

**UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA**

The prediction of condensation flow patterns by using artificial intelligence (AI) techniques

by

Michael Kevin Seal

Submitted in partial fulfilment of the requirements for the degree

MASTER OF ENGINEERING (MEng)

Department of Mechanical and Aeronautical Engineering

University of Pretoria

February 2021

Supervisors: Dr M. Mehrabi and Prof J.P. Meyer

Abstract

Title: The prediction of condensation flow patterns by using artificial intelligence techniques

Supervisors: Dr M. Mehrabi and Prof J.P. Meyer

Department: Mechanical and Aeronautical Engineering

Degree: Master of Engineering (Mechanical Engineering)

Multiphase flow provides a solution to the high heat flux and precision required by modern-day gadgets and heat transfer devices as phase change processes make high heat transfer rates achievable at moderate temperature differences. An application of multiphase flow commonly used in industry is the condensation of refrigerants in inclined tubes. The identification of two-phase flow patterns, or flow regimes, is fundamental to the successful design and subsequent optimisation given that the heat transfer efficiency and pressure gradient are dependent on the flow structure of the working fluid.

This study showed that with visualisation data and artificial neural networks (ANN), a machine could learn, and subsequently classify the separate flow patterns of condensation of R-134a refrigerant in inclined smooth tubes with more than 98% accuracy. The study considered 10 classes of flow pattern images acquired from previous experimental works that cover a wide range of flow conditions and the full range of tube inclination angles. Two types of classifiers were considered, namely multilayer perceptron (MLP) and convolutional neural networks (CNN). Although not the focus of this study, the use of a principal component analysis (PCA) allowed feature dimensionality reduction, dataset visualisation, and decreased associated computational cost when used together with multilayer perceptron neural networks. The superior two-dimensional spatial learning capability of convolutional neural networks allowed improved image classification and generalisation performance across all 10 flow pattern classes. In both cases, the classification was done sufficiently fast to enable real-time implementation in two-phase flow systems. The analysis sequence led to the development of a predictive tool for the classification of multiphase flow patterns in inclined tubes, with the goal that the features learnt through visualisation would apply to a broad range of flow conditions, fluids, tube geometries and orientations, and would even generalise well to identify adiabatic and boiling two-phase flow patterns. The method was validated by the prediction of flow pattern images found in the existing literature.

Keywords: Condensation flow pattern, convolutional neural network, machine learning

Publication

Journal paper

M.K. Seal, S.M.A. Noori Rahim Abadi, M. Mehrabi, J.P. Meyer; Machine learning classification of in-tube condensation flow patterns using visualisation. Submitted to *International Journal of Multiphase Flow* in January 2021 (manuscript number: IJMF-D-20-00448).

Acknowledgements

I would like to thank the following people and organisations for their contribution to the completion of my master's degree:

- my supervisors, Dr M. Mehrabi and Prof J.P. Meyer, for their guidance offered during this research;
- the University of Pretoria, the Department of Mechanical and Aeronautical Engineering, and the National Research Foundation (NRF), for providing me with the opportunity, facilities, and necessary funding to complete this study;
- the departmental administration officers, Ms. T. Evans, Ms. I. Meyer, and Ms. S. Steenberg, for their professional assistance;
- my friends and family members, for the love, support, and encouragement showed towards me during the duration of this study.

Table of contents

Abstract.....	i
Publication	ii
Acknowledgements.....	iii
List of figures.....	vii
List of tables	ix
Nomenclature	xi
Chapter 1. Introduction	1
1.1 Background	1
1.2 Problem statement	2
1.3 Aim	2
1.4 Research objectives	2
1.5 Scope of study.....	3
1.6 Organisation of dissertation	3
Chapter 2. Literature review	4
2.1 Introduction	4
2.2 Condensation heat transfer	4
2.2.1 Modes of condensation	4
2.2.2 In-tube condensation.....	5
2.3 Two-phase flow terminology	5
2.3.1 Intensive properties	5
2.3.2 Non-dimensional parameters	6
2.4 Flow patterns	7
2.4.1 Description of flow patterns	7
2.4.2 Flow pattern detection methods	10
2.4.3 Two-phase flow pattern maps	11
2.5 Experimental work conducted at the University of Pretoria.....	12
2.5.1 Summary of experimental results.....	13
2.5.2 Experimental set-up.....	14
2.6 Artificial intelligence	15
2.7 Machine learning	16

2.7.1	Machine learning methods	16
2.7.2	Machine learning terminology.....	18
2.8	Artificial neural networks.....	19
2.8.1	Artificial neuron	19
2.8.2	Multilayer perceptron neural network	20
2.8.3	Activation functions	21
2.8.4	Training	23
2.8.5	Optimisation algorithms	26
2.8.6	Regularisation	28
2.9	Convolutional neural networks.....	30
2.9.1	Convolutional layers.....	30
2.9.2	Pooling layers	32
2.9.3	Training	33
2.9.4	Data augmentation	34
2.10	Research investigating artificial neural networks for flow pattern identification.....	34
2.11	Summary and conclusions	39
Chapter 3. Methodology		40
3.1	Introduction	40
3.2	Data preparation.....	40
3.2.1	Image acquisition	40
3.2.2	Image pre-processing.....	40
3.2.3	Dataset generation	42
3.3	Principal component analysis	44
3.3.1	Dimensionality reduction by PCA.....	44
3.3.2	Dataset visualisation	48
3.4	Deep learning models	49
3.4.1	Training methodology	49
3.4.2	Multilayer perceptron final model.....	51
3.4.3	Convolutional neural network final model	54
3.5	Conclusion.....	57
Chapter 4. Results and discussion.....		58
4.1	Introduction	58

4.2	Classifier performance criteria.....	58
4.3	Performance of the MLP.....	59
4.4	Performance of the CNN.....	60
4.5	Conclusion.....	66
Chapter 5. Development of an online predictive tool for in-tube multiphase flow patterns		67
5.1	Introduction	67
5.2	Training with image augmentation.....	67
5.3	Results.....	68
5.4	Model validation	69
5.5	Development of a predictive tool.....	70
5.6	Conclusion.....	70
Chapter 6. Conclusions and recommendations.....		72
6.1	Conclusions	72
6.2	Recommendations	73
References		74
Appendix A. Prediction of flow pattern images from the literature.....		A-1
A.1	Introduction	A-1
A.2	Prediction results of flow pattern images from the literature	A-1
A.3	Conclusion.....	A-23

List of figures

Figure 2-1: Schematic representation of the most commonly observed flow patterns in: (a) vertically orientated tubes, (b) horizontally orientated tubes 8

Figure 2-2: Schematic of the experimental set-up [8, 26] 15

Figure 2-3: Differences between the supervised machine learning tasks of: (a) classification, (b) regression 17

Figure 2-4: Structure of an artificial neuron 20

Figure 2-5: Architecture of a standard MLPNN 21

Figure 2-6: Plot of activation function (blue) and corresponding gradient (orange): (a) sigmoid, (b) hyperbolic tangent, (c) ReLu 21

Figure 2-7: Example of a dropout MLP: (a) standard MLP with two hidden layers, (b) MLP with dropout applied in hidden layers 29

Figure 2-8: Example of a convolution with a 3x3-sized kernel filter, stride length of 1, depth of 1 and single layer of padding. 31

Figure 2-9: Example of the max-pooling operation 33

Figure 3-1: Frequency of each of the considered flow pattern classes 42

Figure 3-2: (a) Principal component-wise dataset variance, (b) cumulative dataset variance 46

Figure 3-3: The image dataset projected to its first three principal components, viewed from: (a) the first and second components, (b) the first and third components, (c) the second and third components 49

Figure 3-4: (a) Process of training and evaluating the deep learning models, (b) cross-validation process used to fine-tune model hyperparameters, (c) comprehensive flow chart of the model training process 51

Figure 3-5: MLP final model architecture and hyperparameters 54

Figure 3-6: CNN final model architecture and hyperparameters 57

Figure 4-1: MLPNN mean learning curves as a function of the number of epochs trained: (a) Accuracy, (b) Cross-entropy loss..... 59

Figure 4-2: Normalised confusion matrix for the MLP results..... 60

Figure 4-3: CNN mean learning curves as a function of the number of epochs trained: (a) Accuracy, (b) Cross-entropy loss..... 60

Figure 4-4: Normalised confusion matrix for the CNN results..... 61

Figure 4-5: Comparison between the MLPNN and CNN mean test set F-scores on the 10 flow pattern classes 62

Figure 4-6: Total prediction time per image 62

Figure 5-1: Mean learning curves for the CNN trained with augmented images: (a) Accuracy, (b) Cross-entropy loss..... 68

Figure 5-2: Normalised confusion matrix for the CNN trained on augmented images..... 69

Figure A-1: Three images depicting Bubbly flow for different gas superficial velocities, taken from Bhagwat and Ghajar [15] A-1

Figure A-2: Five bubbly flow pattern images for differing liquid superficial velocities, taken from Bhagwat and Ghajar [15] A-2

Figure A-3: Flow pattern images observed in gas-liquid two-phase flow for: (a) vertical upward, (b) horizontal, (c) upward inclined flow, taken from Bhagwat and Ghajar [9] A-4

Figure A-4: Falling film flow pattern observed for vertical downward flow, taken from Bhagwat and Ghajar [10] A-7

Figure A-5: Images of bubbly flow in horizontal, downward vertical and downward inclined flow, taken from Bhagwat and Ghajar [10] A-8

Figure A-6: Flow patterns during upward vertical flow in a 2.01 mm internal diameter tube, taken from Chen et al. [19] A-9

Figure A-7: Flow patterns during upward vertical flow in a 2.88 mm internal diameter tube, taken from Chen et al. [19] A-10

Figure A-8: Flow patterns during upward vertical flow in a 4.26 mm internal diameter tube, taken from Chen et al. [19] A-12

Figure A-9: Typical flow patterns seen in horizontal two-phase flow, taken from Ghajar and Tang [12].. A-13

Figure A-10: Taitel and Dukler flow pattern map with representative flow pattern photographs, taken from Ghajar and Tang [12]..... A-15

Figure A-11: Depiction of flow patterns seen during two-phase refrigerant flow in horizontal tubes, taken from Roman et al. [13] A-17

Figure A-12: Observed flow patterns from the work of Xing et al. [22] A-19

Figure A-13: Flow pattern images during upward vertical flow, taken from Gao et al. [121] A-21

Figure A-14: Flow patterns depicting the boiling of nitrogen in a horizontal tube, taken from Ohira et al. [14] A-22

List of tables

Table 2-1: Summary of recent works investigating artificial neural networks trained by backpropagation to predict in-tube multiphase flow patterns	36
Table 3-1: Representative images of each of the flow pattern classes	43
Table 3-2: Randomly selected flow pattern images from each class weighted by the first, second, third, 100th, and 500th most significant eigenfigures	47
Table 3-3: Process of image reconstruction by including subsequently more eigenfigures for three randomly selected flow pattern images	48
Table 3-4: Model hyperparameters considered for the MLP cross-validation grid search process.....	53
Table 3-5: Model hyperparameters considered for the CNN cross-validation grid search process.....	55
Table 4-1: Mean performance metrics of the MLP on the test dataset	60
Table 4-2: Mean performance metrics of the CNN on the test dataset	61
Table 4-3: Average execution time per prediction on the test set	62
Table 4-4: Examples of CNN predictions of selected images from the test dataset	64
Table 4-5: Examples of commonly misclassified slug and elongated bubble flow pattern images.....	65
Table 5-1: Examples of image augmentation applied to a Churn flow pattern image	68
Table 5-2: The mean performance metrics for the CNN trained with augmented images	69
Table A-1: Model results of the images depicting bubbly flow for differing gas superficial velocities, taken from Bhagwat and Ghajar [15]	A-2
Table A-2: Model results of the images depicting bubbly flow for differing liquid superficial velocities, taken from Bhagwat and Ghajar [15]	A-3
Table A-3: Model results of the images observed in gas-liquid two-phase flow for vertical upward, horizontal and upward inclined flow, taken from Bhagwat and Ghajar [9]	A-5
Table A-4: Model results of the flow pattern images depicting falling film flow, taken from Bhagwat and Ghajar [10]	A-7
Table A-5: Model results of the images of Bubbly flow for horizontal, downward vertical and downward inclined flow, taken from Bhagwat and Ghajar [10].....	A-8
Table A-6: Model results of the images for upward vertical flow in a 2.01 mm internal diameter tube, taken from Chen et al. [19].....	A-9
Table A-7: Model results of the images for upward vertical flow in a 2.88 mm internal diameter tube, taken from Chen et al. [19]	A-11
Table A-8: Model results of the images for upward vertical flow in a 4.26 mm internal diameter tube, taken from Chen et al. [19].....	A-12
Table A-9: Model results of the typical flow patterns seen in horizontal two-phase flow, taken from Ghajar and Tang [12]	A-13

Table A-10: Model results of the images showing representative flow pattern photographs for each flow pattern in the Taitel and Dukler flow pattern map, taken from Ghajar and Tang [12] A-15

Table A-11: Model results of the images of two-phase refrigerant flow in horizontal tubes, taken from Roman et al. [13]..... A-18

Table A-12: Model results of the observed flow pattern images from Xing et al. [22] A-20

Table A-13: Model results of flow pattern images during upward vertical flow, taken from Gao et al. [121] A-21

Table A-14: Model results of images depicting the boiling of nitrogen in a horizontal tube, taken from Ohira et al. [14] A-23

Nomenclature

Mathematical conventions

x – scalar (normal font)

\mathbf{x} – vector (bold font)

\mathbf{X} – matrix (bold and capitalised font)

\hat{x} - estimated value

Symbols

a	value after activation function
A	area [m^2]
b	bias value
c	contrast
C	convolution matrix
D	tube internal diameter [m]
F	kernel filter matrix
F	f-score performance metric
Fr	Froude number
g	gravitational constant [m/s^2]
G	mass flux [$kg/m^2 \cdot s$]
h	convective heat transfer coefficient [$W/m^2 \cdot K$]
I	input image matrix
J	cost function
k	thermal conductivity [$W/m \cdot K$]
K	number of classes
l	luminance
L	loss function/length [m]
m	mass [kg]
\dot{m}	mass flow rate [kg/s]

N	number of training samples
Nu	Nusselt number
O	output matrix size
P	padding thickness
P	precision performance metric
Pr	Prandtl number
Q	depth
R	recall performance metric
Re	Reynolds number
s	structure
S	stride length
u	fluid velocity
V	eigenvector
w	neural network weight
W	input image size
x	vapour quality/input value
y	output value
z	weighted summation value

Greek symbols

β	exponential decay rate
ε	void fraction
Σ	covariance matrix
η	learning rate
ρ	density [kg/m^3]
δ	error value
μ	mean value/dynamic viscosity [$kg/m \cdot s$]
λ	regularisation constant

σ	covariance
σ^2	variance
θ	parameter/tube inclination angle [°]
Ω	matrix of eigenvectors
Δ	change in

Abbreviations

Abbr	abbreviation
Adam	adaptive moment estimation
AH	annular-horizontal (flow pattern class)
AI	artificial intelligence
ANN	artificial neural network
API	application programming interface
AW	annular-wavy (flow pattern class)
AV	annular-vertical (flow pattern class)
B	bubbly (flow pattern class)
BUA	broadband attenuation ultrasound
C	churn (flow pattern class)
CHMM	continuous hidden Markov model
CNN	convolutional neural network
CPDF	cumulative probability density function
DWT	discrete wavelet transform
EB	elongated bubble (flow pattern class)
ECT	electrical capacitance tomography
ELU	exponential linear unit
EVV	electronic expansion valve
HTC	heat transfer coefficient
I	intermittent (flow pattern class)
MLP	multilayer perceptron
MLPNN	multilayer perceptron neural network
PC	principal component

PCA	principal component analysis
PSD	power spectral density
ReLU	rectified linear unit
SOM	self organising map
S	slug (flow pattern class)
SS	stratified-smooth (flow pattern class)
SSIM	structural similarity index measure
SVM	support vector machine
SW	stratified-wavy (flow pattern class)

Subscripts

a	activation
l	liquid
R	regularisation
v	vapour
opt	optimum
p	pixels
t	training interval

Superscripts

l	layer
-----	-------

Chapter 1. Introduction

1.1 Background

The rapid development in the electronics and power generation industries has placed increasing pressure on thermal management systems because modern-day gadgets and heat transfer devices require high heat flux and precision, while occupying the minimum possible space. Multiphase flow provides a solution because high heat transfer rates can be achieved at moderate temperature differences through the process of phase change. An application of multiphase flow is in-tube condensation of refrigerants in water and air-cooled condensers, which find their place extensively in the air-conditioning, refrigeration, automotive, and process industries.

It has been well documented that the heat transfer coefficients (HTC) and pressure drops occurring within two-phase flow systems are closely related to the local flow structure of the working fluid, which transitions as the fluid condenses along the length of the tube. Local flow patterns occur as the result of a balance between gravitational, shear, and capillary forces; and are influenced by the mass flow rate, vapour quality, tube orientation (inclination), and geometry, as well as the physical properties of the two phases [1]. For instance, annular flow, which is characterised by a phase interface separating a thin liquid film around the perimeter of the tube from a vapour flow in the core region, is the result of flow dominated by shear forces and is usually the result of a combination of high mass flux and vapour quality, as well as being more common in tubes with vertical orientation. On the contrary, lower mass fluxes in horizontally orientated tubes lead to stratified flows, characterised by the collection of condensate at the bottom of the tube as the denser liquid phase is driven down the tube walls by gravitational forces. Due to the different physical characteristics describing these flow patterns, modern developed models, and correlations often only apply to a specific flow pattern [2]. In fact, predicting the transition between flow patterns is analogous to predicting the transition from laminar to turbulent flow in single-phase flow systems [3]. To develop universally accepted two-phase flow models and subsequently optimise the design of heat transfer equipment for condensation applications, a reliable method to determine two-phase flow patterns is required. Hence, a great amount of effort has been made by various researchers for this purpose.

The traditional method of predicting flow patterns is the mapping of the transition boundaries between various flow patterns based on the non-dimensionalisation of experimental results and mechanistic models of the transition criteria. However, even the most advanced flow pattern maps depend on the flow conditions used for their development, cover a limited experimental range, and are subject to which of the numerous flow pattern identification criteria were used for their development [3, 4]. Additionally, most flow pattern maps are produced for adiabatic conditions in horizontal and vertical tube orientations. These maps have often been extrapolated or modified to include the effects of condensation heat transfer and inclination effects; however, with limited success. For these reasons, flow pattern maps are often difficult to interpret and to use.

Due to the advancement in computational power in recent years, artificial intelligence (AI), in particular, the use of deep learning models, has emerged as a popular method of solving complex modern-day

problems. AI has been extensively employed over the past three decades to predict two-phase flow patterns and to develop new flow pattern maps. However, upon review of these works, it is evident that the most focused on adiabatic and boiling two-phase flows in horizontal tubes and channels. Additionally, the existing works rely on well-established models, computer-generated data, and measured thermo-hydraulic parameters, such as pressure and temperature measurements, which can introduce bias and reduce generalisation capability compared with models trained on high dimensional noisy data, such as from visualisation.

1.2 Problem statement

Despite the substantial number of works investigating two-phase flow patterns, it is evident that no flow pattern map, mechanistic model, or other methods that can be used to predict the local flow patterns for in-tube condensation for a wide range of experimental conditions have yet been developed. Additionally, the lack of agreement between authors with regards to the flow pattern identification and description criteria means that even corresponding works are often difficult to interpret and to compare. Therefore, there is a great need for the development of new predictive tools for two-phase flow patterns such that there is a consensus on the successful prediction of the flow pattern. Despite the recent advancements in employing AI for flow pattern identification, it appears that no attempt has been made to create a flow pattern predictive tool based on AI methods trained on visualisation data.

1.3 Aim

This study aimed to train a deep learning model that could successfully predict the flow patterns of in-tube condensation flows based on visualisation data. In the study, both multilayer perceptron (MLP) and convolutional neural networks (CNN) were considered. The performance of the resulting models was evaluated on the ability to predict the flow pattern images of samples acquired from both the University of Pretoria and those found in the literature.

1.4 Research objectives

The research objectives of this investigation were as follows:

- acquire an extensive database of condensation flow pattern images that cover a wide range of experimental conditions and tube orientations;
- pre-process and prepare the image data for subsequent analysis using AI techniques;
- explore and gain insight into the structure of the image data with the use of a principal component analysis (PCA). Also, use the PCA to extract features applicable to the training process with a MLP;
- optimise, train, and compare the performance of the MLP and CNN for the image classification task;
- adjust the training methodology such that the trained model generalises well to predict flow pattern images acquired from both the existing experimental set-up and those which are found in the literature.

1.5 Scope of study

The study investigated the use of artificial neural networks (ANN) trained on visualisation data for the task of in-tube condensation flow pattern classification.

An extensive database of flow pattern images was obtained from previous experimental works investigating the condensation of R-134a refrigerant in inclined smooth tubes at the University of Pretoria's thermoflow laboratory. The image database was pre-processed and prepared to create a dataset applicable to the subsequent analysis using deep learning models. As a result, a dataset containing 3 961 flow pattern images across 10 classes was developed.

A PCA was conducted for dimensionality reduction and feature extraction applicable to the use of a MLP. The resulting analysis allowed insight into the structure of the dataset, and additionally, allowed visualisation of the dataset as the transformed data was projected to a two-dimensional space in a meaningful way.

The deep learning models' hyperparameters were optimised using a five-fold cross-validation and grid search process, and the resulting models' classification performance was evaluated and compared. Finally, with the use of a developed image augmentation strategy, the CNN training process was made more robust such that the learnt features were more applicable to flow pattern images that may be acquired by new experimental set-ups and those already seen in the literature. It was with this robust training process that a predictive tool for in-tube multiphase flow patterns was developed.

1.6 Organisation of dissertation

The remainder of the dissertation consists of five chapters:

The following chapter is the literature review, which provides a comprehensive analysis of condensation flow patterns and the theory of the AI techniques explored in this study. This chapter also reviews the existing flow pattern maps and AI techniques used for flow pattern classification reported in the literature.

Chapter 3 describes the methodology followed in the investigation. This includes the data preparation, the use of a PCA for dimensionality reduction and feature extraction, and the training methodology followed to fine-tune the models' hyperparameters. The chapter includes a complete description of the deep learning models considered in this investigation.

Chapter 4 summarises and compares the results of the two deep learning models on the test dataset.

Chapter 5 describes the process of retraining the final CNN model with an augmented version of the training dataset to develop a classifier that generalises well to images acquired from sources external to the University of Pretoria.

Finally, Chapter 6 concludes the previous chapters by providing a summary of the investigation and its outcomes. The chapter also provides recommendations for future research and investigations.

Chapter 2. Literature review

2.1 Introduction

The objective of this chapter is to give a comprehensive review of condensation flow patterns and the theory of the AI techniques explored in the literature. Firstly, the fundamentals of condensation heat transfer and two-phase flow terminology are discussed. This is followed by an overview of two-phase flow patterns, which includes a description of the physical and visual characteristics defining the flow patterns, the existing flow pattern detection methods, as well as the theory and history of two-phase flow pattern maps. Next, the experimental work investigating condensation in inclined smooth tubes conducted at the University of Pretoria's thermoflow laboratory is summarised. This chapter then reviews AI, and more specifically, the theory of machine learning, deep learning, and ANNs. Lastly, literature reporting on employing ANNs for in-tube two-phase flow pattern identification is reviewed, to identify the main findings and the gaps in the literature.

2.2 Condensation heat transfer

Condensation refers to the change of the physical state of matter from the vapour, or gaseous phase, to the liquid phase, and occurs when the temperature of the vapour phase is reduced below the saturation temperature. Condensation usually occurs when the vapour comes into contact with a surface where the temperature is below the saturation temperature, but may also occur on the free surface of a liquid or in the presence of another vapour phase if the temperature of the liquid or vapour to which it is in contact with is below the saturation temperature. Condensation is a form of convection heat transfer due to the involvement of fluid motion. However, it differs from other forms of convection heat transfer because it depends on the latent heat of vaporisation of the fluid. Heat transfer rates associated with condensation are much greater than those typical with single-phase processes as the large latent heat of vaporisation is absorbed during the change of phase from a vapour to a liquid. Therefore, it is preferred in industrial applications because high heat transfer rates can be achieved with moderate temperature differences [5].

2.2.1 Modes of condensation

Condensation occurs in two typical modes: film and dropwise condensation. During film condensation, condensate wets the cooling surface and forms a liquid film, which grows in thickness as it falls under the influence of gravity or flows along the surface due to an external force. However, the increasing thickness of the liquid film creates resistance to heat transfer because the latent heat of vaporisation must pass through this liquid film before reaching the surface on the other side. On the other hand, dropwise condensation does not suffer from this resistance because instead of the continuous film, numerous droplets of varying diameter move along the surface and allow continuous contact between the surface and the vapour phase. Although higher heat transfer rates are typical with dropwise condensation, maintaining a non-wetting surface in industrial applications is both expensive and not realisable over long periods. For this reason, film condensation is assumed in the design of heat transfer equipment.

2.2.2 In-tube condensation

Most condensation applications encountered in industry involve condensation on the inner surface of tubes, which complicates the heat transfer analysis. Depending on the flow conditions, the distribution and interaction of the liquid and vapour phases may take on many different forms, referred to as flow patterns, or flow regimes. The successful design of heat transfer equipment is highly dependent on the successful identification of the local flow patterns occurring along the length of the tube.

2.3 Two-phase flow terminology

Although the focus of this study is not specifically two-phase flows but rather the classification of two-phase flow patterns, certain terminology is critical to the understanding and description of two-phase flow patterns, and therefore, are defined for completeness. Additionally, as discussed in Chapter 6, the extension of the work done in this study involves the use of the defined properties and parameters because they appear in most heat transfer and pressure drop correlations, where they are used for the development of flow pattern maps, and as the input features to many of the existing methods employing AI to predict flow patterns.

2.3.1 Intensive properties

Intensive properties are bulk properties or local physical properties of a system that are independent of size or mass, and therefore, can be used to compare two-phase systems with differing geometries, flow conditions, and fluids. For this reason, the transition and classification criteria of different flow patterns are usually described in terms of these properties.

2.3.1.1 Vapour quality

In a saturated mixture, vapour quality (x) is defined as the mass fraction of the vapour phase to the total mass fraction, which includes both the liquid and vapour phases. Mathematically, the vapour quality is written as:

$$x = \frac{\dot{m}_v}{\dot{m}_v + \dot{m}_l} \quad (2.1)$$

where \dot{m}_v and \dot{m}_l refer to the mass flow rate of the vapour and liquid phases, respectively. The vapour quality decreases during condensation flow as the vapour phase condenses to the liquid phase.

2.3.1.2 Void fraction

Void fraction (ε) is defined as the fraction of the tube cross-sectional area that is occupied by the vapour phase, mathematically written as:

$$\varepsilon = \frac{A_v}{A_v + A_l} \quad (2.2)$$

where A_v and A_l refer to the tube cross-sectional area that is occupied by the vapour and liquid phases, respectively. Similar to the vapour quality, the void fraction also decreases as the vapour phase condenses

during two-phase flows. However, considering two systems with identical vapour qualities, the void fraction may be considerably different depending on the system pressure, because the vapour phase density is usually significantly lower than that of the liquid phase.

2.3.1.3 Mass flux

Mass flux (G) is defined as the ratio of the total mass flow rate to the tube cross-sectional area, mathematically written as:

$$G = \frac{\dot{m}}{A} \quad (2.3)$$

where \dot{m} and A refer to the total mass flow rate and tube cross-sectional area, respectively. Mass flux can also be described as the mean flow velocity multiplied by the mean density of the two-phase mixture.

2.3.2 Non-dimensional parameters

The number of variables that define a system can be reduced by combining the variables into dimensionless groups, which are usually derived as the ratio between physical quantities. The resulting non-dimensional parameters allow for a reduced amount of experimental data to make correlations of physical phenomena to scalable systems. Therefore, non-dimensional numbers are often used for the development of flow pattern maps and as input to existing AI methods to predict the transition between two-phase flow patterns.

2.3.2.1 Reynolds number

The Reynolds number [6] is defined as the ratio of inertial forces to viscous forces in the fluid and is used to predict the transition from laminar to turbulent flow. Mathematically, it is written as:

$$Re = \frac{\rho u D}{\mu} \quad (2.4)$$

where ρ is the fluid density, u is the fluid velocity, μ is the fluid dynamic viscosity, and D is the tube internal diameter. According to Çengel and Ghajar [5], the flow in a tube is laminar for $Re < 2300$, fully turbulent for $Re > 10000$, and transitional in-between.

2.3.2.2 Prandtl number

The Prandtl number is defined as the ratio of the momentum diffusion rate to the thermal diffusion rate and is an intrinsic property of the fluid. The Prandtl number is often used to characterise heat transfer in fluids. For instance, low Prandtl numbers are associated with free-flowing fluids with high thermal conductivity, which are a good choice for heat transfer. Mathematically, it is written as:

$$Pr = \frac{\mu C_p}{k} \quad (2.5)$$

where μ is the fluid dynamic viscosity, C_p is the specific heat of the fluid, and k is the fluid thermal conductivity.

2.3.2.3 Nusselt number

The Nusselt number is defined as the ratio of the convection heat transfer rate to the conduction heat transfer rate. Heat transfer through a liquid layer is by conduction when the fluid is stationary and by convection when the fluid is in motion. The Nusselt number represents the enhancement of heat transfer through a fluid layer due to convection relative to conduction. Mathematically, it is written as:

$$Nu = \frac{hD}{k} \quad (2.6)$$

where h is the convective HTC, D is the tube internal diameter, and k is the fluid thermal conductivity. For fully developed turbulent flows in smooth tubes, the Nusselt number is most accurately determined with the Dittus and Boetler [7] equation, Equation (2.7), which was presented as an improvement of the Coleburn equation.

$$Nu = 0.0023Re^{0.8}Pr^n, n = 0.3 \text{ for cooling} \quad (2.7)$$

2.3.2.4 Froude number

The Froude number is defined as the ratio of inertial forces to gravitational forces, mathematically written as:

$$Fr = \frac{u}{\sqrt{gD}} \quad (2.8)$$

Where u is the fluid velocity, g is the gravitational acceleration constant, and D is the tube internal diameter.

2.4 Flow patterns

The local distribution and interaction of the liquid and vapour phases within the tube form distinctive geometric flow structures referred to as flow patterns, or flow regimes. The flow patterns are led by different physical phenomena, or more specifically, the balance between gravitational, shear, and capillary forces [8]; and depend on the mass flux, vapour quality, pressure, heat flux, tube orientation (inclination) and geometry, as well as the physical properties of the two phases [1].

2.4.1 Description of flow patterns

There is considerable disagreement in the literature on the naming or categorising of flow patterns. Both the names and the transition criteria are not consistent between authors, which makes the presented data difficult to interpret and to use. Therefore, the classifying criteria used in this study are explained in detail. These criteria are based on the descriptions given by Thome [3] and Thome and Cioncolini [4].

Although this study considered the full range of tube orientations, the description of flow patterns presented in the literature is often described separately for vertical and horizontal tube orientations. For the most part, the same flow patterns are observed in both vertical and horizontal flows, however, except for stratification effects. In horizontal flows, buoyancy forces stratify the liquid phase towards the bottom

of the tube and the vapour phase to the top. However, these forces become negligible with high mass flux. Figure 2-1a schematically depicts the flow patterns most commonly observed in vertical and near-vertical tube orientations, including bubbly, slug, churn, and annular. Similarly, Figure 2-1b schematically depicts the flow patterns most commonly observed in horizontal and near-horizontal tube orientations, including annular, annular-wavy, bubbly, elongated bubbles, intermittent, slug, stratified-smooth, and stratified-wavy. A description of the characteristics of each flow pattern follows.

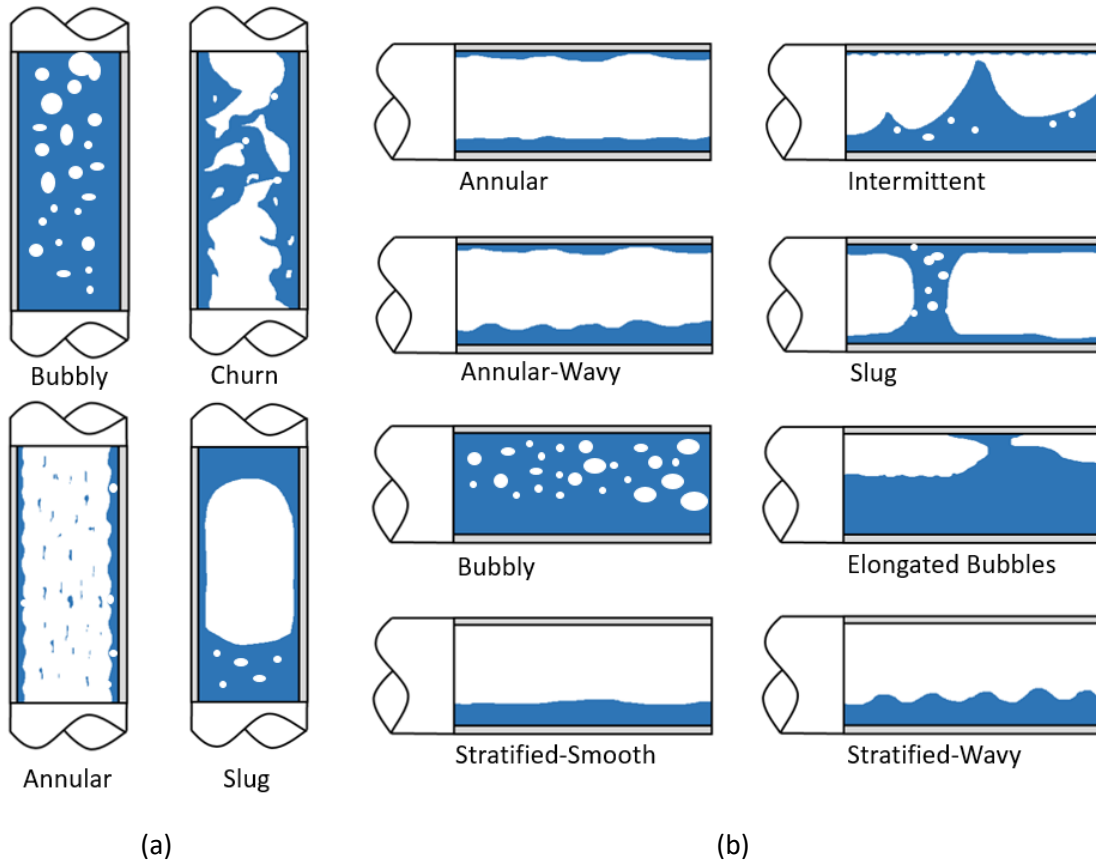


Figure 2-1: Schematic representation of the most commonly observed flow patterns in: (a) vertically orientated tubes, (b) horizontally orientated tubes

2.4.1.1 Bubbly flow pattern

Bubbly (B) flow describes many vapour bubbles with a notably smaller diameter than that of the tube flowing within the continuous liquid phase. This flow pattern is observed for all tube orientations, with the bubbles congregating near the top of the tube due to buoyancy forces for tubes with near-horizontal orientation and with lower mass fluxes [3, 4, 9, 10].

2.4.1.2 Annular flow pattern

The annular flow pattern is described by a continuous annular film of liquid which flows at the internal perimeter of the tube, and which surrounds a central core of vapour which flows at a much higher velocity. The annular flow pattern is dominated by shear forces and is observed in tubes of all orientations at

combinations of high mass flux and high vapour quality. However, distinct differences are observed when comparing tubes of horizontal and vertical orientations. In horizontal flows, the liquid film is often thicker at the bottom of the tube due to gravitational forces, unless the mass flux is increased significantly for which the thickness of the liquid ring becomes uniform [3, 4]. In vertical flows, liquid droplets are usually seen entrained in the vapour core, and vapour bubbles are sometimes seen within the liquid film. Annular flow in vertical tubes containing periodic clouds or wisps of liquid within the vapour core is often referred to as “wispy annular” flow. To distinguish between these differences, this study refers to the flow patterns of annular-horizontal (AH) and annular-vertical (AV). The annular flow pattern is particularly stable and is accompanied by very high HTC, and therefore, is often the most desirable flow pattern for two-phase pipe flows.

2.4.1.3 Annular-wavy flow pattern

The high velocity of the vapour phase flowing in the core region during annular flows may result in interfacial waves on the liquid surface that grow visibly in amplitude, leading to the annular-wavy (AW) flow pattern. Annular-wavy flow may also form as the result of an increasing liquid film thickness at the top of the tube during stratified-wavy flows. In the description of diabatic two-phase flows, the annular-wavy flow pattern is not commonly mentioned because the increased vapour velocity required to form the interfacial waves usually leads to the top surface of the tube becoming dry. However, with condensing flows, the constant phase change and formation of condensate at the perimeter of the tube allow continuity of the liquid film at the top surface [11].

2.4.1.4 Stratified-smooth flow pattern

Stratified-smooth (SS) flow is observed at low mass fluxes in horizontal and near-horizontal tube orientations. Buoyancy forces are completely dominant in this regime, leading to the liquid phase travelling along the bottom of the tube and the vapour phase at the top. The phase interface is smooth without significant waves and usually more than half of the perimeter at the top of the tube is dry or wetted only by the liquid condensate, which quickly flows to the bottom of the tube [4]. This flow pattern is commonly referred to as “stratified” only in the literature [3, 4, 9, 12, 13] but is termed stratified-smooth in this study to distinguish it from stratified-wavy flow.

2.4.1.5 Stratified-wavy flow pattern

The stratified-smooth flow transitions to stratified-wavy (SW) when the gas velocity is increased together with the interfacial shear on the liquid surface leads to the formation of waves, which travel in the direction of the flow and wrap around the perimeter of the tube. The wave amplitudes are notable and depend on the relative velocity between the two phases, but do not reach the top of the tube [4]. A thin liquid film may be seen at the top surface of the tube due to the condensing of the vapour phase.

2.4.1.6 Intermittent flow pattern

With further increased gas velocity, these waves increase in size and intermittently reach and wet the top surface of the tube. This flow pattern is referred to as intermittent flow (I). Waves with smaller amplitudes not reaching the top surface occur between the larger waves, and the rolling motion of the waves often

leads to vapour bubbles becoming entrained within the larger waves [3, 4]. Intermittent flow is also often described as the result of the combination of the elongated bubbles and slug flow regimes. However, for this study, the intermittent flow pattern occurs when the intermittent waves show a chaotic or turbulent nature and when neither the elongated bubbles nor the slug flow patterns are obvious.

2.4.1.7 Elongated bubbles flow pattern

The elongated bubbles (EB) flow pattern, commonly referred to as plug flow in the literature [3, 4, 12-14], is described by the motion of large elongated bubbles that flow along the top of the tube, and therefore, are observed in tubes with a near-horizontal orientation. The diameter of the elongated bubbles is significantly smaller than the diameter of the tube such that the liquid phase is continuous along the bottom of the tube [4].

2.4.1.8 Slug flow pattern

The slug (S) flow pattern occurs in both vertical and horizontal tube orientations. For vertical flows, increased heat and mass fluxes during the bubbly flow regime lead to an increase in void fraction and the close proximity of bubbles, which coalesce to form distinct slugs of vapour with the diameter nearing that of the tube. Bhagwat and Swanand [15] describe the slugs as cylindrical and “bullet-shaped” vapour bubbles, which due to their high velocity, push the liquid in front of them towards the walls creating a film on the tube surface. The large slugs are also commonly referred to as Taylor bubbles in the literature [3, 15, 16]. Smaller bubbles are often seen remaining from the prior bubbly flow, or from vapour that has broken off from the tails of these slugs. In horizontal flows, slugs form when increased gas velocity leads to elongated bubbles growing in diameter. Although the formation of these vapour slugs follows a very different process, the slug flow patterns in both vertical and horizontal tubes are visually similar. Hence, they are grouped into the same class for this study.

2.4.1.9 Churn flow pattern

During the transition between stable slug flow and stable annular flow in vertical tube orientations, collapsing slugs in the vapour core lead to a chaotic fluctuating flow pattern named churn (C) flow. The instability is the result of gravitational and shear forces which act in opposite directions on the thin liquid film between the tube perimeter and the vapour slugs [4]. Chalgeri and Jeong [16] also describe the transition from slug to churn flow as occurring when the nose of a Taylor bubble, or vapour slug, touches the tail of the preceding Taylor bubble. This leads to instability as the vapour slugs are constantly created and destroyed. Churn flow is generally avoided in two-phase systems because of the flow and pressure oscillations that it induces, which may have destructive consequences on piping systems [3].

2.4.2 Flow pattern detection methods

The ability to successfully classify flow patterns depends on determining the transition criteria based on experimental evidence [17]. The experimental techniques available to identify flow patterns are subdivided by Rouhani and Sohal [18] into two principally different approaches. The first includes methods of direct observation, for which visualisation is the most common. The second includes indirect determining methods based on the statistical analysis of fluctuations of measured quantities such as

pressures and void fractions, which are analysed according to certain mathematical models based on the calculation of power spectral density and probability density functions.

Visualisation has traditionally been the most widely employed method [8-10, 12, 13, 15, 19-38] due to its simplicity and inexpensiveness to implement, and is also the method used in this study. In these types of experimental set-ups, the flow is visualised through a transparent section, and the flow patterns are subsequently classified based on the observer's personal judgment. The subjective interpretation of the flow is the major disadvantage of this method because the classification of the flow is conflicted between authors based on the numerous flow pattern identification criteria. In an extensive review paper, Doretta et al. [17] list the major research groups which used the visualisation approach. They emphasise that although the numerous authors use differing technical visualisation solutions, fluids, and test conditions; they all use the same visual test section topology, which includes one or more transparent sections with the same internal diameter as the other tubes, joined smoothly in series, such that the flow is undisturbed. Although the classification of the flow may be subjective, the experimental procedure and analysis sequence followed are fairly consistent.

The indirect determining techniques are based on quantitative objective criteria and are often preferred over direct visualisation. Rouhani and Sohal [18] compare the statistical analysis of the fluctuating character of the flow with obtaining the "signature" of the flow patterns, which show distinctly identifiable trends. However, studies using these methods are less common in the literature because the highly accurate experimental facilities required are not always universally available and are more expensive when compared to the facilities that use the visualization technique [17]. Additionally, the validation of these studies is normally based on visualisation. Hence, the subjectivity of these approaches cannot be completely eliminated.

2.4.3 Two-phase flow pattern maps

It has been well documented that the HTC and pressure drops occurring within two-phase flow systems are closely related to the local flow structure of the working fluid, which transitions as the fluid condenses along the length of the tube. Reiterating that the local flow patterns occur as the result of a balance between gravitational, shear, and capillary forces, it follows that modern-developed heat transfer models for predicting in-tube condensation often only apply to a specific flow pattern [2]. Therefore, an accurate prediction of the local flow pattern is of vital importance to the designers of heat transfer equipment.

To identify the flow pattern in a tube based on the local flow conditions, it is common practice to use a flow pattern map [36, 39-50], which is a diagram displaying the transition boundaries between the various flow patterns. The boundaries are usually plotted on a log-log axis using dimensionless numbers to represent liquid and vapour velocities. Early flow pattern maps were developed by plotting flow pattern observations versus various two-phase parameters until the flow patterns could be separated by distinct zones. More sophisticated maps were later developed through the inclusion of non-dimensional parameters and mechanistic models of the transition phenomena; however, these maps still tend to be biased towards the fluids used for their development [3, 4].

The earliest widely quoted flow pattern maps for vertical upward flows were presented by Fair [39] and Hewitt and Roberts [40], while Baker [41], Mandhane et al. [42], and Taitel and Dukler [43] presented

maps for horizontal flow orientations. Although these flow pattern maps were developed for adiabatic two-phase flows of air and water, they are often extrapolated for the prediction of condensation two-phase flow patterns. A relationship exists between the flow patterns of adiabatic and condensation two-phase flows. However, diabatic flow pattern maps extrapolated for condensation flows show inconsistencies at higher heat flux ranges due to the condensate which forms around the entire perimeter of the tube, which is not the case during adiabatic two-phase flows. To include the effects of heat transfer, numerous modifications to the Taitel and Dukler [43] map have led to the maps of Steiner [44], Kattan et al. [45], and finally, El. Hajal and Thome [46], which is considered the most widely accepted model to predict the condensation of refrigerants in horizontal tubes. This agreement is found among several of the extensive review papers sufficiently summarising the work done in this field [1, 2, 51, 52]. Specific to R-134a refrigerant, Suliman et al. [36] modified the El. Hajal and Thome map to include an improved transition region between the stratified-wavy, and the annular and intermittent flow patterns.

Review papers by Lips and Meyer [47] and Cheng et al. [48] indicate that far fewer studies have investigated the effect of tube inclination on flow patterns. Although some maps can be extrapolated for slight inclinations, the maps of Barnea [49] and Crawford et al. [50] are the only two that deal with the whole range of inclination angles. However, these maps have only been validated with very few experimental conditions. The works of Mohseni and Akhavan-Behabadi [53], Xing et al. [22], and Lips and Meyer [8] show that flow patterns are strongly dependent on tube inclination, and all confirm that none of the existing flow pattern maps can capture the flow patterns of refrigerants in inclined tubes for the full range of inclination angles. Additionally, Lips and Meyer [47], Olivier et al. [29], Meyer et al. [30], and Ewim et al. [32] indicate that an optimal inclination angle that maximises the HTC exists in the range of -10° and -30° (downward flow). These works highlight the need for new predictive tools for flow pattern maps such that heat transfer can be optimised by simply inclining condenser tubes.

In summary, most flow pattern maps presented in the literature were developed based on visualisation studies. All of the models showed dependence on the experimental conditions, the choice of fluids, as well as being subject to the numerous flow pattern identification criteria [54].

2.5 Experimental work conducted at the University of Pretoria

Condensation of R-134a refrigerant in inclined smooth tubes has been the focus of many studies conducted at the thermoflow laboratories at the University of Pretoria's Department of Mechanical Engineering [8, 24-38]. The experimental works together create an extensive database investigating the effects of various parameters and a wide range of flow conditions; including mass fluxes ranging from 50 to $700 \text{ kg/m}^2\text{s}$, refrigerant saturation temperatures between 30°C and 50°C , mean vapour qualities ranging from 0.1 to 0.9, and the full range of tube inclination angles from vertical downwards (-90°) to vertical upwards ($+90^\circ$), with 0° representing the test condenser in a horizontal position.

These studies resulted in an extensive database of two-phase flow images, which were used in this study. This section summarises the experimental results of these studies, and describes the experimental set-up which was used to capture the flow pattern images.

2.5.1 Summary of experimental results

Lips and Meyer investigated the effect of tube inclination on the flow pattern and HTC [8], as well as on the pressure drop and void fraction [26], and studied the effect of gravity forces on the HTC and pressure drop [27] for mass fluxes ranging from 200 to 600 kg/m^2s , refrigerant saturation temperature of 40°C, inlet qualities ranging from 0.1 to 0.9, and the full range of inclination angles. Upon comparison with the flow pattern maps available in the literature, it was found that they did not predict the experimental data well. The flow patterns were found to be strongly dependent on the inclination angle, and therefore, gravitational forces for low mass fluxes or low vapour qualities; however, they remained annular for high mass fluxes ($G \geq 300 kg/m^2s$) and high vapour qualities ($x > 0.6 - 0.7$) regardless of the tube orientation.

Olivier et al. [29] investigated the effect of tube inclination on the void fraction and HTC for mass fluxes ranging from 100 to 400 kg/m^2s , refrigerant saturation temperature of 40°C, inlet qualities ranging from 0.1 to 0.9, and the full range of inclination angles. They found that the void fraction and HTC were also greatly affected by the inclination angle at combinations of low mass fluxes and low vapour qualities. Additionally, they found that void fractions and HTCs increased with downward inclinations with an optimum angle being in the range of -10° and -30°. Interestingly, they observed that at intermediate conditions for mass flux and vapour quality, the void fraction and HTC were independent of the inclination angle despite having observed various flow patterns.

Meyer et al. [30] investigated the effects of the saturation temperature and inclination angle on the HTC for mass fluxes ranging from 100 to 400 kg/m^2s , refrigerant saturation temperatures between 30°C and 50°C, inlet qualities ranging from 0.1 to 0.9, and the full range of inclination angles. The results showed that the HTC was strongly dependent on both the tube inclination angle and the saturation temperature. The increase in saturation temperature generally led to a decrease in HTC and increased the effect of the tube inclination. They too concluded that the inclination angle had a more profound effect on the resulting HTC at combinations of lower mass fluxes and lower vapour qualities and suggested that the optimum inclination angle was in the range of -15° to -30°.

Adelaja et al. [25] investigated the effect of saturation temperatures on the pressure drop in inclined tubes. The study explored mass fluxes ranging from 100 to 400 kg/m^2s , refrigerant saturation temperatures of 30°C, 40°C, and 50°C, inlet qualities ranging from 0.1 to 0.9, and the full range of inclination angles. The pressure drop was seen to be significantly affected by both the inclination angle and the saturation temperature, and had maximum and minimum values for upward flow and downward flow, respectively. The opposite was found for the frictional pressure drop, which had minimum and maximum values for downward and upward flow respectively.

Ewim and Meyer [31-34] investigated the pressure drop, HTC, and resulting flow patterns in horizontal and inclined smooth tubes in the low mass flux range of 50 to 200 kg/m^2s . The studies considered mean vapour qualities ranging from 0.1 to 0.9, a mean refrigerant saturation temperature of 40°C, while the temperature difference between the refrigerant and the tube wall varied from 1°C to 10°C. The temperature difference was found to have a negligible effect on the resulting pressure drop; however, the frictional pressure drop increased with increasing mass flux, temperature difference, and vapour

quality. The tube orientations resulting in the maximum and minimum values of the pressure drop and frictional pressure drop were in agreement with those of Adelaja et al. [25]. The tube inclination was found to significantly affect both the flow patterns and HTC, and the optimum inclination angle which maximised HTC was determined in the range of -15° to -30° , which was in agreement with the studies investigating higher mass fluxes. Additionally, it was found that the HTC was more sensitive to the temperature difference for downward flows than for upward flows. For vertical upward and downward flows, the HTC was almost temperature difference independent. Considering the horizontal tube orientation, the resulting flow patterns were predominantly stratified and stratified-wavy, and the HTC was found to decrease with increasing temperature difference.

2.5.2 Experimental set-up

Figure 2-2 is a schematic of the experimental set-up used in these studies. Although slight modifications were made to accommodate the various flow conditions investigated, the general set-up remained consistent. The set-up consisted of a vapour-compression cycle with a 10 kW nominal cooling capacity.

- The vapour phase was generated by passing the refrigerant R-134a through a water-heated evaporator (12), followed by a suction accumulator (13), and scroll compressor (1).
- The refrigerant then circulated through two high-pressure condensation lines: the bypass line and the test line, each of which had its own electronic expansion valve (EEV) (10 and 11). The bypass line had one water-cooled heat exchanger (3) and was included to control the mass flow, pressure, and temperature at the test line. The test line consisted of three water-cooled condensers: a pre-condenser (4) used to control the inlet vapour quality, a test condenser (7) used to take test measurements, and the post-condenser (9) used to ensure that the refrigerant was fully liquid before entering the EEV (10).
- The test section (7) consisted of a tube-in-tube counter-flow heat exchanger of length 1 488 mm. Water flowed in the annulus and the refrigerant on the inside channel, which was a copper tube with an internal diameter of 8.38 mm.
- Flow visualisation was achieved through the use of cylindrical sight glasses (5) with the same internal diameter positioned at the inlet and outlet of the test condenser and was recorded with a high-speed video camera (200 fps) placed at the exit of the sight glass.
- Strait calming sections of length 500 mm and 400 mm, respectively, were placed before and after the sight glasses to ensure the flow was fully developed at the test condenser inlet and to minimise the disturbance of the exit sight glass.
- Flexible hoses were used at the inlet and exit of the test section to allow the whole range of inclination (θ) of the test condenser from vertical downwards (-90°) to vertical upwards ($+90^\circ$), which was measured using a digital inclinometer calibrated to an accuracy of 0.01° .
- Coriolis mass flow meters (2) were used to measure the mass flow of water and refrigerant through the pre-, test, and post-condensers.
- The heat transfer rate was maintained at 200 W during experimentation by controlling the mass flow rates and water inlet temperature through the annulus with a thermal bath (8).

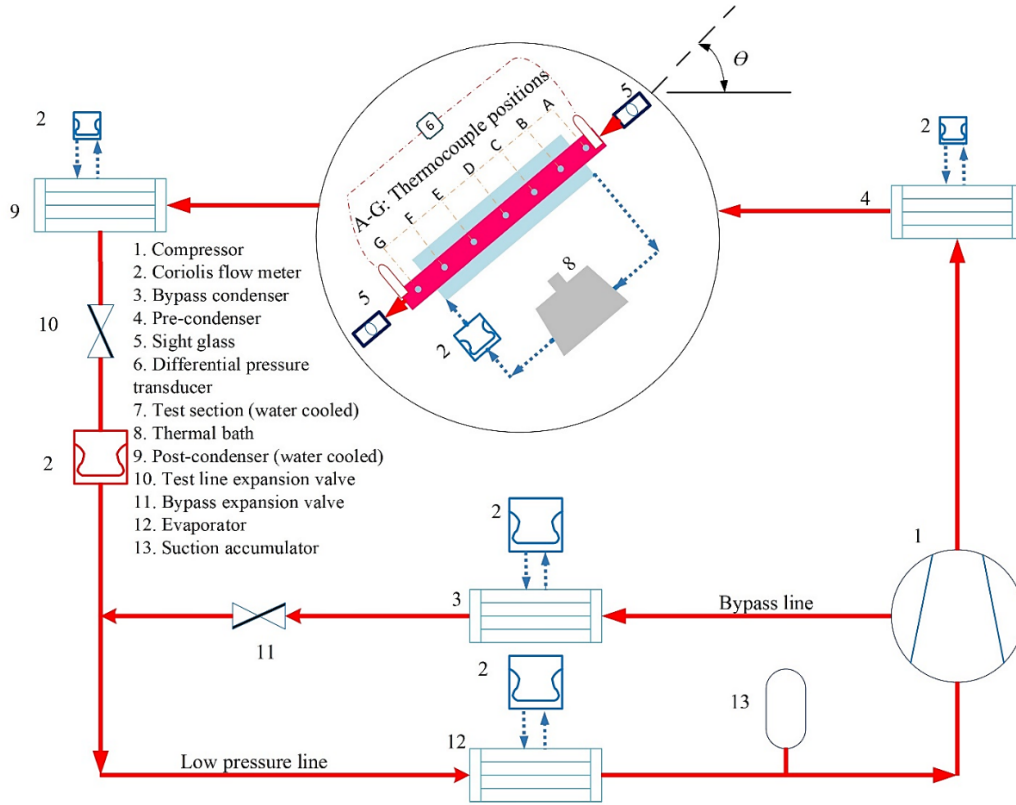


Figure 2-2: Schematic of the experimental set-up [8, 26]

2.6 Artificial intelligence

Artificial Intelligence (AI) can be described as the ability of computers to perform tasks that require human-level capability, or the simulation of human intelligence in machines such that they may think or act like humans. Human intelligence can be described through a process involving four stages: sense, store, process and act [55]. The initial process involves gathering information from human surroundings through sensory organs and then storing the information in memory. By processing the information, humans can distinguish patterns, or links, which allow them to act appropriately and with reasoning when faced with a given task. Similarly, machines are developed to mimic the human brain to sense, store, and process information, such that they may be used to make decisions with the best probability of achieving a certain goal.

This study considered the subset of AI known as machine learning, and more specifically, the subset of machine learning considering deep learning using ANNs. The use of such algorithms has gained momentum in recent times as computational power increases and has shown performance that matches and even exceeds human level capability. For instance, machine learning has been used together with computer vision for the development of automated self-driving vehicles [56] and was seen to outperform humans in detecting objects in visual data during the 2015 ImageNet challenge [57]. Gameplay has often been used to demonstrate the abilities of such algorithms to study and learn as they played, and as such has seen trained neural networks outperform humans in chess [58], and more recently, Go [59].

In the following sections, the concepts and terminology relating to machine learning and ANNs are defined, and then the literature investigating ANNs used for the task of two-phase flow pattern identification is explored.

2.7 Machine learning

Machine learning is an application or subset of AI in which computer algorithms automatically learn and improve through experience. Algorithms trained through machine learning processes can access data and use it to learn for themselves, rather than being explicitly trained or given human assistance. The data normally takes the form of a collection of examples; which may be created by humans, collected from natural processes, or even generated by another algorithm. The data is used to algorithmically develop a statistical model, which can then be used to solve practical problems or conduct some other kind of decision-making under uncertainty [60-62].

2.7.1 Machine learning methods

A machine learning algorithm, or model, is a mathematical expression that represents data in the context of a specific problem, where the aim is to use the data to gain insight. Depending on the task, various learning methods may be employed.

2.7.1.1 Supervised learning methods

Machine learning models trained by supervised learning methods analyse a known dataset, or collection of data examples with their associated labels, to produce an inferred function. Such models trained using labelled data are used to make predictions about unseen or future data. Using the known labelled or target values, the model performance can be evaluated and is normally improved through further training or exposure to a more extensive dataset. For this reason, supervised learning methods are also referred to as a predictive learning approach and are the form of machine learning most widely used in practice [62].

Supervised learning methods applied to data with categorical or discrete target values are referred to as classification tasks, while those with continuous or numerical target values are referred to as regression tasks. The differences between the two supervised learning tasks are shown visually in Figure 2-3. In classification tasks, the algorithm objective is to determine a line or hypersurface which best separates the training examples into a discrete number of considered classes. The model's performance is then judged on its ability to successfully predict the class of new examples based on the input data values. On the other hand, the objective of regression tasks is to determine a line or hypersurface which best follows the training examples, and the performance is judged on the ability of the model to interpolate or extrapolate the target values of new examples [61].

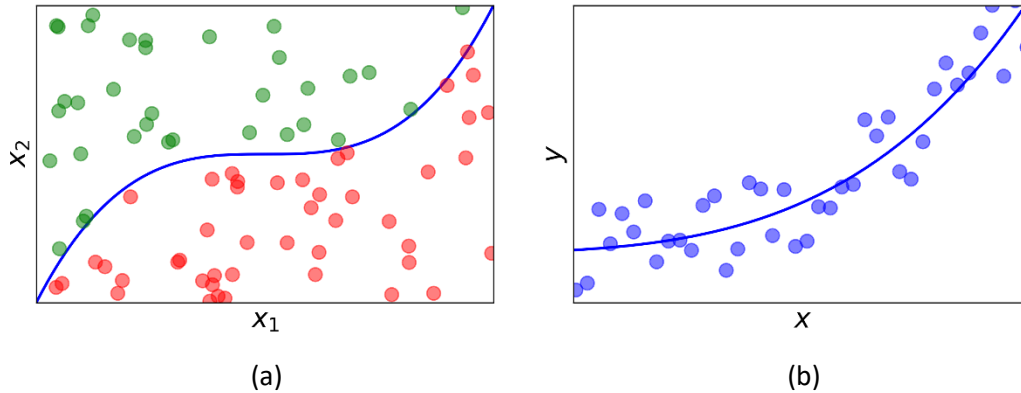


Figure 2-3: Differences between the supervised machine learning tasks of: (a) classification, (b) regression

2.7.1.2 Unsupervised learning methods

In contrast, unsupervised or descriptive learning methods are used to study a dataset that is neither classified nor labelled in an attempt to infer a function that describes a hidden structure, simplifies, or identifies patterns in the data. Unlike supervised learning methods, there is no obvious error metric that can be used to evaluate the performance of such a learning task. Rather, it is used as an exploratory data analysis tool, or to transform data into a new form which can then be used to solve a practical problem [62].

One such unsupervised learning method that was used in this study is dimensionality reduction, where the dimensionality refers to the number of variables, characteristics, or features that are present in the data. In most datasets, the features show some extent of correlation, and therefore some information is redundant or noise, which may negatively impact a machine learning model as the redundant information is learnt during the training process. Dimensionality reduction is used to transform the data into a lower-dimensional space by preserving the most important information and discarding the rest. Besides allowing a more efficient training process, dimensionality reduction also improves performance by reducing the effect of overfitting related to the noise in the data.

2.7.1.3 Semi-supervised learning methods

Semi-supervised learning methods can be thought of as a combination of both supervised and unsupervised learning methods, because both labelled data and unlabelled data are used for training. The use of a large amount of unlabelled data together with a small amount of labelled data can considerably improve learning accuracy. Semi-supervised learning is usually considered when the acquisition of labelled data is expensive or requires a high level of skill, while the acquisition of unlabelled data is relatively straightforward or does not require additional resources [61].

2.7.2 Machine learning terminology

This section defines the common terminology related to machine learning and the data used to train these models.

2.7.2.1 Parameters and hyperparameters

Parameters refer to the variables that form part of machine learning models, which are changed or updated automatically during training based on the training data. In contrast, hyperparameters are the variables or inputs which influence the performance of machine learning models and are manually chosen before the training process rather than being learnt from it. The hyperparameters determine the number of parameters as well as how they are modified during the learning process. They control the shape and size of models as well as the speed and complexity in which they learn. It follows that careful consideration is required when choosing the model hyperparameters, and they are usually determined by considering multiple variants [61].

2.7.2.2 Shallow and deep learning

A shallow learning algorithm learns or updates the model parameters directly from the input features of the data. In contrast, deep learning algorithms contain multiple processing layers, and the parameters related to each layer are learnt from the outputs of proceeding layers rather than from the input features directly. The complexity of models employing deep learning allows these models to learn from representations of the data with multiple levels of abstraction, and therefore, a highly non-linear relationship between the input and output of the data can be learnt. For this reason, deep learning is often referred to as representation learning. The use of deep learning has revolutionised machine learning methods, especially for classification tasks, because the higher levels of representation of the data can automatically highlight the important variations of the data that are necessary for discrimination while suppressing those which are not. Traditional methods required careful engineering and considerable domain expertise such as the precise selection of model hyperparameters to extract such high-level features [63].

2.7.2.3 Training, validation and test sets

A machine learning model trained by a supervised learning process with enough parameters would be able to memorise the data it is trained on, but would not perform well when used to predict new or unseen data. For this reason, a fraction of the data is removed from the training process and used to evaluate the performance of the model without any bias, by using examples that have not been seen before. The fraction used for evaluation is justified based on the size of the dataset. It is common to allocate at least 50 percent of the data for training, with a larger fraction used for datasets that contain fewer data samples [61].

After the collection of a dataset, it is randomly shuffled and split into three distinct sets: training, validation, and test. The training dataset usually consists of the biggest part of the dataset and is used to train the machine learning model by updating the model parameters. The test dataset consists of a fraction of the total dataset that has been set aside to evaluate the predictive or generalisation

performance of the trained model on new or unseen data. Although the data forms part of the original larger dataset, it is used to indicate how well the model will perform on new data acquired from other sources. The training process may include a validation dataset, which is usually of similar size to the test dataset. The validation dataset is used to evaluate the performance of the model while it is being trained because it indicates the expected model performance on the testing dataset. By maximising the predictive performance on the validation dataset, it is expected that the predictive performance will be maximised on the test dataset as well. Additionally, the validation dataset can be used to fine-tune the model hyperparameters, and to determine when sufficient training has taken place such that the model's predictive performance is maximised both on data used for training and on new data [61].

2.7.2.4 Cross-validation

In the process described above, it is assumed that the training dataset contains the necessary information required for consistent generalisation. Therefore, it is assumed that a different split of the data will not drastically improve or degrade the model's generalisation performance. Cross-validation is a method of providing a more robust estimate of the model's performance by taking the average over multiple runs. The most commonly used method of cross-validation is k-fold cross-validation, where the training dataset is partitioned into k groups of similar size called folds. A series of models is then subsequently trained with each of the k folds iteratively acting as the validation dataset, while the remaining k-1 folds of data act as the training dataset. The performance of the model is determined by averaging the performance overall of the k folds. The method allows the data to be used more effectively, because by cycling through the data, all data points are eventually used for both training and evaluating the model. For this reason, cross-validation is often used to fine-tune model hyperparameters. Once the best model hyperparameters have been determined, the entire training dataset can be used to train the final model and then subsequently, test the model performance on the test dataset [64, 65].

2.8 Artificial neural networks

ANNs were developed to mimic the working structure of the human brain, which is composed of neurological processing units called neurons. The connections between these biological neurons give humans the ability to process information and perform complex tasks. Neurological studies found that in humans and other highly intelligent species, that the brain contains significantly more neurons than in lesser intelligent species. The same relationship exists when employing ANNs, where more complex network architectures consisting of thousands or millions of parameters are used to solve complex problems, such as image recognition [55]. The remainder of this section explains the important concepts of ANNs, including their architecture, training and testing methodology, as well as the best practices for their implementation.

2.8.1 Artificial neuron

Similar to the human brain which is composed of biological neurons, artificial neurons are the basic processing units or building blocks of neural networks. An artificial neuron (referred to as a neuron only further on) can be summarised as the combination of two blocks, which are shown in Figure 2-4. The inputs (x_i) to a neuron may be the inputs of the larger network or come from the outputs of other neurons

within the network. Each input is associated with a connection or synaptic weight (w_i), which scales the magnitude of the input to the neuron. In the first block of the neuron, the weighted summation of the inputs is determined, which can be written mathematically as:

$$z = \sum_{i=1}^M w_i x_i + b \quad (2.9)$$

where z is the calculated weighted summation, and b is a scalar term referred to as the bias input, and acts like an intercept added to a linear equation to make the model more general.

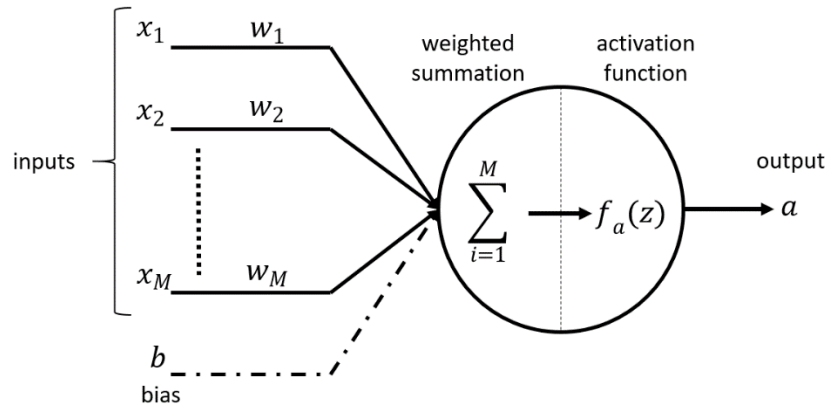


Figure 2-4: Structure of an artificial neuron

The second block is called an activation function, which is a continuous and differentiable function that transforms the weighted summation value. Numerous activation functions may be used depending on the application; however, they are usually used to add non-linearity within the framework of multiple connected neurons [65]. Mathematically, the activation function is written as:

$$a = f_a(z) \quad (2.10)$$

2.8.2 Multilayer perceptron neural network

The simplest structure of a neural network or neural model considers a single layer of linear input neurons connected to a single output neuron, which uses the heavy-side step activation function (or binary threshold), which outputs 1 when $z > 0$ or 0 if $z < 0$. This was the structure of the first neural model introduced by Frank Rosenblatt in 1957 and is referred to as the perceptron [65].

A single perceptron is limited by its linearity; however, by combining multiple processing units, highly non-linear and complex functions can be determined. A multilayer perceptron (MLP) is an ANN which consists of at least three interconnected layers of neurons: an input layer which provides input data to the network, one or more hidden layers (depending on the complexity) where most of the computations happen, and an output layer, which outputs the finished computation. Each neuron in the network is connected to all of the neurons in the previous layer and a bias unit, and the connections between layers are associated with weights, which are updated during network training as the network encodes the relationship between the input and output. The network performance is dependent on the architecture,

or the number of hidden layers and neurons in each of the respective layers. If the architecture is too simple, the network will not be able to learn the complexity of the unknown function; and if too complex, the network may suffer from an over-fitting problem, as the network learns the intricate details of the training data and suffers a decreased generalisation ability on unseen data [66]. The architecture of a standard MLP is shown in Figure 2-5.

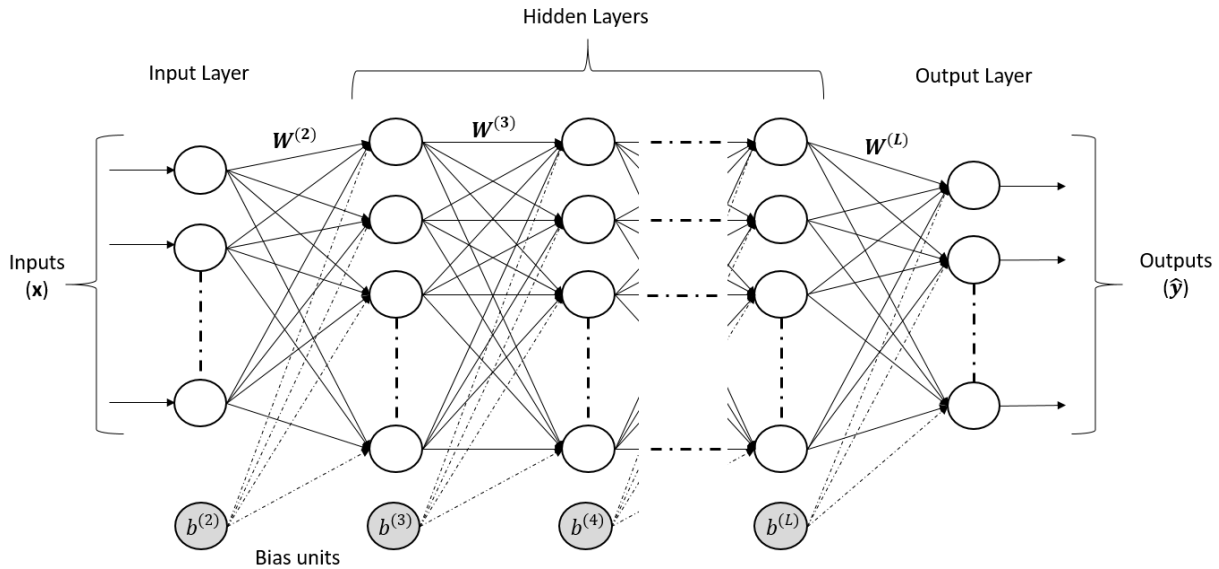


Figure 2-5: Architecture of a standard MLPNN

2.8.3 Activation functions

Activation functions are used at each neuron in an artificial neural network. They are used to help the network learn complex patterns in the data by defining whether a neuron should be activated, as well as determining the output of a neuron based on a given input or set of inputs. However, the choice of these functions requires careful consideration to achieve both good accuracy and an improved learning speed during training [65]. Some of the most popular activation functions used in state-of-the-art neural networks are the sigmoid, hyperbolic tangent, and rectified linear unit (ReLU) activation functions. A plot of each of these functions is shown in Figure 2-6, and their corresponding equations given by Equations (2.11) to (2.13).

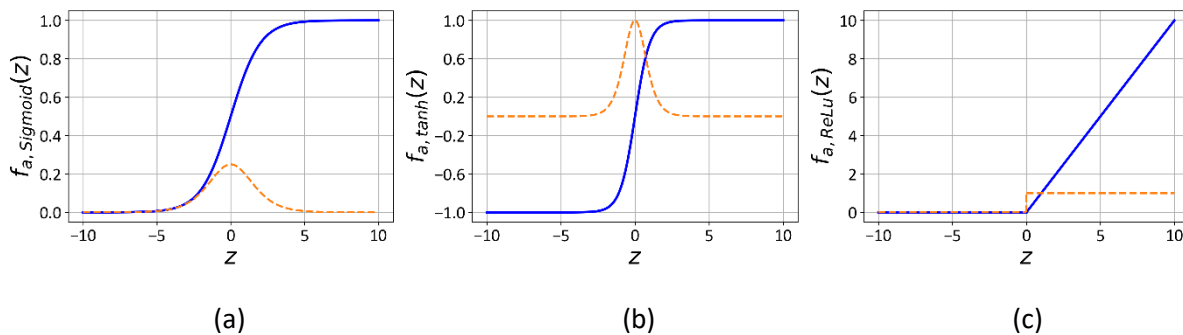


Figure 2-6: Plot of activation function (blue) and corresponding gradient (orange): (a) sigmoid, (b) hyperbolic tangent, (c) ReLu

$$f_{a, \text{sigmoid}}(z) = \frac{1}{1 + e^{-z}} \quad (2.11)$$

$$f_{a, \text{tanh}}(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}} \quad (2.12)$$

$$f_{a, \text{ReLU}}(z) = \max(0, z) \quad (2.13)$$

Referring to Figure 2-6a and Figure 2-6b, the sigmoid and hyperbolic tangent activation functions appear very similar; however, there are important differences between them. The sigmoid activation function is bounded in $[0,1]$, with two asymptotes: $f_{a, \text{sigmoid}}(z) \rightarrow 0$ when $z \rightarrow -\infty$ and $f_{a, \text{sigmoid}}(z) \rightarrow 1$ when $z \rightarrow \infty$. The hyperbolic tangent also has two asymptotes; however, it is bounded in $[-1,1]$. The shape of these curves is almost linear in the range $[-2,2]$, and then becomes almost completely flat thereafter. The result is that for small absolute values nearing 0, the gradient is high and nearly constant, which results in a quick learning rate, or large weight update step. However, at larger absolute values, the gradient is near 0 and results in very small weight updates and corresponding slow learning rates. This problem is referred to as vanishing gradients [55].

The ReLU activation function, depicted in Figure 2-6c, does not suffer from this problem and is considered as the default activation function for modern deep learning neural networks due to its simplicity and ability to accelerate training [56]. The gradient of the ReLU function is constant when the input value is positive and null when it is negative. Consequently, while the input is positive, the ReLU function does not suffer from the problem of vanishing gradients and can always make weight corrections based on the gradient. However, for negative inputs, both the output and gradient of the ReLU function are null, allowing no weight modification. Although this is generally not an issue, some modifications to the ReLU function consider a small negative gradient when the input is negative such as the leaky ReLU, exponential linear units (ELU), and Swish activation functions [65], but were not considered further in this study.

The sigmoid function is a popular choice in the output layer of neural networks because its output perfectly represents a probability value in $[0,1]$. However, for classification tasks, the softmax activation function is generally preferred because its output represents a discrete probability distribution over multiple outputs, corresponding to multiple classes or target labels. The output vector elements are in the range $[0,1]$ and sum to 1. Hence the output represents the probability of the input falling into each of the considered classes. The softmax activation function is given by:

$$f_{a, \text{softmax}}(z_k) = \frac{e^{z_k}}{\sum_{k=1}^K e^{z_k}} \quad , 0 < f_a(z_k) < 1 \quad (2.14)$$

where K refers to the number of categorical classes.

2.8.4 Training

2.8.4.1 Feedforward propagation

MLPs are trained using a feed-forward with a backpropagation algorithm. During the forward pass of information from the input to the output layer, the neurons in each of the hidden and output layers perform a weighted sum of the outputs from the neurons of the respective previous layer. The weighted sum is then considered as the argument to an activation function. Considering Equations (2.9) and (2.10), the feed-forward step for any arbitrary layer in the network can be written mathematically in a vectorised format as:

$$\mathbf{a}^{(l)} = f_a(\mathbf{z}^{(l)}) = f_a(\mathbf{W}^{(l)} \cdot \mathbf{a}^{(l-1)} + \mathbf{b}^{(l)}) \quad (2.15)$$

where the vectorised form of the weighted summation $\mathbf{z}^{(l)}$ is the dot product between the weights matrix of the current layer and the activations (outputs) of the previous layer summed together with the bias unit of the current layer. The activations of the current layer in vectorised form are then determined as the element-wise application of the activation function to the weighted sum. The vectorised form is used to present the feed-forward algorithm, Algorithm 1, where the bold script and superscript bracketing were omitted for simplicity as all variables refer to the vectorised form. In the algorithm, the ReLu and softmax activation functions are assumed for the hidden layers and output layer, respectively.

Algorithm 1: Feedforward algorithm

Require: x : Training sample

Require: W^l, b^l for $l \in [2, L]$: Weights and bias matrices

Set $\mathbf{a}^1 \leftarrow x$ (Initialise activations of Layer 1 as the input training sample)

for $l=2 \dots L-1$ **do**

$\mathbf{z}^l \leftarrow W^l \cdot \mathbf{a}^{l-1} + b^l$ (Determine weighted summation in hidden layers)

$\mathbf{a}^l \leftarrow f_{a, \text{ReLU}}(\mathbf{z}^l)$ (Determine activations in hidden layers)

end for

$\mathbf{z}^L = W^L \cdot \mathbf{a}^{L-1} + b^L$ (Determine weighted summation in output layer)

$\mathbf{a}^L = f_{a, \text{softmax}}(\mathbf{z}^L)$ (Determine activations in output layer)

Return $\mathbf{z}^l, \mathbf{a}^l, W^l, b^l, l \in [1, L]$

2.8.4.2 Cost function

The activations (output) from the output layer ($\mathbf{a}^{(L)} = \hat{\mathbf{y}}_n$) give the prediction of the unknown function (regression) or the estimated probability that the considered n^{th} training sample falls into each of the considered categorical classes (classification). During training by a supervised learning method, the backpropagation algorithm is used to minimise the error between the current estimated ($\hat{\mathbf{y}}$) and labelled (\mathbf{y}) outputs of the training data. The error is minimised by optimising a loss function (L), or when averaged over multiple training samples, it is referred to as a cost function (J). Considering a training set consisting of input samples ($\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N)$) and output labels ($\mathbf{Y} = (\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_N)$), together with the parameters (weights and biases) of the neural network ($\boldsymbol{\theta}$), the cost function is determined as the average of the loss function over either a subset (batch size) or all training samples, mathematically written as:

$$J(\theta) = \frac{1}{N} \sum_{n=1}^N L(x_n, y_n, \theta) = L(X, Y, \theta) \quad (2.16)$$

The best or optimal set of parameters is determined by finding the minimum of the cost function, mathematically given as:

$$\theta_{opt} = \operatorname{argmin}_{\theta} (J(\theta)) \quad (2.17)$$

Considering the loss function for a single sample (input x_n and associated true label y_n), the loss function can also be expressed as being explicitly dependent on the true and predicted label values as:

$$L(x_n, y_n, \theta) = L(\hat{y}_n, y_n) \quad (2.18)$$

which allows the parameters to be embedded into the prediction, and is fundamental to the explanation of the backpropagation algorithm.

For classification tasks employing the softmax activation function in the output layer, the cross-entropy cost function is the most appropriate choice. Minimising the cross-entropy cost function is equivalent to minimising the Kullback-Leiber divergence theorem between the true and predicted distributions over the respective considered classes. Due to its robustness and convexity, it is the cost function that is used in almost all deep learning classification tasks [65]. The cross-entropy cost function is given as:

$$J(\hat{Y}, Y) = \frac{1}{N} \sum_{n=1}^N L(\hat{y}_n, y_n) = -\frac{1}{N} \sum_{n=1}^N \sum_{k=1}^K y_{nk} \log(\hat{y}_{nk}) \quad (2.19)$$

where N is the number of training samples, K is the number of categorical classes, y_{nk} is either a value of 0 or 1, representing whether class label k is the correct classification for the n^{th} training sample, and \hat{y}_{nk} is the output probability of class label k for the n^{th} training sample.

2.8.4.3 Backpropagation

Starting from randomly initialised weights, the backpropagation algorithm is used together with an optimisation algorithm to adjust the network parameters after each training iteration, based on the gradient of the cost function, such that the global error on the training set is minimised. The backpropagation algorithm is used to compute the partial derivatives $\partial L / \partial w$ and $\partial L / \partial b$ of the loss function with respect to the weights or biases in the network for each training sample.

The component-wise partial derivative of the loss function for a single training sample with respect to any weight in the matrix is determined by back-propagating the error at the output layer with the use of the chain rule, and is given by:

$$\frac{\partial L(\hat{y}_n, y_n)}{\partial w_{ji}^{(l)}} = \frac{\partial L(\hat{y}_n, y_n)}{\partial z_j^{(l)}} \frac{\partial z_j^{(l)}}{\partial w_{ji}^{(l)}} = \delta_j^{(l)} \cdot a_i^{(l-1)} = \Delta w_{ij}^{(l)} \quad (2.20)$$

where $\delta_j^{(l)}$ is the local error at the j^{th} node in the l^{th} layer of the network. With the softmax output activation function, the local error at each of the neurons in the output layer is first determined as the difference between the estimated and labelled class outputs:

$$\delta_j^{(L)} = (\hat{y}_j - y_j) \quad (2.21)$$

And for each previous layer (assuming ReLu activation function), the error is determined in terms of the error in the next layer as:

$$\delta_i^{(l)} = w_{ij}^{(l+1)} \cdot \delta_j^{(l+1)} f'_{a,ReLU}(z_i^{(l)}) \quad (2.22)$$

where the derivative of the ReLu activation function is given by:

$$f'_{a,ReLU}(z_j^{(l)}) = \begin{cases} 0, & \text{if } z_j^{(l)} < 0 \\ 1, & \text{if } z_j^{(l)} > 0 \end{cases} \quad (2.23)$$

The same procedure is followed for the bias nodes in each layer, not repeated here, but note that the output activation of the previous layer $a_i^{(l-1)} = 1$ for all bias nodes in the network. Hence, the bias parameter update is based solely on the value of the local error.

The backpropagation algorithm is given as Algorithm 2, where all parameters are again shown in the vectorised form, and with the bold script and brackets in the superscripts omitted for simplicity.

Algorithm 2: Backpropagation Algorithm

Require: $z^l, a^l, W^l, b^l, l \in [1, L]$: The output from the feedforward algorithm

Require: y : Labelled output of training sample

$\delta^L \leftarrow (a^L - y)$ (Compute the error in the output layer)

$\Delta W^L \leftarrow a^{L-1} \cdot \delta^L$ (Compute the gradients w.r.t. the weights of the output layer)

$\Delta b^L \leftarrow \delta^L$ (Compute the gradient w.r.t the bias of the output layer)

for $l = L-1, \dots, 2$ **do**

$\delta^l \leftarrow (W^{l+1})^T \cdot \delta^{l+1} \odot f'_{a,ReLU}(z^l)$ (Compute the error in the hidden layers)

$\Delta W^l \leftarrow a^{l-1} \cdot \delta^l$ (Compute gradients w.r.t. the weights of the hidden layer)

$\Delta b^l \leftarrow \delta^l$ (Compute gradients w.r.t. the bias of the hidden layer)

end for

Return $\Delta W^l, \Delta b^l, l \in [2, L]$

The combined effect over a batch or all training samples is then determined by computing the average. This can be expressed mathematically in vectorised form as:

$$\frac{\partial J(\hat{Y}, Y)}{\partial W} = \frac{1}{N} \sum_{n=1}^N \frac{\partial L(\hat{y}_n, y_n)}{\partial W} = \Delta W \quad (2.24)$$

$$\frac{\partial J(\hat{Y}, Y)}{\partial \mathbf{b}} = \frac{1}{N} \sum_{n=1}^N \frac{\partial L(\hat{y}_n, y_n)}{\partial \mathbf{b}} = \Delta \mathbf{b} \quad (2.25)$$

2.8.5 Optimisation algorithms

The computed gradients allow the optimisation of the cost function by updating the model parameters towards the minimum (θ_{opt}).

2.8.5.1 Gradient descent optimisation algorithm

The simplest algorithm for optimising the weights and biases is that of gradient descent, which minimises the cost function by iteratively moving in the direction of steepest descent defined by the negative of the cost function gradient. The gradient descent algorithm is given as:

$$\theta_{t+1} = \theta_t - \eta \frac{\partial J(\hat{Y}, Y)}{\partial \theta_t} = \theta_t - \eta \Delta \theta_t \quad (2.26)$$

where θ refers to either the weights or bias parameters in the vectorised form: \mathbf{W} or \mathbf{b} ; $\Delta \theta$ is the partial derivative or gradient of the loss function with respect to the chosen parameter; η is a hyperparameter of the model known as the learning rate, which defines the step size taken during each optimisation iteration; and the subscript t refers to the index of the training iteration. The model parameters can be updated based on the error of a single training sample, or based on the average over multiple or even all training samples at once. The former is known as stochastic optimisation, while the latter is referred to as batch optimisation. In either case, the optimisation process is repeated until all samples included in the training set have been used, which is referred to as an epoch of training. Training continues for multiple epochs until the optimum or a sufficient optimum value is reached.

The batch size and learning rate are important model hyperparameters that need to be carefully chosen (or fine-tuned) as they determine the speed at which the network learns, as well as the final accuracy of the trained model. The inclusion of hidden layers, the non-linearity of the output function, and the significant amount of parameters associated with deep learning models lead to a complex solution space containing multiple local minima and saddle points (flat surfaces in the solution space). Hence, the optimisation algorithm may converge to suboptimal solutions. A high value for the learning rate could lead to a diverging solution as the network parameters are updated too quickly, while a model with a learning rate that is too small would require significant computation to train and may get stuck in suboptimal locations. Likewise, a batch size that is too small may place significant emphasis on a few training samples which may be outliers or non-dominant samples and degrade the generalisation performance of the model, while a large batch size can significantly decelerate training speeds.

2.8.5.2 Adam optimisation algorithm

Even with the fine-tuning of model hyperparameters, the standard gradient descent algorithm may still converge to suboptimal solutions. To improve the performance of deep learning models, many modifications to the gradient descent algorithm as well as new optimisation algorithms have been proposed to speed up convergence in problems that require a large number of parameters, and to avoid

the instabilities associated with ill-conditioned systems [65, 67]. Gradient perturbation is a method where small homoscedastic noise components are added to the gradients to avoid the issue of vanishing gradients. However, the randomness associated with this method makes it difficult to fine-tune model hyperparameters. The inclusion of a momentum term, which is computed based on both the current and previous gradients, is another and more robust method of improving the performance of the gradient descent algorithm when plateaus are encountered in the solution space. RMSProp is an optimisation algorithm that is based on the concept of momentum. This algorithm adaptively optimises each of the model parameters separately by computing exponentially weighted moving averages of the changing speed of parameters. By optimising each parameter separately, the speed of slowly changing weights can be increased, while those that are changing too quickly and likely to become unstable can be decreased.

In this study, the adaptive moment estimation (Adam) [67] optimisation algorithm was used. The Adam method also computes adaptive learning rates for each of the network's parameters and was proposed as an improvement to the RMSProp algorithm. The method stores an exponentially decaying average of past and past squared gradients, \mathbf{m}_t and \mathbf{v}_t , computed as:

$$\mathbf{m}_t = \beta_1 \mathbf{m}_{t-1} + (1 - \beta_1) \Delta \theta_t \quad (2.27)$$

$$\mathbf{v}_t = \beta_2 \mathbf{v}_{t-1} + (1 - \beta_2) \Delta \theta_t^2 \quad (2.28)$$

where β_1 and β_2 are hyperparameters that control the exponential decay rate of the computed averages. \mathbf{m}_t and \mathbf{v}_t are the first- and second-order moments (mean and uncentered variance) of the gradients respectively, and as they are initialised as vectors of 0's, they are biased towards 0 during the initial weight update steps, and especially so when the decay rates are small (β_1 and β_2 chosen close to 1). This bias is corrected through computing bias-corrected estimates of the first- and second-order moments, $\hat{\mathbf{m}}_t$ and $\hat{\mathbf{v}}_t$ respectively:

$$\hat{\mathbf{m}}_t = \frac{\mathbf{m}_t}{1 - \beta_1} \quad (2.29)$$

$$\hat{\mathbf{v}}_t = \frac{\mathbf{v}_t}{1 - \beta_2} \quad (2.30)$$

The bias-corrected moment estimates are then used to update the network parameters, computed as:

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\hat{\mathbf{v}}_t} + \varepsilon} \hat{\mathbf{m}}_t \quad (2.31)$$

where ε is a very small number added to avoid the division by 0 during implementation. Values of 0.9 for β_1 , 0.999 for β_2 , 10^{-8} for ε , and 0.001 for η are suggested by the authors [67]; however, these hyperparameters can also be fine-tuned according to the problem. Algorithm 3 gives the Adam optimisation algorithm used to update all the parameters for a single update step.

Algorithm 3 Adam optimisation algorithm

Require: η : learning rate
Require: $\beta_1, \beta_2 \in [0,1)$: Exponential decay parameters for moment estimates
Require: θ_t : Parameter matrix/vector at current time step
Require: $\Delta\theta_t$: gradient of parameter at current time step
Require: m_{t-1} : 1st moment vector from previous time step
Require: v_{t-1} : 2nd moment vector from previous time step
 $m_t \leftarrow \beta_1 \cdot m_{t-1} + (1 - \beta_1)\Delta\theta$ (Update biased 1st moment estimate)
 $v_t \leftarrow \beta_2 \cdot v_{t-1} + (1 - \beta_2)\Delta\theta^2$ (Update biased 2nd moment estimate)
 $\hat{m}_t \leftarrow m_t/(1 - \beta_1)$ (Compute non-biased 1st moment estimate)
 $\hat{v}_t \leftarrow v_t/(1 - \beta_2)$ (Compute non-biased 2nd moment estimate)
 $\theta_t = \theta_{t-1} - \eta \cdot \hat{m}_t/(\sqrt{\hat{v}_t} + \epsilon)$ (Update parameters)
Return θ_t, m_t, v_t

2.8.6 Regularisation

As previously mentioned, creating a complex network architecture with too many free parameters may lead to overfitting. However, too few parameters may lead to the inability to learn the relationship between input and output (underfitting). Additionally, the high capacity of deep learning models can become problematic when the structure of the model required to learn the training set is not proportional to that required to generalise well. Regularisation can be used to lessen the effects of overfitting while ensuring sufficient parameters and model complexity.

2.8.6.1 L1 and L2 regularisation

The most common method of implementing regularisation is by adding a non-negative function of the weights to the cost function. The function acts as a penalty term and ensures that the network parameters are as small as possible, which, in turn, prevents the network from becoming highly dependent on specific features of the training set. Mathematically, the modified cost function can be written as:

$$L_R(\mathbf{X}, \mathbf{Y}, \boldsymbol{\theta}) = L(\mathbf{X}, \mathbf{Y}, \boldsymbol{\theta}) + \lambda g(\boldsymbol{\theta}) \quad (2.32)$$

where the lambda parameter (λ) is a specified constant which controls the strength of the regularisation and is another hyperparameter of the resulting model which needs to be considered. A value of 0 for the constant would assume no regularisation of weights while increasing this value penalizes complexity and keeps the network's weights small. Increasing the constant term to a very high value will lead to an underfitted solution because the penalty term will drive all parameter values to 0.

The L2 regularisation term is composed of the squared magnitude, or L2-norm of the model parameters ($g(\boldsymbol{\theta}) = \|\boldsymbol{\theta}\|_2^2$). L2 regularisation is also referred to as weight decay or weight shrinkage because it forces weights to decay towards 0 (but not exactly 0), which prevents the model from being dominated by a few parameters. On the other hand, the L1 regularisation term is composed of the absolute magnitude, or L1-norm of the model parameters ($g(\boldsymbol{\theta}) = \|\boldsymbol{\theta}\|_1$). L1 regularisation shrinks the less important features' coefficients to 0, which removes some parameters altogether and it is useful to

compress a model that may have too many free parameters, which would otherwise likely model noise in the dataset [65].

2.8.6.2 Dropout regularisation

Dropout is a regularisation technique for deep learning was first introduced by Hinton et al. [68] and Srivastava et al. [69]. To implement dropout, a percentage of neurons (and their respective connections) in a specified layer or layers in the network are randomly removed during each training step, and the remaining active network connection weights are updated based on the classification performance and associated cost function gradient. An example of dropout is shown in Figure 2-7, which indicates that a network containing two hidden layers each with five neurons is reduced to a network with hidden layers consisting of two and three neurons respectively. Implementing dropout allows for a different sub-network of the original model to be updated after each training step, while the performance of the model on a validation or test set is evaluated using the complete original model. The regularisation effect works in two ways: firstly, training with only a subset of the model with reduced capacity is less likely to overfit the data and prevents the model from memorising the interdependence between neurons; and secondly, the overlap between many sub-networks which have each been trained on a different subset of the training data allows for an averaged prediction.

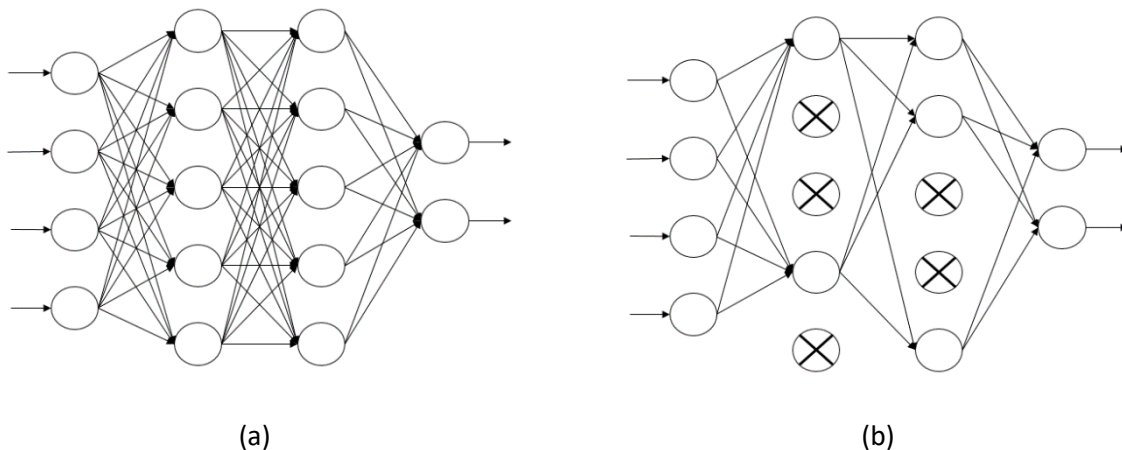


Figure 2-7: Example of a dropout MLP: (a) standard MLP with two hidden layers, (b) MLP with dropout applied in hidden layers

Apart from reducing overfitting, dropout also allows the use of an increased learning rate, which is preferred to allow the model to explore more of the solution space. Finally, based on their test studies, the authors [68, 69] recommended that the dropout rate in hidden layers should be either set by validation methods or simply should be set at 0.5, while the dropout rate in the input layer (if used) be much lower. They also suggest the use of improved optimisation algorithms and L2 regularisation together with dropout networks.

2.9 Convolutional neural networks

Convolutional neural networks (CNNs), first introduced by LeCun et al. [70], have been described as the most accurate of the AI techniques for performing visual processing tasks. In the same manner in which MLPs took inspiration from the neural structure of the human brain, CNNs take inspiration from the visual cortex, which is the region of the brain responsible for processing visual input. For this reason, CNNs are usually the best choice in the fields of computer vision and image classification. Compared with MLPs, which require a flattened vector of input features, CNNs have the advantage of learning from two- or three-dimensional spatial information of the input [71]. The feature extraction process takes place with the use of convolutional layers that pass a set of independent filters over the images, transforming the images and extracting geometric patterns. During training, the filters are adjusted using the backpropagation algorithm such that meaningful information is extracted. Hence, the values of the filters are the parameters (weights) that need to be optimised during training. Examples of the use of image filters are blurring, sharpening, edge detection, curve detection, or boundary detection. Early convolutional layers extract low-level features such as edges or curves. However, by stacking multiple convolutional layers on top of each other, more abstract and in-depth information can be extracted, such as shapes (the combination of curves and edges). An additional advantage of CNNs is that parameter values are shared across multiple neurons, allowing the extraction of similar features irrespective of their position within the input images. The use of multiple convolutional layers, however, leads to an exponential increase in parameters that need to be trained. To reduce the required amount of parameters and associated computation in the network, pooling layers are used to reduce the spatial size of images between successive convolutional layers. After the feature extraction in the convolutional layers, one or more fully connected layers and an output layer is used to determine which high-level features most strongly correlate with a specific class output.

2.9.1 Convolutional layers

Kernel filters are used in convolutional layers to extract features based on spatial information. The parameters (weights) are the filter values that are shared across the entire convolutional layer by sliding the filters over the input matrices, resulting in convolutional matrices, which are commonly referred to as feature maps. The values in the convolutional matrices are the result of the sum of the element-wise scalar product between the kernel filter values and those of the portion of the image matrix to which it is applied. Mathematically, the convolved value at index $[i, j]$ (with kernel filter of size $f_1 \times f_2$) is determined as:

$$C_{i,j} = f_a \left(\sum_{u=1}^{f_1} \sum_{v=1}^{f_2} I_{i-1+u, j-1+v} F_{u,v} + b \right) \quad (2.33)$$

where C refers to the convolution matrix, F refers to the kernel filter matrix with indices u and v , I refers to the original or input image matrix, b refers to the bias input for the kernel filter, and $f_a(\cdot)$ refers to the activation function.

The performance of these filters to extract meaningful information is subject to four hyperparameters: size, depth, stride, and padding [72]. The description of these hyperparameters is simplified through the use of an example, shown in Figure 2-8.

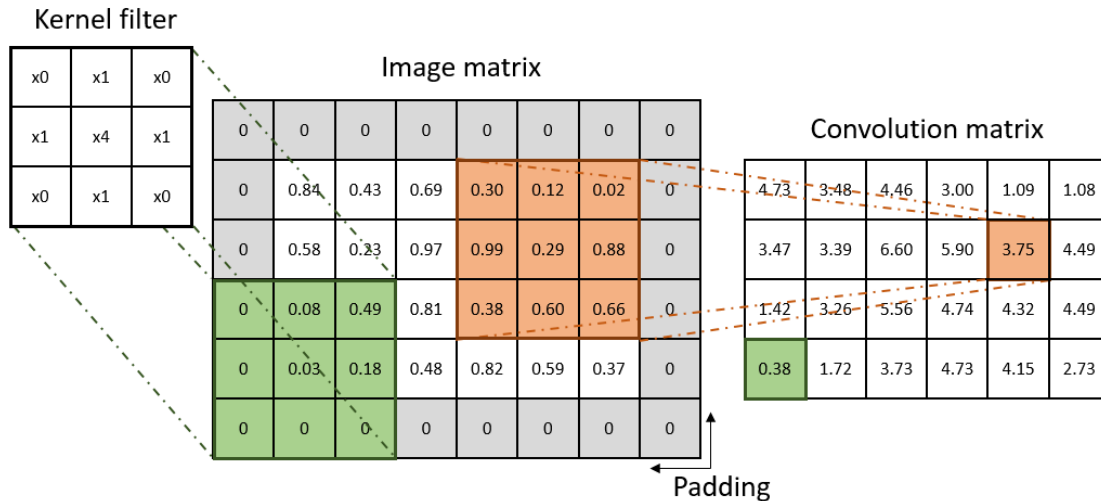


Figure 2-8: Example of a convolution with a 3x3-sized kernel filter, stride length of 1, depth of 1 and single layer of padding.

The size (F) refers to the area of the filters, which are chosen based on the spatial size of the features that need to be extracted. Often larger sized filters are used in earlier convolutional layers to extract lower-level features, and smaller sized filters in latter layers are used to extract higher-level features. In the example, a filter size of 3x3 is considered.

Depth (Q) refers to the number of filters (or channels) applied to the input within the same convolutional layer. For instance, in the first convolutional layer, a depth including one or three colour channels is used depending on whether the input is greyscale or in colour (red, green and blue). In subsequent layers, increasing the depth allows for a combination of different filters to extract the necessary amount of information to sufficiently characterise the data. In the example, a depth of 1 pertaining to a single filter is shown for simplicity. Considering multiple kernel filters ($Q > 1$) each with its own respective set of weights and bias units, Equation (2.33) can be represented in a vectorised format for the q^{th} convolution matrix formed as a result of the q^{th} kernel filter as:

$$C_q = f_a(I * F_q + b_q) \quad (2.34)$$

For $q=1,2,\dots,Q$, and where $*$ refers to the convolution operation. Subsequent convolutional layers will receive multiple convolutional matrices from the respective previous layer as input, with the number equal to the number of kernel filters used in the previous convolutional layer. In this case, the convolutional matrix is formed by the sum of the convolution operation on all of the Q input images, determined as:

$$C = f_a \left(\sum_{q=1}^Q I_q * F_q + b \right) \quad (2.35)$$

It is through this summation over multiple filters and feature maps that more complex and in-depth features are extracted.

Stride (S) refers to the step size taken by the kernel filter when moving through the input during convolution. A step size of 1 corresponds to the kernel filter moving to every unique location of the input (as in the example shown), while a step size of 2 refers to the filter moving two pixels at a time through the input. A larger stride reduces the size of the convolution matrix by a factor of s^2 . Hence, it reduces the number of parameters and associated computations in the network, with the effect of less overlap between neighbouring pixels in the convolution matrices. To determine the convolutional matrix considering any stride length, Equation (2.33) can be adjusted as:

$$C_{i,j} = f_a \left(\sum_{u=1}^{f_1} \sum_{v=1}^{f_2} I_{si-s+u,sj-s+v} F_{u,v} + b \right) \quad (2.36)$$

Padding (P) (pixels with value 0) can be added to the boundaries of the input image(s) for each convolutional layer such that the spatial size of the image is preserved between subsequent convolutions. Performing convolution without the padding would result in the outermost layer(s) of pixels being dropped from the image with each convolutional layer. Although the padding can be excluded in the design of a CNN, it is usually expected that the boundary information of images is of vital importance to the classification performance. The width of padding required to maintain the spatial size of an image with any kernel size is determined as:

$$P = \frac{F - 1}{2} \quad (2.37)$$

Alternatively, the size of the output (O) can be determined as a function of the input images size (W), the kernel filter size (F), the stride length (S) and the amount of padding (P) as:

$$O = \frac{W - F + 2P}{S + 1} \quad (2.38)$$

Ideally, O should be an integer to ensure that the kernel filters neatly extend to the edges of the input, which otherwise may cause problematic issues during training because the content at the outermost edges of the filters is degraded. For this reason, it is also good practice to choose an input image size to a CNN which is divisible multiple times by a factor of 2, such that it is easy to maintain appropriate sizing of convolutional matrices between multiple layers.

2.9.2 Pooling layers

Stacking multiple convolutional layers on top of one another allows for highly abstract features to be extracted, and, in turn, to categorise highly complex and noisy data. However, the use of multiple convolutional layers leads to a huge amount of features that need to be optimised. To reduce the number

of required parameters and associated computation in the network, pooling layers are used, which are usually placed between successive convolutional layers. Pooling layers reduce the spatial size of feature maps by reducing a neighbourhood of pixel values to a single pixel value that meets certain criteria. Such criteria include retaining the maximum pixel value (max-pooling), minimum pixel value (min-pooling), mean pixel value (mean-pooling), or the summation of neighbouring pixels (sum-pooling). In either case, a pooling layer with a neighbourhood region size of $n_p \times n_p$ pixels will reduce the size of an image by a factor of n_p^2 [66]. Apart from allowing dimensionality reduction, pooling layers reduce the sensitivity of the feature maps to slight translations or distortions with an effect that is proportionate to the size of the pooling region. By preventing the network from learning the specific locations of features, the effects of overfitting are reduced, allowing a more robust training process [65].

Of the mentioned pooling criteria, max-pooling is the most commonly applied pooling operation, because it keeps the most dominant features, which are the most likely to allow for image recognition or classification. Additionally, max-pooling with a neighbourhood size of 2x2 pixels is most commonly used to avoid discarding too much of the data at once, while still allowing a dimensionality reduction of four. Figure 2-9 shows an example of max-pooling applied to the output convolutional matrix seen previously in Figure 2-8, which shows that the pixel with the maximum value in each neighbourhood containing 2x2 pixels is extracted.

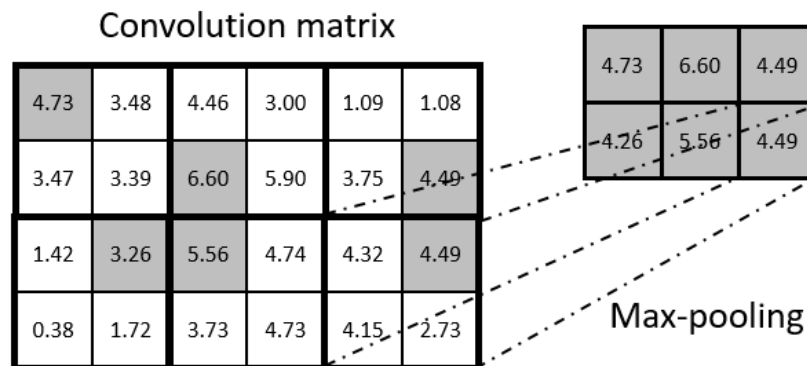


Figure 2-9: Example of the max-pooling operation

2.9.3 Training

The output of the last convolutional layer is flattened from a set of Q matrices into a one-dimensional array, which is then fed as input to a fully connected neural network. The fully connected layers are identical to the MLP previously described, with the extracted features from the convolutional layers forming the input layer of the MLP. The feedforward step, determination of the cost function, backpropagation, and weight update in the fully connected layers are the same as those previously described for the MLP. However, the backpropagation and resulting kernel filter weight updates in the convolutional layers follow a more complicated method. Firstly, it is noted that the backpropagated loss at the input to the fully connected layers (previously referred to as δ) is also the loss at the output of the last convolutional layer ($\frac{\partial \mathcal{L}}{\partial C}$ or ΔC), after being reshaped back into a set of matrices. Secondly, the nature of the convolution operation leads to the resulting convolution matrices being dependent on every value

of the kernel filter. Hence, the backpropagation of the loss to the previous convolutional layer and the determination of the kernel filter weight gradients also takes the form of a convolution operation. The gradient to update the filter weights (shown for a single filter) is determined with the convolution between the input and the loss gradient at the output of the convolutional layer:

$$\frac{\partial L}{\partial \mathbf{F}} = \Delta \mathbf{F} = \mathbf{I} * \Delta \mathbf{C} = \mathbf{I} * \frac{\partial L}{\partial \mathbf{C}} \quad (2.39)$$

or in element wise format as:

$$\frac{\partial \mathcal{L}}{\partial F_{u,v}} = \sum_i \sum_j \frac{\partial \mathcal{L}}{\partial C_{i,j}} \frac{\partial C_{i,j}}{\partial F_{u,v}} = \sum_i \sum_j \frac{\partial \mathcal{L}}{\partial C_{i,j}} I_{i-1+u, j-1+v} \quad (2.40)$$

Similarly, the gradient of the loss at the input to the convolutional layer can be determined as:

$$\frac{\partial \mathcal{L}}{\partial I_{k,h}} = \sum_i \sum_j \frac{\partial \mathcal{L}}{\partial C_{i,j}} \frac{\partial C_{i,j}}{\partial I_{k,h}} \quad (2.41)$$

Note the indices k and h are used here once-off to differentiate the input gradient indices from those of the output. It turns out that this operation can be determined through a convolution operation between the kernel filter rotated by 180 degrees and the loss gradient at the output:

$$\frac{\partial \mathcal{L}}{\partial I_{i,j}} = F_{ROT180^\circ} * \Delta \mathcal{C} \quad (2.42)$$

where the 180° rotated matrix is formed by flipping the matrix both vertically and horizontally. To backpropagate the error through the pooling layers, the error is assigned to the index of the larger matrix which it came from, while the other indices of the larger matrix are left unchanged because they had no contribution to the error. To perform such a task, a mask is stored in memory of the indices containing the selected values in each region during the forward pass in the pooling layers.

2.9.4 Data augmentation

Data augmentation is a technique in which the size of a dataset is increased by making minor alterations or adding slightly modified copies to the existing data. In the case of image data augmentation, examples are random translation, flipping, scaling, zooming, rotating, cropping, altering brightness or contrast, or even adding noise to the images. Apart from increasing the size of the dataset, data augmentation also acts as a regulariser and helps to prevent overfitting because the model is shown slightly different modifications of the original training examples during each epoch of training. By carefully considering the choices of image augmentation during training, a CNN model can be trained to generalise better to unseen images that may look slightly different from those used during the training process.

2.10 Research investigating artificial neural networks for flow pattern identification

ANNs have been used extensively in the past three decades to deal with issues involving flow and heat transfer. Cong et al. [73] published a review paper that summarises the applications of ANNs for predicting

issues such as flow patterns, pressure drop, void fraction, critical heat flux, and HTC. They point out that the initial step during the analysis of two-phase flow systems is the prediction of flow patterns because the other flow phenomena usually depend on the resulting flow patterns. Subsequently, they found that 16.33% of the literature investigating ANNs and two-phase systems published at that time (2012) was devoted to such a task. Upon analysis of the review, and more specifically the ANNs trained by backpropagation to predict multiphase flow patterns, it can be seen that most studies focused on adiabatic and boiling flows in vertical and horizontal tubes and channels, and mainly considered air-water two-phase flows. Additionally, most of the studies relied on well-established models, computer-generated data, or measured thermo-hydraulic parameters such as pressure and temperature measurements, which can introduce bias and reduce generalisation capability compared with models trained on high dimensional noisy data, such as from visualisation. The studies published since the review article seem to follow the same trend. For instance, Ozbayoglu and Yuksel [74], Inoue et al. [75], Al-Naser et al. [76] and Chandrasekaran and Kumar [77] used dimensionless numbers; Massignan et al. [78], Figueiredo et al. [79] and Baba Musa and Hoi [80] used ultrasonic attenuation data; Hanafizadeh et al. [81] used transient flow pressure signals, and Roman et al. [13] used electrical capacitance tomography (ECT) permittivity data as input data to ANNs to predict in-tube multiphase flow patterns. A summary of the literature investigating ANNs trained by backpropagation to predict multiphase flow patterns is given in Table 2-1.

Table 2-1: Summary of recent works investigating artificial neural networks trained by backpropagation to predict in-tube multiphase flow patterns

Researchers	Year	Input	Output	Tube orientation	Fluid	Heat transfer	Accuracy	Additional remarks
Chandrasekaran and Kumar [77]	2018	Dimensionless numbers: superficial velocities of pressure, density and viscosity	Four flow patterns: mist, stratified, slug, bubbly	Upward and downward flow in inclined tubes	Air-kerosene	None	83%	Results of MLP compared with random forest algorithm (85.4%), logistic regression (73.7%) and support vector machines (79.8%)
Ezzatabadipour et al. [82]	2017	Fluid properties and pipe conditions	Six flow patterns: annular, bubbly, dispersed bubbly, intermittent, stratified-smooth, stratified-wavy	Full range of inclination angles	See remarks	See remarks	83.87%, 83.34% and 85.97% for three tests considered	Study investigated an extensive experimental database from Shoham [83] (5 676 data points) comparing a large range of flow conditions and pipe orientations
Al-Naser et al. [76]	2016	Dimensionless numbers: liquid Reynolds number, gas Reynolds number, pressure drop multiplier	Four flow patterns: annular, dispersed bubbly, intermittent, stratified	Horizontal pipes with varying diameter	Liquid-gas flow	None	>97%	Pre-processing stage using natural logarithmic normalisation to reduce the overlap between flow patterns
Baba Musa and Hoi [80]	2016	Ultrasonic signals recorded from a Doppler sensor extracted into features by applying power spectral density (PSD) and discrete wavelet transform (DWT)	Four flow patterns: slug, elongated bubbles, stratified, stratified-wavy	Horizontal	Air-water	None	95.8% (DWT) 87.5% (PSD)	Only 24 test data points considered; clamp on ultrasound sensor is non-invasive and non-radioactive; perspex pipe allows flow visualisation
Figueiredo et al. [79]	2016	Ultrasonic acoustic attenuation data measured by 4 transducers	Six flow patterns: bubbly, cap bubbly, stable slug, unstable slug, churn, annular	Vertical upward flow in one- and two-inch acrylic pipes	Two-phase (oil-air) and four-phase (oil-water-air-sand)	None	98.3% flow pattern prediction, 4.2% gas-volume-fraction variation	Data produced as a result of non-invasive multiphase flow metering in the oil industry
Roman et al. [13]	2016	Electrical capacitance tomography (ECT) permittivity data obtained from tomograms	Six flow patterns: bubbly, plug, slug, stratified-wavy, annular, transitional	Horizontal	R-134a	Boiling	98.1%, with 99% of flow patterns classified within one flow pattern	High-speed images were obtained for human classification of flow patterns (visualisation technique)

Hanafizadeh et al. [81]	2016	Transient flow pressure signals	Four flow patterns: annular, churn, slug, annular	Upward vertical	Air-water	None	Proposed flow regime map can characterise the considered flow regimes	Neural network outputs used for sketching new flow regime map; reasonable agreement when compared with the map of Hewitt and Roberts; visualisation used to characterise flow patterns through transparent acrylic glass
Massignan et al. [78]	2014	Broadband attenuation ultrasound (BUA) index	Six flow patterns: bubbly, dispersed bubbly, slug, churn, wavy, annular	Vertical	Air-water	None	Method does not work for churn, wavy and annular flows; good prediction between bubbly to slug flow regimes	Non-invasive method; plexiglas sections allow flow visualisation
Inoue et al. [75]	2013	Dimensionless numbers: Reynolds number, Froude number, Weber number, pressure rate, superficial velocity ratio	Five flow patterns: discrete bubbly, stratified, slug, intermittent, annular	Model applicable to vertical, inclined and horizontal pipe orientations	Data extracted from multiple literature sources	None	All flow regimes accurately classified with only a small deviation ($R^2 = 0.9651$)	Training procedure conducted with a Levenberg-Marquardt algorithm
Ozbayoglu and Yuksel [74]	2012	Dimensionless numbers: liquid and gas superficial Reynolds number	Seven flow patterns: stratified, plug, slug, churn, annular-wavy, dispersed bubbly, dispersed annular	Horizontal eccentric annulus	Air-water	None	90.38%	Gas Reynolds number was scaled to reduce the issue of overlapping between flow patterns; backpropagation neural network outperformed nearest neighbourhood and classification tree methods
Rosa et al. [21]	2010	4 statistical moments (mean, standard deviation, skewness, kurtosis) and probability density functions of line averaged void fraction readouts from a single-wire electrical resistivity probe	Six flow patterns: bubbly, spherical cap, slug, unstable slug, semi-annular, annular	Upward vertical flow	Air-water	None	96-100% for the six flow patterns	Flow patterns identified by visual inspection through a plexiglas pipe
Lee et al.[84]	2008	100 inputs from 100 Hz Naquist sampling of bubble size sorted by a probability distribution	Five flow patterns: bubbly, cap bubbly, slug, churn annular.	Upward and downward vertical flow	Air-water	None	All flow regimes predicted well except annular	Transitional regimes identified when multiple output neurons were numerous excited; instantaneous (within 1s) and objective flow regime

							flow due to lack of data	identification; flow regimes in good agreement with Mishima-Ishii criteria for upward flow; downward slug flow in reasonable agreement with Usui criteria
Jing et al. [85]	2008	Scattering ray energy probabilities	Three flow patterns: homogeneous, annular, slug	Vertical	Gas-liquid	None	81.8%	Non-intrusive method
Selli and Selegim [86]	2007	Gabor coefficients obtained from pressure gradient signals	Five flow patterns: stratified-smooth, stratified-wavy, intermittent, annular, bubbly	Horizontal	Air-water	none	100% in steady-state conditions with specified detection level; intermittent and stratified-wavy identified with most certainty	On-line identification of flow regimes; used a previously trained ANN; training signals sampled from Taitel and Dukler map
Hernández et al. [87]	2006	Statistical parameters of the cumulative probability density function (CPDF) of the bubble chord length measured by a conductivity probe	Five flow patterns: bubbly, cap bubbly, slug, churn-turbulent, annular	Upward vertical	Air-water	Adiabatic	95%	On-line identification of flow regimes; good agreement with visual flow map; pre-processing of CPDF signals by PCA
Sunde et al. [88]	2005	Statistical moments of pixel intensity data obtained from radiographic and visible light images	Four flow patterns: bubbly, annular, slug, churn	Upward vertical flow metal and plastic pipes	Air-water	Boiling	95% - Bubbly and annular predicted with high confidence, slug and churn flows often misclassified	Allowed for an online, non-intrusive method to classify flow regimes in a metal pipe
Hervieu [89]	2002	Power spectral density of signals delivered by multi-electrode impedance sensor	Six flow patterns: bubbly, intermittent, annular, stratified-smooth, stratified-wavy, stratified-rugged	Horizontal	Air-water	None	>80%	Non-invasive measurement technique; flow patterns compared with Taitel and Dukler map
Mi et al. [90, 91]	1998, 2001	Statistics (mean and standard deviation) of impedance signals of area averaged void fractions	Five flow patterns: single liquid, annular, bubbly, slug, churn	Horizontal and vertical	Air-water	None	96.4%, 94.3%	Flow patterns observed through transparent Lucite tube; non-intrusive method to determine flow regimes

Apart from the use of supervised methods, the use of ANNs trained by unsupervised methods to produce self-organising maps (SOMs) has also been the subject of numerous studies [84, 92-98].

Although ANNs seem to be the most popular of the AI methods to predict multiphase flow patterns, additional methods applying AI techniques have also been considered. For instance, Wiedemann et al. [99] used wire-mesh sensor data with fuzzy clustering for the prediction of air-water two-phase flows in horizontal tubes. Ghanbarzadeh et al. [100] used an adaptive neuro fuzzy inference system (ANFIS) to identify flow patterns based on image textural features in vertical air-water systems. Zhang and Wang [101] used electrical capacitance tomography (ECT) and support vector machines (SVMs) to classify oil gas two-phase flow patterns. Mahvash and Ross [102] applied continuous hidden Markov models (CHMMs) to identify two-phase flow patterns of air-water applied to local void fraction signals collected from single step index multimode optical fibre probes located at the centre and mid-length of the tube. Finally, Zhang and Wang [103] applied empirical mode decomposition (EMD) and Hilbert-Huang transform (HHT) to sound signals from non-return valves under different conditions to produce gas-liquid two-phase flow pattern maps constructed with the coordinates of the energy of intrinsic mode functions (IMF) and Hilbert marginal spectrum (98.1%).

From the literature investigated, it appears that no study has yet investigated the use of visualisation data as input data to any ANN or alternative AI technique to predict in-tube multiphase flow patterns.

2.11 Summary and conclusions

This chapter started with a brief revision of the fundamental concepts relating to condensation heat transfer and two-phase flow terminology. The main flow patterns observed in horizontal and vertical tube orientations were then discussed according to the identification criteria given by Thome [3] and Thome and Cioncolini [4].

An investigation into the existing flow pattern maps revealed that the most widely accepted map used to predict the condensation of refrigerants in horizontal tubes is the El. Hajal and Thome [46] map, which was slightly improved for the prediction of R-134a refrigerant by Suliman et al. [36]. Despite the improvements, it was concluded that the general trend for this map and the others is that they show dependence on the range of experimental conditions and the choice of fluids used for their development.

The main findings of the experimental work on the condensation of R-134a refrigerant in smooth inclined tubes conducted at the University of Pretoria's thermoflow laboratory were then summarised. The experimental set-up used to capture the flow pattern images in these studies was also discussed.

The theory of ANNs was then given, followed by a review of the literary works investigating the use of ANNs trained by backpropagation algorithms for in-tube two-phase flow pattern identification. From the review, it was evident that there exists a gap in the literature concerning condensation in inclined tubes, and additionally, to the best of the author's knowledge, it appears that no study has yet investigated the use of visualisation data to predict the flow patterns for in-tube two-phase flows.

Chapter 3. Methodology

3.1 Introduction

The purpose of this chapter is to describe the analysis sequence followed in this study. Firstly, the data preparation methods followed to generate the flow pattern image datasets are described. Secondly, the method of PCA is defined, implemented and the results discussed. The analysis allows the visualisation of the image dataset because the extracted features can be projected to two-dimensional space in a meaningful way. The chapter concludes with a description of the deep learning models investigated in this study, as well as the training methodology followed for their implementation.

3.2 Data preparation

Data preparation is fundamental to ensure that deep learning models perform optimally, and includes the collection, cleaning, and transforming of raw data before being used for training or analysis. The correct preparation of data allows for efficient analysis, limited errors, and repeatable results [55].

3.2.1 Image acquisition

This study used ANNs (MLPs and CNNs) trained by supervised machine learning methods for the task of image classification. Such models are trained using labelled data. Hence, the first step is the acquisition of flow pattern images and their associated flow pattern class labels.

Section 2.5 discussed the experimental work conducted at the thermoflow laboratory at the University of Pretoria as well as the experimental set-up used to capture the flow pattern images during these studies. As a result of these studies, it was possible to acquire 3 961 flow pattern images covering a wide range of flow conditions and the full range of tube orientations.

3.2.2 Image pre-processing

Machine learning models are not trained or evaluated using raw data, or data acquired in its original format. Instead, the data needs to be transformed (pre-processed) to meet the requirements of the given task. Hence, before the images are supplied to the algorithms for training, a series of pre-processing steps are taken to enhance the quality of the data and make it suitable for model training such that meaningful features may be learnt and while ensuring that the required computation remains at an acceptable level.

3.2.2.1 Representing images in numerical format

Images converted to a numerical representation take the form of multidimensional arrays. The array size is determined by the number of pixels, corresponding to the original resolution of the image; and the element values are scalars in the range $[0,255]$, corresponding to the pixel intensities. The experimental images were obtained in RGB (colour) format, and subsequently, can originally be described by $3N_p$ features; where N_p is the number of pixels, and the factor 3 corresponds to the three colour channels (red, green and blue). The dimensionality of the images is immediately reduced by a factor of 3 by converting the images to greyscale (pixel intensity corresponding to black/white),

because the goal of the supervised learning process is the identification of flow patterns rather than the identification of colours.

3.2.2.2 Removing duplicate images from the dataset

It was important to find and remove duplicate images from the dataset because the experimental images originated from multiple sources. The inclusion of these duplicates would otherwise be problematic for two reasons: firstly, training on duplicate images allows the model multiple opportunities per epoch to learn patterns specific to the duplicates, which introduces bias and reduces the model's ability to generalise to new images; and secondly, if duplicate images become part of both the training and test datasets, the performance of the model will be exaggerated, as it would be evaluated on its ability to classify images that have been seen during the training process.

To identify the duplicates, a measure of the structural similarity index (SSIM) [104] was determined between each pair of images in the dataset. The SSIM measures the similarity between two images based on a comparison of the luminance (l), contrast (c), and structure (s). Hence, it gives a robust method to detect similar images despite slight modifications or distortions of brightness, contrast, or translation. An SSIM value of 1 represents two images completely identical images. In this study, a threshold value of 0.9 was used, and if the SSIM measure between two images was higher than the threshold, the images were manually inspected and removed when duplicates were confirmed. Mathematically, considering two images (x and y) with equal size, the three components of the SSIM can be determined using Equations (3.1) to (3.3):

$$l(x, y) = \frac{2\mu_x\mu_y + c_1}{\mu_x^2 + \mu_y^2 + c_1} \quad (3.1)$$

$$c(x, y) = \frac{2\sigma_x\sigma_y + c_2}{\sigma_x^2 + \sigma_y^2 + c_2} \quad (3.2)$$

$$s(x, y) = \frac{\sigma_{xy} + c_3}{\sigma_x\sigma_y + c_3} \quad (3.3)$$

where μ_x and μ_y refer to the mean pixel value in the images or the considered window of the images, σ_x^2 and σ_y^2 are the variance of the images, and σ_{xy} is the covariance between the two images. c_1 , c_2 and c_3 are variables to stabilise the division with a weak denominator. From the three determining components, the SSIM is then determined by Equation (3.4). In this study, equal weighting was assigned to each of the components ($\alpha = \beta = \gamma$), and Equation (3.4) can be simplified as Equation (3.5).

$$SSIM(x, y) = [l(x, y)^\alpha \cdot c(c, y)^\beta \cdot s(x, y)^\gamma] \quad (3.4)$$

$$SSIM(x, y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)} \quad (3.5)$$

It is noted that the reported number of acquired images, namely 3 961, was the number after the removal of duplicate images.

3.2.2.3 Cropping and downsampling

Each of the images is identically cropped to remove any unnecessary background at the top and bottom of the tube and then realigned to ensure all images are squarely orientated. Hobold and Da Silva [105] investigated the use of visualisation to predict pool-boiling regimes. They report that downsampling can be used as an initial method of dimensionality reduction while maintaining a large amount of the dataset variance. In this study, each of the images was downsampled to the dimensions of 96 by 256 pixels ($N_p = 24576$). As mentioned in Section 2.9.1, it is good practice to choose an input image size to a CNN which is divisible multiple times by a factor of 2 to avoid the degradation of image boundary information between convolutional and pooling layers. Hence, the specific size was selected to reduce the number of initial features by roughly a factor of 2, while maintaining the original aspect ratio of the images as close as possible, and ensuring that the dimensions were suitable as input to a CNN.

3.2.2.4 Normalising

Finally, each of the images was normalised to the range [0,1] such that each image was represented by the same brightness scale. Additionally, normalisation helps speed up convergence during neural network training.

3.2.3 Dataset generation

Once the images were pre-processed, they were labelled into 10 distinct flow pattern classes according to the flow pattern descriptions given in Section 2.4.1. Representative images for each of the flow pattern classes are shown in Table 3-1. Figure 3-1 shows a frequency histogram of the number of images in each of the classes. It can be seen that the number of images ranged from a minimum of 207 for the slug class to a maximum of 597 for the stratified-wavy class, with the mean being 396 images.

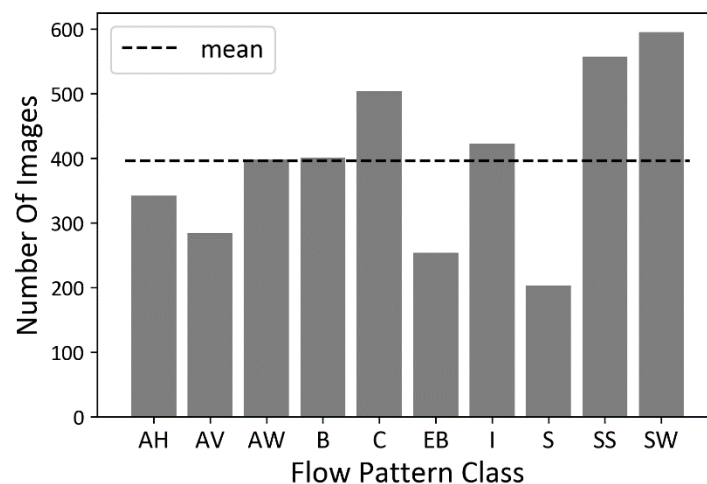
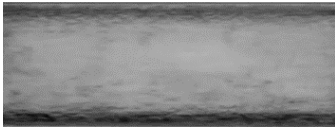
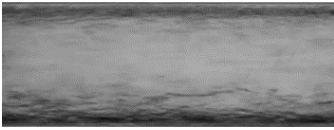
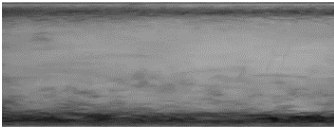
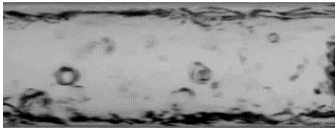
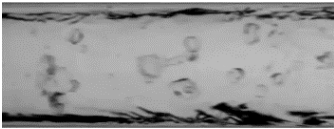
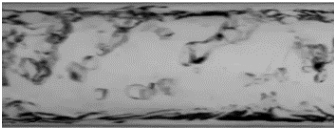
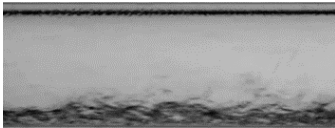
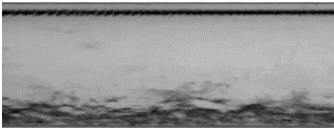
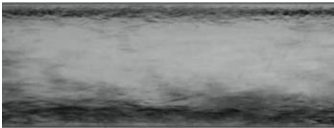
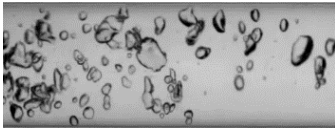
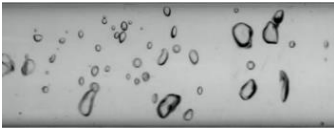
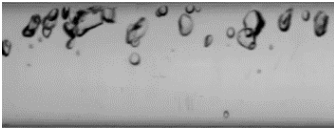
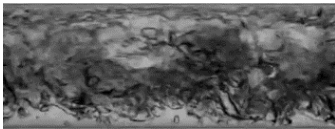
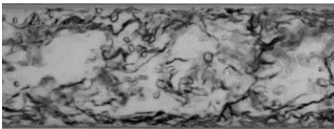
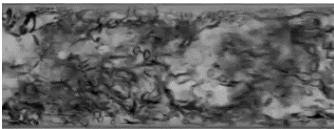
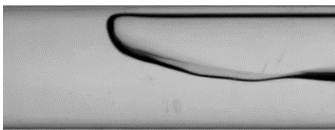
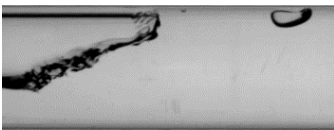

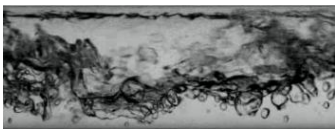
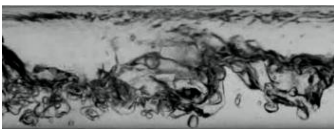
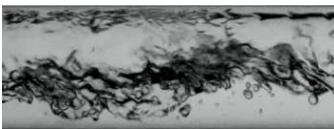
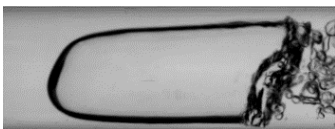

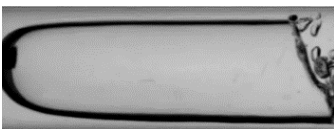
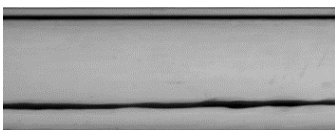
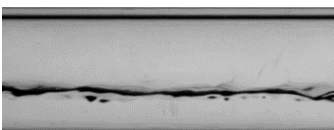
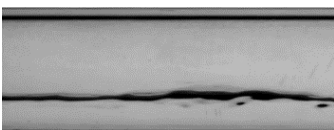
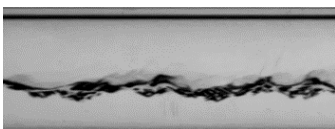

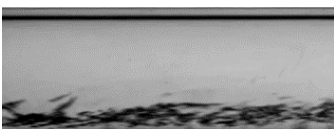


Figure 3-1: Frequency of each of the considered flow pattern classes

Table 3-1: Representative images of each of the flow pattern classes

Flow pattern	Abbr	Representative images		
Annular-horizontal	(AH)			
Annular-vertical	(AV)			
Annular-wavy	(AW)			
Bubbly	(B)			
Churn	(C)			
Elongated bubbles	(EB)			
Intermittent	(I)			
Slug	(S)			
Stratified-smooth	(SS)			
Stratified-wavy	(SW)			

Once the images were labelled, they were randomly separated into two datasets: a training dataset consisting of 75% of the images, to be used for training the various deep learning models considered in this study; and a test dataset consisting of the remaining 25% of the images, to be used for testing the models' performance on classifying or predicting unseen condensation flow pattern images. Although the images were randomly assigned to the two folders, they were separated in a stratified

manner such that a 75%/25% ratio was consistent among all ten classes of images. A balanced number of samples for each class in both the training and test datasets ensures that both datasets are an accurate reflection of the original dataset.

The training process considered a five-fold cross-validation process to determine the optimal choice of model hyperparameters. To accommodate this process, the training dataset was duplicated and randomly split in a stratified manner into five equal subsets, each consisting of 20% of the training dataset (15% of the original dataset). The subsets were used to find the optimal set of model hyperparameters, and then the full training dataset was used to train the final model.

3.3 Principal component analysis

MLPs require a one-dimensional array of input features, meaning that the two-dimensional image array needs to be flattened from a matrix to a vector before being fed as input to an MLP. Additionally, MLPs require one perceptron unit for each input feature. Hence, using the images in their current form (96 x 256) would require an extensive number of parameters, which, in turn, would create an expensive computational task to train and would likely lead to an overfitted solution. Apart from the resulting inefficiency, another problem is that MLPs are not image translation invariant, meaning that they are highly sensitive to slight translations of the input images because the spatial information is lost when the array is flattened into a vector.

PCA is a useful tool for both dimensionality reduction and feature extraction, which makes it possible to use MLPs trained on the resulting features for the task of image classification. Dimensionality reduction allows for the resulting computation to be lowered to an acceptable level, and the extraction of features based on spatial information improves the model generalisation capability. An added benefit of PCA is that it allows for visualisation of the dimensionally reduced dataset because the data can be projected onto a two- or three-dimensional space in a meaningful way. The visualisation allows insight into the structure of the dataset and gives information that can aid in the subsequent analysis.

3.3.1 Dimensionality reduction by PCA

PCA is a useful tool for dimensionality reduction because it reduces the number of features of the data, while maintaining the variance of the original dataset. PCA applies an orthogonal linear transformation to the dataset to a new coordinate system in which the vector space is composed of the eigenvectors of the covariance matrix (Σ) of the dataset. Each eigenvector represents the successive identification of the axis of greatest variance in the data (the principal components (PC)). The newly defined coordinate system optimally explains the dataset variance because the first principal component (dataset eigenvector with largest corresponding eigenvalue) explains the greatest variance, and each of the proceeding principal components explains a reduced amount of variance [62, 72]. The dimensionality of the data is reduced by retaining a fraction (K/N_p) of the principal components that retain an appropriate amount of the variance and discarding the rest.

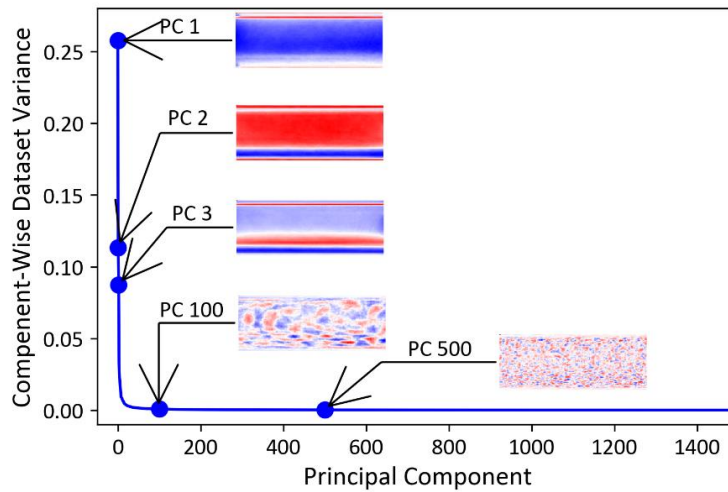
Mathematically, the implementation for the PCA is as follows [65]:

- 1) Starting with the image dataset $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \dots, \mathbf{x}_M\}$, create a matrix $\mathbf{X} \in \mathbb{R}^{M \times N_p}$ with each of the M image samples flattened into shape $(1 \times N_p)$ as rows

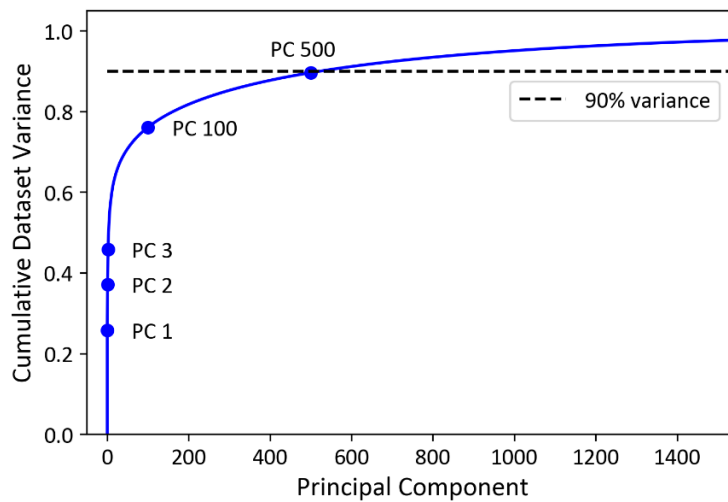
- 2) Compute the covariance matrix of the image dataset, $\Sigma = \mathbf{X}^T \mathbf{X}$
- 3) Compute the eigen decomposition of the covariance matrix, $\Sigma = \mathbf{V} \mathbf{\Omega} \mathbf{V}^T$, where $\mathbf{V} \in \mathbb{R}^{N_p \times N_p}$ is a matrix of the eigenvectors and $\mathbf{\Omega}$ is a diagonal matrix containing the eigenvalues
- 4) Select the largest K eigenvalues from $\mathbf{\Omega}$ and corresponding eigenvectors from \mathbf{V} , where $K < N_p$
- 5) Create a matrix $\mathbf{A} \in \mathbb{R}^{N_p \times K}$ with the K eigenvectors as columns
- 6) Project the dataset to the lower dimensional space through the linear transformation $\mathbf{Z} = \mathbf{X} \mathbf{A}$, where the resulting transformed dataset $\mathbf{Z} \in \mathbb{R}^{M \times K}$ has reduced dimensionality by the ratio K/N_p

Each eigenvector is of the same dimensionality (N_p) as the original images used to construct the covariance matrix, and therefore, can also be viewed as images with the same resolution. In the field of computer vision, these figures are named eigenfigures, or eigenfaces when applied to the problem of face recognition [62, 106]. The eigenfigures themselves form a basis set of all images used to construct the covariance matrix and can be thought of as the ingredients for the construction of the flow pattern images. In fact, an image in the dataset can be reconstructed through the sum of the eigenfigures weighted by the values of the principal component relating to each eigenfigure for the considered image.

These concepts can be better understood through visual media. Figure 3-2a shows the percentage of the dataset variance explained by each of the first 1 500 principal components (or eigenfigures), and Figure 3-2b shows the cumulative explained variance as a function of the number of included components. It can be seen that the first principal component accounts for roughly 26% of the dataset variance; however, the variance explained by subsequent components decays quickly below 1% within a few components. This trend is also evident in Figure 3-2b as the cumulative variance shows an asymptotic convergence towards 1. As a result, the inclusion of only five components is sufficient to explain more than 50% of the dataset variance, the inclusion of 100 components retain roughly 76%, and by using 500 principal components, over 90% of the total dataset variance can be retained and allows a dimensionality reduction of 50x from the original 24576 greyscale pixels. In Figure 3-2a, the eigenfigures corresponding to the first, second, third, 100th, and 500th principal components are also shown, where the blue and red colours show the spatial region of the eigenfigures contributing positively and negatively to the corresponding eigenvector.



(a)

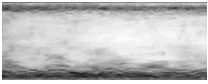

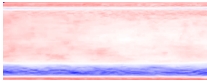
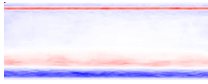
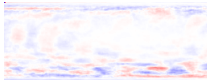
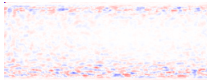
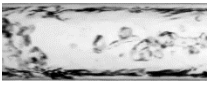

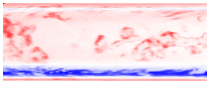
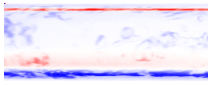
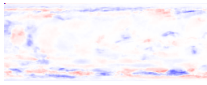
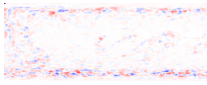
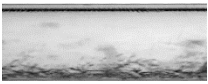

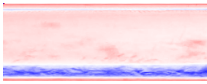
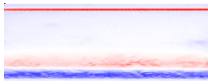
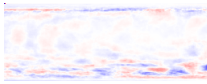
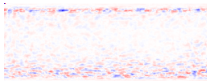
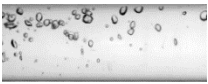
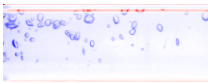
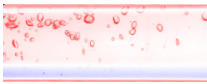
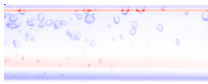

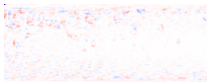
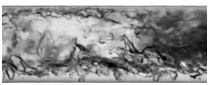
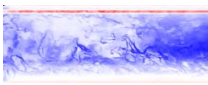
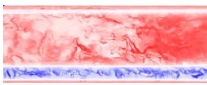
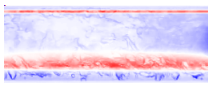
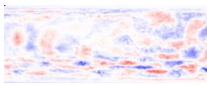
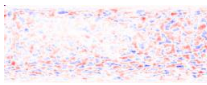

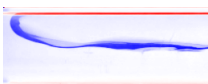


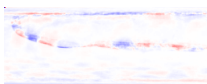
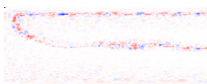
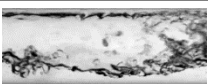
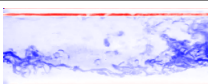
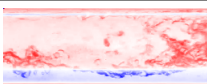
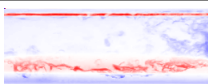
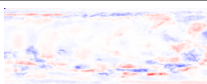
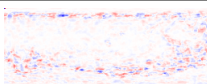
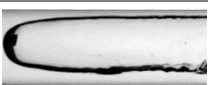
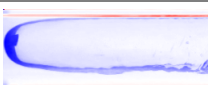
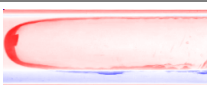

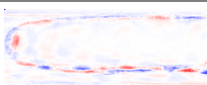
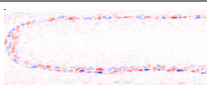
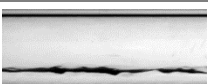

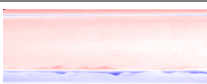

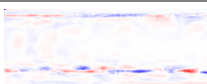
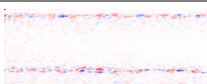
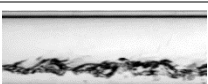
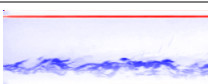
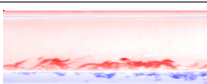

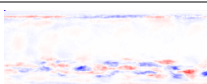
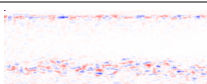


(b)

Figure 3-2: (a) Principal component-wise dataset variance, (b) cumulative dataset variance





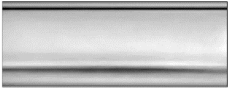
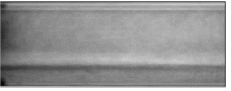

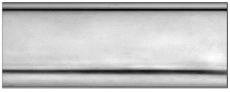
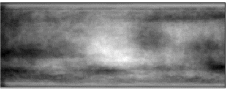

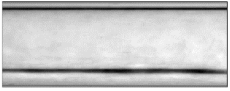
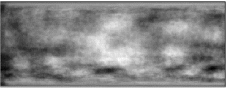
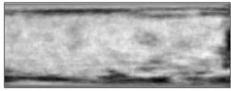
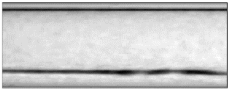
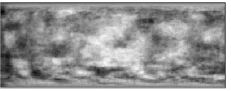
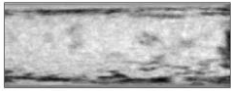
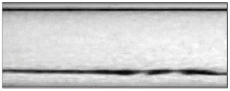
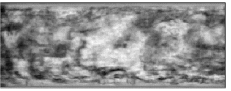
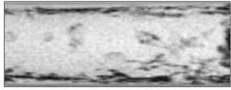
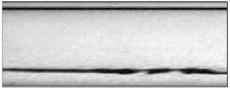
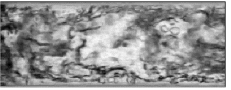
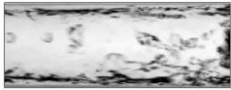
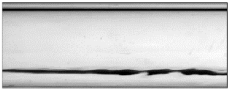
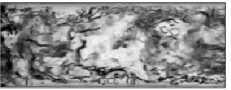
To better explain the significance of the eigenfigures, Table 3-2 shows a randomly chosen image from each class weighted by each of the considered eigenfigures, respectively. It appears that the first eigenfigure places a lot of weight on the horizontal line seen at the top of the tube in the stratified, stratified-wavy, elongated bubble, and annular-wavy flow patterns. This observation shows that the training process may be biased towards the experimental set-up of the dataset because the horizontal line at the top of the tube is a characteristic of the experimental set-up rather than a characteristic of the specific flow patterns. The second and third eigenfigures place more weight on the location of the interface at the bottom of the tube, with the blue areas being more prevalent with the annular flow patterns and the red being more prevalent with the stratified flow patterns. The 100th and 500th eigenfigures appear to model noise in the dataset, with the size of the noise structures becoming smaller with less significant eigenfigures. This observation justifies the exclusion of less significant eigenfigures from the transformed dataset and subsequent training process because learning based on the noise is likely to lead to an overfitted solution. Therefore PCA also assists as a method of regularisation, and the choice of the number of principal components to include in the analysis is a hyperparameter that needs to be considered.

Table 3-2: Randomly selected flow pattern images from each class weighted by the first, second, third, 100th, and 500th most significant eigenfigures

Principal Component / Eigenfigure		1	2	3	100	500
Class	Original images	Weighted images				
AH						
AV						
AW						
B						
C						
EB						
I						
S						
SS						
SW						

Finally, Table 3-3 shows the process of reconstructing an image by including subsequently more eigenfigures. The summation of the eigenfigures weighted by the corresponding principal component values for a given image allows the original image to be recovered. The process is shown for a randomly selected image from the annular-vertical, stratified, and churn flow patterns. It can be seen that as more eigenfigures are subsequently included, the clarity of the image improves. Additionally, even with very few included eigenfigures, the differences between the images from the different classes can be seen. The stratified-smooth flow pattern image can be identified using just 10 eigenfigures, while the churn flow pattern image requires more eigenfigures because it is more dependent on the noise structures.

Table 3-3: Process of image reconstruction by including subsequently more eigenfigures for three randomly selected flow pattern images

Number of included eigenfigures	Reconstructed images		
	AV	SS	C
3			
10			
50			
100			
250			
500			
1000			
all			

3.3.2 Dataset visualisation

Another benefit of PCA is that a significant amount of the variance is captured by two or three components, and it can subsequently be plotted onto two- or three-dimensional space in a meaningful way. Figure 3-3 shows the image dataset projected to the first three principal components, which collectively compromise for 46% of the total variance in the dataset. Using just three principal components, there is a definite separation between the 10 classes of data as they appear to form distinctive groups in two-dimensional space. Additionally, the flow pattern classes showing similar characteristics can be identified in the plot. For instance, the three classes of annular flow patterns (shown in blue, red and magenta) form neighbouring groupings in all three of the plots. On the other hand, the overlap between the classes suggests that it may be difficult to train a classifier with high accuracy. For instance, the bubbly and slug flow pattern groups (shown in cyan and orange) show a large degree of overlap in all three of the figures and therefore are likely to be misclassified for each other.

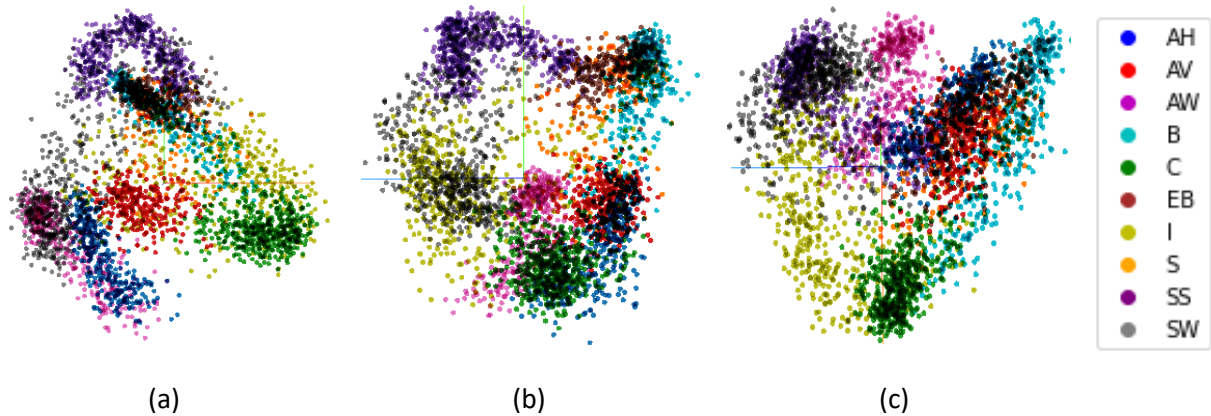


Figure 3-3: The image dataset projected to its first three principal components, viewed from: (a) the first and second components, (b) the first and third components, (c) the second and third components

3.4 Deep learning models

In this study, both MLPs and CNNs were employed for the task of image classification. The MLP was trained using the extracted principal components from the PCA, while the CNN was trained directly on the pre-processed images in the training dataset. In both cases, the choice of model hyperparameters needed to be fine-tuned to optimise the models' predictive performance. This section summarises the training methodology followed to find the best set of model hyperparameters for both models, and then subsequently describes the specific set of hyperparameters investigated and selected for each of the models.

The models presented in this section were implemented using the Keras™ application programming interface (API) in TensorFlow™, which is an end-to-end open-source machine learning platform run using Python™ [107]. For any model attributes not specified, the default settings of the TensorFlow models can be assumed.

3.4.1 Training methodology

Figure 3-4 algorithmically depicts the training methodology followed in this study. Figure 3-4a shows the process of training and evaluating the deep learning models based on the performance of both the training and test datasets. The process of data preparation leading to the formation of the labelled (original) dataset; and subsequent generation of training, test, and cross-validation datasets was described in Section 3.2. With the datasets prepared, the models can be trained and their performance evaluated. However, to ensure that the models are trained to achieve the best possible performance, it is important to fine-tune the model hyperparameters, which is equivalent to the process of optimising the model for the specific task and the specific dataset.

The hyperparameters specific to each of the models considered in this investigation are discussed in the following sections. However, in both cases, a five-fold cross-validation and grid search process was implemented to determine the optimal model hyperparameters, which is shown in Figure 3-4b. To implement cross-validation, the training set was partitioned into five equal subsets, referred to as folds. Each of the five folds successively acted as the test set, while the remaining four folds acted as the training set. The cost function associated with a specific set of model hyperparameters is determined as the average of the cost function over the five folds. The optimal set of hyperparameters

was determined by iteratively evaluating the average cost function for every unique set of model hyperparameters based on a grid search of all considered hyperparameters within a specified range.

Figure 3-4c shows the model training process used to update the model parameters.

- The training process started with the initialisation of the model parameters (weights). In this investigation, the network weights were automatically initialised on the Tensorflow platform with the Xavier [108] weight initialisation method. The method is the default method on the platform, and initialises the weights from a uniform distribution, within specific limits determined by the number of input and output units forming the weight matrix for a respective layer. The weight initialisation method was left as the default because the method has been validated to substantially improve neural network convergence for well-established image classification tasks, and copes well with preventing the issue of vanishing gradients in deep learning models [65].
- Following the parameter initialisation, the model was trained for a specified number of epochs (Epoch*), with the training dataset shuffled before each epoch of training to ensure that there was no correlation between successive training samples.
- For each epoch of training, the model parameters were updated multiple times depending on the number of batches (Batch*), which was determined as the number of samples in the training set divided by the number of training samples included in each batch (batch size hyperparameter).
- For each batch of samples, the model parameters were updated according to the training process outlined in Sections 2.8.4 and 2.8.5:
 - firstly, the model outputs are determined by the forward propagation of the input (X) to the model (Section 2.8.4.1, Algorithm 1);
 - secondly, the loss function per training sample and corresponding cost function for the batch are then determined by comparing the model outputs (\hat{Y}) to the labelled (true) outputs (Y) of the training data (Section 2.8.4.2);
 - thirdly, the gradient of the cost function at each layer in the neural network model is determined using the backpropagation algorithm (Section 2.8.4.3 and Algorithm 2 for MLP, Section 2.9.3 for CNN);
 - finally, the model parameters are updated using the Adam optimisation algorithm (Section 2.8.5.2, Algorithm 3).
- The training process is ended when this process has been repeated for the specified number of batches and epochs.
- Once the model has been trained, its performance is evaluated on a test or validation dataset.

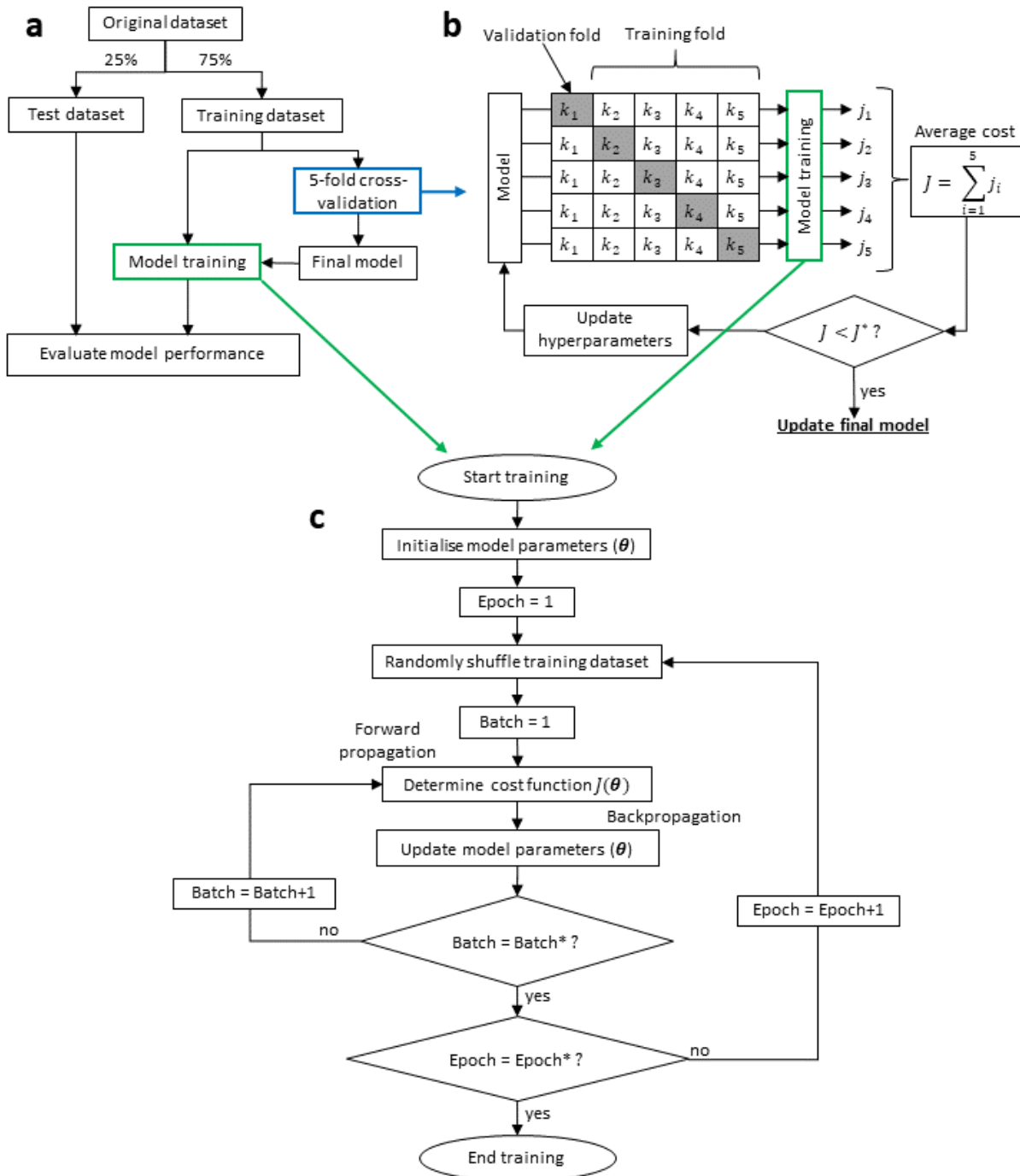


Figure 3-4: (a) Process of training and evaluating the deep learning models, (b) cross-validation process used to fine-tune model hyperparameters, (c) comprehensive flow chart of the model training process

3.4.2 Multilayer perceptron final model

The design of an MLP requires the consideration of many model hyperparameters relating to the model architecture, training, regularisation, and input feature section.

Specific to the architecture, the complexity of the model and the resulting number of model parameters are determined by the number of hidden layers and the number of neurons in each layer. Additionally, the use of the principal components as the input to the model adds an additional

hyperparameter because the number of components to be included needs to be chosen. Section 3.3.1 showed that the first principal components accounted for a significant proportion of the total dataset variance, while higher principal components seemed to model noise. Consequently, enough components need to be included to capture the variance in the dataset, while excluding some of the components may reduce the effects of overfitting because only the most significant features of the dataset are available to the algorithm during training.

Considering the training process, the batch size and learning rate control the speed at which the model learns, as well as the final accuracy the model achieves. The batch size considers the number of training samples used for each iteration to determine the cost function and subsequently, update the model parameters. The amount by which the parameters are updated per batch is determined by the learning rate coupled together with the choice of optimisation algorithm (and associated hyperparameters). As mentioned in Section 2.8.5.1, the combination of these hyperparameters is fundamentally important in optimising the models' performance due to the complexity of the solution space associated with deep learning models.

The choice of activation function in each layer in the network and the choice of loss function are important hyperparameters also related to training. The ReLu activation function (Equation (2.13)) is used in the hidden layers. As discussed in Section 2.8.3, the ReLu function has a favourable ability to accelerate training in deep learning models because it is less susceptible to the problem of vanishing gradients [55]. The softmax function (Equation (2.14)) is used in the output layer together with the cross-entropy loss function (Equation (2.19)), which, as discussed in Sections 2.8.3 and 2.8.4.2, are well suited to multi-class classification problems.

The hyperparameters controlling the regularisation of the model include the L2 constant term (λ in Equation (2.32)), which controls the strength of the L2 regularisation penalty term, and the choice of dropout probability ratio and where in the model dropout is implemented.

To determine the optimal combination of model hyperparameters, an automated search is usually used which considers hyperparameter combinations within a specified range. There are three typical strategies [55, 61, 109]:

- *Exhaustive search*: All possible hyperparameter combinations are explored. However, the method is only feasible for small models and small datasets due to the required computation.
- *Random search*: Hyperparameter combinations are randomly selected.
- *Grid search*: Hyperparameter combinations are plotted on a grid (matrix).

As mentioned in the previous section, Section 3.4.1, the training methodology used in this study makes use of five-fold cross-validation using the grid search strategy. The range of hyperparameters was first determined by the author through trial and error; by determining the range beyond which the model performance was significantly degraded. The specific combinations explored were then plotted in a matrix, where it is noted that a more elaborate or detailed grid search could have led to an improved solution.

Table 3-4 summarises the model hyperparameters considered during the grid search cross-validation process for the MLP. Network architectures with one, two, and three hidden layers were considered, with the number of neurons in each layer being equal to the number of principal components used as input features, for which values between 50 and 750 were considered. The learning rate was considered in the range $[1e^{-5}, 1e^{-2}]$, L2 regularisation constant term in the range $[1e^{-8}, 1e^{-2}]$

implemented on the weights of each of the hidden layers, and the batch size varied between four and 128. The dropout probability ratio of 0.5 was implemented between each of the hidden layers and was not part of the cross-validation process. The combination of hyperparameters for the final model were determined as the combination which had the lowest average cost.

Table 3-4: Model hyperparameters considered for the MLP cross-validation grid search process

Hyperparameter	Considered range / value	Final model
Architecture		
Number of hidden layers	[1,2,3]	2
Number of neurons per hidden layer	[50,100,250,500,750]	100
Number of included principal components	[50,100,250,500,750]	100
Training		
Batch size	[4,8,16,32,64,128]	32
Learning rate	[$1e^{-5}$, $1e^{-4}$, $1e^{-3}$, $1e^{-2}$]	$1e^{-4}$
Weight initialisation method	-	Xavier [108]
Optimisation algorithm	-	Adam, $\beta_1 = 0.9$, $\beta_2 = 0.999$
Hidden layer activation function	-	ReLU
Output layer activation	-	Softmax
Loss function	-	Cross-entropy
Regularisation		
Dropout	-	0.5 – implemented between hidden layers
L2 regularisation constant (λ)	[$1e^{-8}$, $1e^{-7}$, $1e^{-6}$, $1e^{-5}$, $1e^{-4}$, $1e^{-3}$, $1e^{-2}$]	$1e^{-5}$
Resulting model parameters to train = 21 210		

The final MLP model is shown in Figure 3-5. The network architecture consisted of two hidden layers of 100 neurons in each layer. Hence, only 100 principal components, accounting for 76% of the dataset variance, led to the lowest average value for the cost function during the cross-validation process. Additionally, a batch size of 32, a learning rate of $1e^{-4}$, a dropout probability ratio of 0.5 implemented on the weights connecting the two hidden layers, and an L2 regularisation constant of $1e^{-5}$ implemented on each of the two hidden layers were determined. To train the MLP model during the cross-validation process and training of the final model, the Adam optimisation algorithm was used. The parameters associated with the algorithm, namely β_1 and β_2 , corresponding to the exponential

decay rates for the first- and second-moment estimates respectively, were left as the default values of 0.9 and 0.999, as suggested by the authors [67], and which are also the default values in the Tensorflow platform. Also depicted in the figure is the model prediction at the output layer, which shows how the neurons in the output layer were activated to give the prediction (\hat{y}) for a given training sample.

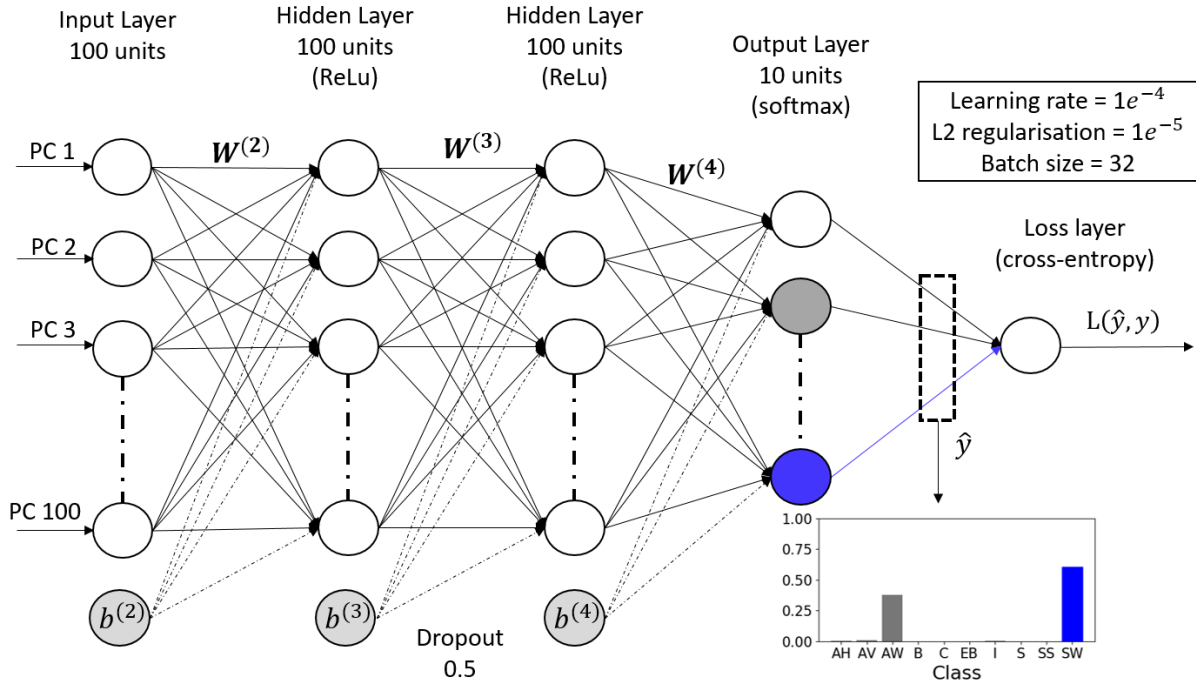


Figure 3-5: MLP final model architecture and hyperparameters

3.4.3 Convolutional neural network final model

The design of the CNN followed a similar process as previously described for the MLP, except that there were significantly more hyperparameters which needed to be considered due to the feature extraction process in the convolutional layers.

Firstly, it is noted that the fully connected layers of a CNN and associated hyperparameters are the same as those previously described for an MLP. The output from the convolutional layers forms the input to the fully connected layers. Hence, the convolutional layers can be thought of as a replacement to the PCA, extracting features to be used as the input to the MLP. The aim of using the convolutional layers is that the extracted features are more applicable to the task of image classification because they are determined from two-dimensional spatial information.

The additional hyperparameters associated with the architecture of the convolutional layers include the number of convolutional layers, the number of kernel filters in each layer, the size of the filters, the stride length, and the thickness of the padding used. Additionally, the use of max-pooling layers and associated hyperparameters of the max-pooling region size and stride length also needs to be considered. It is noted that the hyperparameters associated with the convolutional and pooling layers were discussed in Section 2.9.1 and Section 2.9.2, respectively.

These hyperparameters and those previously described of the fully connected layers were again determined with the cross-validation grid search process, while also aiming to minimise the number of parameters in the resulting network to keep the computation at an acceptable level. Therefore, if

the resulting average cost function during the cross-validation process was not improved by more than 5%, the set of hyperparameters with the lesser associated computational requirement was selected.

Table 3-5 summarises the model hyperparameters considered during the cross-validation grid search process for the CNN. For the network architecture, three and four convolutional layers each consisting of between four and 64 convolutional kernel filters were considered. For the first convolutional layer, kernel filter sizes of 3x3, 5x5, and 7x7 were considered together with a stride length of either 1 or 2, while the kernel filter size in subsequent layers was maintained at 3x3 with a stride length of 1. Max-pooling layers with a region size of 2x2 pixels and a stride length of 2 were used between each of the convolutional layers to reduce the computation in the network and were not part of the validation process. Only a single fully connected layer consisting of between 250 and 5000 neurons was considered. The learning rate was considered in the range $[1e^{-5}, 1e^{-2}]$, L2 constant term was in the range $[1e^{-8}, 1e^{-2}]$ implemented on the weights of the fully connected layer, and the batch size varied between 4 and 128. A dropout layer with a dropout probability ratio of 0.5 was implemented before the fully connected layer and was again not part of the cross-validation process.

Table 3-5: Model hyperparameters considered for the CNN cross-validation grid search process

Hyperparameter	Considered range / value	Final model
Architecture		
Input image size	-	96x256 pixels
Number of convolutional layers	[3,4]	3
Number of filters per convolutional layer	[4,8,16,32,64]	8,16,16 in three layers respectively
Kernel filter size in first layer	[3x3,5x5,7x7]	5x5
Stride length	[1,2] - in first layer, 1 in subsequent layers	2 in first layer, 1 in subsequent layers
Maxpooling region size	-	2x2 (with stride 2)
Padding	-	Padding to maintain image spatial size
Number of fully connected layers	[1,2]	1
Number of neurons per fully connected layer	[250,500,1000,2500,5000]	1000
Training		
Batch size	[4,16,32,128]	32
Learning rate	$[1e^{-5}, 1e^{-4}, 1e^{-3}, 1e^{-2}]$	$1e^{-4}$
Weight initialisation method	-	Xavier [108]

Optimisation algorithm		Adam, $\beta_1 = 0.9, \beta_2 = 0.999$
Convolutional layer activation function	-	ReLU
Fully connected layer activation function	-	ReLU
Output layer activation	-	Softmax
Loss function	-	Cross-entropy
Regularisation		
Dropout	-	0.5 – implemented between hidden layers
L2 regularisation constant (λ)	$[1e^{-8}, 1e^{-7}, 1e^{-6}, 1e^{-5},$ $1e^{-4}, 1e^{-3}, 1e^{-2}]$	$1e^{-6}$
Resulting model parameters to train = 1 550 706		
3696 in convolutional layers		
1 547 010 in fully connected layers		

The resulting final model for the CNN is shown in Figure 3-6. The architecture consisted of three convolutional and max-pooling layers with eight, 16 and 16 convolutional kernels of size 5×5 , 3×3 and 3×3 in each layer, respectively. A stride length of 2 was used in the first convolutional layer and a stride length of 1 was used in the second and third convolutional layers, respectively. A max-pooling region size of 2×2 and a stride length of 2 were used in each layer to reduce the spatial size of the feature maps between the convolutional layers. The result was feature maps of size 6 by 16. After flattening the extracted features into a vector, they were fed as input to a fully connected layer containing 1 000 neurons, and then an output layer with 10 neurons. A dropout layer with a dropout probability of 0.5 placed before the fully connected layer and an L2 penalty term with an L2 constant of $1e^{-6}$ implemented on the fully connected layer were included for network regularisation. Training took place with a batch size of 32, learning rate $1e^{-4}$, and with the ReLU activation function at each layer in the network except for the output layer, for which the softmax activation function was used. The cross-entropy loss function was optimised with the Adam optimisation algorithm and with the TensorFlow default learning hyperparameters of 0.9 and 0.999 for β_1 and β_2 , respectively.

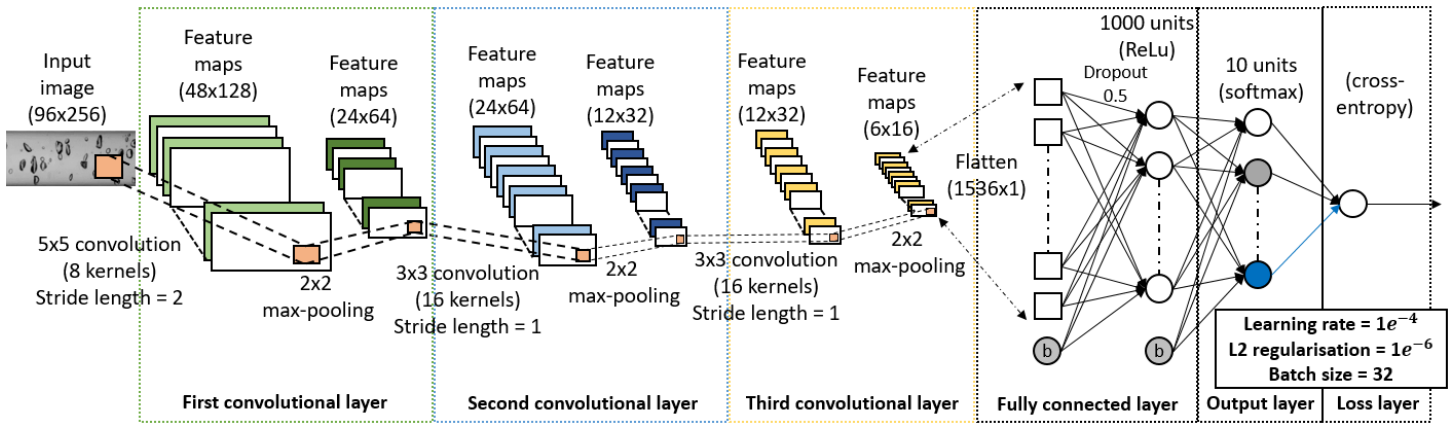


Figure 3-6: CNN final model architecture and hyperparameters

3.5 Conclusion

This chapter described the analysis sequence followed to develop and train the ANNs investigated in this study. A total of 3 961 unique flow pattern images were obtained, pre-processed, and labelled into 10 distinct flow pattern classes, which formed the original or total image dataset. From the original dataset, the training, cross-validation, and test datasets were then formed. Representative images from each of the considered classes were also shown.

The PCA allowed for dimensionality reduction and the extraction of features which were applicable to training the MLP. Additionally, the PCA allowed insight into the structure of the image dataset because it showed how the variance of the dataset was distributed in high-dimensional space and allowed the visualisation of the data in two-dimensional space. It was concluded that the data should be relatively straightforward to classify because the classes formed distinctive groups when projected onto the first three principal components. However, due to the overlap between classes, achieving a high classification accuracy may pose a difficult task.

The chapter concluded with a description of the deep learning models and the training methodology followed for their implementation. The method followed to fine-tune the model hyperparameters were discussed, and the resulting final model architectures and associated hyperparameters were then presented.

Chapter 4. Results and discussion

4.1 Introduction

The purpose of this chapter is to present and discuss the results of the deep learning models. First, the criteria used to evaluate the performance of the classifiers are discussed, and then the results for both the MLP and the CNN are given.

4.2 Classifier performance criteria

Many metrics have been developed to evaluate classifier performance [110]. It is of vital importance to adopt the correct performance measures to make accurate conclusions and comparisons about classifier performance for a particular problem. For this reason, several performance metrics were included in this investigation, such as accuracy, precision, recall, and F-score.

- *Accuracy* is defined as a fraction of correctly classified identifications across all classes and is useful to summarise the model's performance. However, *accuracy* can be misleading, especially when describing imbalanced datasets, as is the case in this paper where the number of images in the classes ranged between 207 and 597.
- *Precision* is defined as a fraction of true positives (correct prediction of the positive class) divided by the total number of positive identifications and is the most appropriate metric when the focus is to minimise false positives (incorrect prediction of the positive class). More simply put, precision is a measure of the ability of a classifier to predict only the positive instances for each of the classes.
- *Recall* is defined as a fraction of the true positives that were identified and is the most appropriate metric when the focus is to minimise false negatives (incorrect prediction of the negative class).

It would not be difficult to implement a classifier with either 100% precision or 100% recall for a specific class. However, implementing a classifier with an excellent prediction for all classes requires both high precision and high recall.

- The *F-score* provides a balance between precision and recall into a single metric, and is mathematically defined as:

$$F = \frac{2PR}{P + R} \quad (4.1)$$

where P and R refer to the precision and recall metrics scores respectively. The F-score is always lower than the arithmetic mean between precision and recall and is zero if either metric is zero.

After determining the architecture and other hyperparameters of the final models from the five-fold cross-validation and grid search process, the final models were trained using the full training dataset. To achieve a reliable representation of the classifiers' performance in this section, the MLP was run 50 times for 200 epochs, and the CNN was run 50 times for 50 epochs, with randomly initialised weights for each training run.

4.3 Performance of the MLP

Figure 4-1 shows the mean learning curves and the 95% confidence bands of the training and test set accuracy and loss as a function of the number of training epochs.

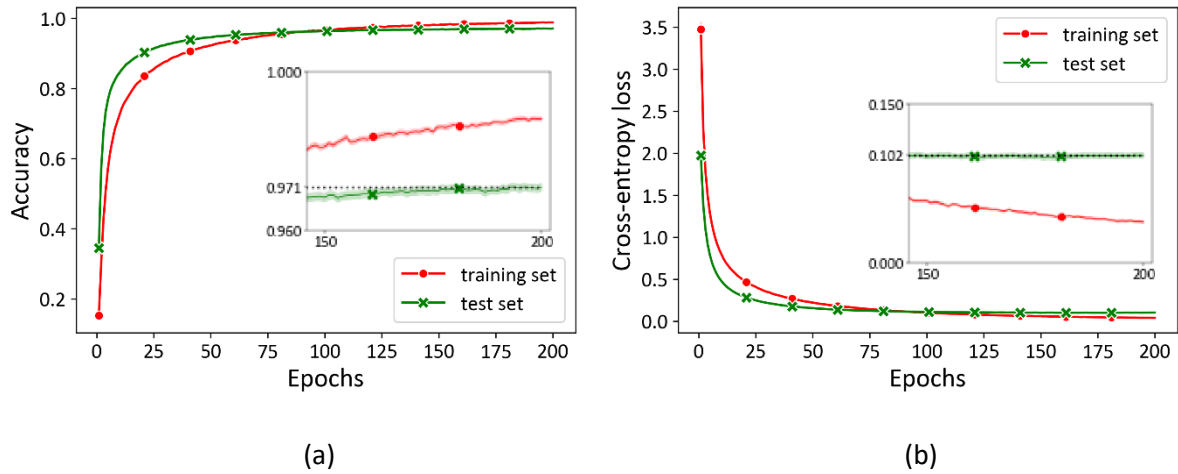


Figure 4-1: MLPNN mean learning curves as a function of the number of epochs trained: (a) Accuracy, (b) Cross-entropy loss

Initially, the test set performance exceeded that of the training set due to the influence of the dropout layer. This is because the test set score during training was determined using all of the neurons in the network, while the training score was determined using only the neurons in the network that was active for the considered training iteration. After training for 200 epochs, the test set loss is seen to have converged for the last 50 epochs of training, while the training set loss asymptotically approached 0. A similar trend is seen for the mean accuracy curves, with the test set accuracy reaching a nearly constant value of 0.971, while the training set accuracy asymptotically approached 1. This converging behaviour indicates that the model hyperparameters were well determined to ensure that the model was regularised and prevented overfitting. Additionally, it is noted that the results are highly repeatable because the 95% confidence band can only be seen when zooming into the learning curves, and even then are very narrow.

Table 4-1 presents the mean performance metric results on the test set for each of the flow pattern classes. The results shown should be considered together with the normalised confusion matrix, given by Figure 4-2. A confusion matrix, commonly referred to as a misclassification table, shows the combination of the predicted and actual classes. The rows in the table show the instances of a predicted class, while the columns show the instances of a true class. The diagonal terms represent the overlap between predicted and true classes (correct classification) and those off the diagonal show where samples were misclassified. Additionally, normalising the confusion matrix represents a percentage measure of the results such that imbalanced classes can be compared. By comparing the performance metrics with the confusion matrix, it is possible to get an indication of which classes are most easily confused. For instance, the lowest metric was the precision score for the slug flow pattern class of 0.8125. With reference to the confusion matrix, it can be seen that 5% and 4% of the elongated bubbles and bubbly flow pattern classes, respectively, were misclassified as slug flow, and are the main contributors to the low score. The result makes sense because many of the slug flow pattern images also show bubbles at the tails of the slugs; and the images from the slug and elongated bubbles flow pattern classes show distinct similarities, which also led to 3% of slug images being misclassified

as elongated bubbles. Overall, the test set accuracy score of 97.09% is impressive, considering that the MLP was efficiently trained on only the first 100 principal components of the training dataset.

Table 4-1: Mean performance metrics of the MLP on the test dataset

Class	Precision	Recall	F-Score	Accuracy
AH	1.0000	0.9931	0.9965	0.9709
AV	0.9983	0.9804	0.9893	
AW	0.9952	0.9940	0.9946	
B	0.9738	0.9545	0.9641	
C	0.9770	0.9522	0.9644	
EB	0.9568	0.9496	0.9532	
I	0.9387	0.9651	0.9517	
S	0.8125	0.9579	0.8793	
SS	0.9961	0.9781	0.9870	
SW	0.9773	0.9755	0.9764	

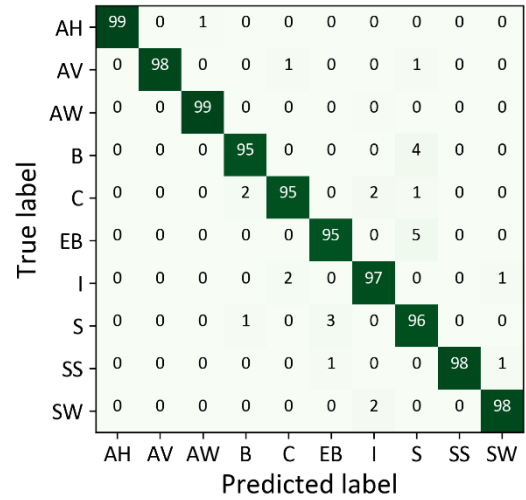


Figure 4-2: Normalised confusion matrix for the MLP results

4.4 Performance of the CNN

Figure 4-3 shows the mean learning curves and the 95% confidence bands of the training and test set accuracy and loss as a function of the number of training epochs.

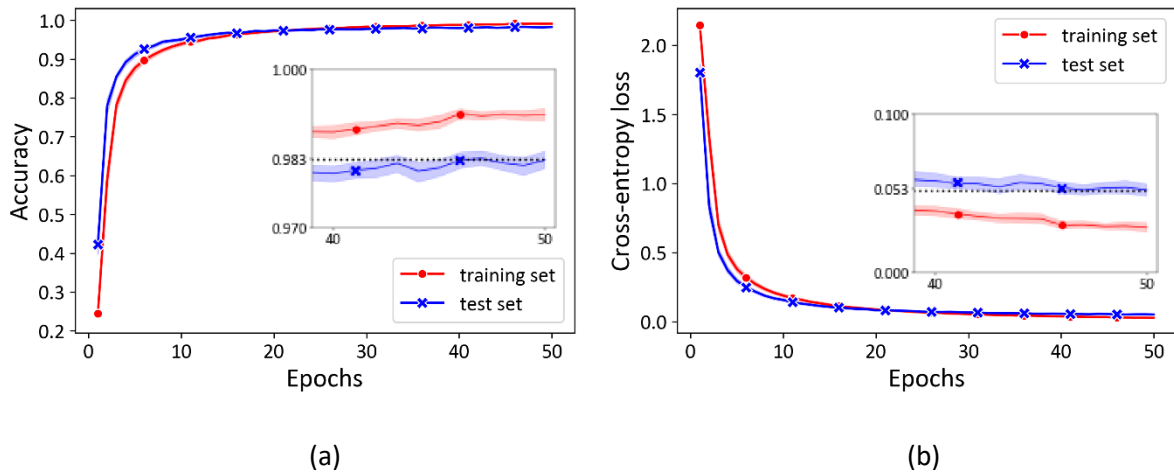


Figure 4-3: CNN mean learning curves as a function of the number of epochs trained: (a) Accuracy, (b) Cross-entropy loss

A similar trend is seen when comparing the learning curves for the CNN with those of the MLP. Initially, the test set performance exceeded that of the training set until the test set curves converged towards near-constant values and the training curves asymptotically approached 0 and 1 for the loss and accuracy, respectively. However, compared with the learning curves seen for the MLP, the training process was accelerated for each training epoch for the CNN due to the significant increase in model

parameters. Additionally, the results were significantly improved over the MLP after just 50 epochs of training because the mean accuracy increased from 0.971 to 0.983, and the mean loss decreased from 0.102 to 0.053. The improvement is significant because the misclassification rate decreased by 41%. The width of the 95% confidence interval bands was greater than those seen for the MLP; however, they are still very narrow and it can be concluded that the results are highly repeatable.

The mean performance metrics on the test set are presented in Table 4-2, and the normalised confusion matrix is shown in Figure 4-4. All of the metrics were above 95% with the exception of the slug flow pattern class, which again suffered from a low precision score of 0.8586 because the bubbly and elongated bubbles flow pattern classes were again commonly misclassified as slug flow. It is noted that the CNN achieved near-perfect test scores for all of the annular flow pattern classes.

Table 4-2: Mean performance metrics of the CNN on the test dataset

Class	Precision	Recall	F-Score	Accuracy
AH	0.9988	0.9988	0.9988	0.9832
AV	1.0000	0.9981	0.9991	
AW	0.9990	0.9963	0.9977	
B	0.9897	0.9652	0.9773	
C	0.9989	0.9770	0.9878	
EB	0.9589	0.9504	0.9547	
I	0.9664	0.9929	0.9794	
S	0.8586	0.9624	0.9075	
SS	0.9928	0.9843	0.9885	
SW	0.9940	0.9889	0.9914	

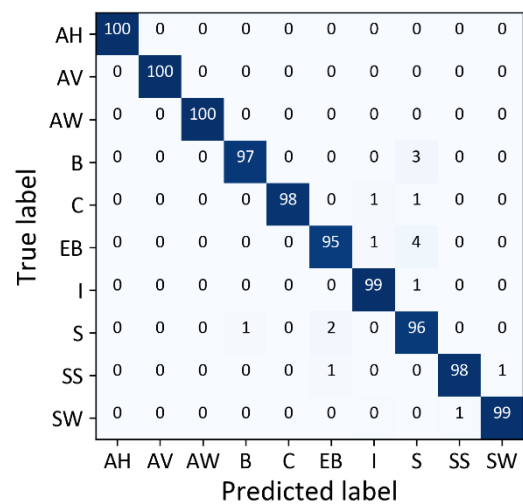


Figure 4-4: Normalised confusion matrix for the CNN results

Figure 4-5 shows a comparison of the F-scores on each of the 10 flow pattern classes between the MLP and the CNN. The performance of the CNN exceeded that of the MLP for the prediction of all 10 flow pattern classes, with the most significant improvements being for the churn, intermittent, and slug flow pattern classes. The results justify the increased computational expense associated with the CNN and highlights the favourable ability of CNNs to extract features based on two-dimensional spatial information.

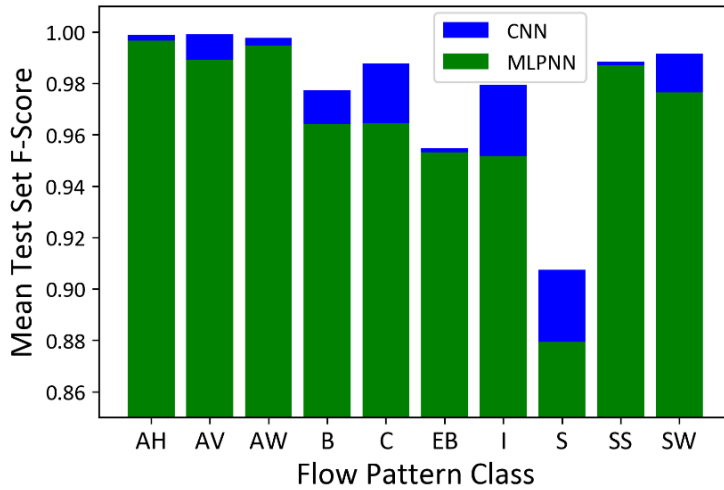


Figure 4-5: Comparison between the MLPNN and CNN mean test set F-scores on the 10 flow pattern classes

Table 4-3 displays the average execution time per prediction on the test set for the MLP and CNN, respectively. The total time per prediction was determined by predicting the entire test dataset 30 times and then determining the average prediction time per image. It was found that the prediction time using the MLP was 0.046ms, and the prediction time using the CNN was 0.477ms; which was an order of magnitude longer than for the MLP. When repeating the analysis including the time required for pre-processing the images (0.077ms); and additionally, in the case of the MLP, to flatten the image array into a vector and compute the PCA transformation to the first 100 principal components (0.041ms); it was found that the MLP was more than three times quicker per prediction. This relationship is depicted in Figure 4-6. However, the prediction time for the CNN (0.554ms) was still sufficiently fast to allow the trained model to be coupled to real-time flow pattern identification. In fact, in Section 2.5.2 it was specified that the high-speed video camera was capable of capturing images at 200fps (one image every 5ms). Hence, the prediction time was quicker than that required by the camera to capture images. The calculations were done on a personal computer (Intel(R) Core™ i7-3970X CPU @ 3.50GHz, 32GB RAM) and Python 3.8.

Table 4-3: Average execution time per prediction on the test set

Average time per test set image (ms)		
	MLP	CNN
Prediction	0.046	0.477
Pre-processing	0.077	0.077
PCA transform	0.041	
Total	0.164	0.554

Classifier	MLP prediction (ms)	PCA transform (ms)	pre-processing (ms)	Total (ms)
MLP	0.046	0.041	0.077	0.164
CNN	0.477	0	0.077	0.554

Figure 4-6: Total prediction time per image

Table 4-4 demonstrates the prediction of the CNN on select images from the test dataset. For each image considered, the prediction probability is shown for all 10 of the considered classes, with the total probability from all 10 classes summing to 1, which, as described in Section 2.8.3, is the result of the output layer of the CNN using the softmax activation function. For each figure, the blue text above the flow pattern images is associated with a correct prediction, while the red text is associated with an incorrect prediction. Similarly, the bar chart associated with each prediction contains bars in blue colour representing a fraction of the prediction towards the correct class; red colour representing a fraction of the prediction towards the incorrect class that was incorrectly predicted with the highest probability; and grey colour representing an incorrect class that shared part of the prediction but was not the highest-scoring class. Viewing the prediction in this way allows additional insight. For instance, the slug flow pattern image that was misclassified as bubbly flow could be accepted as being classified into the bubbly flow pattern class because dispersed bubbles are seen covering at least 50% of the image. The bubbly flow pattern image that was misclassified as churn flow contains many dispersed bubbles that are seen interacting and with a slightly turbulent nature, which makes the misclassification plausible. The second annular-horizontal flow pattern prediction contains roughly 18% prediction for the annular-wavy flow pattern class, and likewise, the first annular-wavy flow pattern prediction contains roughly 45% prediction for the annular-horizontal flow pattern class. These predictions show that the transition between flow patterns can also be identified by this analysis when the images contain features that may be specific to multiple classes.

It was noted that the slug and elongated bubble flow pattern classes were the most commonly misclassified by the CNN. Table 4-5 shows some examples of commonly misclassified flow pattern images from these classes. The results were generated by training the final CNN model four successive times with the same combination of hyperparameters as specified in Table 3-5. Each time, the trained model was used to predict the test set. The examples highlight the importance of evaluating the performance of such a deep learning model based on multiple reinitialised runs. Additionally, they highlight the stochastic nature of CNNs, which add randomness during learning by randomly initialising the model weights and shuffling the training samples before each epoch of training.

Table 4-4: Examples of CNN predictions of selected images from the test dataset


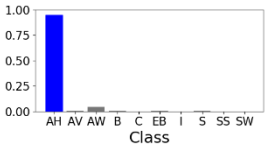

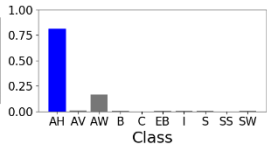
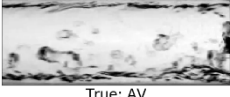
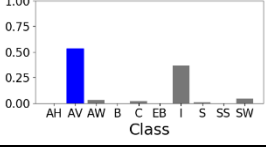
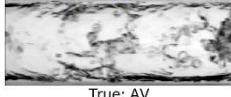
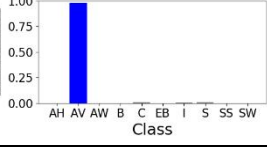

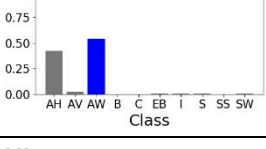

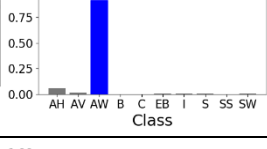
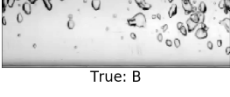
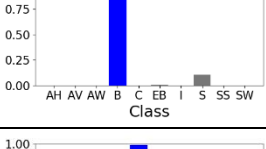
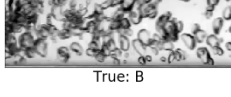
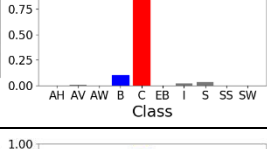
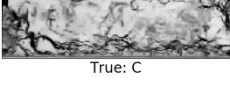
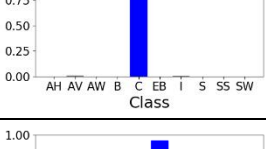

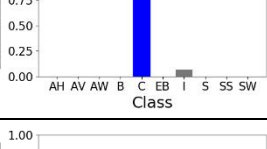

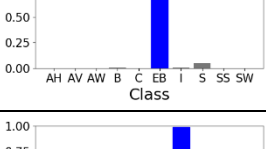
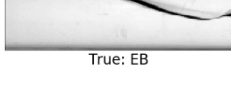
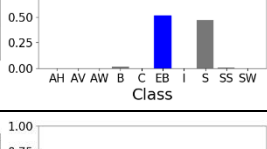
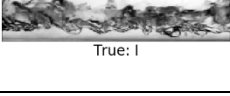
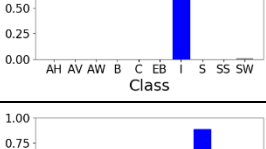
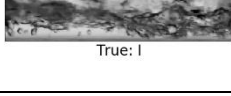
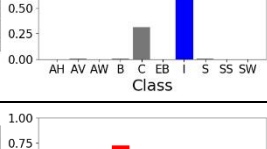
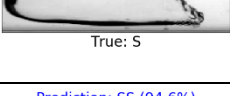
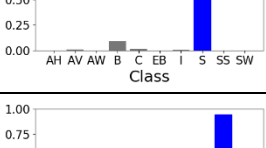
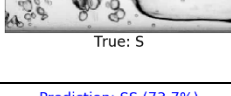
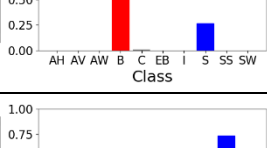
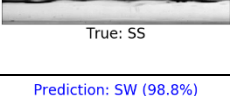
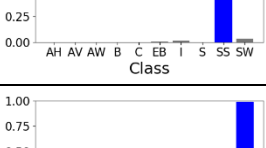
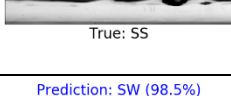
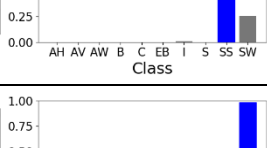
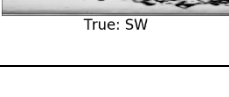
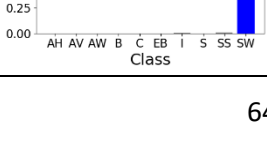
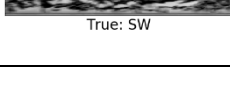
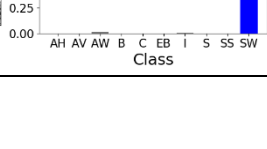
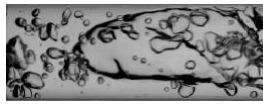
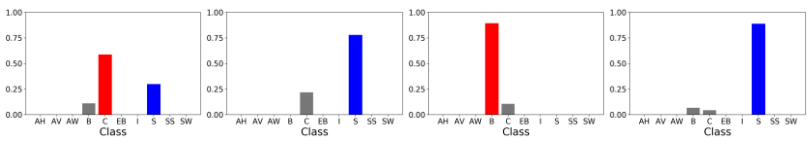
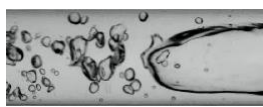
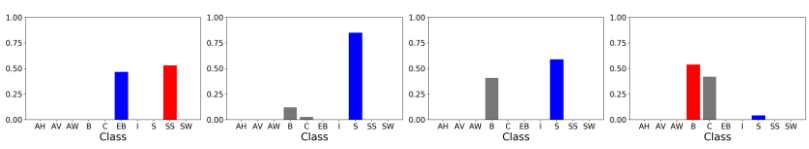
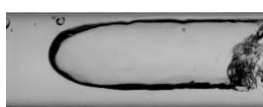
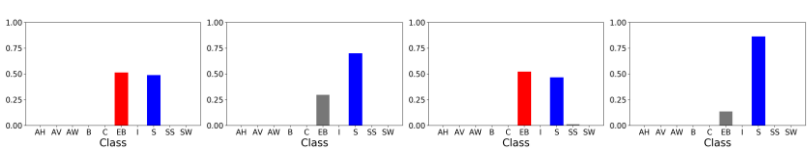
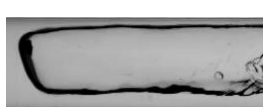
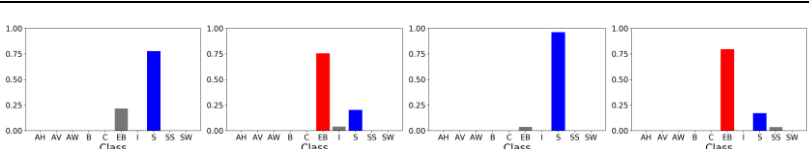

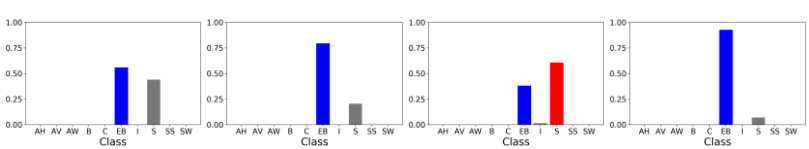

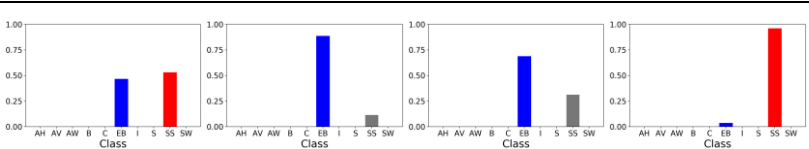
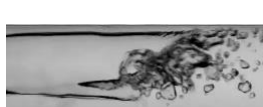
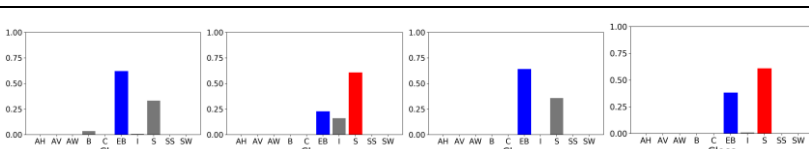

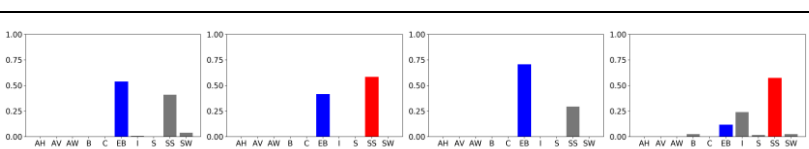
Class	Prediction	
AH	<p>Prediction: AH (94.7%)</p>  <p>True: AH</p> 	<p>Prediction: AH (81.2%)</p>  <p>True: AH</p> 
	<p>Prediction: AV (53.6%)</p>  <p>True: AV</p> 	<p>Prediction: AV (97.7%)</p>  <p>True: AV</p> 
AW	<p>Prediction: AW (53.9%)</p>  <p>True: AW</p> 	<p>Prediction: AW (92.0%)</p>  <p>True: AW</p> 
	<p>Prediction: B (88.9%)</p>  <p>True: B</p> 	<p>Prediction: C (84.1%)</p>  <p>True: B</p> 
C	<p>Prediction: C (99.0%)</p>  <p>True: C</p> 	<p>Prediction: C (93.6%)</p>  <p>True: C</p> 
	<p>Prediction: EB (94.4%)</p>  <p>True: EB</p> 	<p>Prediction: EB (51.4%)</p>  <p>True: EB</p> 
I	<p>Prediction: I (99.1%)</p>  <p>True: I</p> 	<p>Prediction: I (67.6%)</p>  <p>True: I</p> 
	<p>Prediction: S (88.7%)</p>  <p>True: S</p> 	<p>Prediction: B (72.7%)</p>  <p>True: S</p> 
SS	<p>Prediction: SS (94.6%)</p>  <p>True: SS</p> 	<p>Prediction: SS (73.7%)</p>  <p>True: SS</p> 
	<p>Prediction: SW (98.8%)</p>  <p>True: SW</p> 	<p>Prediction: SW (98.5%)</p>  <p>True: SW</p> 

Table 4-5: Examples of commonly misclassified slug and elongated bubble flow pattern images

Class	Image	Predictions
S		
		
		
		
EB		
		
		
		

4.5 Conclusion

In this chapter, the results of the MLP and CNN were presented and compared. For both models, the resulting learning curves indicated that the hyperparameters were well defined and that the resulting models were not overfitting the data.

The MLP was found to predict the test dataset with an accuracy of 97.09%, which was concluded to be impressive considering the low associated computational expense. The CNN improved the classification performance for all 10 of the flow pattern classes, and the overall prediction accuracy improved to 98.3%. This result highlighted the favourable ability of CNNs to extract features based on two-dimensional spatial information.

The time per prediction was then compared, which showed that the MLP was able to make predictions more than three times faster than the CNN. In either case, it was concluded that the prediction time per image for both of the considered models was sufficiently fast to allow real-time flow pattern prediction when coupled to an experimental set-up.

Finally, examples of predictions of the CNN model were shown for select images from the test dataset. The favourable ability of using the softmax activation function in the output layer of the models was seen, as the predictive probability per class gave additional insight because images with incorrect or low prediction scores could be justified.

Chapter 5. Development of an online predictive tool for in-tube multiphase flow patterns

5.1 Introduction

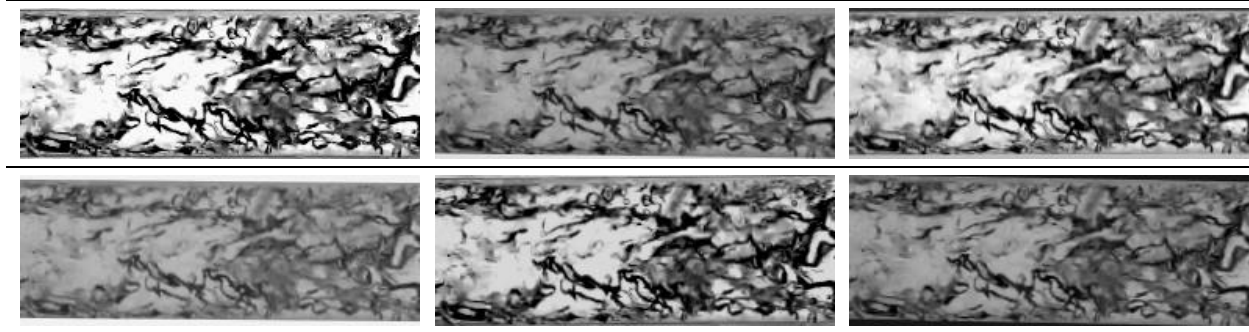
This chapter describes the process of retraining the final CNN model with an augmented version of the training dataset to develop a classifier that generalises well to images acquired from sources external to the University of Pretoria. The developed model is validated with flow pattern images acquired from the literature. The analysis sequence allows for the development of a predictive tool that can be used to identify multiphase flow and subsequently classify in-tube flow patterns.

5.2 Training with image augmentation

Although the final CNN model performed well for all of the considered flow pattern classes, the images of in-tube flow patterns found in the literature that were published by authors [9-13, 15, 16, 19-23, 111-114] external to the University of Pretoria showed distinct differences from those that have already been investigated. Apart from the different flow conditions, tube geometries, and inclination angles; the images had differing levels and ranges of brightness and contrast, the camera was positioned at different orientations, the images were often presented slightly skewed, and often with borders of different colours where it was difficult to distinguish whether the border was part of the experimental apparatus or due to poor image presentation. To develop a tool that could accurately identify the different flow patterns despite these differences, and without the user needing to spend significant time pre-processing images before prediction, it is necessary to make the training process more robust to include these differences.

The Keras API allows the user to write a unique image pre-processing function that can be used for image augmentation during the training process. The result is that a new augmented version of the dataset could be used for each epoch of training. The goal of training with augmented images was that the features that are distinct to the flow patterns are learnt during training rather than the features that may be unique to the experimental set-up used to generate the images previously investigated. Additionally, a model trained on augmented images would have improved generalisation to images acquired from external sources without hindering the generalisation to the images used thus far (the current test dataset). The written function used in this study randomly shifted the range and level of brightness, randomly rotated images within three degrees, applied borders of different colours and thickness, and randomly altered the contrast of images. An example of the image augmentation process is shown in Table 5-1, where a churn flow pattern image is shown with six examples of the applied image augmentation function.

Table 5-1: Examples of image augmentation applied to a Churn flow pattern image



5.3 Results

Employing the same final CNN model architecture and other hyperparameters, as described in Section 3.4.3, the training process was repeated using the image augmentation function applied to the training dataset, while the test dataset was left unchanged. The training process was repeated 50 times for 100 epochs, and the mean learning curves are shown in Figure 5-1.

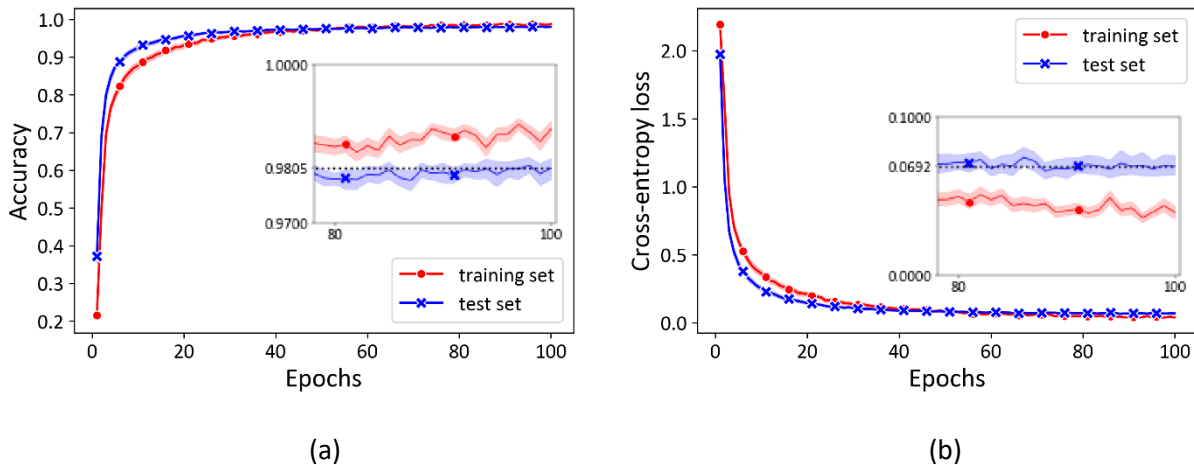


Figure 5-1: Mean learning curves for the CNN trained with augmented images: (a) Accuracy, (b) Cross-entropy loss

Again, a similar trend was seen when comparing the learning curves with those previously presented. Initially, the test set performance exceeded that of the training set until the test set curves converged towards near-constant values and the training curves asymptotically approached 0 and 1 for the loss and accuracy, respectively. However, training using augmented images led to a slower rate of learning because the model was exposed to a new augmented version of the image dataset for each epoch in the training process. The augmented images also led to slight fluctuations in the learning curves and a 95% confidence band with increased width. Furthermore, the image augmentation also aided in regularising the model, and it was evident that the gap between the training and test learning curves was less than previously seen for the CNN trained on the original images. Comparing the results with those previously seen, there was a slight degradation on the test dataset as the mean accuracy decreased from 0.9832 to 0.9805 and

the mean loss increased from 0.0530 to 0.0692. Additionally, when comparing the mean performance metrics and normalised confusion matrix on the test set, given in Table 5-2 and Figure 5-2, respectively, with those of the CNN trained without image augmentation, there are only subtle differences. The general trend was that the F-score was slightly decreased for all classes except for the intermittent and stratified-wavy classes, which had slight improvements. It is also noted that the annular-wavy and churn flow patterns were classified with 100% precision, and the annular-horizontal flow pattern was classified with 100% recall over the 50 training runs.

Table 5-2: The mean performance metrics for the CNN trained with augmented images

Class	Precision	Recall	F-Score	Accuracy
AH	0.9833	1.0000	0.9916	0.9805
AV	0.9986	0.9977	0.9981	
AW	1.0000	0.9859	0.9929	
B	0.9843	0.9698	0.9770	
C	1.0000	0.9621	0.9807	
EB	0.9593	0.9487	0.9540	
I	0.9698	0.9945	0.9820	
S	0.8333	0.9623	0.8932	
SS	0.9899	0.9833	0.9866	
SW	0.9960	0.9874	0.9916	

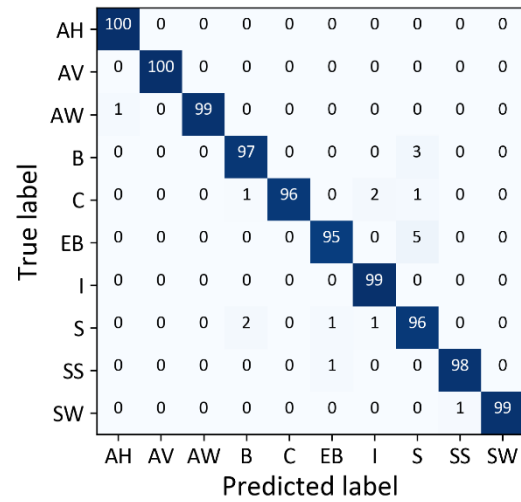


Figure 5-2: Normalised confusion matrix for the CNN trained on augmented images

5.4 Model validation

To validate the improved generalisation performance, the retrained model was used to predict the flow pattern classes of images found in the literature [9, 10, 12, 13, 15, 19-23]. Appendix A includes these images and the corresponding model prediction. The images cover a wide range of experimental conditions and tube orientations. Additionally, many of the obtained images also showed flow patterns of boiling and adiabatic flows. The general trend seen was that the model generalised very well to the images found in the literature, and in many of the cases where an image was misclassified, part of the prediction was towards the correct flow pattern class. A more substantial description is given in the appendix for each of the considered images. The name given to a specific flow pattern image in the literature often was different from the term used in this study. For this reason, each image is shown with the corresponding model prediction, the class label given to the figure in the corresponding literature source, and the class label that was assigned to the image by the author of this study.

5.5 Development of a predictive tool

Having validated the model's ability to generalise well to images obtained from sources external to the University of Pretoria's experimental set-up, it was decided to develop a predictive tool that can be used to classify the flow patterns of newly acquired images. Hence, create a tool for users to upload images and to have the flow pattern class predicted by the final model returned to the user.

To ensure meaningful feedback on the use of the predictive tool, a CNN model was trained to first identify that the image was representative of multiphase flow and that it was adequately pre-processed. This was done to ensure that randomly acquired images and those which do not meet the pre-processing requirements were rejected from the analysis rather than being given an arbitrary class prediction. Hence, the new model was used for a binary classification problem.

To train the model to identify multiphase flow, the original dataset was divided into a new training and test dataset consisting of 70% and 30% of the original image dataset, respectively. To obtain images that depict things other than multiphase flow, images were downloaded from the Google Open Image Dataset V6 [115]. The Open Image Dataset consists of 9.2 million images covering 600 classes. However, for this study, the images were simply labelled as not showing the presence of multiphase flow and were added to the training and test datasets in equal proportion to the number of condensation images in each set, respectively. The model was trained for 50 epochs. For each epoch, the training dataset consisted of the 70% (2 584 condensation images) of the original dataset together with an equal amount of images randomly selected from the Open Images Dataset. Therefore, a different subset of the Open Images Dataset was selected for each epoch of training. Image augmentation was applied to the entire training set during training. The augmentation included randomly rotating, translating, flipping, cropping, zooming, and adjusting the contrast and brightness of the images. The image augmentation strategy employed was more random than what was previously used because the goal of the training process was simply to detect the presence of multiphase flow, and the specific flow pattern was not of any importance.

The trained model was found to successfully predict the presence of multiphase flow for all of the condensation images in the test dataset, as well as all of the images acquired from the literature, which were cropped by the author of this study. Additionally, the model successfully predicted 94% of the random images, which was considered sufficient as it was not the purpose to use the predictive tool for images that do not show multiphase flow. Although the image augmentation process allowed for images that were slightly skewed or poorly cropped, those that were very poorly cropped were also correctly rejected from the analysis.

The predictive tool using both models was then published online. To obtain an accurate prediction, the user should aim to pre-process (crop and align) the images according to the methods specified in Section 3.2.2. Hence, crop the images with as little as possible of the tube border exposed and having square alignment.

5.6 Conclusion

Through the use of data augmentation, the training process was made more robust to include the types of differences seen in flow pattern images found in the literature. The results of the retrained model were

compared with the model described in the previous section, where it was found that the image augmentation process led to a slight degradation in performance on the test dataset. However, validating the new model with the images found in the literature, showed that the model generalised well to the images obtained from sources external to the University of Pretoria's experimental set-up. The analysis sequence allowed for the development of a predictive tool that can be used to identify multiphase flow and subsequently classify in-tube flow patterns of newly acquired images.

Chapter 6. Conclusions and recommendations

6.1 Conclusions

This study investigated the use of artificial intelligence, and more specifically, the deep learning models of multilayer perceptron and convolution neural networks for the task of image classification. The models were applied to the flow pattern images of condensation flow in inclined smooth tubes.

Before commencing the investigation, a literature review was conducted on condensation heat transfer, condensation flow patterns, and the theory of artificial intelligence and deep learning models. Based on the investigation into the existing flow pattern prediction methods, it was concluded that the flow pattern maps were not adequate to predict flow patterns for a wide range of experimental conditions and tube inclinations. Additionally, the prediction methods based on artificial intelligence were mainly focused on adiabatic and boiling flows in horizontal and vertical tube orientations, and mainly considered air-water two-phase flows. Most of the studies also relied on well-established models, computer-generated data, and measured parameters, which could introduce bias and reduce generalisation capability compared with models trained on high dimensional noisy data, such as from visualisation.

After completing the literature review, a flow pattern image dataset was obtained from existing experimental works. The dataset was prepared for the subsequent analysis and included flow pattern images covering 10 distinct classes for a wide range of test conditions. The use of a principal component analysis allowed for dimensionality reduction and the extraction of relevant features that could be used for training a multilayer perceptron neural network. The analysis also allowed for meaningful insights into the structure of the dataset. The projection to a lower-dimensional space allowed visualisation of the projected principal components, and the analysis of eigenfigures gave insight into the features of the data constituting most of the variance. A five-fold cross-validation and grid search method were then implemented to fine-tune the model hyperparameters, and the resulting models were trained on the training dataset.

Upon analysis of the results, the multilayer perceptron neural network successfully classified 97.1% of the images in the test dataset. The result was impressive considering the low computational expense associated with the method, and the fact that the model was trained on only the first 100 most significant features extracted from the principal component analysis. The results of the convolutional neural network showed an improvement of the classification accuracy to 98.3%, and an improved prediction across all 10 of the considered flow pattern classes. Comparison of the results justified the additional computation and verified the favourable ability of convolutional neural networks to extract features based on two-dimensional spatial information for image classification tasks. An analysis of the prediction time required per image in the test dataset revealed that both methods were sufficiently fast to enable real-time flow pattern prediction.

The convolutional neural network was then retrained using a uniquely developed image augmentation strategy. This made the model more applicable to predicting the flow patterns of images that would typically be found in the existing literature, or which would originate from sources external to the

University of Pretoria's experimental set-up. The improved generalisation capability of the model was validated using images obtained from the literature. Based on the success of the results, it was decided to develop a predictive tool for users to upload newly acquired images of in-tube multiphase flow patterns. The predicted flow pattern class from the final model in this investigation is returned to them by the predictive tool.

In summary, the results of this investigation showed that the prediction of flow patterns based on visualisation data allowed for a fast and non-intrusive means of classifying flow patterns with high accuracy. The time per prediction revealed that this type of system could be coupled to existing experimental set-ups to allow for a real-time flow pattern predictive method. The developed predictive tool allows users to predict the flow patterns of their own images, which can be used for comparison with the existing identification. This may aid in developing a universal standard and reduce the existing subjectivity by which flow patterns are identified and described.

6.2 Recommendations

The impressive initial results of using visual data to predict flow patterns provide reason for further investigation and analysis.

As is the case with most deep learning models and with supervised machine learning processes in general, the inclusion of more training data originating from a wider image database is likely to improve the classification performance, especially regarding the generalisation capability when applied to images from new experimental set-ups. More data should also be included to validate the choice of model hyperparameters.

The existing methodology could be improved by exploring a more elaborate range of model hyperparameters. This study was limited by the exponential increase in computational expense when expanding these models.

The model performance and an analysis of the feature extraction should be compared with state-of-the-art convolutional neural network models which performed well on standardised image classification tasks. Such models are LeNet [116], ResNets [117], GoogLeNet (Inception) [118], Network in Network [119], and VGG [120].

The use of visual data could be coupled with thermo-hydraulic data, intensive properties, or non-dimensional parameters to develop and improve flow pattern maps. Additionally, visual data with or without these additional parameters should be used to predict the heat transfer coefficient and pressure drop occurring within two-phase systems.

References

1. Dalkilic, A., & Wongwises, S. (2009). Intensive literature review of condensation inside smooth and enhanced tubes. *International Journal of Heat and Mass Transfer*, 52 (15-16), 3409-3426.
2. Liebenberg, L., & Meyer, J. P. (2008). Refrigerant condensation flow regimes in enhanced tubes and their effect on heat transfer coefficients and pressure drops. *Heat Transfer Engineering*, 29 (6), 506-520.
3. Thome, J. R. (2010). Two-phase flow patterns. In *Heat Transfer Engineering Data Book III: Enhanced heat transfer design methods of tubular heat exchangers* (pp. 307-332), Essen, Germany: Publico.
4. Thome, J. R., & Cioncolini, A. (2015). Two-phase flow pattern maps for macrochannels. In *Encyclopedia of Two-Phase Heat Transfer and Flow: Fundamentals and Methods* (pp. 5-45), Singapore: WSPC.
5. Çengel, Y. A., & Ghajar, A. J. (2015). *Heat and mass transfer: Fundamentals & applications* (5th ed.). New York: McGraw Hill Education.
6. Reynolds, O. (1883). An experimental investigation of the circumstances which determine whether the motion of water shall be direct or sinuous, and of the law of resistance in parallel channels. *Proceedings of the Royal Society of London*, 35 (224-226), 84-99.
7. Dittus, F., & Boelter, L. (1930). University of California publications on engineering. 2, 371.
8. Lips, S. P., & Meyer, J. P. (2012a). Experimental study of convective condensation in an inclined smooth tube. Part I: Inclination effect on flow pattern and heat transfer coefficient. *International Journal of Heat and Mass Transfer*, 55 (1), 395-404.
9. Bhagwat, S. M., & Ghajar, A. J. (2016). Experimental investigation of non-boiling gas-liquid two phase flow in upward inclined pipes. *Experimental Thermal and Fluid Science*, 79, 301-318.
10. Bhagwat, S. M., & Ghajar, A. J. (2017). Experimental investigation of non-boiling gas-liquid two phase flow in downward inclined pipes. *Experimental Thermal and Fluid Science*, 89, 219-237.
11. Coleman, J. W., & Garimella, S. (2003). Two-phase flow regimes in round, square and rectangular tubes during condensation of refrigerant R134a. *International Journal of Refrigeration*, 26 (1), 117-128.
12. Ghajar, A., & Tang, C. (2009). Advances in void fraction, flow pattern maps and non boiling heat transfer two-phase flow in pipes with various inclinations. *Advances in Multiphase Flow and Heat Transfer*, 1.
13. Roman, A. J., Kreitzer, P. J., Ervin, J. S., Hanchak, M. S., & Byrd, L. W. (2016). Flow pattern identification of horizontal two-phase refrigerant flow using neural networks. *International Communications in Heat and Mass Transfer*, 71, 254-264.
14. Ohira, K., Nakayama, T., Takahashi, K., Kobayashi, H., Taguchi, H., & Aoki, I. (2015). Pressure drop and heat transfer characteristics of boiling nitrogen in square pipe flow. *Physics Procedia*, 67, 675-680.

15. Bhagwat, S. M., & Ghajar, A. J. (2012). Similarities and differences in the flow patterns and void fraction in vertical upward and downward two phase flow. *Experimental Thermal and Fluid Science*, 39, 213-227.
16. Chalgeri, V. S., & Jeong, J. H. (2019a). Flow patterns of vertically upward and downward air-water two-phase flow in a narrow rectangular channel. *International Journal of Heat and Mass Transfer*, 128, 934-953.
17. Doretti, L., Zilio, C., Mancin, S., & Cavallini, A. (2013). Condensation flow patterns inside plain and microfin tubes: A review. *International Journal of Refrigeration*, 36 (2), 567-587.
18. Rouhani, S. Z., & Sohal, M. S. (1983). Two-phase flow patterns: A review of research results. *Progress in Nuclear Energy*, 11(3), 219-259.
19. Chen, L., Tian, Y., & Karayiannis, T. (2006). The effect of tube diameter on vertical two-phase flow regimes in small tubes. *International Journal of Heat and Mass Transfer*, 49 (21-22), 4220-4230.
20. Ghajar, A. J., & Tang, C. C. (2007). Heat transfer measurements, flow pattern maps, and flow visualisation for non-boiling two-phase flow in horizontal and slightly inclined pipe. *Heat Transfer Engineering*, 28 (6), 525-540.
21. Rosa, E. S., Salgado, R. M., Ohishi, T., & Mastelari, N. (2010). Performance comparison of artificial neural networks and expert systems applied to flow pattern identification in vertical ascendant gas-liquid flows. *International Journal of Multiphase Flow*, 36 (9), 738-754.
22. Xing, F., Xu, J., Xie, J., Liu, H., Wang, Z., & Ma, X. (2015). Froude number dominates condensation heat transfer of R245fa in tubes: Effect of inclination angles. *International Journal of Multiphase Flow*, 71, 98-115.
23. Narcy, M., & Colin, C. (2015). Two-phase pipe flow in microgravity with and without phase change: Recent progress and future prospects. *Interfacial Phenomena and Heat Transfer*, 3, 1-17.
24. Adelaja, A. O., Dirker, J., & Meyer, J. P. (2016). Convective condensation heat transfer of R134a in tubes at different inclination angles. *International Journal of Green Energy*, 13 (8), 812-821.
25. Adelaja, A. O., Dirker, J., & Meyer, J. P. (2017). Experimental study of the pressure drop during condensation in an inclined smooth tube at different saturation temperatures. *International Journal of Heat and Mass Transfer*, 105, 237-251.
26. Lips, S. P., & Meyer, J. P. (2012). Experimental study of convective condensation in an inclined smooth tube. Part II: Inclination effect on pressure drops and void fractions. *International Journal of Heat and Mass Transfer*, 55 (1-3), 405-412.
27. Lips, S. P., & Meyer, J. P. (2012b). Effect of gravity forces on heat transfer and pressure drop during condensation of R134a. *Microgravity Science and Technology: An International Journal for Microgravity and Space Exploration Related Research*, 24 (3), 157-164.
28. Lips, S. P., & Meyer, J. P. (2011). Experimental study of convective condensation of R134a in an inclined tube. *8th International Conference on Heat Transfer, Fluid Mechanics and Thermodynamics (HEFAT)*, Pointe Aux Piments, Mauritius.
29. Olivier, S. P., Meyer, J. P., De Paepe, M., & De Kerpel, K. (2016). The influence of inclination angle on void fraction and heat transfer during condensation inside a smooth tube. *International Journal of Multiphase Flow*, 80, 1-14.

30. Meyer, J. P., Dirker, J., & Adelaja, A. O. (2014). Condensation heat transfer in smooth inclined tubes for R134a at different saturation temperatures. *International Journal of Heat and Mass Transfer*, 70, 515-525.
31. Ewim, D., & Meyer, J. P. (2019). Pressure drop during condensation at low mass fluxes in smooth horizontal and inclined tubes. *International Journal of Heat and Mass Transfer*, 133, 686-701.
32. Ewim, D., Meyer, J. P., & Abadi, S. N. R. (2018). Condensation heat transfer coefficients in an inclined smooth tube at low mass fluxes. *International Journal of Heat and Mass Transfer*, 123, 455-467.
33. Ewim, D. R. E. (2019). Condensation inside horizontal and inclined smooth tubes at low mass fluxes, PhD thesis, University of Pretoria, Pretoria, South Africa.
34. Meyer, J. P., & Ewim, D. (2018). Heat transfer coefficients during the condensation of low mass fluxes in smooth horizontal tubes. *International Journal of Multiphase Flow*, 99, 485-499.
35. Suliman, R., Kyembe, M., & Meyer, J. (2010). Experimental investigation and validation of heat transfer coefficients during condensation of R-134a at low mass fluxes. *7th International Conference on Heat Transfer, Fluid Mechanics and Thermodynamics (HEFAT)*, Antalya, Turkey.
36. Suliman, R., Liebenberg, L., & Meyer, J. P. (2009). Improved flow pattern map for accurate prediction of the heat transfer coefficients during condensation of R-134a in smooth horizontal tubes and within the low-mass flux range. *International Journal of Heat and Mass Transfer*, 52 (25), 5701-5711.
37. Van Rooyen, E., Christians, M., Liebenberg, L., & Meyer, J. P. (2010). Probabilistic flow pattern-based heat transfer correlation for condensing intermittent flow of refrigerants in smooth horizontal tubes. *International Journal of Heat and Mass Transfer*, 53 (7-8), 1446-1460.
38. Adelaja, A. O., Dirker, J., & Meyer, J. P. (2015). Experimental investigation of frictional pressure drop in inclined tubes. *11th International Conference on Heat Transfer, Fluid Mechanics and Thermodynamics (HEFAT)*, Kruger National Park, South Africa.
39. Fair, J. R. (1960). What you need to design thermosiphon reboilers. *Petroleum Refiner*, 39 (2), 105-124.
40. Hewitt, G. F., & Roberts, D. (1969). Studies of two-phase flow patterns by simultaneous X-ray and flash photography. Atomic Energy Research Establishment, Harwell, England, United Kingdom.
41. Baker, O. (1953). *Design of pipelines for the simultaneous flow of oil and gas*. Fall meeting of the petroleum branch of the American Institute of Mining, Metallurgical, and Petroleum Engineers (AIME), Dallas, Texas.
42. Mandhane, J., Gregory, G., & Aziz, K. (1974). A flow pattern map for gas - liquid flow in horizontal pipes. *International Journal of Multiphase Flow*, 1 (4), 537-553.
43. Taitel, Y., & Dukler, A. E. (1976). A model for predicting flow regime transitions in horizontal and near horizontal gas-liquid flow. *American Institute of Chemical Engineers (AIChE) Journal*, 22 (1), 47-55.
44. Steiner, D. (1976). Heat transfer and pressure drop for boiling liquid nitrogen flowing in a horizontal tube. *Cryogenics*, 16 (7), 387-398.
45. Kattan, N., Favrat, D., & Thome, J. (1998). Flow boiling in horizontal tubes: Part 1-development of a diabatic two-phase flow pattern map. *Journal of heat transfer*, 120 (1), 140-147.

46. El Hajal, J., Thome, J. R., & Cavallini, A. (2003). Condensation in horizontal tubes, part 1: Two-phase flow pattern map. *International Journal of Heat and Mass Transfer*, 46 (18), 3349-3363.
47. Lips, S. P., & Meyer, J. P. (2011). Two-phase flow in inclined tubes with specific reference to condensation: A review. *International Journal of Multiphase Flow*, 37 (8), 845-859.
48. Cheng, L., Ribatski, G., & Thome, J. (2008). Two-phase flow patterns and flow-pattern maps: Fundamentals and applications. *Applied Mechanics Reviews*, 61 (5), 050802.
49. Barnea, D. (1987). A unified model for predicting flow-pattern transitions for the whole range of pipe inclinations. *International Journal of Multiphase Flow*, 13 (1), 1-12.
50. Crawford, T., Weinberger, C., & Weisman, J. (1985). Two-phase flow patterns and void fractions in downward flow Part I: Steady-state flow patterns. *International Journal of Multiphase Flow*, 11 (6), 761-782.
51. Cavallini, A., Censi, G., Del Col, D., Doretti, L., Longo, G., Rossetto, L., & Zilio, C. (2003). Condensation inside and outside smooth and enhanced tubes - a review of recent research. *International Journal of Refrigeration*, 26, 373-392.
52. Miyara, A. (2008). Condensation of hydrocarbons - A review. *International Journal of Refrigeration*, 31 (4), 621-632.
53. Mohseni, S. G., Akhavan-Behabadi, M. A., & Saeedinia, M. (2013). Flow pattern visualisation and heat transfer characteristics of R-134a during condensation inside a smooth tube with different tube inclinations. *International Journal of Heat and Mass Transfer*, 60, 598-602.
54. Thome, J. (2005). Update on advances in flow pattern based two-phase heat transfer models. *Experimental Thermal and Fluid Science*, 29 (3), 341-349.
55. Deshpande, A., & Kumar, M. (2018). *Artificial intelligence for big data: Complete guide to automating big data solutions using artificial intelligence techniques*. Birmingham, UK: Packt Publishing.
56. Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. Cambridge, Massachusetts: The MIT Press.
57. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., . . . Fei-Fei, L. (2015). ImageNet large scale visual recognition challenge. *International Journal of Computer Vision*, 115 (3), 211-252.
58. Campbell, M., Hoane, A. J., & Hsu, F.-h. (2002). Deep blue. *Artificial Intelligence*, 134 (1), 57-83.
59. Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., van den Driessche, G., . . . Hassabis, D. (2016). Mastering the game of Go with deep neural networks and tree search. *Nature*, 529 (7587).
60. Burkov, A. (2019). *The hundred-page machine learning book*: Andriy Burkov.
61. Burkov, A. (2020). *Machine learning engineering*: Andriy Burkov.
62. Murphy, K. P. (2012). *Machine learning: A probabilistic perspective*. Cambridge, Massachusetts: The MIT Press.
63. LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, 521 (7553), 436-444.
64. Beyeler, M. (2017). *Machine learning for OpenCV*. Birmingham, UK: Packt Publishing.
65. Bonaccorso, G. (2018). *Mastering machine learning algorithms: Expert techniques to implement popular machine learning algorithms and fine-tune your models*. Birmingham, UK: Packt Publishing.

66. Zaccone, G., & Karim, M. R. (2018). *Deep Learning with TensorFlow: Explore neural networks and build intelligent systems with Python* (2nd ed.). Birmingham: Packt Publishing.
67. Kingma, D. P., & Ba, J. A. (2019). A method for stochastic optimization. *ArXiv preprint*, ArXiv:1412.6980.
68. Hinton, G. E., Srivastava, N., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. R. (2012). Improving neural networks by preventing co-adaptation of feature detectors. *ArXiv preprint*, ArXiv:1207.0580.
69. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15 (1), 1929-1958.
70. LeCun, Y., Boser, B. E., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W. E., & Jackel, L. D. (1990). Handwritten digit recognition with a back-propagation network. *Advances in Neural Information Processing Systems*.
71. Poletaev, I., Tokarev, M. P., & Pervunin, K. S. (2020). Bubble patterns recognition using neural networks: Application to the analysis of a two-phase bubbly jet. *International Journal of Multiphase Flow*, 126.
72. Hearty, J. (2016). *Advanced machine learning with Python*. Birmingham, UK: Packt Publishing.
73. Cong, T., Su, G., Qiu, S., & Tian, W. (2013). Applications of ANNs in flow and heat transfer problems in nuclear engineering: A review work. *Progress in Nuclear Energy*, 62, 54-71.
74. Ozbayoglu, A. M., & Yuksel, H. E. (2012). Analysis of gas-liquid behavior in eccentric horizontal annuli with image processing and artificial intelligence techniques. *Journal of Petroleum Science and Engineering*, 81, 31-40.
75. Inoue, E. H., Carvalho, R. D. M., Estevam, V., Bannwart, A. C., & Fileti, A. M. F. (2013). Development of a neural network for the identification of multiphase flow patterns. *Proceedings of the IASTED International Conference Intelligent Systems and Control*, Marina del Ray, USA.
76. Al-Naser, M., Elshafei, M., & Al-Sarkhi, A. (2016). Artificial neural network application for multiphase flow patterns detection: A new approach. *Journal of Petroleum Science and Engineering*, 145, 548-564.
77. Chandrasekaran, S., & Kumar, S. (2018). Flow pattern and liquid holdup prediction in multiphase flow by machine learning approach. *Environmental Science: An Indian Journal*, 14 (4).
78. Massignan, J. P. D., Neves-Jr, F., Ofuchi, C. Y., Arruda, L. V. R., da Silva, M. J., & Morales, R. E. M. (2014). Broadband ultrasound attenuation technique applied to two phase flow pattern recognition. *Journal of Control, Automation and Electrical Systems*, 25 (5), 547-556.
79. Figueiredo, M. M. F., Goncalves, J. L., Nakashima, A. M. V., Fileti, A. M. F., & Carvalho, R. D. M. (2016). The use of an ultrasonic technique and neural networks for identification of the flow pattern and measurement of the gas volume fraction in multiphase flows. *Experimental Thermal and Fluid Science*, 70, 29-50.
80. Abbagoni, B. M., & Yeung, H. (2016). Non-invasive classification of gas-liquid two-phase horizontal flow regimes using an ultrasonic Doppler sensor and a neural network. *Measurement Science and Technology*, 27 (8).

81. Hanafizadeh, P., Eshraghi, J., Taklifi, A., & Ghanbarzadeh, S. (2016). Experimental identification of flow regimes in gas-liquid two phase flow in a vertical pipe. *Meccanica : An International Journal of Theoretical and Applied Mechanics (AIMETA)*, 51 (8), 1771-1782.
82. Ezzatabadipour, M., Singh, P., Robinson, M. D., Guillen-Rondon, P., & Torres, C. (2017). Deep learning as a tool to predict flow patterns in two-phase flow. *ArXiv preprint*, ArXiv:1705.07117.
83. Shoham, O. (1982). Flow pattern transition and characterization in gas-liquid two phase flow in inclined pipes. Tel-Aviv University, Israel.
84. Lee, J. Y., Ishii, M., & Kim, N. S. (2008). Instantaneous and objective flow regime identification method for the vertical upward and downward co-current two-phase flow. *International Journal of Heat and Mass Transfer*, 51 (13-14), 3442-3459.
85. Jing, C., Bai, Q., & Liu, B. (2008). Improved void fraction measurement by flow regime identification for gas liquid two-phase flows. *Proceedings of SPIE - The International Society for Optical Engineering*, 7128.
86. Selli, M. F., & Seleglim, P. (2007). Online Identification of horizontal two-phase flow regimes through gabor transform and neural network processing. *Heat Transfer Engineering*, 28 (6), 541-548.
87. Hernández, L., Juliá, J. E., Chiva, S., Paranjape, S., & Ishii, M. (2006). Fast classification of two-phase flow regimes based on conductivity signals and artificial neural networks. *Measurement Science and Technology*, 17 (6), 1511-1521.
88. Sunde, C., Avdic, S., & Pázsit, I. (2005). Classification of two-phase flow regimes via image analysis and a neuro-wavelet approach. *Progress in Nuclear Energy*, 46 (3), 348-358.
89. Hervieu, E. (2002). Identification of gas-liquid flow regimes from a space-frequency representation by use of an impedance probe and a neural network. *American Society of Mechanical Engineers (ASME) 2002 Joint U.S.-European Fluids Engineering Division Conference*.
90. Mi, Y., Ishii, M., & Tsoukalas, L. H. (1998). Vertical two-phase flow identification using advanced instrumentation and neural networks. *Nuclear Engineering and Design*, 184 (2), 409-420.
91. Mi, Y., Ishii, M., & Tsoukalas, L. H. (2001). Flow regime identification methodology with neural networks and two-phase flow models. *Nuclear Engineering and Design*, 204 (1), 87-100.
92. Sawant, P., Schelegel, J., Paranjape, S., Ozar, B., Hibiki, T., & Ishii, M. (2008). Flow regime identification in large diameter pipe. *Proceedings of the 16th International Conference on Nuclear Engineering: Thermal Hydraulics Instrumentation and Controls*.
93. Julia, J. E., Ozar, B., Dixit, A., Jeong, J.-J., Hibiki, T., & Ishii, M. (2009). Axial development of flow regime in adiabatic upward two-phase flow in a vertical annulus. *Journal of Fluids Engineering*, 131 (2).
94. Julia, J. E., Ozar, B., Jeong, J.-J., Hibiki, T., & Ishii, M. (2011). Flow regime development analysis in adiabatic upward two-phase flow in a vertical annulus. *International Journal of Heat and Fluid Flow*, 32 (1), 164-175.
95. Paranjape, S., Chen, S.-W., Hibiki, T., & Ishii, M. (2011). Flow regime identification under adiabatic upward two-phase flow in a vertical rod bundle geometry. *Journal of Fluids Engineering*, 133 (9).
96. Tambouratzis, T., & Pázsit, I. (2009). Non-invasive on-line two-phase flow regime identification employing artificial neural networks. *Annals of Nuclear Energy*, 36 (4), 464-469.

97. Tambouratzis, T., & Pàzsit, I. (2010). A general regression artificial neural network for two-phase flow regime identification. *Annals of Nuclear Energy*, 37 (5), 672-680.
98. Liu, L., & Bai, B. (2019). Flow regime identification of swirling gas-liquid flow with image processing technique and neural networks. *Chemical Engineering Science*, 199, 588-601.
99. Wiedemann, P., Döß, A., Schleicher, E., & Hampel, U. (2019). Fuzzy flow pattern identification in horizontal air-water two-phase flow based on wire-mesh sensor data. *International Journal of Multiphase Flow*, 117, 153-162.
100. Ghanbarzadeh, S., Hanafizadeh, P., & Hassan Saidi, M. (2012). Intelligent image-based gas-liquid two-phase flow regime recognition. *Journal of Fluids Engineering*, 134 (6).
101. Zhang, L., & Wang, H. (2010). Identification of oil-gas two-phase flow pattern based on SVM and electrical capacitance tomography technique. *Flow Measurement and Instrumentation*, 21 (1), 20-24.
102. Mahvash, A., & Ross, A. (2008). Application of CHMMs to two-phase flow pattern identification. *Engineering Applications of Artificial Intelligence*, 21 (8), 1144-1152.
103. Zhang, M., & Wang, L. (2017). Identification of gas-liquid two-phase flow patterns using non-return valve sound signals. *2017 Chinese Automation Congress (CAC)*, 771-776.
104. Wang, Z., Bovik, A. C., Sheikh, H. R., & Simoncelli, E. P. (2004). Image quality assessment: From error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13 (4), 600-612.
105. Hobold, G. M., & da Silva, A. K. (2018). Machine learning classification of boiling regimes with low speed, direct and indirect visualization. *International Journal of Heat and Mass Transfer*, 125, 1296-1309.
106. Turk, M., & Pentland, A. (1991). Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, 3 (1), 71-86.
107. Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., . . . Isard, M. (2016). Tensorflow: A system for large-scale machine learning. *12th {USENIX} Symposium on Operating Systems Design and Implementation (OSDI)*.
108. Glorot, X., & Bengio, Y. (2010). Understanding the difficulty of training deep feedforward neural networks. *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*.
109. Raschka, S., & Mirjalili, V. (2017). *Python machine learning: machine learning and deep learning with Python, scikit-learn, and Tensorflow*, Birmingham, UK: Packt Publishing.
110. Hand, D. J. (2012). Assessing the performance of classification methods. *International Statistical Review / Revue Internationale de Statistique*, 80 (3), 400-414.
111. Chalgeri, V. S., & Jeong, J. H. (2019b). Flow regime identification and classification based on void fraction and differential pressure of vertical two-phase flow in rectangular channel. *International Journal of Heat and Mass Transfer*, 132, 802-816.
112. Milkie, J. A., Garimella, S., & Macdonald, M. P. (2016). Flow regimes and void fractions during condensation of hydrocarbons in horizontal smooth tubes. *International Journal of Heat and Mass Transfer*, 92, 252-267.
113. Ghajar, A. J., & Tang, C. C., (2012). Void fraction and flow patterns of two-phase flow in upward and downward vertical and horizontal pipes. *Advances in Multiphase Flow and Heat Transfer*, 4, 175-201.

114. Zhao, Y., Bi, Q., & Hu, R. (2013). Recognition and measurement in the flow pattern and void fraction of gas-liquid two-phase flow in vertical upward pipes using the gamma densitometer. *Applied Thermal Engineering*, 60 (1-2), 398-410.
115. Kuznetsova, A., Rom, H., Alldrin, N., Uijlings, J., Krasin, I., Pont-Tuset, J., . . . Kolesnikov, A. (2020). The Open Images Dataset v4. *International Journal of Computer Vision*, 1-26.
116. LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., & Jackel, L. D. (1989). Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1 (4), 541-551.
117. He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. *Proceedings of the IEEE conference on computer vision and pattern recognition*.
118. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., . . . Rabinovich, A. (2015). Going deeper with convolutions. *Proceedings of the IEEE conference on computer vision and pattern recognition*.
119. Lin, M., Chen, Q., & Yan, S. (2013). Network in network. *ArXiv preprint*, ArXiv:1312.4400.
120. Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *ArXiv preprint*, ArXiv:1409.1556.
121. Gao, Z.-K., Liu, M.-X., Dang, W.-D., & Cai, Q. (2020). A novel complex network-based deep learning method for characterizing gas-liquid two-phase flow. *Petroleum Science*.

Appendix A. Prediction of flow pattern images from the literature

A.1 Introduction

This section presents images of flow patterns obtained from the literature. The resulting prediction of the final convolutional neural network model is given for each of the flow pattern images. A brief description is given for each image presented, and then the model prediction, the class label given to the image in the corresponding literature source, and the class label assigned to the image by the author of this study are provided in tabular format.

Note that the blue colour in the presented prediction bar charts refers to the flow pattern class that was predicted with the highest probability by the model, without any information given about whether the prediction was correct.

A.2 Prediction results of flow pattern images from the literature

The first image considered, Figure A-1, is from the work of Bhagwat and Ghajar [15], which shows three images of bubbly flow for differing gas superficial velocities. The model could predict all of the bubbly images correctly.

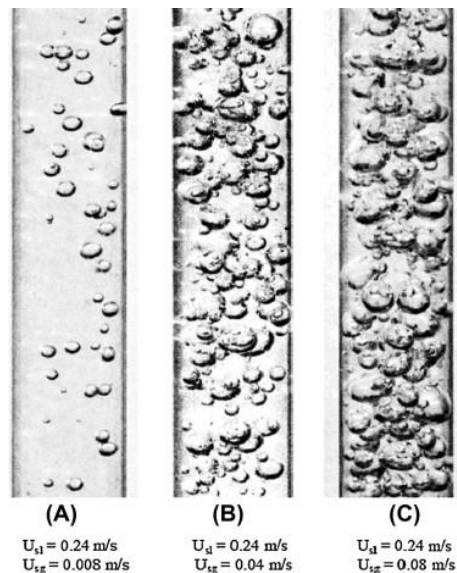


Figure A-1: Three images depicting Bubbly flow for different gas superficial velocities, taken from Bhagwat and Ghajar [15]

Table A-1: Model results of the images depicting bubbly flow for differing gas superficial velocities, taken from Bhagwat and Ghajar [15]

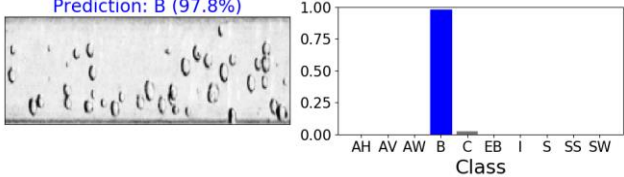
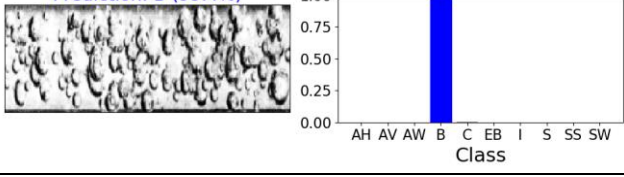
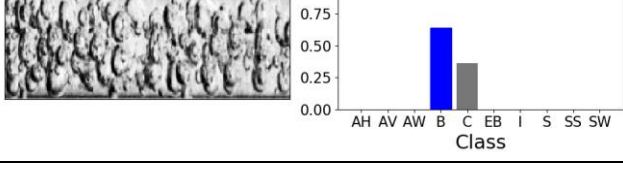
Image with prediction	Class label from literature source	Class label according to this research	Predicted class label
<p>Prediction: B (97.8%)</p> 	Bubbly	Bubbly	Bubbly
<p>Prediction: B (99.4%)</p> 	Bubbly	Bubbly	Bubbly
<p>Prediction: B (63.9%)</p> 	Bubbly	Bubbly	Bubbly

Figure A-2 depicts five bubbly flow pattern images, also taken from the work of Bhagwat and Ghajar [15], where the effects of liquid superficial velocity were investigated. The model could predict all five of the bubbly flow pattern images correctly.

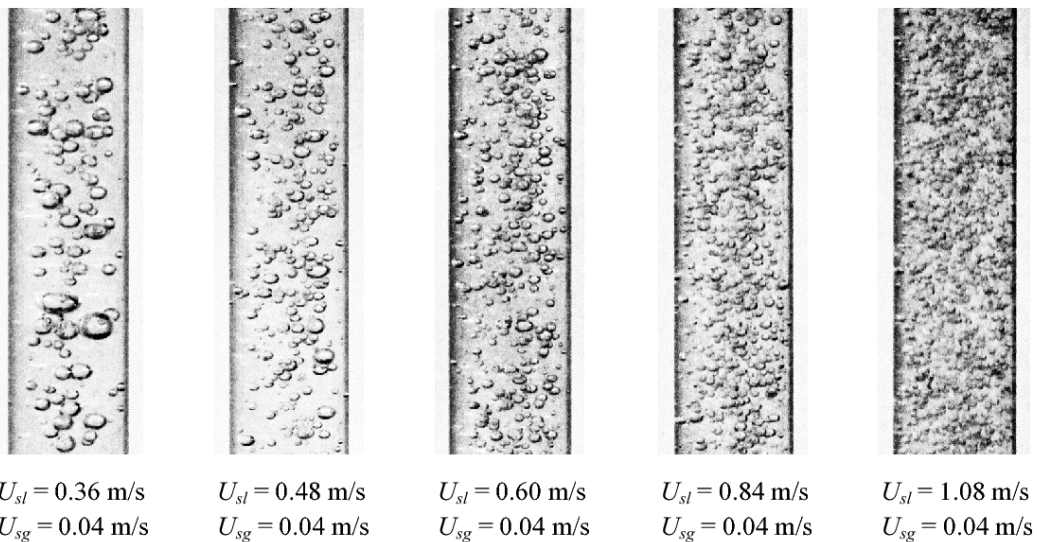


Figure A-2: Five bubbly flow pattern images for differing liquid superficial velocities, taken from Bhagwat and Ghajar [15]

Table A-2: Model results of the images depicting bubbly flow for differing liquid superficial velocities, taken from Bhagwat and Ghajar [15]

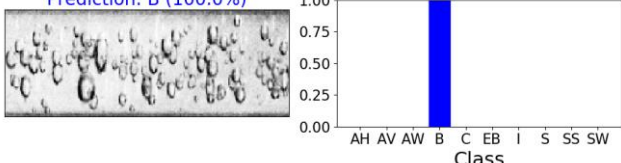
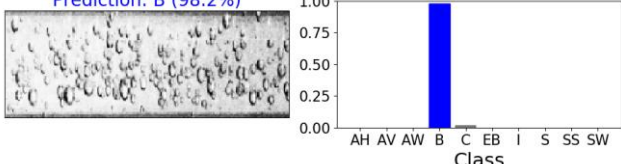
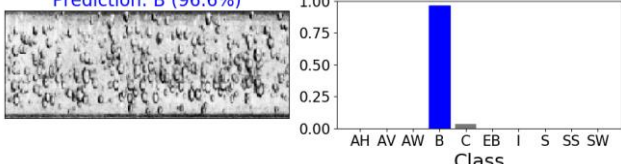
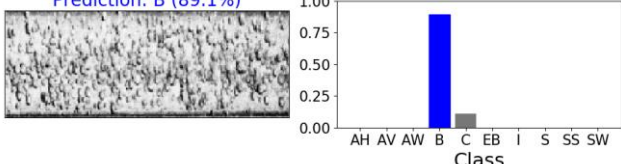
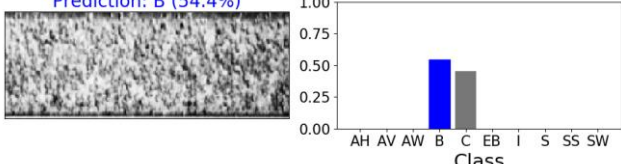
Image with prediction	Class label from literature source	Class label according to this research	Predicted class label
<p>Prediction: B (100.0%)</p> 	Bubbly	Bubbly	Bubbly
<p>Prediction: B (98.2%)</p> 	Bubbly	Bubbly	Bubbly
<p>Prediction: B (96.6%)</p> 	Bubbly	Bubbly	Bubbly
<p>Prediction: B (89.1%)</p> 	Bubbly	Bubbly	Bubbly
<p>Prediction: B (54.4%)</p> 	Bubbly	Bubbly	Bubbly

Figure A-3 depicts the different flow patterns observed in gas-liquid two-phase flow for vertical upward, horizontal, and upward inclined flow. The images are found in the work of Bhagwat and Ghajar [9]. The model misclassified the flow patterns for upward vertical flow, and instead predicted churn flow for all four of the considered images. The result could be due to the poor image lighting seen in the figure. The model showed improved prediction for the horizontal and upward inclined flows.

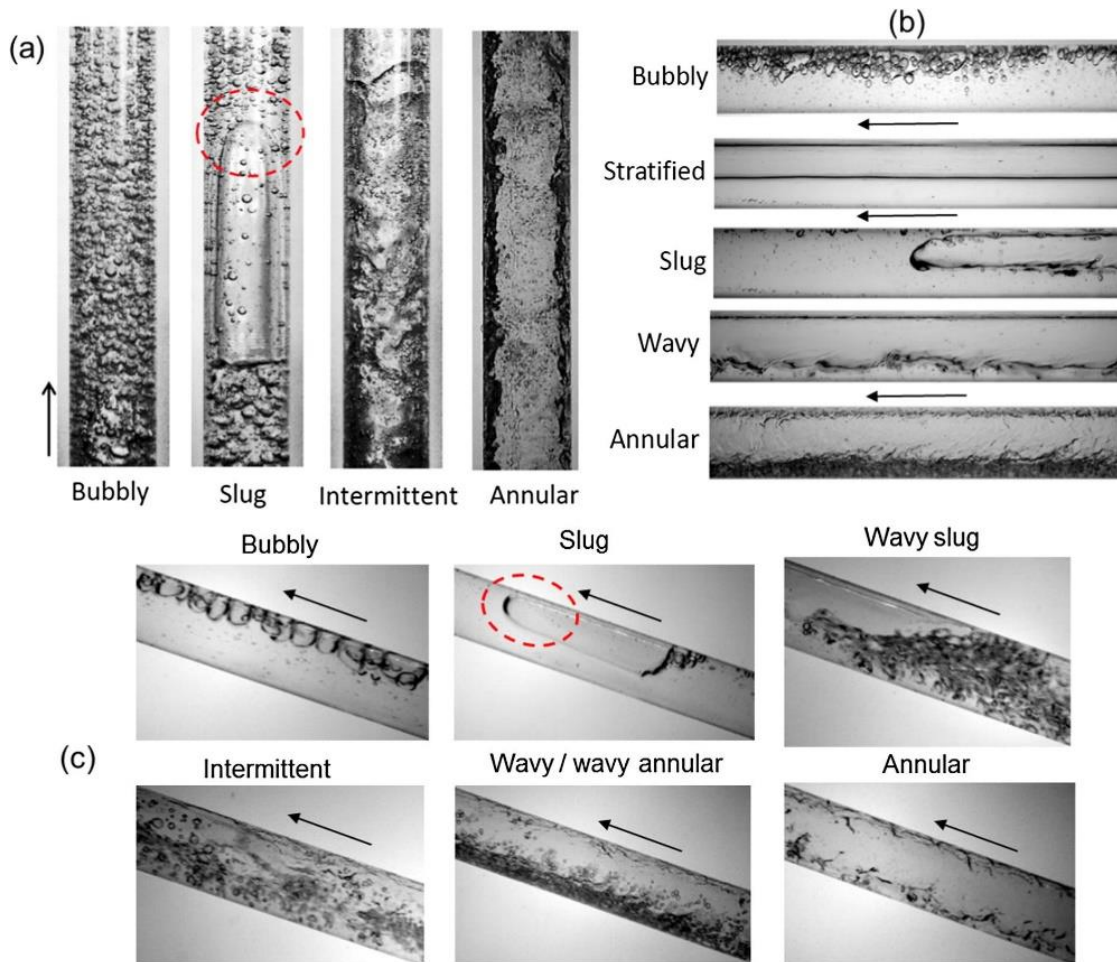
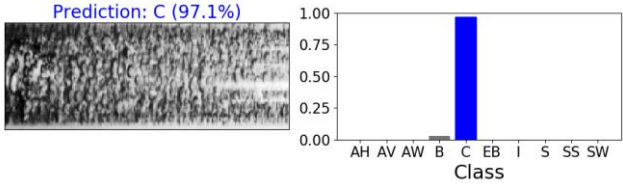
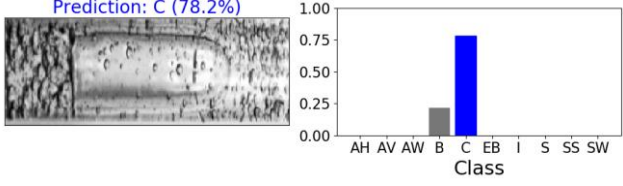
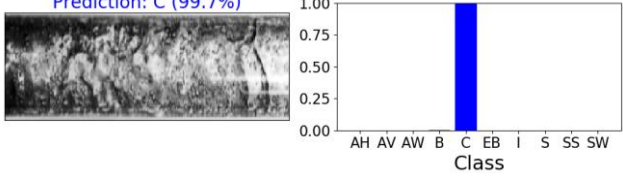
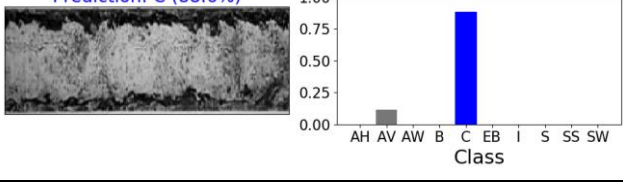
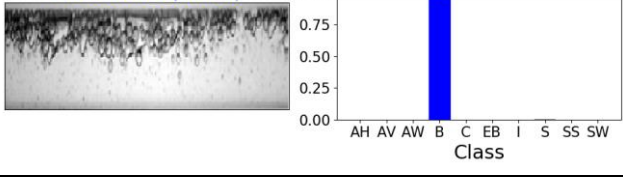
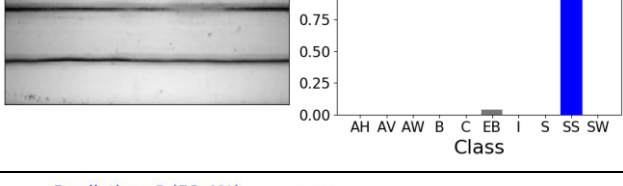
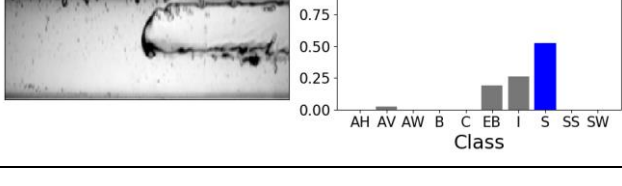


Figure A-3: Flow pattern images observed in gas-liquid two-phase flow for: (a) vertical upward, (b) horizontal, (c) upward inclined flow, taken from Bhagwat and Ghajar [9]

Table A-3: Model results of the images observed in gas-liquid two-phase flow for vertical upward, horizontal and upward inclined flow, taken from Bhagwat and Ghajar [9]

Image with prediction	Class label from literature source	Class label according to this research	Predicted class label
<p>Prediction: C (97.1%)</p> 	Bubbly	Bubbly	Churn
<p>Prediction: C (78.2%)</p> 	Slug	Slug	Churn
<p>Prediction: C (99.7%)</p> 	Intermitted	Churn	Churn
<p>Prediction: C (88.6%)</p> 	Annular	Annular-vertical / Churn	Churn
<p>Prediction: B (99.7%)</p> 	Bubbly	Bubbly	Bubbly
<p>Prediction: SS (96.0%)</p> 	Stratified	Stratified-smooth	Stratified-smooth
<p>Prediction: S (52.4%)</p> 	Slug	Slug	Slug

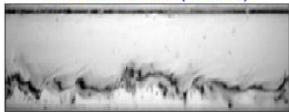
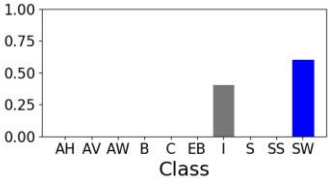
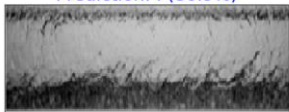
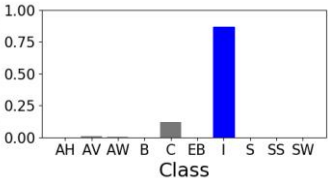

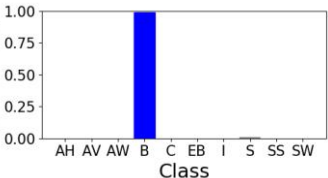

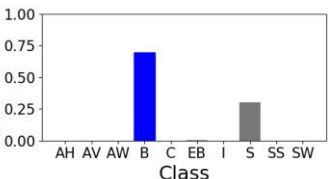

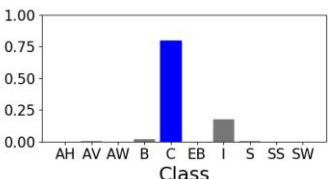
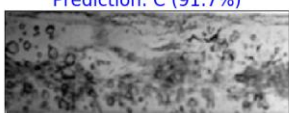
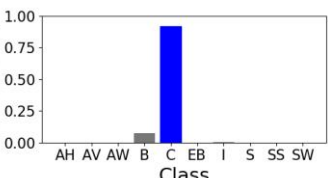
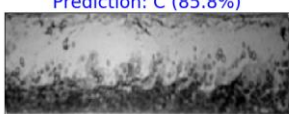
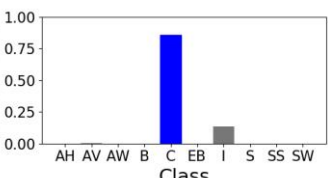
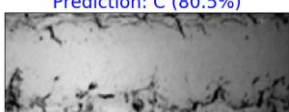
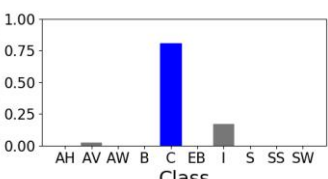
<p>Prediction: SW (59.8%)</p>  	Wavy	Stratified-wavy	Stratified-wavy
<p>Prediction: I (86.9%)</p>  	Annular	Annular-wavy / Intermittent	Intermittent
<p>Prediction: B (99.0%)</p>  	Bubbly	Bubbly	Bubbly
<p>Prediction: B (69.4%)</p>  	Slug	Slug	Bubbly
<p>Prediction: C (79.9%)</p>  	Wavy / Slug	Bubbly / Intermittent / Churn	Churn
<p>Prediction: C (91.7%)</p>  	Intermittent	Bubbly / Churn / Intermittent	Churn
<p>Prediction: C (85.8%)</p>  	Wavy / Wavy-annular	Stratified-wavy / Annular-wavy	Churn
<p>Prediction: C (80.5%)</p>  	Annular	Annular-vertical / Churn	Churn

Figure A-4 depicts three images of the falling film flow pattern for vertical downward flow, found in the work of Bhagwat and Ghajar [10]. The observed flow pattern is described as the annular-vertical flow pattern in this study. The model correctly predicted the first two images as the annular-vertical flow pattern, and the third image as the annular-wavy flow pattern. In the case of all three images, the annular flow pattern was predicted.

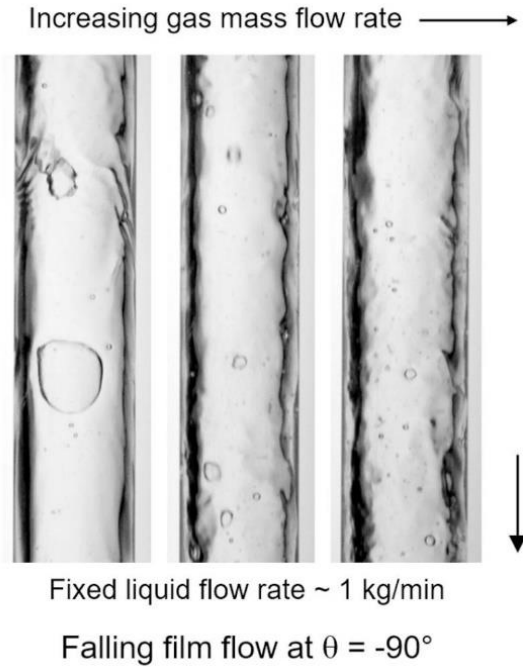
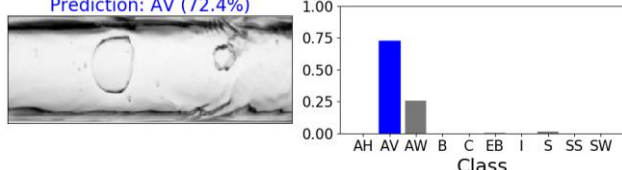
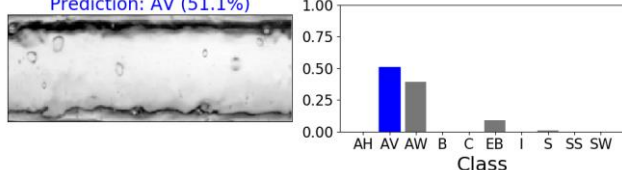


Figure A-4: Falling film flow pattern observed for vertical downward flow, taken from Bhagwat and Ghajar [10]

Table A-4: Model results of the flow pattern images depicting falling film flow, taken from Bhagwat and Ghajar [10]

Image with prediction	Class label from literature source	Class label according to this research	Predicted class label
<p>Prediction: AV (72.4%)</p> 	Falling film	Annular-vertical	Annular-vertical
<p>Prediction: AV (51.1%)</p> 	Falling film	Annular-vertical	Annular-vertical

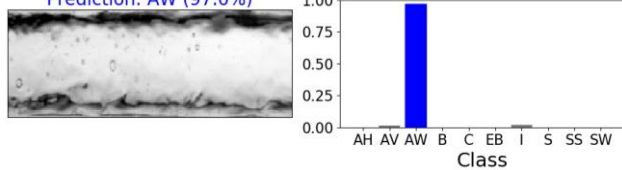
<p>Prediction: AW (97.0%)</p> 	Falling film	Annular-vertical	Annular-wavy
---	--------------	------------------	--------------

Figure A-5 depicts images of bubbly flow observed in horizontal, vertical downward and downward inclined flow. The figure is found in the work of Bhagwat and Ghajar [10]. The model could correctly predict all of the images as the bubbly flow pattern.

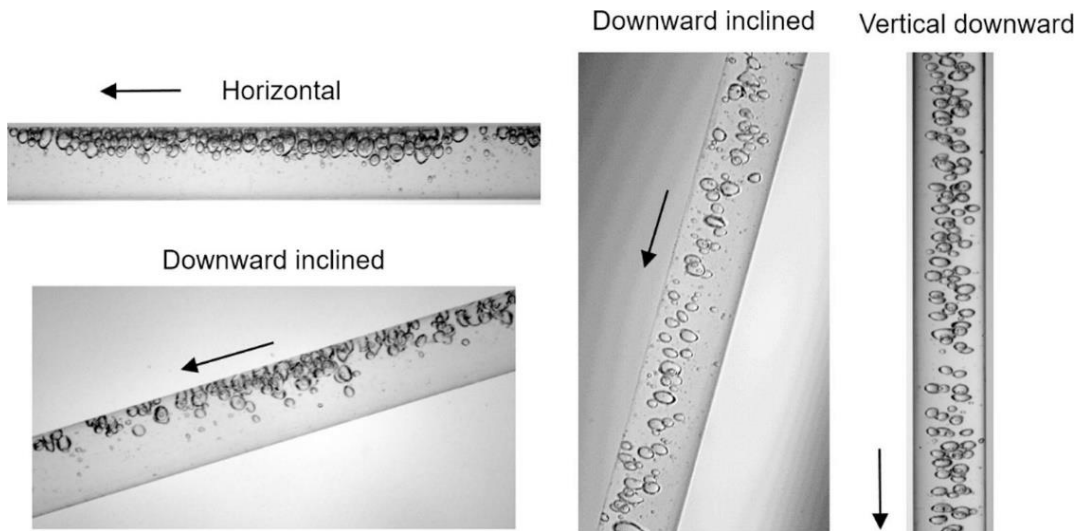
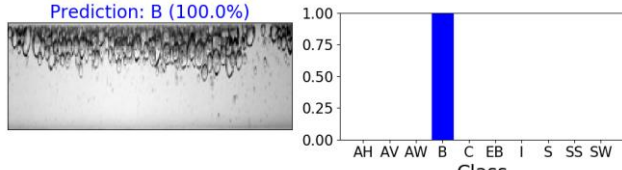
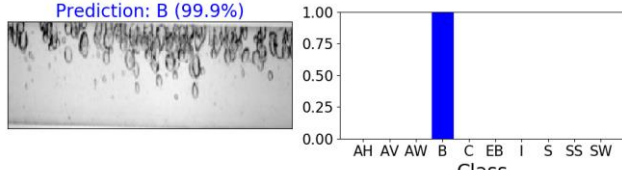


Figure A-5: Images of bubbly flow in horizontal, downward vertical and downward inclined flow, taken from Bhagwat and Ghajar [10]

Table A-5: Model results of the images of Bubbly flow for horizontal, downward vertical and downward inclined flow, taken from Bhagwat and Ghajar [10]

Image with prediction	Class label from literature source	Class label according to this research	Predicted class label
<p>Prediction: B (100.0%)</p> 	Bubbly	Bubbly	Bubbly
<p>Prediction: B (99.9%)</p> 	Bubbly	Bubbly	Bubbly

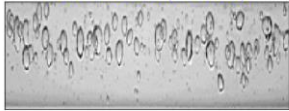
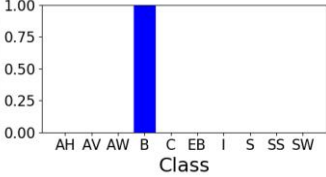
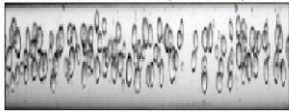
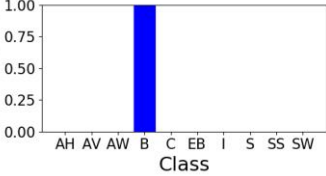
<p>Prediction: B (100.0%)</p>  	Bubbly	Bubbly	Bubbly
<p>Prediction: B (100.0%)</p>  	Bubbly	Bubbly	Bubbly

Figure A-6, Figure A-7, and Figure A-8 depict the flow pattern images observed in upward vertical two-phase flow of R-134a refrigerant considering tubes with internal diameters of 2.01 mm, 2.88 mm and 4.24 mm, respectively. The figures are found in the work of Chen et al. [19]. The model could predict all the flow pattern images correctly according to the flow pattern descriptions given in this study. The model results of the three figures are shown in Table A-6, Table A-7, and Table A-8, respectively.

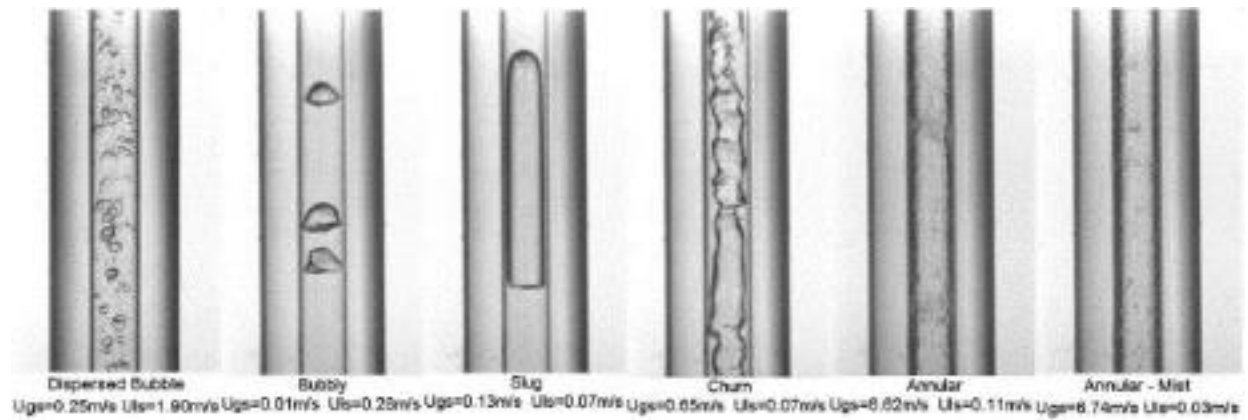

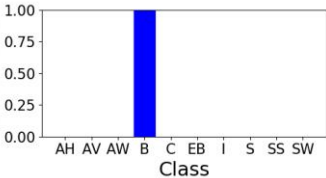

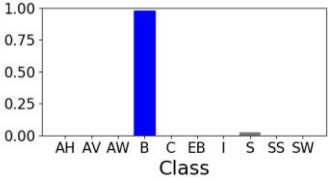
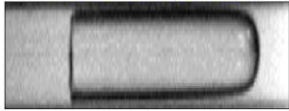
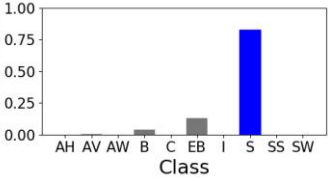

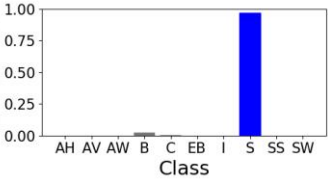
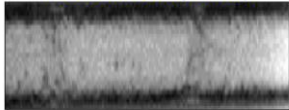
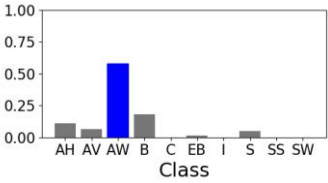
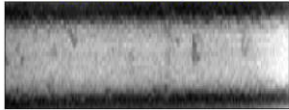
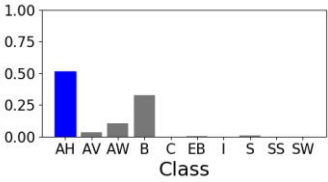


Figure A-6: Flow patterns during upward vertical flow in a 2.01 mm internal diameter tube, taken from Chen et al. [19]

Table A-6: Model results of the images for upward vertical flow in a 2.01 mm internal diameter tube, taken from Chen et al. [19]

Image with prediction	Class label from literature source	Class label according to this research	Predicted class label
<p>Prediction: B (100.0%)</p>  	Dispersed bubbles	Bubbly	Bubbly

<p>Prediction: B (97.8%)</p>  	Bubbly	Bubbly	Bubbly
<p>Prediction: S (82.8%)</p>  	Slug	Slug	Slug
<p>Prediction: S (97.2%)</p>  	Churn	Churn / Slug	Slug
<p>Prediction: AW (58.0%)</p>  	Annular	Annular-vertical	Annular-wavy
<p>Prediction: AH (51.6%)</p>  	Mist	Annular-vertical	Annular-horizontal

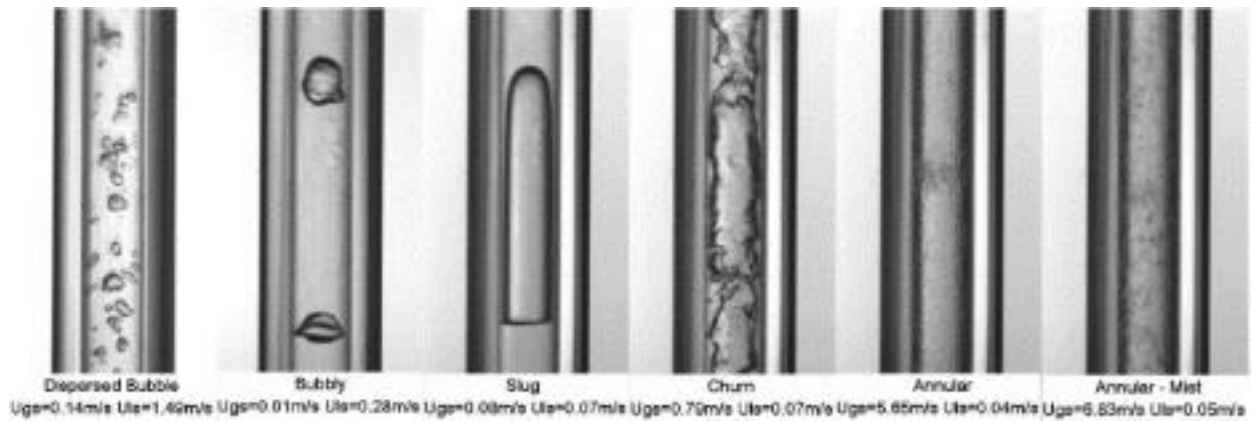


Figure A-7: Flow patterns during upward vertical flow in a 2.88 mm internal diameter tube, taken from Chen et al. [19]

Table A-7: Model results of the images for upward vertical flow in a 2.88 mm internal diameter tube, taken from Chen et al. [19]

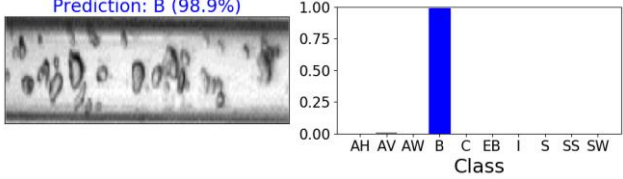
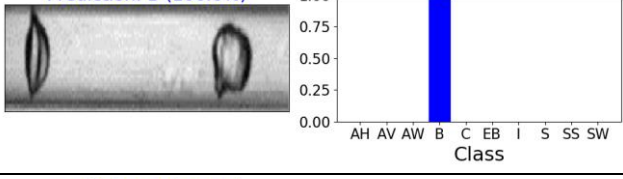
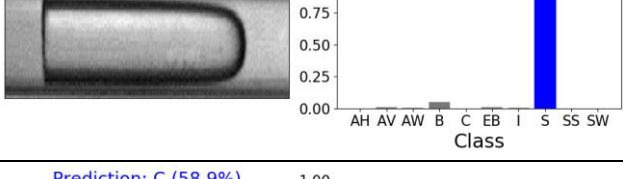
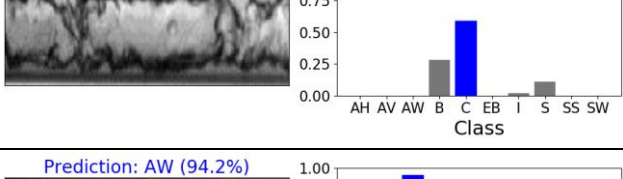
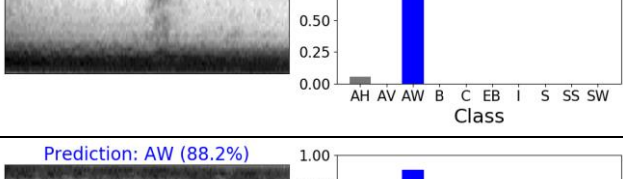
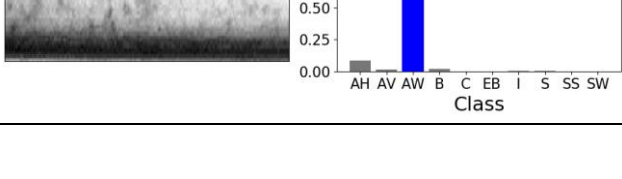
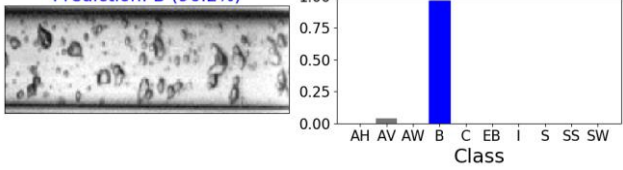
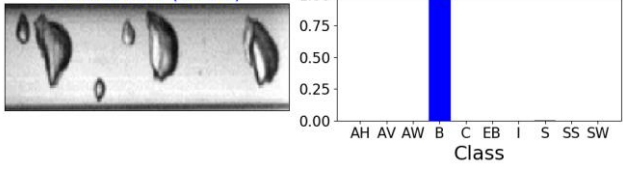
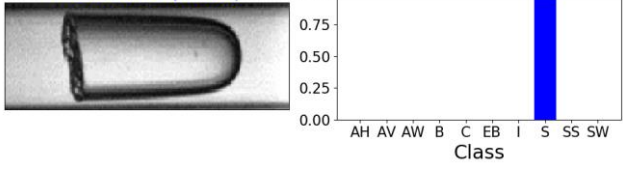
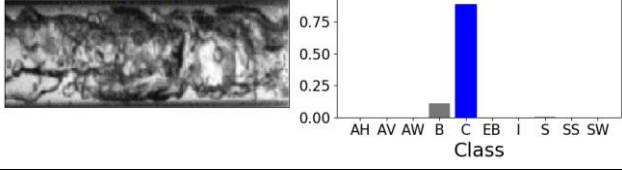
Image with prediction	Class label from literature source	Class label according to this research	Predicted class label
Prediction: B (98.9%) 	Dispersed Bubbles	Bubbly	Bubbly
Prediction: B (100.0%) 	Bubbly	Bubbly	Bubbly
Prediction: S (92.6%) 	Slug	Slug	Slug
Prediction: C (58.9%) 	Churn	Churn	Churn
Prediction: AW (94.2%) 	Annular	Annular-vertical	Annular-wavy
Prediction: AW (88.2%) 	Annular-Mist	Annular-vertical	Annular-wavy



Figure A-8: Flow patterns during upward vertical flow in a 4.26 mm internal diameter tube, taken from Chen et al. [19]

Table A-8: Model results of the images for upward vertical flow in a 4.26 mm internal diameter tube, taken from Chen et al. [19]

Image with prediction	Class label from literature source	Class label according to this research	Predicted class label
Prediction: B (96.2%) 	Dispersed Bubbles	Bubbly	Bubbly
Prediction: B (99.9%) 	Bubbly	Bubbly	Bubbly
Prediction: S (99.9%) 	Slug	Slug	Slug
Prediction: C (88.9%) 	Churn	Churn	Churn

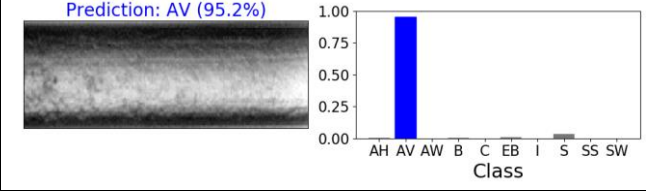
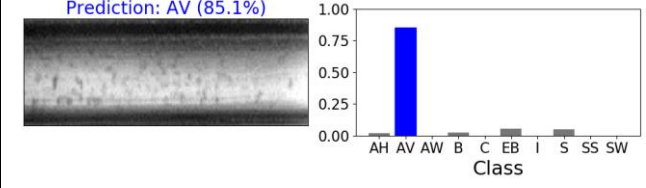
	Annular	Annular-vertical	Annular-vertical
	Mist	Annular-vertical	Annular-vertical

Figure A-9 is taken from the work of Ghajar and Tang [12], and depicts the flow pattern images typically observed in horizontal two-phase flow. The authors refer to the plug flow pattern, which is the equivalent of the elongated bubble flow pattern in this study. Similarly, they refer to the stratified-wavy flow pattern simply as “wavy”. Although there are some discrepancies between the class labels given to the respective images by the authors of the above work as well as the author of this study, the model predicted all the flow pattern images correctly in line with the class names used in this work.

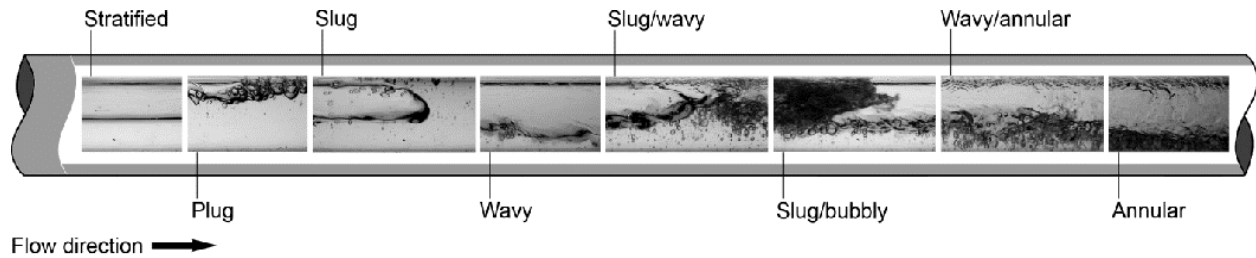
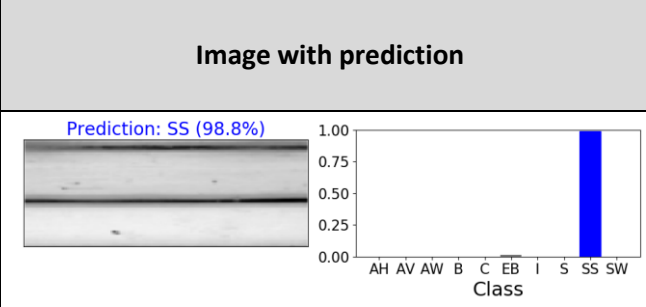
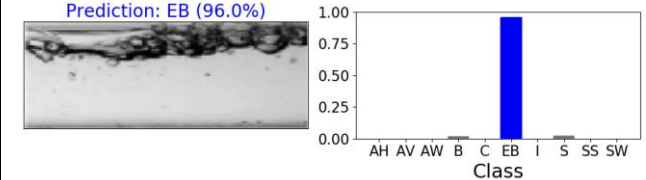


Figure A-9: Typical flow patterns seen in horizontal two-phase flow, taken from Ghajar and Tang [12]

Table A-9: Model results of the typical flow patterns seen in horizontal two-phase flow, taken from Ghajar and Tang [12]

Image with prediction	Class label from literature source	Class label according to this research	Predicted class label
	Stratified	Stratified-smooth	Stratified-smooth
	Plug	Elongated bubbles / Bubbly	Elongated bubbles

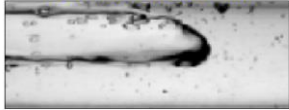
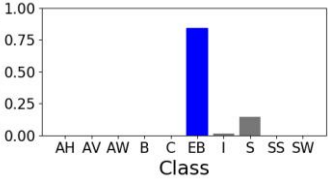
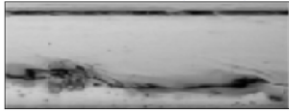
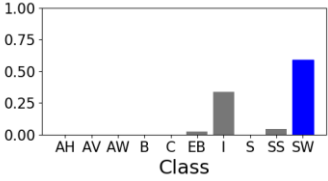
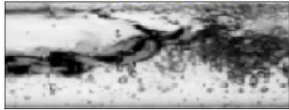
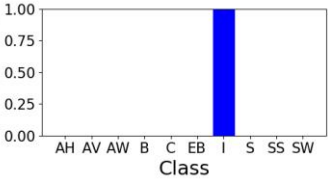

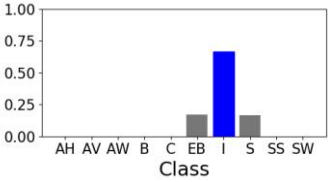
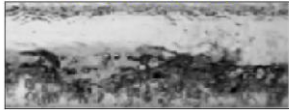
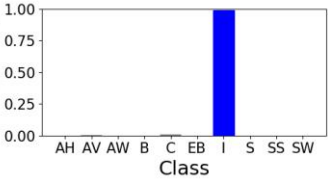
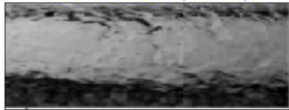
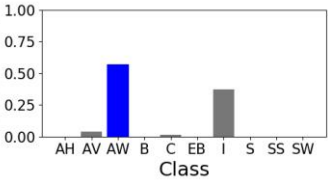
<p>Prediction: EB (84.4%)</p>  	Slug	Slug / Elongated bubbles	Elongated bubbles
<p>Prediction: SW (59.1%)</p>  	Wavy	Stratified-wavy	Stratified-wavy
<p>Prediction: I (99.9%)</p>  	Slug / Wavy	Intermittent	Intermittent
<p>Prediction: I (66.6%)</p>  	Slug / Bubbly	Intermittent	Intermittent
<p>Prediction: I (98.8%)</p>  	Wavy / Annular	Intermittent / Stratified-wavy	Intermittent
<p>Prediction: AW (57.0%)</p>  	Annular	Annular-wavy / Intermittent	Annular-wavy

Figure A-10 is also taken from the work of Ghajar and Tang [12]. In the figure, it appears that the same flow pattern images are shown as those in Figure A-9, except that the images have been cropped differently. Additionally, the images are presented in the region of the Taitel and Dukler [43] flow pattern map corresponding to their respective class labels. The model predictions did not change based on the different cropping, except for of the plug flow pattern image, which was now predicted as bubbly flow.

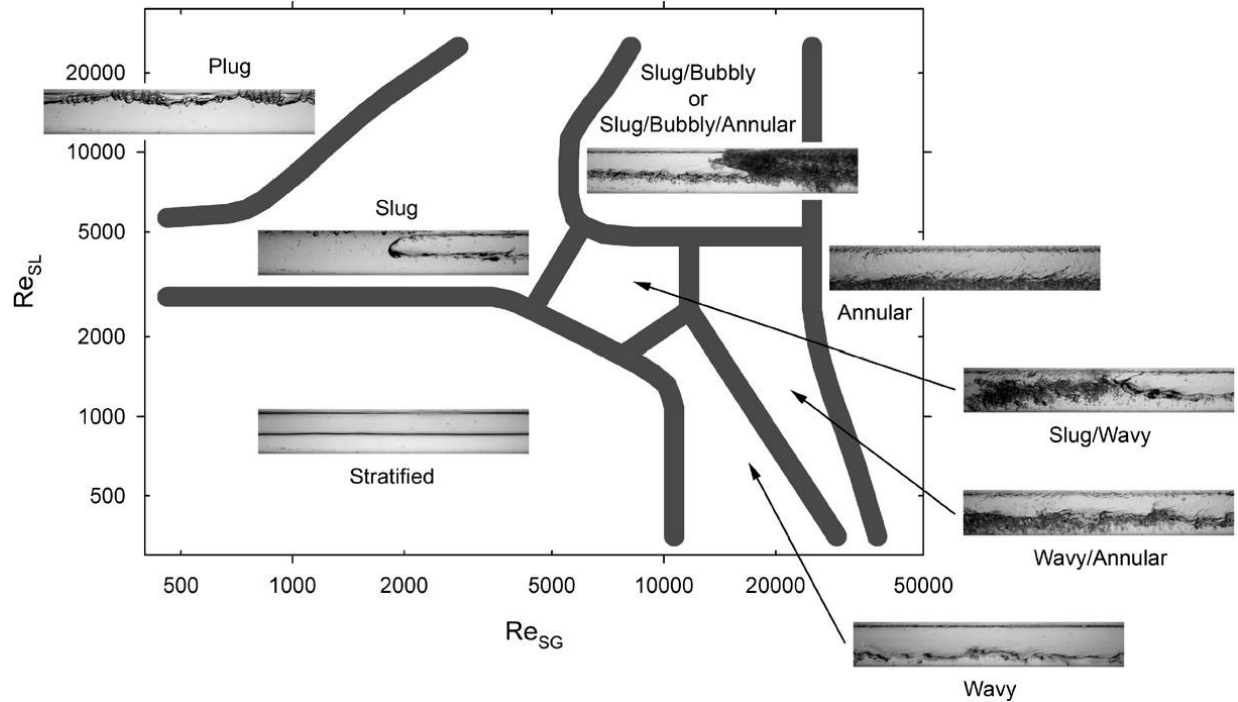
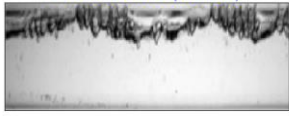
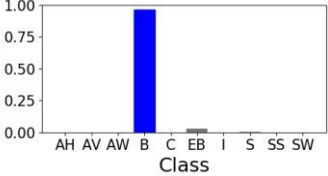
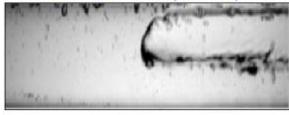
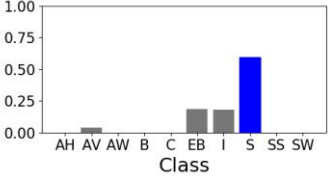


Figure A-10: Taitel and Dukler flow pattern map with representative flow pattern photographs, taken from Ghajar and Tang [12]

Table A-10: Model results of the images showing representative flow pattern photographs for each flow pattern in the Taitel and Dukler flow pattern map, taken from Ghajar and Tang [12]

Image with prediction	Class label from literature source	Class label according to this research	Predicted class label
Prediction: B (96.3%)  	Plug	Elongated bubbles / Bubbly	Bubbly
Prediction: S (59.7%)  	Slug	Slug / Elongated bubbles	Slug


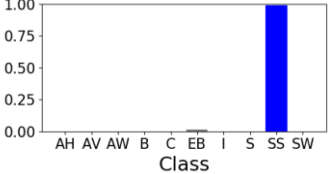

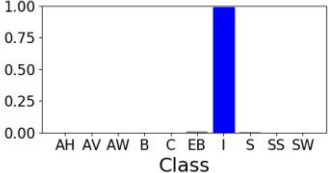
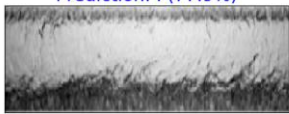
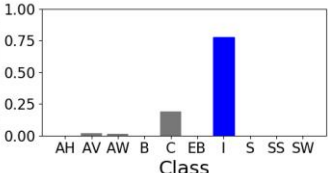
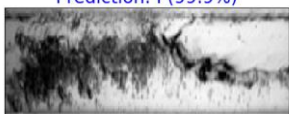
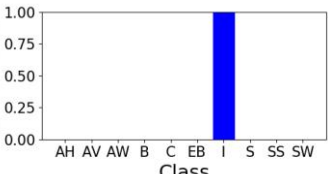
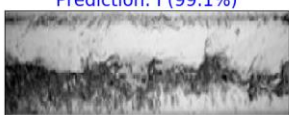
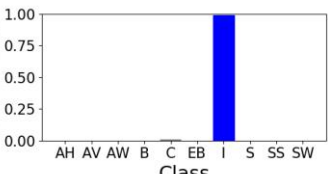
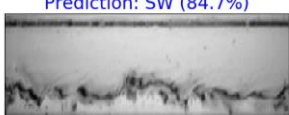
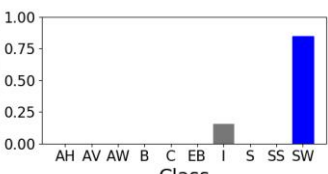
<p>Prediction: SS (98.8%)</p>  	Stratified	Stratified-smooth	Stratified-smooth
<p>Prediction: I (98.8%)</p>  	Slug / Bubbly / Annular	Intermittent	Intermittent
<p>Prediction: I (77.9%)</p>  	Annular	Annular-wavy / Intermittent	Intermittent
<p>Prediction: I (99.9%)</p>  	Slug / Wavy	Intermittent	Intermittent
<p>Prediction: I (99.1%)</p>  	Wavy / Annular	Intermittent	Intermittent
<p>Prediction: SW (84.7%)</p>  	Wavy	Stratified-wavy	Stratified-wavy

Figure A-11 depicts the flow patterns observed in horizontal flow of R-134a refrigerant, taken from the work of Roman et al. [13]. The model correctly predicted the bubbly, elongated bubbles, slug, and stratified-smooth flow pattern images; however, it misclassified the stratified-wavy flow pattern image as elongated bubbles. It is again noted that the elongated bubble flow pattern is referred to as plug flow by the authors.










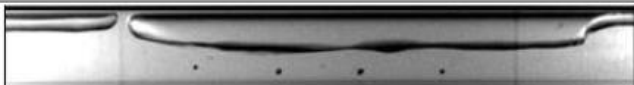



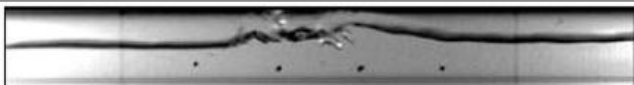

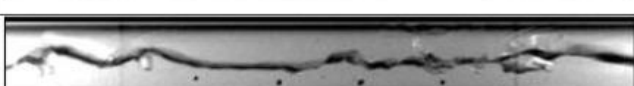


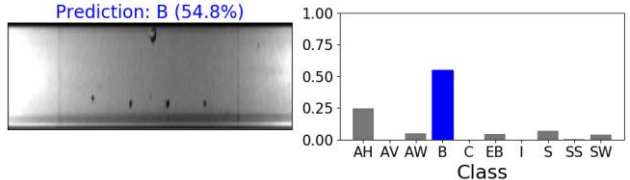
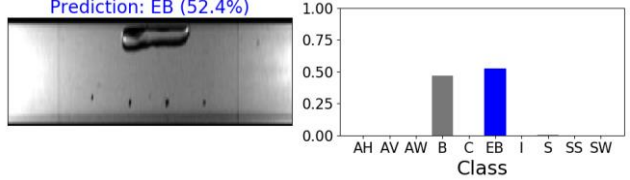
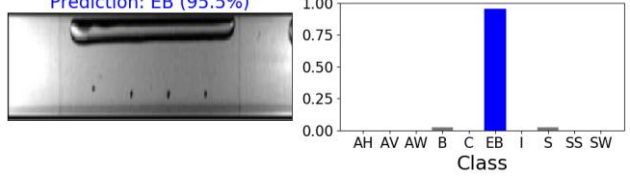
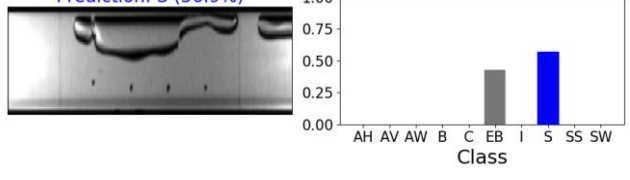
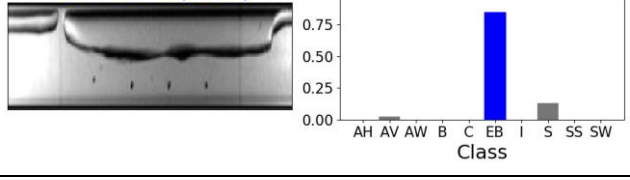
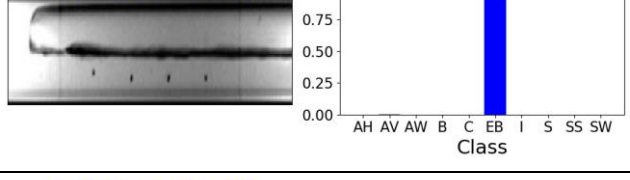
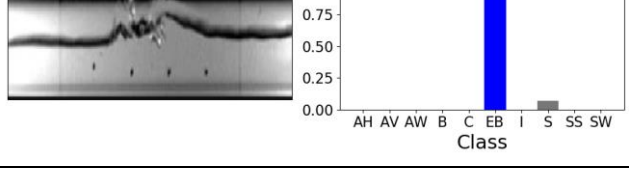
Flow pattern	Drawing of cross section viewed along tube axis	Representative image-viewed perpendicular to tube axis	Time averaged void fraction for flow pattern	Flow rate (L/min) for image	Heater power (W) for image
Bubbly			0.01–0.1	0.5	25
Bubbly - transitional			0.1–0.15	0.5	75
Plug			0.15–0.25	0.3	30
Plug - transitional			0.25–0.40	0.3	50
Slug			0.40–0.55	0.3	75
Slug - transitional			0.55–0.60	0.3	125
Stratified wavy			0.60–0.65	0.3	180
Stratified wavy - transitional			0.65–0.75	0.2	250
Annular			0.75–0.95	0.1	300

Figure A-11: Depiction of flow patterns seen during two-phase refrigerant flow in horizontal tubes, taken from Roman et al. [13]

Table A-11: Model results of the images of two-phase refrigerant flow in horizontal tubes, taken from Roman et al. [13]

Image with prediction	Class label from literature source	Class label according to this research	Predicted class label
<p>Prediction: B (54.8%)</p> 	Bubbly	Bubbly	Bubbly
<p>Prediction: EB (52.4%)</p> 	Bubbly - transitional	Bubbly / Elongated bubbles	Elongated bubbles
<p>Prediction: EB (95.5%)</p> 	Plug	Elongated bubbles	Elongated bubbles
<p>Prediction: S (56.9%)</p> 	Plug - transitional	Elongated bubbles	Slug
<p>Prediction: EB (84.7%)</p> 	Slug	Elongated bubbles	Elongated bubbles
<p>Prediction: EB (99.8%)</p> 	Slug - transitional	Elongated bubbles	Elongated bubbles
<p>Prediction: EB (92.9%)</p> 	Stratified-Wavy	Elongated bubbles / Intermittent	Elongated bubbles

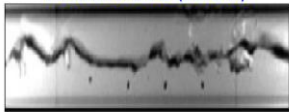
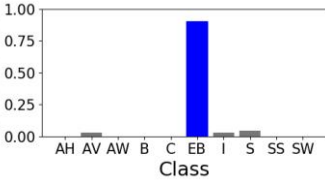
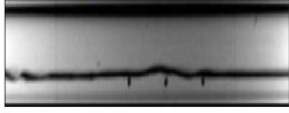
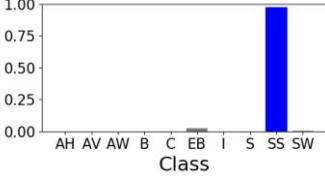
<p>Prediction: EB (90.2%)</p>  	Stratified-Wavy – transitional	Intermittent / Stratified-wavy	Elongated bubbles
<p>Prediction: SS (97.4%)</p>  	Stratified	Stratified- smooth	Stratified- smooth

Figure A-12 depicts the flow pattern images seen during condensation two-phase flow of R245fa refrigerant for a range of different flow conditions. The figure is found in the work of Xing et al. [22]. The model predictions were correct, except for the stratified-smooth flow pattern image which was predicted as stratified-wavy.







Flow pattern	Parameters	Photos
Stratified-wavy (SW)	$G = 199.0 \text{ kg/m}^2\text{s}$ $x = 0.108$ $\theta = 0^\circ$	
Stratified-smooth (SS)	$G = 196.9 \text{ kg/m}^2\text{s}$ $x = 0.104$ $\theta = -15^\circ$	
Intermittent flow (I)	$G = 399.9 \text{ kg/m}^2\text{s}$ $x = 0.054$ $\theta = 15^\circ$	
Churn flow (C)	$G = 400.0 \text{ kg/m}^2\text{s}$ $x = 0.049$ $\theta = 90^\circ$	
Falling film (F)	$G = 300.3 \text{ kg/m}^2\text{s}$ $x = 0.109$ $\theta = -90^\circ$	
Annular flow (A)	$G = 401.4 \text{ kg/m}^2\text{s}$ $x = 0.399$ $\theta = 90^\circ$	

Figure A-12: Observed flow patterns from the work of Xing et al. [22]

Table A-12: Model results of the observed flow pattern images from Xing et al. [22]

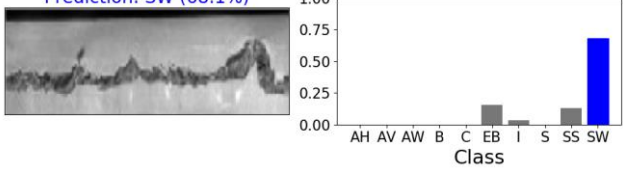
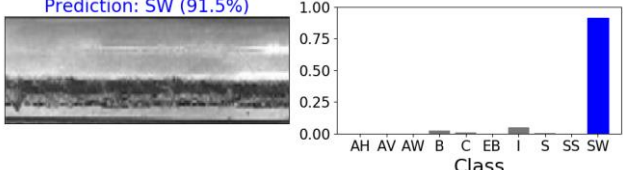
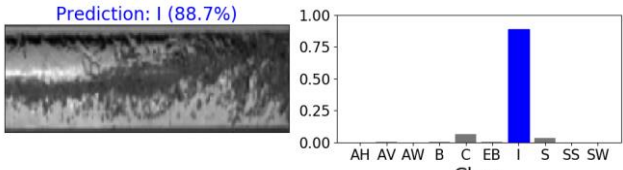
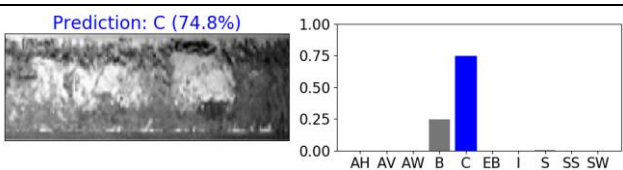
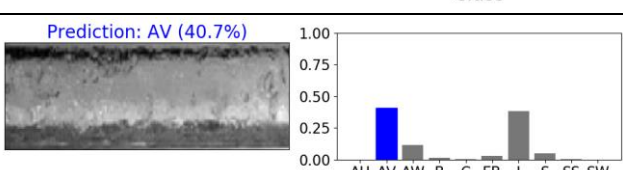
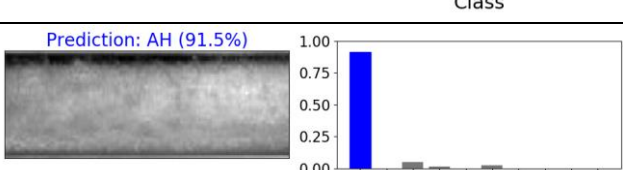
Image with prediction	Class label from literature source	Class label according to this research	Predicted class label
<p>Prediction: SW (68.1%)</p> 	Stratified-wavy	Intermittent / Stratified-wavy	Stratified-wavy
<p>Prediction: SW (91.5%)</p> 	Stratified-smooth	Stratified-smooth / Stratified-wavy	Stratified-wavy
<p>Prediction: I (88.7%)</p> 	Intermittent	Intermittent	Intermittent
<p>Prediction: C (74.8%)</p> 	Churn	Churn	Churn
<p>Prediction: AV (40.7%)</p> 	Falling film	Annular-vertical	Annular-vertical
<p>Prediction: AH (91.5%)</p> 	Annular	Annular-horizontal	Annular-horizontal

Figure A-13 depicts flow pattern images of the transition between bubbly and slug flow, taken from the work of Gao et al. [121]. In the figure, the images have a high brightness level, distinctly different from the images originally used in this study, yet the model could accurately predict the presence of bubbly and slug flow, respectively. This shows that the image augmentation strategy employed improved the generalisation capability of the model.

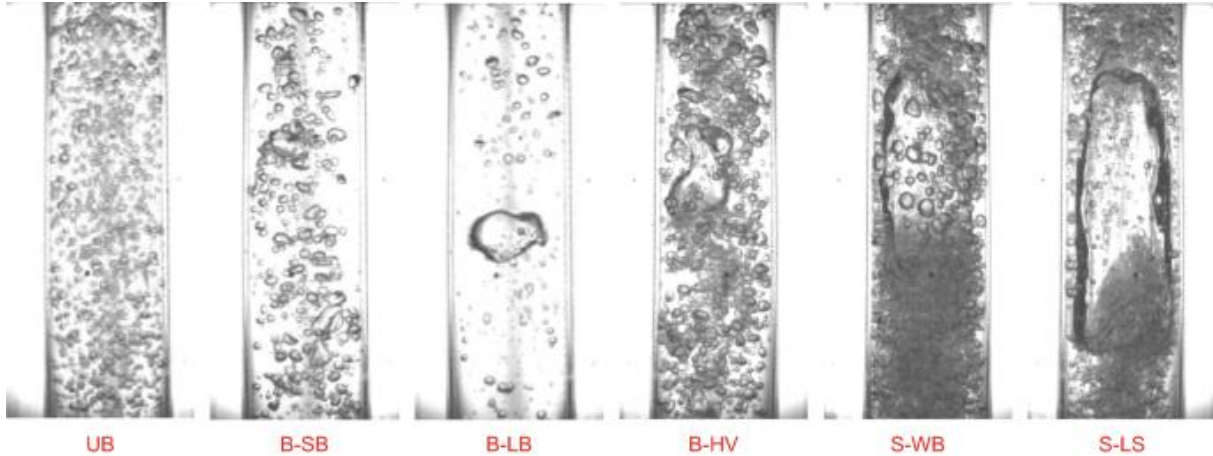
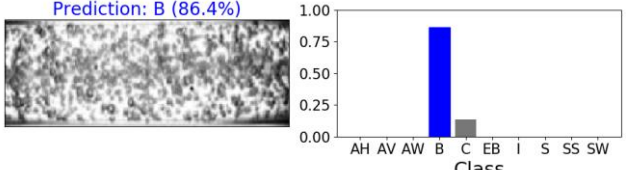
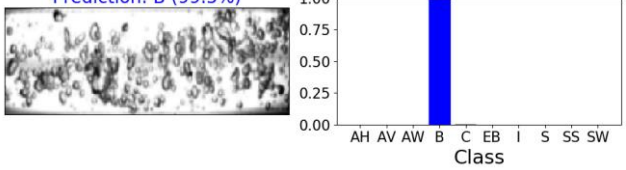
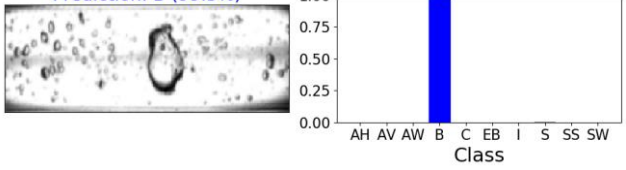


Figure A-13: Flow pattern images during upward vertical flow, taken from Gao et al. [121]

Table A-13: Model results of flow pattern images during upward vertical flow, taken from Gao et al. [121]

Image with prediction	Class label from literature source	Class label according to this research	Predicted class label
<p>Prediction: B (86.4%)</p> 	Uniform bubble flow	Bubbly	Bubbly
<p>Prediction: B (99.5%)</p> 	Bubble flow with small bubbles	Bubbly	Bubbly
<p>Prediction: B (99.5%)</p> 	Bubble flow with large bubbles	Bubbly	Bubbly


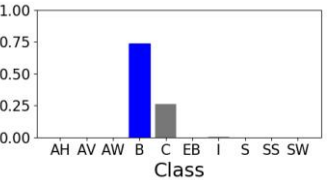
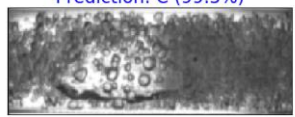
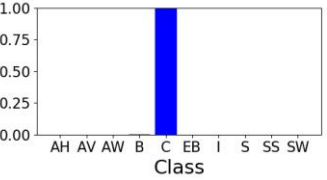

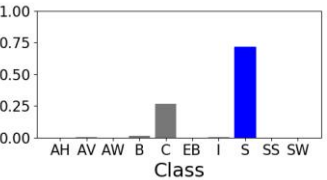
<p>Prediction: B (73.5%)</p>  	Bubble flow with high velocity	Bubbly / Churn	Bubbly
<p>Prediction: C (99.5%)</p>  	Slug flow wrapped in bubbles	Churn	Churn
<p>Prediction: S (71.6%)</p>  	Slug flow with large slugs	Slug	Slug

Figure A-14 depicts the flow pattern images seen during the boiling of nitrogen in a horizontal tube, found in the work of Ohira et al. [14]. The model could correctly predict all the flow patterns according to the definitions of this study. Therefore, this result showed that learning from visualisation data allows improved generalisation from condensation two-phase flows to boiling two-phase flows.

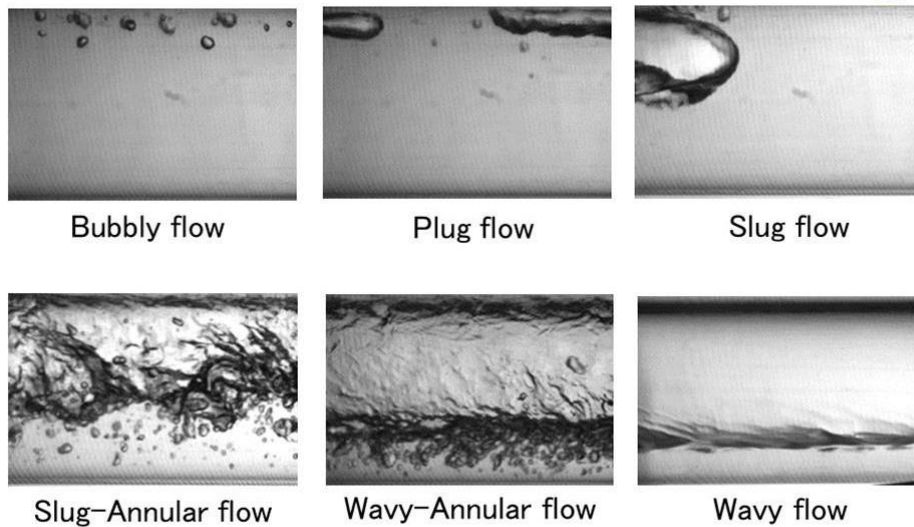
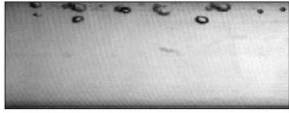
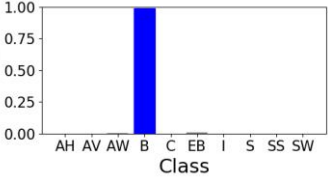
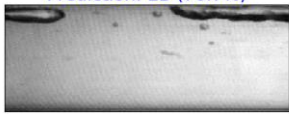
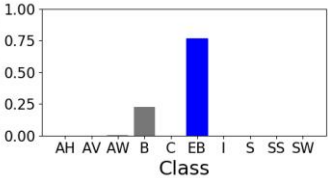
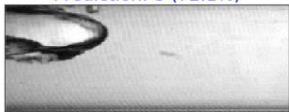
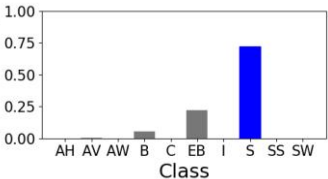
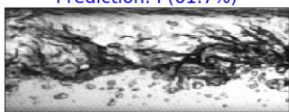
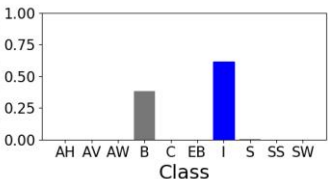
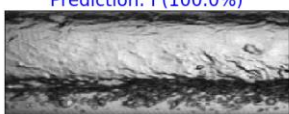
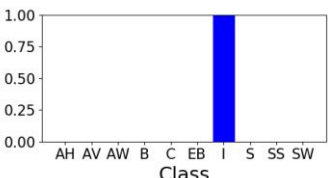
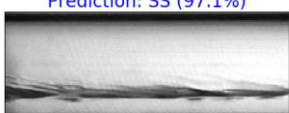
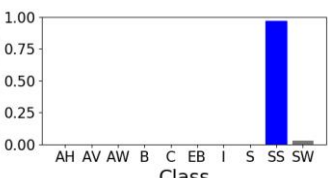


Figure A-14: Flow patterns depicting the boiling of nitrogen in a horizontal tube, taken from Ohira et al. [14]

Table A-14: Model results of images depicting the boiling of nitrogen in a horizontal tube, taken from Ohira et al. [14]

Image with prediction	Class label from literature source	Class label according to this research	Predicted class label
<p>Prediction: B (99.0%)</p>  	Bubbly	Bubbly	Bubbly
<p>Prediction: EB (76.7%)</p>  	Plug	Elongated bubbles	Elongated bubbles
<p>Prediction: S (72.1%)</p>  	Slug	Slug	Slug
<p>Prediction: I (61.7%)</p>  	Slug / Annular	Intermittent	Intermittent
<p>Prediction: I (100.0%)</p>  	Wavy / Annular	Intermittent / Stratified-wavy	Intermittent
<p>Prediction: SS (97.1%)</p>  	Wavy	Stratified-smooth	Stratified-smooth

A.3 Conclusion

This section has presented flow pattern images from the literature. The prediction of the final model presented in this investigation was shown for each of the images, respectively, to validate the generalisation capability of the model. In general, the results showed that the model generalised well to

flow pattern images obtained from sources external to the University of Pretoria's experimental set-up, as well as to images that showed distinct differences to those originally investigated in this study, such as differing tube geometries, brightness, contrast, and choice of fluids.