advanced techniques can address pressing environmental challenges.

Bibliography

[1] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. B. Ommer, "High-resolution image synthesis with latent diffusion models," pp. 10 674–10 685, 2021, ISSN: 10636919. DOI: 10.1109/CVPR52688.2022.01042. [Online]. Available: https://doi.org/10.1109/CVPR52688.2022.01042.

[2] J. Redmon, S. K. Divvala, R. B. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 779–788. DOI: 10.1109/CVPR.2016.91. [Online]. Available: https://doi.org/10.1109/CVPR.2016.91.

[3] M. Tan and Q. V. Le, "EfficientNet: Rethinking model scaling for convolutional neural networks," vol. abs/1905.11946, 2019. arXiv: 1905.11946. [Online]. Available: https://arxiv.org/abs/1905.11946.

[4] J. Devlin, M. W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *North American Chapter of the Association for Computational Linguistics*, Association for Computational Linguistics, 2019, pp. 4171–4186. DOI: 10.18653/v1/N19-1423. arXiv: 1810.04805. [Online]. Available: https://doi.org/10.18653/v1/N19-1423.

[5] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, pp. 1735–1780, 1997. DOI: 10.1162/neco.1997.9.8.1735. [Online]. Available: https://doi.org/10.1162/neco.1997.9.8.1735.

[6] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, pp. 436–444, 7553 2015, ISSN: 1476-4687. DOI: 10.1038/nature14539. [Online]. Available: https://doi.org/10.1038/nature14539.

[7] T. B. Brown et al., "Language models are few-shot learners," in *Advances in Neural Information Processing Systems*, vol. 33, Curran Associates, Inc., 2020, pp. 1877–1901. arXiv: 2005.14165. [Online]. Available: https://proceedings.neurips.cc/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf.

[8] J. Mao, S. Shi, X. Wang, and H. Li, "3d object detection for autonomous driving: A comprehensive survey," *International Journal of Computer Vision*, vol. 131, pp. 1909–1963, 2023. DOI: 10.1007/s11263-023-01774-z. [Online]. Available: https://doi.org/10.1007/s11263-023-01774-z.

[9] I. O. Tolstikhin et al., "MLP-Mixer: An all-MLP architecture for vision," *arXiv preprint arXiv:2105.01601*, vol. abs/2105.01601, 2021. arXiv: 2105.01601. [Online]. Available: https://arxiv.org/abs/2105.01601.

[10] J. Deng, W. Dong, R. Socher, L. J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009, pp. 248–255. DOI: 10.1109/CVPR.2009.5206848. [Online]. Available: https://doi.org/10.1109/CVPR.2009.5206848.

[11] T.-Y. Lin et al., "Microsoft COCO: Common objects in context," in *European Conference on Computer Vision*, 2014, pp. 740–755. DOI: 10.1007/978-3-319-10602-1_48. [Online]. Available: https://doi.org/10.1007/978-3-319-10602-1_48.

[12] Y. Wang, Q. Yao, J. T. Kwok, and L. M. Ni, "Generalizing from a few examples: A survey on few-shot learning," *ACM Comput. Surv.*, vol. 53, 3 Jun. 2020, ISSN: 0360-0300. DOI: 10.1145/3386252. [Online]. Available: https://doi.org/10.1145/3386252.

[13] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, pp. 1345–1359, 10 2010. DOI: 10.1109/TKDE.2009.191.

[14] C. Shorten and T. M. Khoshgoftaar, "A survey on image data augmentation for deep learning," *Journal of Big Data*, vol. 6, no. 1, p. 60, 2019, ISSN: 2196-1115. DOI: 10.1186/s40537-019-0197-0. [Online]. Available: https://doi.org/10.1186/s40537-019-0197-0.

[15] T. Hospedales, A. Antoniou, P. Micaelli, and A. Storkey, "Meta-learning in neural networks: A survey," *IEEE Transactions on Pattern Analysis & Machine Intelligence*, vol. 44, pp. 5149–5169, 09 Sep. 2020, ISSN: 1939-3539. DOI: 10.1109/TPAMI.2021.3079209. [Online]. Available: https://doi.org/10.1109/TPAMI.2021.3079209.

[16] L. Deng, "The MNIST database of handwritten digit images for machine learning research," *IEEE Signal Processing Magazine*, vol. 29, pp. 141–142, 6 2012, ISSN: 10535888. DOI: 10.1109/MSP.2012.2211477.

[17] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *International Conference on Machine Learning*, 2017, pp. 1126–1135. [Online]. Available: https://proceedings.mlr.press/v70/finn17a.html.

[18] G. Koch, R. Zemel, and R. Salakhutdinov, "Siamese neural networks for one-shot image recognition," in *International Conference on Machine Learning (ICML) Deep Learning Workshop*, vol. 2, 2015. [Online]. Available: https://www.cs.cmu.edu/~rsalakhu/papers/oneshot1.pdf.

[19] J. Snell, K. Swersky, and R. S. Zemel, "Prototypical networks for few-shot learning," in *Advances in Neural Information Processing Systems*, vol. 30, Curran Associates, Inc., 2017, pp. 4077–4087. [Online]. Available: https://proceedings.neurips.cc/paper/2017/file/cb8da6767461f2812ae4290eac7cbc42-Paper.pdf.

[20] F. Sung, Y. Yang, L. Zhang, T. Xiang, P. H. S. Torr, and T. M. Hospedales, "Learning to compare: Relation network for few-shot learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, 2017, pp. 1199–1208. DOI: 10.1109/cvpr.2018.00131. [Online]. Available: https://doi.org/10.1109/cvpr.2018.00131.

[21] 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*, vol. 378, pp. 686–707, Feb. 2019, ISSN: 10902716. DOI: 10.1016/J.JCP.2018.10.045. [Online]. Available: https://doi.org/10.1016/J.JCP.2018.10.045.

[22] ANSYS, Inc., *ANSYS fluent*, version 2023 R1, Canonsburg, PA, 2023. [Online]. Available: https://www.ansys.com/products/fluids/ansys-fluent.

[23] J. Pérez et al., "Physics-informed long-short term memory neural network performance on holloman high-speed test track sled study," in *American Society of Mechanical Engineers, Fluids Engineering Division (Publication) FEDSM*, vol. 2, American Society of Mechanical Engineers Digital Collection, 2022, ISBN: 9780791885840. DOI: 10.1115/FEDSM2022-86953. [Online]. Available: https://dx.doi.org/10.1115/FEDSM2022-86953.

[24] N. Snow and I. D. Center, *Why glaciers matter | national snow and ice data center*, 2023. [Online]. Available: https://nsidc.org/learn/parts-cryosphere/glaciers/why-glaciers-matter.

[25] B. Aryal, K. E. Miles, S. A. V. Zesati, and O. Fuentes, "Boundary aware U-Net for glacier segmentation," in *Proceedings of the Northern Lights Deep Learning Workshop*, vol. 4, May 2023. DOI: 10.7557/18.6789.

[26] S. R. Bajracharya and B. R. Shrestha, "The status of glaciers in the hindu kush-himalayan region," International Centre for Integrated Mountain Development (ICIMOD), Tech. Rep., 2011. DOI: 10.53055/icimod.551. [Online]. Available: https://doi.org/10.53055/icimod.551.

[27] A. Gardner et al., *MEASURES its_live regional glacier and ice sheet surface velocities, version 2*, 2024. DOI: 10.5067/JQ6337239C96. [Online]. Available: https://nsidc.org/data/nsidc-0776/versions/2.

[28] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE Transactions on Signal Processing*, vol. 45, pp. 2673–2681, 11 1997. DOI: 10.1109/78.650093. [Online]. Available: https://doi.org/10.1109/78.650093.

[29] P. J. Werbos, "Backpropagation through time: What it does and how to do it," *Proceedings of the IEEE*, vol. 78, pp. 1550–1560, 10 1990. DOI: 10.1109/5.58337. [Online]. Available: https://doi.org/10.1109/5.58337.

[30] K. Fukushima, "Visual feature extraction by a multilayered network of analog threshold elements," *IEEE Transactions on Systems Science and Cybernetics*, vol. 5, pp. 322–333, 4 1969, ISSN: 21682887. DOI: 10.1109/TSSC.1969.300225. [Online]. Available: https://doi.org/10.1109/TSSC.1969.300225.

[31] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems*, https://proceedings.neurips.cc/paper_files/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf, vol. 25, Curran Associates, Inc., 2012, pp. 1097–1105.

[32] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," *arXiv preprint arXiv:1505.04597*, vol. abs/1505.04597, 2015. DOI: 10.1007/978-3-319-24574-4_28. arXiv: 1505.04597. [Online]. Available: https://doi.org/10.1007/978-3-319-24574-4_28.

[33] L. R. Dice, "Measures of the amount of ecologic association between species," *Ecology*, vol. 26, no. 3, pp. 297–302, 1945. DOI: 10.2307/1932409. [Online]. Available: https://doi.org/10.2307/1932409.

[34] H. Kervadec, J. Dolz, J. Yuan, C. Desrosiers, E. Granger, and I. B. Ayed, "Boundary loss for highly unbalanced segmentation," in *Proceedings of the 2nd International Conference on Medical Imaging with Deep Learning*, vol. 102, PMLR, 2019, pp. 285–296. [Online]. Available: https://proceedings.mlr.press/v102/kervadec19a.html.

[35] A. D. Weiss, *Topographic position and landforms analysis*, Poster presentation, ESRI User Conference, San Diego, CA, The Nature Conservancy, 2001.

[36] J. P. Wilson and J. C. Gallant, *Terrain Analysis: Principles and Applications*. New York: Wiley, 2000.

[37] L. W. Zevenbergen and C. R. Thorne, "Quantitative analysis of land surface topography," *Earth Surface Processes and Landforms*, vol. 12, no. 1, pp. 47–56, 1987. DOI: 10.1002/esp.3290120107.

[38] G. Chander, B. L. Markham, and D. L. Helder, "Summary of current radiometric calibration coefficients for Landsat mss, tm, ETM+, and EO-1 ALI sensors," *Remote Sensing of Environment*, vol. 113, pp. 893–903, 2009. DOI: 10.1016/j.rse.2009.01.007.

[39] J. W. Rouse, R. H. Haas, J. A. Schell, and D. W. Deering, "Monitoring vegetation systems in the great plains with ERTS," in *Third Earth Resources Technology Satellite-1 Symposium*, 1973, pp. 309–317.

[40] S. K. McFeeters, "The use of the normalized difference water index (NDWI) in the delineation of open water features," *International Journal of Remote Sensing*, vol. 17, no. 7, pp. 1425–1432, 1996. DOI: 10.1080/01431169608948714.

[41] D. K. Hall, G. A. Riggs, and V. V. Salomonson, "Development of methods for mapping global snow cover using moderate resolution imaging spectroradiometer data," *Remote Sensing of Environment*, vol. 54, no. 2, pp. 127–140, 1995. DOI: 10.1016/0034-4257(95)00137-K. [Online]. Available: https://doi.org/10.1016/0034-4257(95)00137-K.

[42] A. Kendall, Y. Gal, and R. Cipolla, "Multi-task learning using uncertainty to weigh losses for scene geometry and semantics," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 7482–7491. DOI: 10.1109/CVPR.2018.00781. [Online]. Available: https://doi.org/10.1109/CVPR.2018.00781.

[43] A. Kolmogorov, "Sulla determinazione empirica di una legge di distribuzione," *Giornale dell'Istituto Italiano degli Attuari*, vol. 4, pp. 83–91, 1933.

[44] N. Smirnov, "Table for estimating the goodness of fit of empirical distributions," *The Annals of Mathematical Statistics*, vol. 19, no. 2, pp. 279–281, 1948.

Appendix A

Appendix: Ablations and Extra Analysis

A.1 Statistical Feature Analysis

To gain insight into the discriminative potential of each input feature, we performed two-sample Kolmogorov-Smirnov (K-S) tests [43, 44] between the distributions of the three semantic classes: Background, Clean Ice, and Debris-Covered Ice. The K-S statistic measures the maximum distance between the empirical cumulative distribution functions (ECDF) of two samples.

This statistical analysis has limitations. The calculations were performed on a sampled subset of the dataset (skipping intermediate slices to reduce computational load), and the underlying ground truth masks may contain minor labeling imperfections. Furthermore, univariate statistics cannot fully capture the complex, non-linear interactions exploited by deep learning models. Therefore, these values are simplified heuristic indicators for our feature selection strategy, not definitive proofs of feature optimality.

Tables A.1 and A.2 present the K-S statistics for spectral and physical features, respectively. All reported tests yielded statistically significant p-values ($p < 0.001$), though significance is easily achieved with large sample sizes. A higher statistic generally suggests a stronger signal for differentiating between the class pairs.

These results suggest that spectral features, particularly NDSI and Saturation, appear effective at distinguishing Clean Ice from both Background and Debris Ice. However, they show lower potential for separating Debris Ice from Background, with all K-S statistics for the BG vs. DCI comparison remaining relatively low (below 0.35). This indicates that spectral information alone may be insufficient for robust debris-covered glacier segmentation, pointing towards the need for additional feature types.

The trends in Table A.2 suggest that topographic features, specifically Roughness and Slope, provide stronger discrimination between Background and Debris-Covered Ice compared to spectral features in this analysis. Velocity also appears to offer a useful signal for distinguishing Debris Ice from the static Background. While subject to the sampling and labeling limitations mentioned above, these patterns support the hypothesis that combining spectral data (strong for Clean Ice) with topographic/dynamic data (strong for Debris Ice) creates a more robust feature set for glacier mapping.

Table A.1: Kolmogorov-Smirnov statistics for Spectral Features. (BG: Background, CI: Clean Ice, DCI: Debris Ice)

Feature	**BG vs. CI**	**BG vs. DCI**	**CI vs. DCI**
Band 1	**0.632**	0.196	**0.643**
Band 2	0.620	0.172	0.637
Band 3	0.608	0.149	0.629
Band 4	0.563	0.091	0.587
Band 5	0.507	0.111	0.612
Band 6 (VCID1)	0.587	0.151	0.641
Band 6 (VCID2)	0.587	0.151	0.642
Band 7	0.516	0.126	0.636
NDVI	0.260	**0.328**	0.241
NDWI	0.281	**0.289**	0.181
NDSI	**0.743**	0.203	**0.811**
Hue (H)	0.534	**0.269**	0.577
Saturation (S)	**0.732**	0.129	**0.805**
Value (V)	0.593	0.106	0.630

Table A.2: Kolmogorov-Smirnov statistics for Topographic & Dynamic Features. (BG: Background, CI: Clean Ice, DCI: Debris Ice)

Feature	**BG vs. CI**	**BG vs. DCI**	**CI vs. DCI**
Elevation	**0.359**	0.227	**0.579**
Slope	**0.328**	**0.703**	**0.427**
Velocity	0.321	**0.393**	0.137
Velocity X	0.147	0.278	0.143
Velocity Y	0.212	0.277	0.183
Flow Accumulation	0.069	0.283	0.256
TPI	0.124	0.215	0.113
Roughness	**0.328**	**0.709**	**0.435**
Plan Curvature	0.057	0.112	0.077

A.2 Ablation Study

This study was developed to isolate each element of all proposed strategies to assess their independent contributions to the final models. Due to the models all sharing architecture with the SOTA baseline [25], the small percentage of labeled pixels, the randomness of training, and potentially mislabeling limitations, these runs shouldn't be used to fully evaluate the merit of each physics-guided strategy.

Definitions:

- **B0-B7:** Multispectral Landsat-7 Satellite Channels (raw spectral bands)
- **DEM:** Elevation and slope derived from the Digital Elevation Model
- **Spec:** Spectral Indices including NDVI (Normalized Difference Vegetation Index), NDWI (Normalized Difference Water Index), and NDSI (Normalized Difference Snow Index)
- **Phys:** Physics-derived channels from Chapter 3: Flow Accumulation, TPI (Topographic Position Index), Roughness, and Plan Curvature
- **Vel:** Velocity channels (magnitude and directional components)
- **Vel Loss:** Velocity loss regularization term
- **IoU:** Intersection over Union - measures overlap between predicted and actual glacier areas
- **Prec:** Precision - proportion of correctly identified glacier pixels out of all pixels predicted as glacier
- **Rec:** Recall - proportion of actual glacier pixels that were correctly identified

A.2.1 Complete Feature Ablation

Table A.3: Complete feature ablation study showing the impact of different input components on model performance.

B0-B7	DEM	Spec	Phys	Vel	Vel Loss	DCI IoU	DCI Prec	DCI Rec	CI IoU	CI Prec	CI Rec
✓	✓	✓	✓	✓	✓	**46.07**	**91.53**	**56.16**	**70.58**	**80.79**	**95.97**
✓	✓	-	✓	-	-	32.41	70.26	37.56	70.58	80.79	84.81
✓	-	-	-	✓	✓	16.82	91.53	17.09	51.13	52.25	95.97
✓	-	-	-	✓	-	30.85	57.28	40.07	67.78	75.20	87.28
✓	-	✓	-	-	-	21.62	72.19	23.59	64.54	69.56	89.95
✓	✓	-	-	-	-	37.03	62.71	47.48	67.89	75.26	87.40
✓	-	-	-	-	-	31.93	70.69	36.80	70.22	80.77	84.32

A.2.2 Individual Physics Components

Table A.4: Individual physics component ablation study isolating the contribution of each physics-derived feature.

B0-B7	Flow Acc	TPI	Rough	Plan Curv	DCI IoU	DCI Prec	DCI Rec	CI IoU	CI Prec	CI Rec
✓	✓	✓	✓	✓	**32.41**	70.26	**37.56**	**70.58**	**80.79**	84.81
✓	✓	-	-	-	14.56	24.90	25.96	67.33	78.68	82.36
✓	-	✓	-	-	26.08	**81.02**	27.78	63.60	69.63	88.02
✓	-	-	-	✓	12.91	39.63	16.07	63.48	68.80	**89.16**

A.2.3 Velocity Components Analysis

Table A.5: Velocity component analysis comparing the impact of velocity channels versus velocity loss regularization.

B0-B7	Vel	Vel Loss	DCI IoU	DCI Prec	DCI Rec	CI IoU	CI Prec	CI Rec
✓	✓	✓	**41.91**	66.40	**53.20**	65.85	72.36	**87.98**
✓	✓	-	32.40	70.25	37.56	**70.78**	**82.27**	83.52
✓	-	-	25.45	**76.74**	27.57	70.22	80.77	84.32

Vita

Jose Guadalupe Perez Zamora graduated from Mission Early College High School, El Paso, Texas, in 2015. He then entered The University of Texas at El Paso to pursue a Bachelor of Science in Computer Science after receiving the BUILDing Scholars research scholarship. That scholarship allowed him to co-author a publication in the Frontiers in Systems Neuroscience journal and graduate with Cum Laude honors in 2018. He is now pursuing a Ph.D. in Computer Science at The University of Texas at El Paso under his research advisor Dr. Olac Fuentes at the Vision and Learning Laboratory. He is particularly interested in the application of neural networks to other disciplines and has worked with faculty and engineers from other disciplines such as neuroscience, mechanical engineering, and geology to bring the models and problem-solving capabilities of AI to other fields, especially those with limited data available.

He spent one and a half years as an intern at the Johns Hopkins University Applied Physics Lab during his Ph.D., working under the guidance of Dr. Pedro Rodriguez, Dr. Luis Puche, Beatrice Garcia, Kevin Chiou, and Malik Jackson to work on several team projects involving the deployment of AI models (MLOps) using Prefect, and the training and evaluation of models using PyTorch. These included different CNN models for multi-spectral satellite image classification, and a triplet-loss based model for real-time one-shot image classification.

During his time as a Ph.D. student, he worked briefly as a Research Assistant helping with a research project in the field of empirical game theory with Dr. Cristopher Kiekintveld, for which he co-authored a publication. He spent many semesters as a Teaching Assistant, providing guidance and tutoring to students in Data Structures, Machine Learning, and Deep Learning courses at both the undergraduate and graduate level. He was a recipient of the Google-CAHSI Dissertation Award in 2023, and has also received several travel awards to present for the Biomedical Engineering Society (BMES), Society for Neuroscience (SFN), and Computing Alliance of Hispanic Serving Institutions (CAHSI) conferences. His work has been published at the Frontiers in Systems Neuroscience journal, SPIE Defense + Commercial Sensing Symposium, and the ASME Fluids Engineering Division.