2024

Effective Drag Coefficient Prediction on single-view 2D Images of Snowflakes

Cameron Hudson
University of Vermont

Effective Drag Coefficient Prediction on single-view 2D Images of Snowflakes

A Thesis Presented

by

Cameron Hudson

to

The Faculty of the Graduate College

of

The University of Vermont

In Partial Fulfillment of the Requirements
for the Degree of Master of Science
Specializing in Complex Systems and Data Science

January, 2024

Defense Date: November 13, 2023
Thesis Examination Committee:

Safwan Wshah, Ph.D., Advisor
Chris Danforth, Ph.D., Chairperson
Jeffrey Marshall, Ph.D.
Holger Hoock, DPhil, Dean of the Graduate College

Abstract

The drag coefficient of snowflakes is an crucial particle descriptor that can quantify the relationships with the mass, shape, size, and fall speed of snowflake particles. Previous studies has relied on estimating and improving empirical correlations for the drag coefficient of particles, utilizing 3D images from the Multi-Angled Snowflake Camera Database (MASCDB) to estimate snowflake properties such as mass, geometry, shape classification, and rimming degree. However, predictions of the drag coefficient with single-view 2D images of snowflakes has proven to be a challenging problem, primarily due to the lack of data and time-consuming, expensive methods used to estimate snowflake shape factors such as sphericity and convex hull. In this paper, we propose a cost-effective and time-efficient approach to address the challenges in predicting the drag coefficients from single-view 2D images of falling snowflakes. Our method combines EfficientNetB7 for image preprocessing to remove the background and border from snowflake images, Kernel Principal Component Analysis (KPCA) to extract meaningful features from the snowflake images, and Machine Learning methods, namely Random Forests, XGBoost models, Multilayer Perceptron (MLP) models, and MLP models trained on distinct Reynolds number flow regimes, to predict drag coefficients using the Locatelli and Hobbs dataset. Through comprehensive evaluation, our model achieved a mean squared error of 0.195, outperforming most existing empirical correlations. Moreover, an evaluation of the feature importance using mean decrease impurity (MDI) showed that the KPCA feature extraction added influential and meaningful data points to our machine learning models.

To family, friends, and professors that helped along the way. Thank you.

Acknowledgements

This material is based upon work supported by the Broad Agency Announcement and Cold Regions Research and Engineering Laboratory (ERDC-CRREL) under Contract No. W913E522C0003. We are thankful for data collected and maintained by John Locatelli and Peter Hobbs, and for A.J. Heymsfield for providing the Locatelli and Hobbs Dataset.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Snowflakes are naturally formed particles with very complex shapes, occurring in a wide variety of forms and crystal forms. Due to differences in atmospheric conditions such as temperature, supersaturation, and turbulence, ice crystals will undergo different forms of growth including vapor deposition and aggregation as they fall. Upon reaching ground level, frozen hydrometeors have already grown by the previously mentioned processes, ergo making certain mature forms more common, such as aggregates and rimed particles, i.e., graupel, while other habits such as hexagonal plates and columnar crystals are less common, tending to occur mostly in the early stages of growth in clouds. In any case, the shape and microphysical properties of ice particles can vary widely, making them difficult to quantify and predict. One method of approaching the prediction of their properties is by observing the drag coefficient, a non-dimensional value that describes a shape's resistance flowing through a fluid such as air. With a constant shape, the drag coefficient varies with the Reynolds number (the ratio between the inertial and viscous forces of a fluid flow) and gives a measure of the "scale" of the flow involving an object. By establishing these non-dimensional quantities, it is possible to quantify the relationship between the mass of a falling particle, its fall speed, its shape, and its drag coefficient, at any scale. But, it is a rather expensive and time-consuming process to get these snowflake features. By the current means with snowflake data, one must record the mass and fall speed to get the drag coefficient, which limits simple data collection of snowflakes. If a team wants to record data and start building their own database that includes the drag coefficient, one would need to invest in equipment to get 3D snowflake structures [1] to get the

mass [8] or save each snowflake to melt down to get the mass [2]. Then use IR emitter-detector pairs to estimate the fall speed [1], additional equipment like photomultiplier tubes to record the fall speed [2], or with cameras with a frame per second above 100, by observing the distance the particle traveled between frames. Currently, the only simple, cost-effective, and easy-to-use methods to obtain the drag coefficient of falling particles is with empirical correlations. There exist empirical correlations that require us to obtain a variety of particle descriptors such as the sphericity or convex hull, which requires additional equipment to obtain. There also exists simpler empirical correlations to obtain drag coefficient like Heymsfield and Westbrook [3], that just require the 2D snowflake image to obtain features such as Reynolds number and area (see equation 2.1). In this thesis, we give an alternative method to estimating the drag coefficient that requires just a single-view 2D snowflake image, its shape class, and Reynolds number. We seek to accomplish this by investigating different machine learning algorithms such as EfficientNet Convolution Neural Network (CNN) [9], random forests [10], XGBoost models [11], and multi-layer perceptron (MLP) [12] neural network, that use the Reynolds number, shape classes, and 2D snowflake image to obtain the drag coefficient of snowflakes.

Drag coefficient prediction of falling snowflakes is challenging because of the lack of data to support the problem. Several studies have collected data on the shape, size, mass, and fall velocity of snow particles, but lack drag coefficient data. There also exists uncertainty in this collected data that propagates and causes uncertainty in the drag coefficient data. Locatelli and Hobbs [2] recorded the mass and fall speed, along with high-quality snowflake images, diameter, fall velocity, air density, air viscosity, shape class, and Reynolds number for each snowflake, but only contains 128

snowflakes. These variables were then used to calculate the drag coefficient for each snowflake (see equation 3.1). The most common snowflake image data that is collected is with three cameras, called the Multi-Angle Snowflake Camera (MASC). The MASC captures three images of the same falling snowflakes at the same time, making it possible to recreate the snowflakes in 3D. This piece of equipment is expensive, which makes it inaccessible to most research groups, but a dataset of around a million multi-angle snowflake images collected from around the world exists, called the Multi-Angle Snowflake Camera Database (MASCDB) [1], along with many snowflake descriptors that can be used to calculate the drag coefficient. Many researchers take advantage of the 3D structure that is possible with the MASCDB [1], such as Leinonen et al [8] and Leinonen and Berne [13]. In general, researchers have found using 3D snowflake structures has improved their methods, such as Köbschall et al. (2023) [14], that gives a new estimation for the drag coefficient that uses artificially generated 3D snowflakes using code from Leinonen et al 2013 [15], where the drag coefficient and shape factors are known. They performed tank experiments by placing 3D printed artificial generated snowflakes at the bottom of the tank in flow regime-glycerol mixtures, to control the Reynolds Number, and allowing to fall upward. The fall speed, orientation of snowflake, and 2D projected areas were captured. They improved the estimation of drag coefficient by creating a convex hull volume, a 3D convex polygon with straight edges, that contain all snowflake values while minimizing area, around the 3D snowflakes. The features sphericity, cross-wise sphericity, and length-wise sphericity were calculated from the convex hull 3D geometry. These features were then inserted into Hölzer and Sommerfeld [4], Ganser [5], and Haider and Levenspiel [7] accordingly to calculate the drag coefficient. These drag coefficients were

then compared to the drag coefficient calculated with sphericity values from the original snowflake geometry. The application of the convex hull geometry was observed to improve the accuracy of the prediction of the drag coefficient compared to the use of the original geometry [14]. Since the convex hull and sphericity values are required, this method is inapplicable to snowflake data captured with single-view 2D camera setups. In this thesis, our objective is to present an approach that accurately predicts the drag coefficient without the convex hull or sphericity values. This is achieved by using features extracted using PCA [16] and KPCA [17] from the snowflake image. These extracted features will be used with Reynolds number and shape class to accurately predict the drag coefficient using various machine learning models [10–12].

A comprehensive classification of ice crystal habits, nearly ubiquitous in the field of cloud microphysics, was created by Magono and Lee [18] by analyzing $30,000$ microscopic photographs and classifying the snowflakes into 80 different shape classes. This classification scheme was also applied to the snowflakes collected by Locatelli and Hobbs [2]. Extracting features from the snowflake image proved to be helpful in shape classification by Leinonen and Berne [13], but has yet to be shown for the estimation of the drag coefficient using snowflake images, but is believed to help in the prediction from Leinonen and Berne's analysis. We seek to show this by extracting features from the snowflake image and estimating the drag coefficient with machine learning methods and showing that the features are meaningful and helpful. We can understand what features are meaningful in the prediction the drag coefficient, by a feature importance analysis on the random forest with a process called mean decrease impurity (MDI) [19, 20]. MDI calculates the feature importance by counting the number of times a feature is used to split a node in the tree (see Chapter 4). This

analysis can illustrate whether the extracted features enhance the prediction accuracy, as evidenced in the study by Leinonen and Berne 2020 [13]. Our goal is to improve prediction by leveraging modern machine learning techniques on the snowflake image itself with the Reynolds number and shape class from Locatelli and Hobbs [2], to outperform empirical correlations that explore the relationships between mass, fall speed, and shape factors [3–7].

The main objective of this thesis is to leverage single-view 2D snowflake images to enhance the prediction of the drag coefficient of falling snowflakes. This is achieved by applying machine learning methods on snowflake descriptors along with extracted features from snowflake images. To the best of our knowledge, this thesis is the first that leverages the snowflake image via dimensional reduction to predict the drag coefficient. This method is cost-efficient, fast, and easy to use, allowing for drag coefficient estimation to be more accessible to single-view 2D image datasets. For this, four main steps were taken for drag coefficient estimation. Firstly, data was sourced from Locatelli and Hobbs [2]. Every snowflake image in the dataset was then processed by the deep learning algorithm EfficientNetB7 CNN [9] to remove the background and border, as having extra noise in the background would negatively affect the dimensional reduction on the images. Secondly, the drag coefficient was calculated for each snowflake image in the Locatelli and Hobbs [2] dataset. Thirdly, dimension reduction methods such as PCA [16] and KPCA [17] were applied to all the processed snowflake images to further compress the image to one or more features. Lastly, various machine learning methods such as random forests [10], XGBRegressor models [11], Multi-Layer Perceptron (MLP) Neural Networks [12], and an ensemble of MLP models trained on different Reynolds number regimes were trained to

predict drag coefficient on snowflakes. An analysis of the number of features extracted versus the mean square error (MSE) for the model is presented. Then, the drag coefficient estimation is compared to drag coefficient correlations found in literature (Heymsfield and Westbrook [3], Hölzer and Sommerfeld [4], Ganser [5], Ganser with Leith [6] shape factor, and Haider and Levenspiel [7]). Additionally, geometric features such as sphericity, cross-wise sphericity, and length-wise sphericity were predicted by a customized Physics-Based Deep Learning model that allows our predictions to be compared against the state-of-the-art empirical correlation models that use the different types of sphericity (Hölzer and Sommerfeld [4], Ganser [5], Ganser with Leith [6] shape factor, and Haider and Levenspiel [7]).

1.2 Problem Statement

As stated above, this thesis is concerned with the challenges with drag coefficient prediction on single-view 2D images. We propose a method to remove the background and border of single-view 2D images with EfficientNetB7 [9], use KPCA [17] to extract features from the images, to be used with the Reynolds number and shape class on an various amount of machine learning methods to predict the drag coefficient on the Locatelli and Hobbs dataset [2]. This method is cost-effective, as it uses purely tabular data, and is time-efficient. Researchers can capture single-view 2D images of falling snowflakes, where Reynolds number and shape class can be determined from visual analysis of each snowflakes, to accurately predict the drag coefficient, avoiding expensive equipment and methods to produce 3D images to extract convex hull geometry and sphericity quantities.

1.3 Contributions

The contributions of this thesis are (1) We propose to use the snowflake image as a feature via dimensional reduction algorithms for improving the prediction of drag coefficient by advanced machine/deep learning methods. This process reduces the cost and time previously required to calculate the drag coefficient using different shape factors such as sphericity or convex hull. (2) We improve and set benchmarks for machine learning and deep learning models for the prediction of the drag coefficient on single-view 2D image data. (3) A comprehensive study is conducted on variables that are critical for drag coefficient prediction via feature importance analysis. It is shown that the extracted features contribute greatly and increase the accuracy of the prediction models compared with the existing empirical correlations mentioned above. The first two contributions are addressed in Chapter 3 and the last contribution is addressed in Chapter 4.

1.4 Thesis Overview

Chapter 2 contains a literature review where we provide an in-depth review of various empirical correlations and which we are using to compare our model against, summaries of the two main databases used: Locatelli and Hobbs [2] and MASCDB [1], and previously used machine learning methods applied to snowflake image data. The main results of this thesis are covered in Chapter 3, where we present our method of removing the background on single-view 2D snowflake images, feature extraction methods used, and machine learning methods used. Chapter 4 covers an in-depth

discussion of results and methods, containing the feature importance analysis of our model. Finally, in Chapter 5, we summarize the thesis and present ideas for future research.

Chapter 2

Preliminaries

2.1 Literature Review

2.1.1 Empirical Correlations

During the last half century, there have been many efforts to improve the prediction and correlation of the drag coefficient of generic particles. Researchers have used different predictors to correlate with the drag coefficient such as the Reynolds numbers and sphericity, which is defined as the ratio between the surface area of the volume equivalent sphere around the snowflake and the snowflake itself; how close the snowflake is to a sphere [4, 7]. To improve drag coefficient correlations, more predictors have been incorporated such as using the convex hull volume geometry of the snowflake to extract sphericity, cross-wise, and length-wise sphericity [14]. Researchers have also used the Newtons/Stokes shape factor [5], min and max values of inertia tensor [21], porosity, the measure of the void spaces in a material [22, 23], and permeability, the measure of the ease with which fluids will flow though a porous rock [24]. Additionally, researchers have used the combination of variables flatness, elongation, and diameter [25], and the combination of the variables nominal diameter, Reynolds Number, the ratio of surface-equivalent-sphere to the nominal diameters, and circularity [26, 27] to improve estimations on drag coefficient. Most of these descriptors require additional equipment and costs to be obtained, such as convex hull geometry, different sphericity values, and most shape factors. It may make the empirical correlation more accurate, but it makes empirical correlation more difficult to use overall and researchers may not have the equipment or funding to obtain the descriptors necessary to use the different empirical correlations. The aim of this thesis

is to find and present a simple and effective way to obtain the drag coefficient using cost-effective and time-efficient methods on a single-view 2D image. This method doesn't require additional equipment such as the 3D snowflakes captured with the Multi-Angled Snowflake Camera or methods that use the 3D structure to estimate shape factors such as the ones mentioned above. This method seeks to be budget friendly for budget limited researchers.

The empirical correlations that our model will be comparing against are (Eq. 2.1) Heymsfield and Westbrook [3], (Eq. 2.2) Hölzer and Sommerfeld [4], (Eq. 2.3) Ganser [5], and (Eq. 2.4) with Leith shape factor correction [6], and (Eq. 2.5) Haider and Levenspiel [7], shown below. In the following formulas, $A_r^k = \frac{\text{Area}}{(\pi/4)\times\text{diameter}^2}$, ψ = Sphericity, $\psi_\parallel$ = Length-wise Sphericity, $\psi_\perp$ = Cross-wise Sphericity, and Re = Reynolds number. It is important to note that some correlations focus on the drag coefficient of solely particles, and not snowflake particles. The Heymsfield and Westbrook [3] correlation focuses on ice particles, including stellar and dendritic crystals, needles, open bullet rosettes, and low-density aggregates. The Hölzer and Sommerfeld [4] correlation focuses on non-sphericical particles, including particle shapes such as cubes, cube-octahedrons, octahedrons, tetrahedrons, isometric particles, and disks and plates. They do not focus on snowflake particles. For the Ganser and Leith corretion [5,6] correlations, the focus is also on general sphericical and non-spherical particles, using shapes such as Hölzer and Sommerfeld. Finally, the Haider and Levenspiel [7] focuses also on sphericical and non-spherical, including solid particles that are not porous unlike most snowflakes.

$$C_d = C_0(1 + \frac{\delta_0}{\sqrt{Re}})^2,\ C_d^* = C_d A_r^k,\ \text{w/}\ C_0 = 0.35, \delta_0 = 8.0, k = 0.5 \tag{2.1}$$

$$C_D = \frac{8}{\mathrm{Re}}\frac{1}{\sqrt{\psi_\parallel}} + \frac{16}{\mathrm{Re}}\frac{1}{\sqrt{\psi}} + \frac{3}{\sqrt{\mathrm{Re}}}\frac{1}{\psi^{\frac{3}{4}}} + 0.4210^{0.4(-\log\psi)^{0.2}}\frac{1}{\psi_\perp} \tag{2.2}$$

$$\begin{gathered} C_D = \frac{24 * k_S}{\mathrm{Re}}(1 + 0.1118(\frac{\mathrm{Re} * k_N}{k_S})^{0.6567}) + \frac{(0.4305 * k_N)}{(1 + \frac{3305}{\frac{(\mathrm{Re} * k_n)}{k_S}})}, \\ k_S = \frac{1}{3} + \frac{2}{3}\sqrt{\psi}, \\ k_N = 10^{1.8148(-log\psi)^{0.5743}} \end{gathered} \tag{2.3}$$

$$\begin{gathered} C_D = \frac{24 * k_S}{\mathrm{Re}}(1 + 0.1118(\frac{\mathrm{Re} * k_N}{k_S})^{0.6567}) + \frac{(0.4305 * k_N)}{(1 + \frac{3305}{\frac{(\mathrm{Re} * k_n)}{k_S}})}, \\ k_S = \frac{1}{3}\sqrt{\psi_\perp} + \frac{2}{3}\sqrt{\psi}, \\ k_N = 10^{1.8148(-log\psi)^{0.5743}} Mul \end{gathered} \tag{2.4}$$

$$\begin{gathered} C_D = \frac{24}{\mathrm{Re}}(1 + C_1\mathrm{Re}^{C_2}) + \frac{C_3}{(1 + \frac{C_4}{\mathrm{Re}})}, \\ C_1 = \exp(2.33 - 6.46\psi + 2.45\psi^2), \\ C_2 = 0.096 + 0.556\psi, \\ C_3 = \exp(4.90 - 13.89\psi + 18.42\psi^2 - 10.26\psi^3), \\ C_4 = \exp(1.47 + 12.26\psi - 20.73\psi^2 - 15.89\psi^3) \end{gathered} \tag{2.5}$$

Since we can convert our drag coefficient estimate to mass by the following: $\text{Mass} = (\frac{\text{C}_\text{d}}{\text{Acceleration due to Gravity}}) \times 0.5 \times \text{Air Density} \times \text{Velocity}^2 \times \text{Area}$, we can compare our model to mass prediction methods as well. We will be comparing our mass estimation to Leinonen et al 3DGAN [8], Matrosov et al [28] and Baker and Lawson [29].

2.1.2 MASCDB and Drag Calculation

The Multi-Angle Snowflake Camera (MASC) is a recent development in snow research that uses three high-resolution, synchronized cameras at different angles to observe falling hydrometeors. A large volume of MASC data was published in Grazioli et al 2022 [1] and has proven to be immensely valuable to the scientific community, spawning a multitude of efforts in quantifying and predicting snow properties [8, 13, 30–32]. The data does not contain measured mass values or drag coefficients of the falling snowflakes. Instead, this analysis is done by Leinonen et al [8] with their 3DGAN model, which uses Generative Adversarial Networks (GANs) [33], specifically the Wasserstein GAN with gradient penalty (WGAN-GP) [34, 35], to estimate the mass of snowflakes in MASC images. The 3DGAN includes a generator and discriminator. The generator constructs a 3D structure based on the triplet snowflake images while the discriminator checks that the constructed structure matches the triplet images, represented by the Wasserstein distance between the generated image and the real image. The 3DGAN researchers acknowledge that this process has drawbacks such as the formation of the 3D structure may not be representative of real snowflake formation, the 3D structure may not be entirely accurate, and loss of textural information by using silhouettes of the triplet MASCDB images instead of grayscale MASCDB

images. In the MASCDB paper [1], the authors also mention that this process of mass estimation only works if all three of the images are good quality, and are of a larger particle size. Lastly, the 3DGAN was only validated on 3D printed snowflakes, where mass was known, and not the MASCDB snowflakes. This can account for the scatter in Figures 2.1, 2.2 (see Chapter 4 for additional review of the MASCDB).

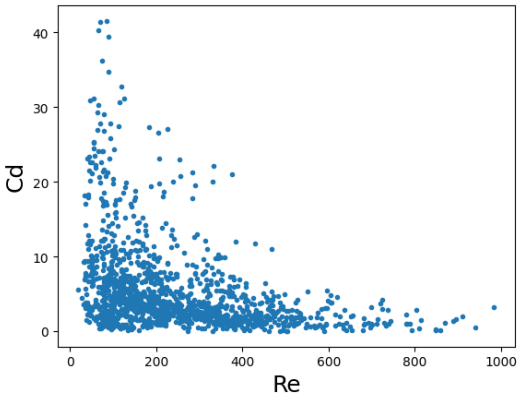

Figure 2.1: Reynolds Number vs Drag Coefficient of Snowflakes in MASCDB [1]

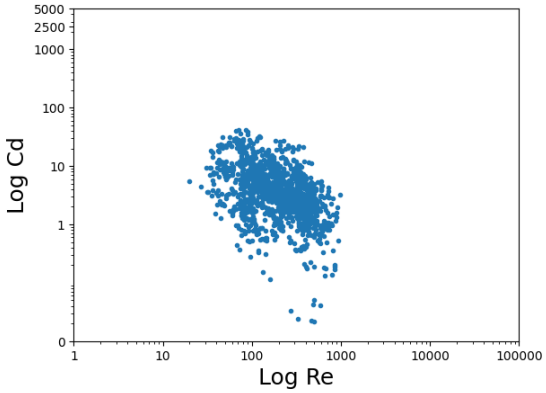

Figure 2.2: Log-Log Plot of Reynolds Number vs Drag Coefficient of Snowflakes in MASCDB [1]

Additionally, the fall velocity measurements in the MASCDB are scattered because a windshield was not used in the majority of measurements of fall velocity. This has caused some data points to have unrealistic velocity values. Fitch et al 2021 [36] discusses the limitations of this effect and gives an interval of fall velocities to filter out, such as if the wind speed is greater than 5 m/s and the fall velocity is below 0.45 m/s. This is also noted in the MASCDB original paper by Grazioli et al [1].

2.1.3 Locatelli and Hobbs Dataset Validation

Locatelli and Hobbs [2] recorded the mass and fall speed, along with high-quality snowflake images, diameter, fall velocity, air density, air viscosity, shape class, and Reynolds number for each snowflake. A windshield was used in their data collection on the fall velocity. With this being said, the fall velocity recordings are not subject to the same uncertainty as the MASCDB, overall reducing error on our drag coefficient calculation. They recorded the fall speed of snowflakes via photomultiplier tubes and then caught the snowflakes on a plastic sheet, to reduce shattering or bouncing of snowflakes. To obtain the mass, they took the snowflakes on the plastic sheet and cooled them to below 0 degrees Celsius and microphotographs were taken to measure the the dimensions of the snowflakes. The particles were then melted and photographs were taken of the water droplets that were formed. The mass was then calculated by measuring the diameter of the droplets and by using a mass-size relationship that was made with ice spheres made of varying sizes. Specifically, the ice spheres were melted, the diameter of their droplets was recorded, and then they were frozen again. They were dropped in oil and melted again, recording their true spherical diameter to get mass. A regression curve was fit for the many varying-sized ice spheres, which were used to calculate snowflake masses of the melted snowflake diameters. Mass-size relationships were formed for each 15 shape classes found. This method proved to be very effective in finding the mass of each snowflake, reducing uncertainty in drag coefficient estimate.

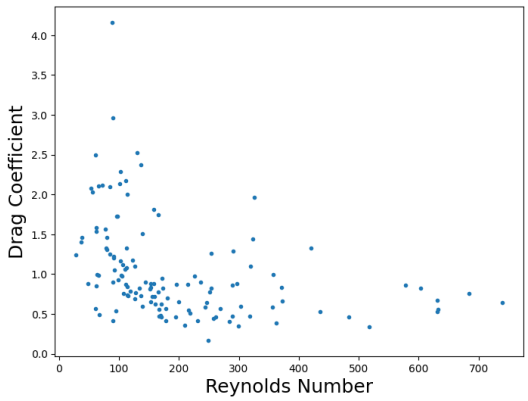

Figure 2.3: Reynolds Number vs Drag Coefficient of Snowflakes in Locatelli and Hobbs (1974) [2]

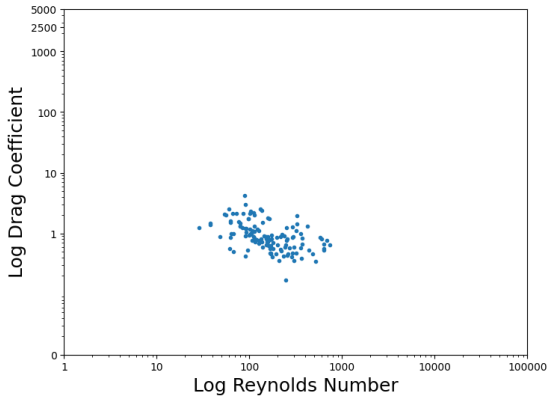

Figure 2.4: Log Reynolds Number vs Log Drag Coefficient of Snowflakes in Locatelli and Hobbs (1974) [2]

2.1.4 Comparison of Databases

The Locatelli and Hobbs [2] and the MASCDB [1] are both good options for our drag coefficient model, both give high quality images and have the data necessary to calculate the drag coefficient. The Locatelli and Hobbs database gives an accurate drag coefficient when using their mass and fall speed recordings. This can be seen in figures 2.3, 2.4 that show the drag coefficient compared to the Reynolds number for the Locatelli and Hobbs dataset [2]. Figure 2.3 follows a clear trend as the Reynolds number increases, the drag coefficient decreases. There is some scatter in Figure 2.3, but the log-log plot of the drag coefficient and Reynolds number show that the data is densely packed with no major outliers present. The major downside of using Locatelli and Hobbs [2] is the limited data present, with only 128 data points. The MASCDB [1] has around three million snowflakes images and data points but has scattered nature which is troublesome for a drag coefficient prediction model. This can be seen in figures 2.1, 2.2, which have a great amount of scatter while having

some outliers present. Further, figure 2.1 doesn't follow closely to known trends of the drag coefficient of particles found in literature, such as presented in Köbschall et al [14]. Even though the MASCDB has far more images, because of the nature in MASCDB [1], the Locatelli and Hobbs [2] dataset is preferred and was used for training and validation data. Refer to chapter 4 for testing on the MASCDB [1].

2.1.5 Machine Learning Approaches

Many existing studies explore and incorporate machine learning techniques for various snowflake descriptor estimation and classification. Most of this work has been done on the MASCDB [1], using three-view 2D images or 3D images. A majority of the of machine learning works has been on the shape classification of snowflakes. Leinonen and Berne [13] use unsupervised shape classification, meaning shape classification with no shape labels. They were able to accomplish this task by training a GAN that maps each MASC image into a vector of latent variables and reconstructs the snowflake. K-medoids [37] (similar to K-means) classified the images into many groups and hierarchical clustering reduced the number of groups found with K-medoids into 6 or 16 different groups. The reconstructed snowflakes were then compared to other snowflakes in its group to see if the latent variables were meaningful in the K-medoids and hierarchical clustering algorithms. There have been more approaches to snowflake shape classification. Hicks et al 2019 [30] use a supervised learning method with a CNN model to classify snowflakes into five different shape classes with an accuracy of 93.4%. Praz et al [31] achieved similar results with classification among 10 shape classes with an accuracy of 94.7% on MASCDB [1], with a regularized multinomial logistic regression (MLR) model. They also used MLR mod-

els to estimate the rimming degree of snowflakes and classification to detect melting snow with equally good results. Leinonen et al [8] estimate mass by using a separate neural network, trained just for mass prediction. They achieved this by using three 2D images as input. The images are downsampled, concatenated, and passed through dense blocks, to give the mass output. They achieved a result of an NRMSE (normalized root mean squared error) of 42 on 3D printed snowflakes. These methods mainly use the MASCDB [1] and most take advantage of the 3D images within the database. In this thesis, we seek to use single-view 2D, gray-scaled, images that would replicate data captured without the use of Multi-Angle Snowflake Camera, such as a single camera or GoPro. This is done by extracting features from the images, and not using the images themselves as input in the prediction models. This method seeks to be more cost-effective in capturing snowflake data and seeks to better analyze and estimate the drag coefficient.

2.1.6 Reynolds Number and Flow Regimes

The Reynolds number is an important descriptor of falling particles that impacts the drag coefficient [21, 38]. The Reynolds number is calculated using equation 2.6 and in the case of falling snowflakes, the fluid used is the atmosphere. In the tank experiments by Köbschall et al [14], the fluid density and viscosity would change respectively with the fluid density and viscosity of the water-glycerol.

$$\mathrm{Re} = \frac{\text{Density of Fluid} \times \text{Terminal Velocity} \times \text{Particle Diameter}}{\text{Viscosity of Fluid}} \tag{2.6}$$

In order to use equation 2.6, one would need to have measurements of the particle diameter and velocity. Both can be obtained via simple camera setup, capturing falling snowflakes with with a FPS above 100. This is currently being investigated by our team with a GoPro, a macro lens, and a wooden tunnel with a wind-shielded, snowflake capture box. With this process, one can calculate the Reynolds number, define the shape class, and get high quality, single-view 2D snowflake images, all necessary for our model inputs.

The Reynolds number of a particle has a great impact on how that particle falls and flows through the atmosphere. Tagliavini et al (2022) [38] breaks up Reynolds number into two groups, low (Reynolds number (Re) $\leq$ 400) and moderate/high (Reynolds number (Re) $>$ 400). In the low Re group, snowflakes are characterized with a steady flow, meaning that the velocity or orientation does not change over time, remaining in the same orientation as it falls. Then for moderate/high Reynolds number snowflake flakes, tend to have an unsteady flow, characterized by their chaotic falling nature, changing velocity, and unknown final orientation. Tagliavini et al (2021) [21] ran simulations with a Delayed-Detached Eddy Simulation model on falling snowflakes at varying Reynolds numbers. It was found that for almost all snowflakes with low Reynolds number, characterized here as Reynolds number (Re) $\leq$ 250, had steady flow. Because of the steady nature of the fall, it was relatively easy for their model to predict, only having $\pm$7% experimental error in predicting drag coefficient. At moderately high Reynolds numbers (Reynolds number (Re) $>$ 250, snowflakes with unsteady flow, their model struggled to predict the drag coefficient, with errors

obtained larger than 30%. It is clear from Tagliavini et al [21, 38] that the Reynolds number has a great impact on how a snowflake falls and the drag coefficient. They present low and moderately high Reynolds number regimes with differences in flow that ultimately effect their drag coefficient prediction for the two regimes. The thesis seeks to expand on this work by splitting up our model into two different Reynolds number regimes, and training two different models to accurately predict the drag coefficient, creating an ensemble method. With just a single model, the model will try and understand the relationships between Reynolds number, shape, and image, but might fall short due to the major differences in those relationships in the different regimes. See Chapter 3 for results on single and ensemble methods.

2.1.7 Overview of Related Methods

We propose to use single-view 2D, gray-scaled images from Locatelli and Hobbs [2] with physics methods to extract the drag coefficient from the dataset and use machine learning to perform a cost-efficient and easy-to-use snowflake drag coefficient prediction. This is done by extracting features from the images, which has been shown to help in Leinonen and Berne [13], and without 3D convolution models, such as Leinonen et al [8] uses for mass predictions and Hicks et al [30] uses for shape classification. By extracting features from the single-view 2D image, we avoid extracting additional snowflake descriptors such as the convex hull or shape factors that various empirical correlations use, while still maintaining accuracy in the drag coefficient prediction. Our results will be summarized and compared directly to various drag coefficient correlations to address our model's performance, which is summarized in Chapter 4.

Chapter 3

Drag Coefficient Prediction

3.1 Preprocessing Data

3.1.1 Drag Coefficient Calculation

This thesis focuses on using the snowflake image itself to improve the prediction of the drag coefficient. The Locatelli and Hobbs [2] dataset includes the snowflake image (see Figure 3.1 for original snowflake image), mass, diameter, fall velocity, air density, air viscosity, shape class, and Reynolds number for each snowflake with identifying numbers for labeling sake. In this thesis, mass, diameter, fall velocity, air density, and air viscosity were not used from the dataset because they have dimensional units; only dimensionless features can be used to predict drag, as itself is dimensionless. Therefore, we are using the Reynolds number and shape classifier from the dataset. The drag coefficient was calculated for each snowflake by equation 3.1, with reference area being the area of the snowflake in the 2D images.

$$C_D = \frac{\text{Mass} \times \text{Acceleration due to Gravity}}{0.5 \times \text{Air Density} \times \text{Velocity}^2 \times \text{Reference Area}} \tag{3.1}$$

3.1.2 Background Removal

The initial step was removing the background of the snowflakes for feature extraction. The snowflakes given by Locatelli and Hobbs (1974) [2] had a non-black background and a colored border around the snowflake edge. Most modern snowflake datasets have their snowflakes against a black background, such as the MASCDB ([1]Grazioli et al 2022). As shown in Figure 3.1, there are multiple parts of the background that are grey, adding extra noise to the image. Feature extraction on the original snowflake

images would add unnecessary information from the image background and border. To get an accurate feature reduction on the image, the only non-black pixels of the snowflake image should be the snowflake itself. To remove the background of the snowflakes, neural networks that specialize in object detection were used. U2-Net [39] and EfficientNetB7 [9] were tested to remove the background of the snowflake, without removing sections of the snowflake. In our experiment, U2-Net [39] could not catch the subtleties of the edges of the snowflakes, resulting in some unremoved background. EfficientNetB7 [9] found and removed the subtleties of the snowflake and succeeded in removing the rest of the background. Therefore, EfficientNetB7 was adopted to process the background of all the snowflakes. Both models failed to remove the blue border Locatelli and Hobbs [2] drew around the snowflakes. Thus, a custom image processing function was made to remove certain colors of blue from the image that were only found on the border. This was done by inspecting the RGB quantities of the border colors and replacing certain blue pixels with black pixels by filtering them from the identified RGB value range. In this manner, the Locatelli and Hobbs [2] snowflake images were just the images of the snowflake with a black background, matching the snowflakes in MASCDB [1]. One sample processed snowflake image is visualized in Figure 3.2. However, the images are of size 2128×3256, which is too large to fit in the deep learning detection model. To improve computational run time, while still keeping a high-quality image, the images were scaled down to 256×256. Each image was inspected manually to assess the quality of the images. The images that lacked quality were removed. Data points with missing data, or outliers, for certain features were also removed. This lowered the data count to 128 points.

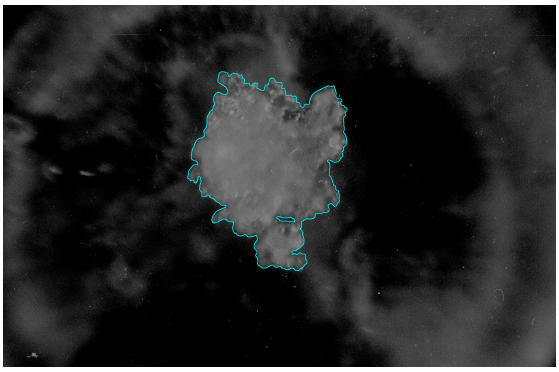

Figure 3.1: Original Snowflake Image from Locatelli and Hobbs (1974)

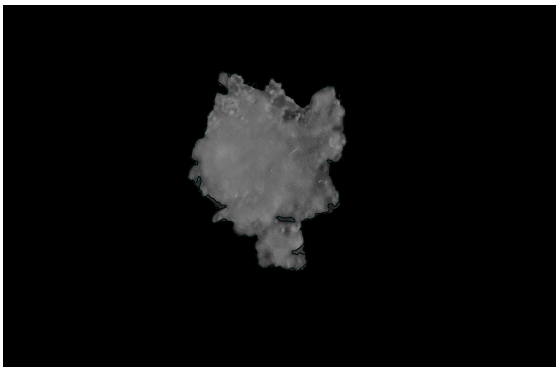

Figure 3.2: Snowflake from Fig 1. with Background and Border Removed

3.1.3 Feature Extraction on 2D Images

For feature reduction, Principal Component Analysis (PCA) [16] and Kernel-Principal Component Analysis (KPCA) [17] were considered. PCA uses the covariance matrix and eigenvalues to reduce the image array to a lower dimension. Kramer (1991) [40] demonstrated that the combination of non-linear PCAs and neural networks perform well together. Cao et al (2003) [41] compared PCA, KPCA, and ICA (independent component analysis), and presented that KPCA performed the best for feature extraction/reduction. Cramer et al [42] found similar results to Cao et al [41], that found that non-linear PCA (KPCA) via 'rbf' kernel performed the best in their feature extraction comparison between KPCA and autoencoders. Both methods are cost-effective and robust but it is clear from literature [41, 42] that KPCA is better for our use case than PCA.

The Radial Basis Function (RBF) [43] kernel empowers PCA to capture nonlinear patterns in data, whereas the original PCA only captures linear patterns in data. Thus, the RBF kernel allows a more accurate feature reduction on images with com-

plex structures, such as snowflakes. To get the data ready for KPCA, the images were flattened to a single array containing all the pixel values of the image. Normalization was then performed to standardize the pixel data to a mean of 0 and a variance of 1. This will ensure that all the pixels of the snowflake will contribute more equally to the KPCA analysis, preventing dominance from any particular feature due to its scale, such as the black background. The number of features extracted from KPCA was tested for the best results. In total, the number of features extracted was limited to 1, 3, and 5. The extracted features were merged with each data point in our dataset, leaving us with three datasets, one for each number of features extracted.

3.2 Machine Learning Methods

To test if feature extraction is pulling meaningful features, different machine learning models were made to test the accuracy of models with the different numbers of extracted features. These models were compared to models with no image feature reduction feature to evaluate if the extracted features were improving the model. The different models tested were (1) random forests [10], (2) XGBRegressor [11], (3) Multi-Layer Perceptron (MLP) neural network [12], and (4) Ensemble of two MLP models split at Reynolds number of 158. Sklearn [44], XGBoost [11], and PyTorch [45] packages were used for implementation respectively. Mean Squared Error (MSE) is adopted for evaluating the accuracy. The shape class in the Locatelli and Hobbs [2] dataset is represented as a string, thus one hot encoding was used to transform shape strings into classes represented by integers.

3.2.1 Random Forests

Random Forest [10] is a type of ensemble method that combines decision trees to create a more powerful predictive model. A set number of decision trees are made and each tree is trained with bootstrapped samples (also known as bagging), a random sample of training data with replacement, allowing for each tree to be trained on different but similar samples of the data. The trees are then averaged for the final regression prediction. Initial testing was done with a random forest model with default hyperparameters to create a control model to test against. After, models were trained with random search and grid search with cross-validation. Randomized search was done on the hyperparameters: n_estimators, max_features, max_depth, min_samples_split, min_samples_leaf, and bootstrapped samples or not. The data was split into train and test sets by a ratio of 75% to 25% with a random state of 42. The most optimal hyperparameters were saved and used for error calculations.

3.2.2 Extreme Gradient Boosting

XGBoost (Extreme Gradient Boosting) [11] has been known to perform very well on tabular data. Given the feature reduction on the snowflake, our dataset is purely tabular, making XGBoost an excellent model for drag coefficient prediction. For our case, the XGBoostRegressor model is implemented. The great performance of XGBoostRegressor is because it creates an ensemble of sequential decision trees which are added one by one to correct prediction errors made by previous decision trees. This process is known as boosting, hence the name of the model. XGBoost differs from random forests mainly by using boosting over bagging to create its decision trees

and add trees sequentially rather than averaging the results of each decision tree. Again, an initial model is created with XGBoostRegressor to be used as a control with Reynolds number, shape class and without any KPCA features. Both were done with the grid search of variables: learning_rate, max_depth, and n_estimators with a KFold cross-validation of 5. The data was split into train and test sets by a ratio of 75% to 25% with a random state of 42. The best estimator for each cross-validation and hyperparameter search was used for testing.

3.2.3 Multi-Layer Perceptron

Multi-Layer Perceptron (MLP) [12] models are a type of neural network that is used across the machine learning field to tackle regression and classification problems. MLP models consist of an input layer, hidden layers, and an output layer. The input layer takes in the data and has nodes that corresponds to a feature in our dataset. The input layer feeds into multiple hidden layers where the learning is done. Each node has bias and each connection between the nodes has a corresponding weight. By training the model we are adjusting these weights and bias during a process called forward propagation. This is done by using activation functions, in our case, ReLU functions, that take the summed output of the hidden layers to produce an output that is used in the next hidden layer, or output layer. The activation function, ReLu, introduce non-linearity to our model. The output layer takes in the input from the last hidden layer and produces a final prediction for our model. In our case, we are predicting drag coefficient, therefore our output layer returns a single node, which is the prediction. During the training process, the backpropagation compares our

prediction values at each layer and adjusts weights and biases to minimize our error (mean squared error, equation 3.2.3).

$$\text{MSE} = \frac{1}{N}\sum_{i=1}^{N}(x_i - y_i)^2 \tag{3.2}$$

To accommodate the need that the input data has different numbers of features because KPCA is employed for image feature extraction, we design three variant models which take 3, 5, and 7 input features for the number of KPCA output of 1, 3, 5 respectively and the remaining is for Reynolds number and shape classifier. We employ a general MLP architecture that consists of five hidden layers with a ReLU activation function between every layer and dropout layers with a probability of 0.2 after the first layer and the second layer. Each layer has $(n \times 2) + 1$ nodes more than the previous layer for the first half (scaling) and $\frac{(n-1)}{2}$ nodes for the second half (descaling) of the model.

3.2.4 Ensemble of MLP Models Split at Reynolds Number

Objects and snowflakes flow differently with different Reynolds Numbers which affects the type of fall of snowflakes. With a low Reynolds Number, the snowflake is in a steady flow regime. But with moderate to high Reynolds number (Reynolds number (Re) > 400), the flow regimes turn unsteady [38]. Fully training a machine learning model on both different types of flow regimes might have negative impacts on the results, as physics acting on the snowflakes is different. Creating an ensemble model of the

two different flow regimes, the MLP models can accurately learn the different flow regimes, and not fight each other by altering weights in the original MLP model. The MLP model was split at a Reynolds Number of 158 (splitting the dataset at 400 causes too little data in the moderate to high Reynolds number group, causing the model to overfit the data), for a relatively even split of 65 data points below Re = 158 and 62 above Re = 158. This split still increases the prediction of the model without overfitting the data. The same architecture as the full MLP model was used for the lower and upper flow regimes.

3.2.5 Training the Machine Learning Models

For all the machine learning models, the Reynolds Number and the KPCA extracted feature(s) were standardized with Min Max Scaler. standardizing input data is a common practice in deep learning model training and can improve the convergence of the model. A custom PyTorch [45] dataset class was created to load the data and change the data type of features to float32. The "id" label for each snowflake was included in the custom dataset. This proved to be necessary for combining the test datasets with additional data for result comparisons. The data was split into 75% and 25% denoted as the training set and testing set respectively. During training, the batch size was set to 8 with random shuffling of the data. The Adam optimizer [46] with a learning rate starting at $1e-3$ and Equation 3.2.3 was used in the training process of the model. A learning rate scheduler was used with a step size of 100 and gamma of 0.5, thus halving the learning rate every 100 epoch steps. These configurations are also applied to the ensemble MLP model. The dataset was split

at Re = 158, each loaded with its own custom dataset and data loader. The models were trained sequentially, each using a unique optimizer with the same setting as training the MLP model. A custom class was built that reads in Reynolds number of the input and predicts with the correct model, being sent to the lower model if Re $\leq$ 158 or upper model if Re $>$ 158.

3.3 Overview of Results

3.3.1 Results from Machine Learning Testing

Every model discussed in the Methodology section was evaluated on the Locatelli and Hobbs [2] dataset. First testing was done on the random forest algorithm. A baseline model was run and evaluated with cross-validation and a random search for hyperparameter optimization on the Reynolds number and shape class. Shown in Table 3.1, this model gave a baseline test Mean Squared Error (MSE) of 0.236. It was found with the models ran with one and three features extracted from the images via KPCA, gave slight improvements on test MSE with our best random forest model performing with a test MSE of 0.230 on a single feature extracted from the images. From the random forest models experiment, it concluded that the smaller number of features extracted, the more accurate our prediction model became. Our random forest models outperform the baseline model, showing that KPCA feature extraction from the Locatelli and Hobbs [2] images is improving the model.

We saw a similar trend in the XGBoosterRegressor models. As shown in Table 3.2, the baseline model was run and evaluated on Reynolds Number and shape and gave

a test MSE of 0.232. Running models with one, three, and five feature extraction, the model with three features extracted outperformed the models with one and five feature extraction, with a test MSE of 0.209. The models with one and five features extraction did not outperform the control model.

Moving to the most complex model that was tested and evaluated, the multi-layer perceptron. Shown in Table 3.3, a baseline model was made with just Reynolds Number and shapes, which was the worst model performance out of all the baselines tested and evaluated, with a test MSE of 0.591. Our results found that the more features we had, the better the model did at prediction, the opposite of our other models. Random Forest performed best with one feature extraction, XGBoostRegressor performed best with three feature extraction, and the MLP model performed best with five features extracted. Our best model performance resulted in a test MSE of 0.320. The MLP model was not outperforming the random forest and XGBoost models. The split MLP model at Reynolds number=158 started to outperform all the models tested. Shown in Table 3.4, a baseline model was performed to get a test MSE of 0.625. As we added the extracted features, the model started to outperform the other methods. The best test MSE achieved was 0.195 with 5 features extracted. This model outperformed the rest of the models and tests, becoming our best model. Figure 3.3 shows the prediction from the best model compared to the actual drag in the testing data. The blue line is $Y = x$, thus if a point is closer to the blue line, the more accurate the prediction is. Figure 3.4 shows the error plot for Figure 3.3, showing the squared difference between the real drag and the predicted drag. The closer a point is to line $Y = 0$, the more accurate the data point is. The split MLP model proved to outperform the single MLP model, showing that the ensemble

method benefited from learning in distinct flow regimes rather than one model trying to capture the nature of two different flow regimes.

Using our best model, being the MLP ensemble model with five features extracted from snowflake images, our best MSE was achieved of 0.195. To check if our model is outperforming empirical correlations in literature [3–7], each empirical correlation was calculated with the Locatelli and Hoobs [2] dateset. For Heymsfield and Westbrook [3], the Reynolds number and area ratio, defined as the area of the particle projected normal to the flow divided by the area of the circumscribing disc, were used from the test set to calculate the drag coefficient by Equation (2.1), achieved a test MSE of 0.269, failing to beat our model. See Figure 3.5 for direct comparison between correlation and model. The next correlations tested, Hölzer and Sommerfeld [4], Ganser [5,6], and Haider and Levenspiel [7], all required sphericity, cross-wise sphericity, and length-wise sphericity to calculate. A custom physics-based neural network was made to extract these features from the image. Running this model on the Locatelli and Hobbs [2] images gave the sphericity variables needed. Using these, Hölzer and Sommerfeld [4] correlation, Equation (2.2), achieved a test MSE of 0.221. See Figure 3.6. Next, Ganser [5] and the Ganser with Leith shape factor [6], Equation (2.3) and Equation (2.4), respectfully, achieved a test MSE of 0.282 and 0.286. See Figure 3.7 and 3.8, respectfully. Lastly, Haider and Levenspiel [7], Equation (2.5) achieved a test MSE 0f 0.220, the best that we have seen from the drag correlations. See Figure 3.9. Table 3.5. summarizes the comparisons of our model and the empirical correlations.

Table 3.1: Regression Results on Drag Coefficient with Random Forest Cross Validation of 3 and Random Search Hyperparameter Optimization

# of Features	Test MSE
0 - Baseline	0.236
1	**0.230**
3	0.231
5	0.242

Table 3.2: Regression Results on Drag Coefficient with XGBoostRegressor KFold of 5 with Grid Search Hyperparameter Optimization

# of Features	Test MSE
0 - Baseline	0.232
1	0.234
3	**0.209**
5	0.244

Table 3.3: Regression Results on Drag Coefficient with MLP Models

# of Features	Test MSE
0 - Baseline	0.591
1	0.399
3	0.327
5	**0.320**

Table 3.4: Regression Results on Drag Coefficient with Ensemble MLP Models split at Reynolds Number 158

# of Features	Test MSE
0 - Baseline	0.625
1	0.212
3	0.211
5	**0.195**

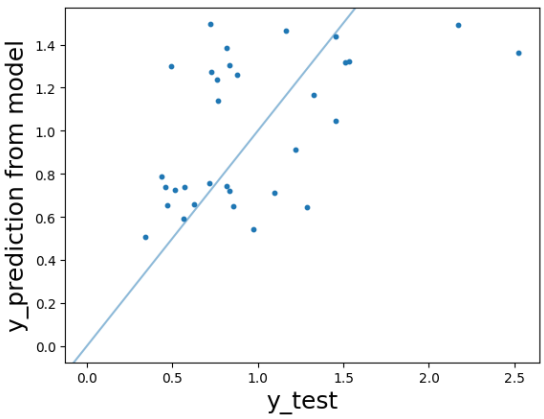

Figure 3.3: Locatelli and Hobbs [2] Dataset Drag Coefficient Prediction vs Actual Drag Coefficient

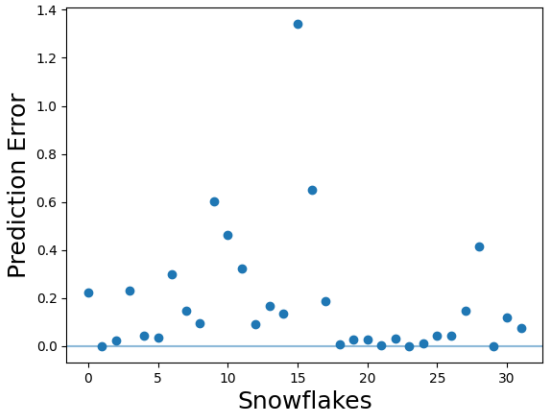

Figure 3.4: Error Plot for the Locatelli and Hobbs [2] Dataset Drag Coefficient Prediction vs Acutal Drag Coefficient

3.3.2 Comparison to Empirical Correlations

Using our best model, being the MLP ensemble model with five features extracted from snowflake images, our best MSE was achieved of 0.195. To check if our model is outperforming empirical correlations in literature [3–7], each empirical correlation was calculated with the Locatelli and Hoobs [2] date set. For Heymsfield and Westbrook [3], the Reynolds number and area were used from the test set to calculate the drag coefficient by Equation (2.1), achieved a test MSE of 0.269, failing to beat our model. See Figure 3.5 for direct comparison between correlation and model. The next correlations tested, Hölzer and Sommerfeld [4], Ganser [5, 6], and Haider and Levenspiel [7], all required sphericity, cross-wise sphericity, and length-wise sphericity to calculate. A custom physics-based neural network was made to extract these features from the image. Running this model on the Locatelli and Hobbs [2] images gave the sphericity variables needed. Using these, Hölzer and Sommerfeld [4] correlation, Equation (2.2), achieved a test MSE of 0.221. See Figure 3.6. Next,

Table 3.5: Drag Coefficient Prediction Comparison versus empirical correlations

Model/Correlation	Test MSE
Our Model (Split MLP)	**0.195**
Heymsfield and Westbrook (2010)	0.269
Hölzer and Sommerfeld (2008)	0.221
Ganser (1993)	0.282
Ganser (1993) w/ Leith Shape Factor	0.286
Haider and Levenspiel (1988)	0.220

Ganser [5] and the Ganser with Leith shape factor [6], Equation (2.3) and Equation (2.4), respectfully, achieved a test MSE of 0.282 and 0.286. See Figure 3.7 and 3.8, respectfully. Lastly, Haider and Levenspiel [7], Equation (2.5) achieved a test MSE 0f 0.220, the best that we have seen from the drag correlations. See Figure 3.9. Table 3.5. summarizes the comparisons of our model and the empirical correlations.

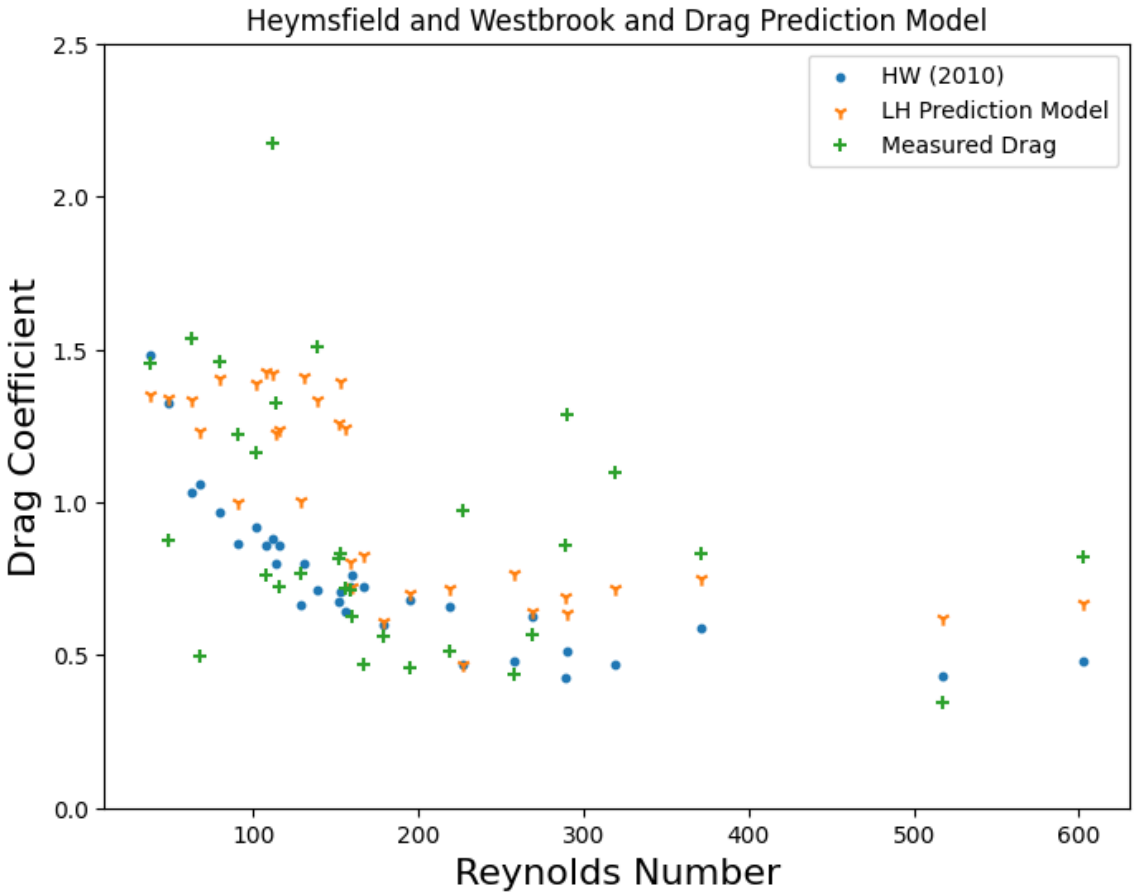

Figure 3.5: Heymsfield and Westbrook [3] Drag Coefficient estimation compared to our models drag prediction with the real drag coefficient present.

3.3.3 Comparison on Mass Prediction

Mass and drag coefficient can be converted back and forth, with Eq. 3.3, as we did with Locatelli and Hobbs [2] mass recordings to obtain the drag coefficient data. Our drag coefficient predictions can be converted back into the mass by the following equation.

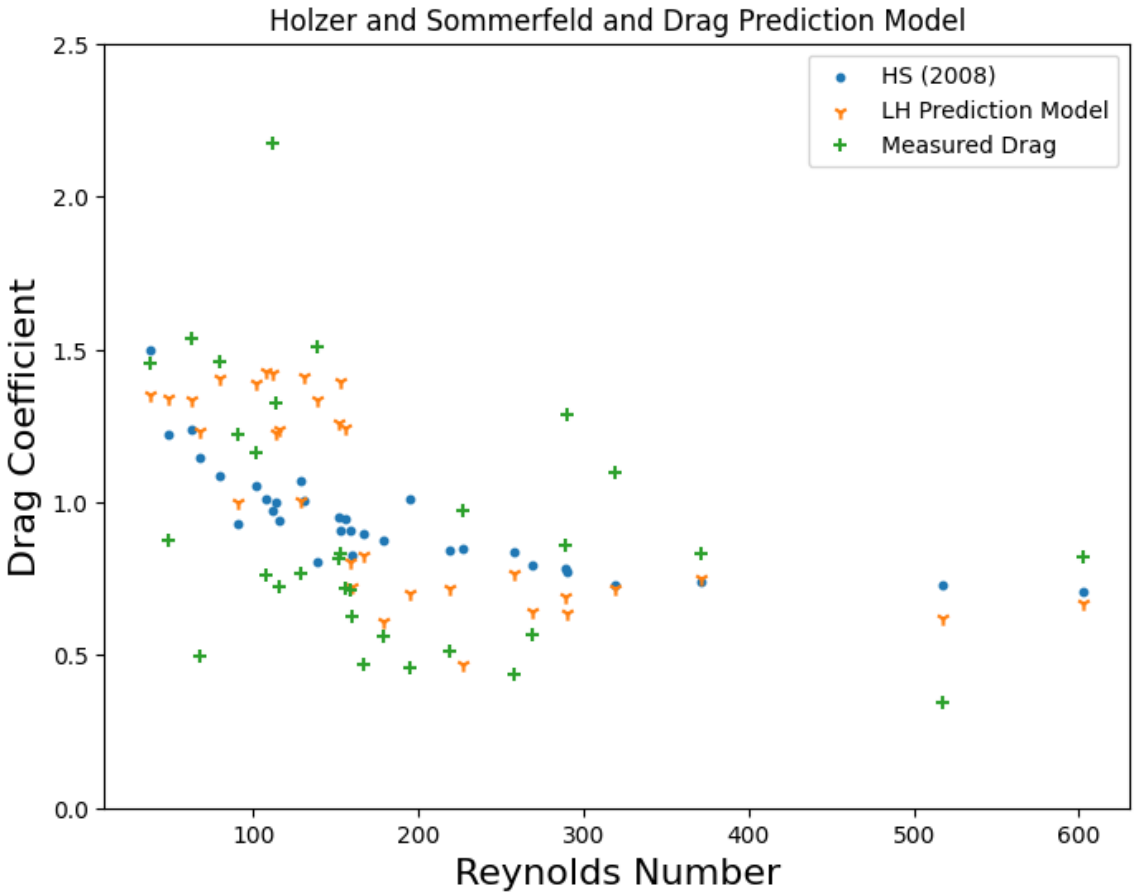

Figure 3.6: Hölzer and Sommerfeld [4] Drag Coefficient estimation compared to our models drag prediction with the real drag coefficient present.

$$\text{Mass} = \frac{C_D}{\text{Acceleration Due to Gravity}} \times 0.5 \times \text{Air Density} \times \text{Velocity}^2 \times \text{Area} \quad (3.3)$$

Equation 3.3 allows us to do a mass estimation comparison with the 3DGAN model made by Leinonen et al [8]. Unfortunately, given that their model performs on 3D data, a direct comparison of their model on the Locatelli and Hobbs [2] is not possible. In Leinonen et al [8] they present their mass estimations against 3D printed

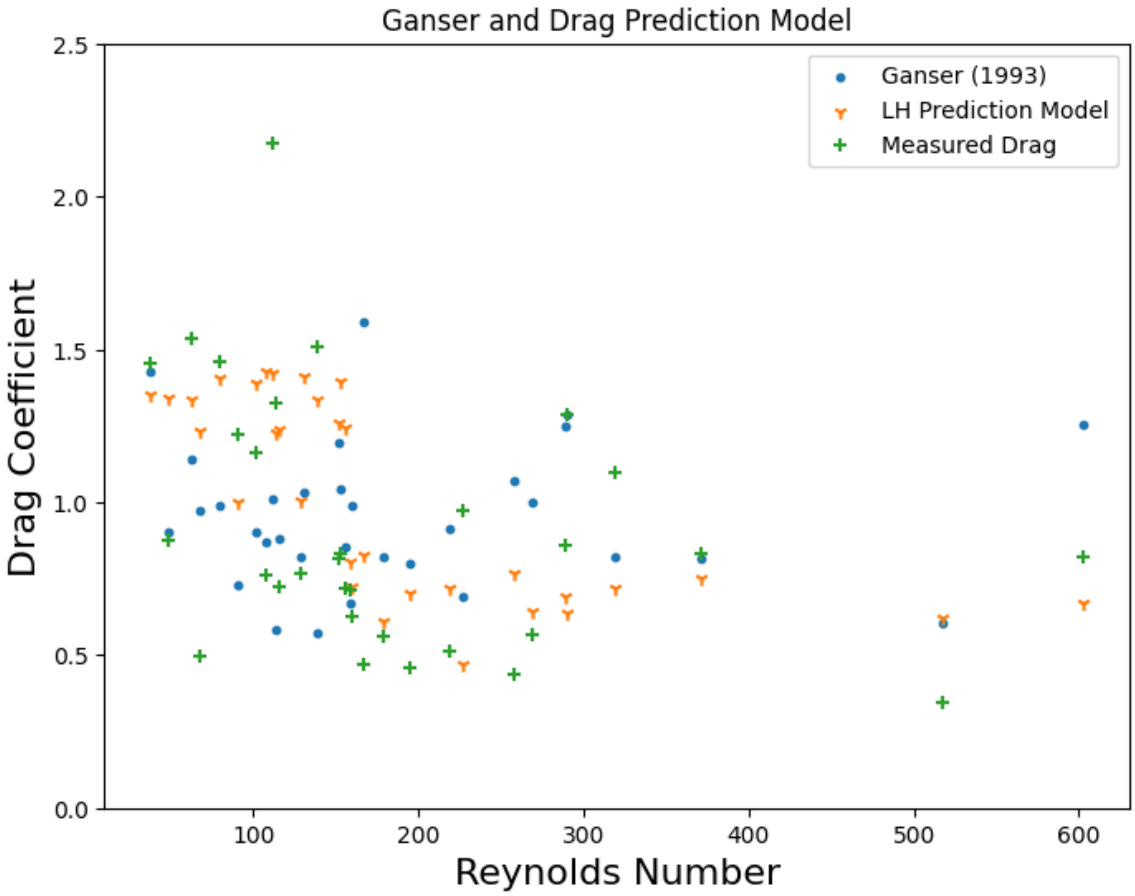

Figure 3.7: Ganser [5] Drag Coefficient estimation compared to our models drag prediction with the real drag coefficient present.

snowflakes giving a test NRMSE (normalized root mean squared error) of 42.000. With our model, we achieved a mass estimation test NRMSE of 46.714. See Figure 3.10 and Figure 3.11 for a comparison of predicted mass compared to reference mass and Reynolds number, respectfully.

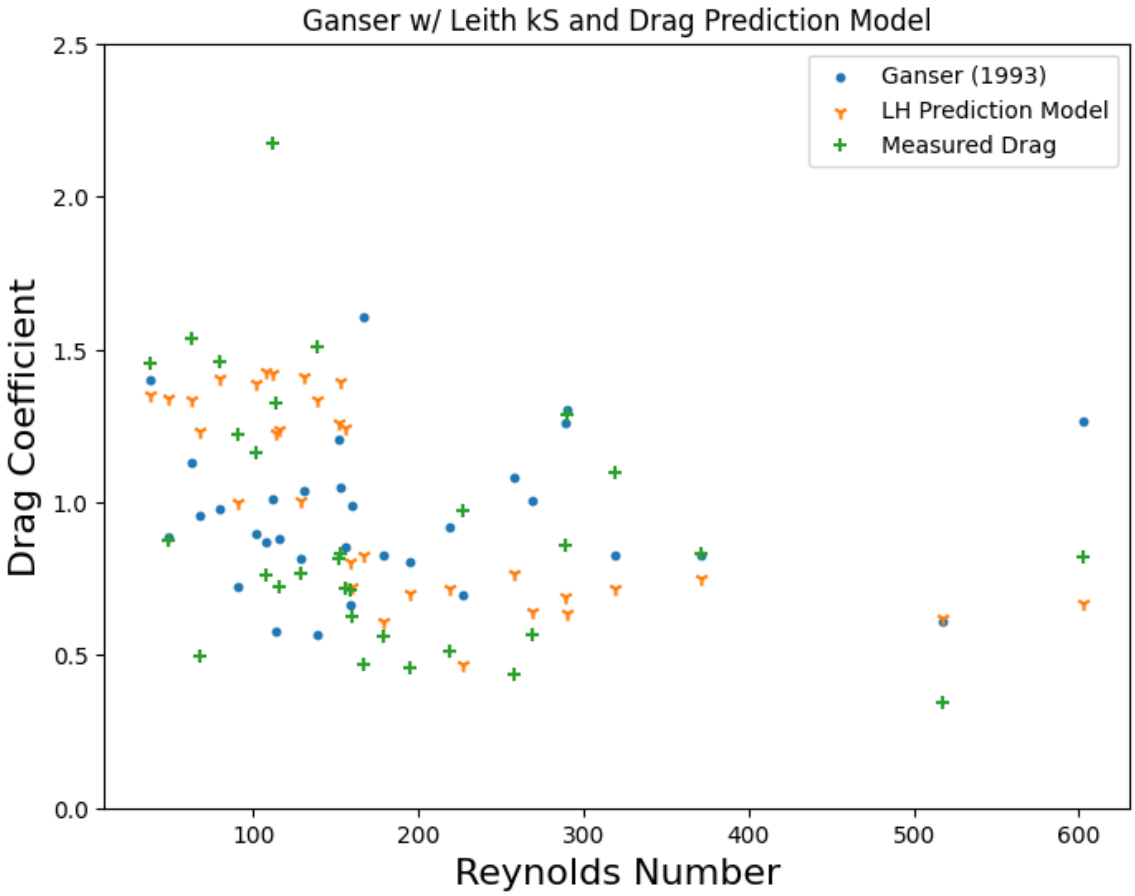

Figure 3.8: Ganser [5] with Leith shape factor [6] Drag Coefficient estimation compared to our models drag prediction with the real drag coefficient present.

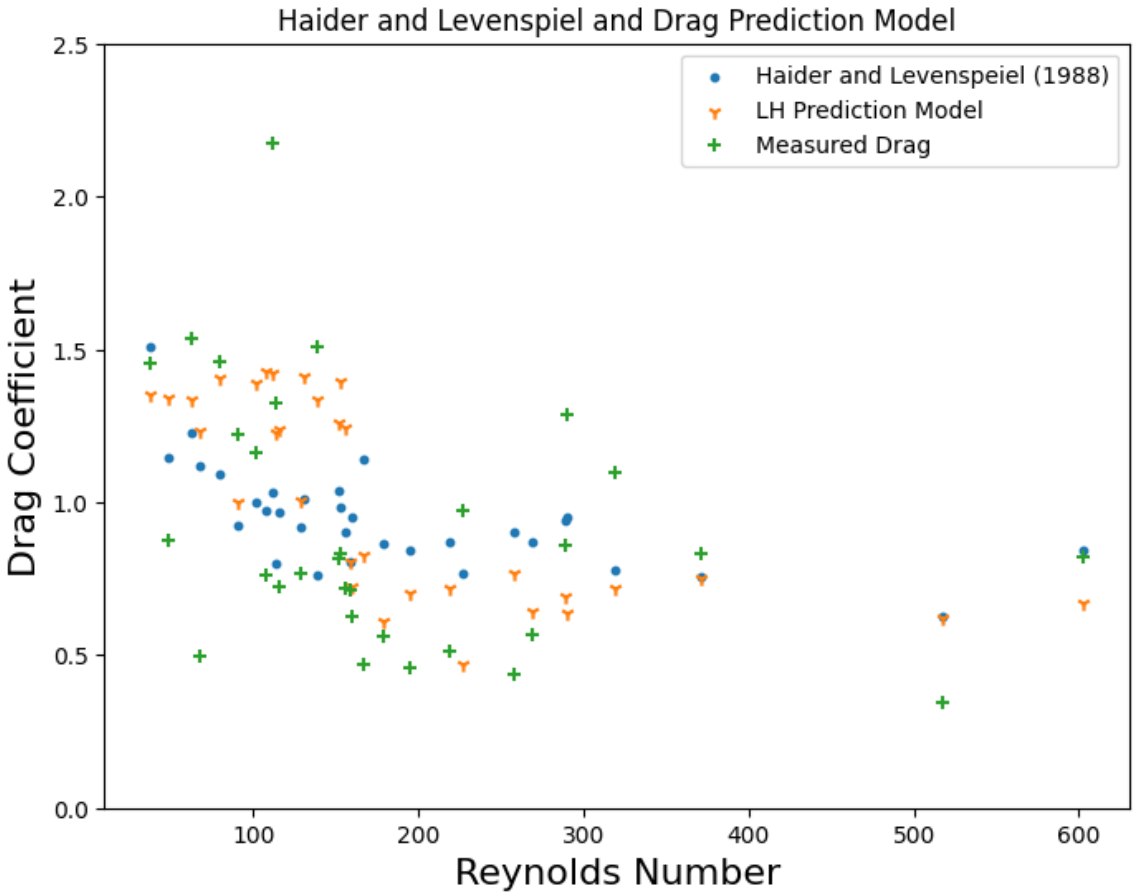

Figure 3.9: Haider and Levenspiel [7] Drag Coefficient estimation compared to our models drag prediction with the real drag coefficient present.

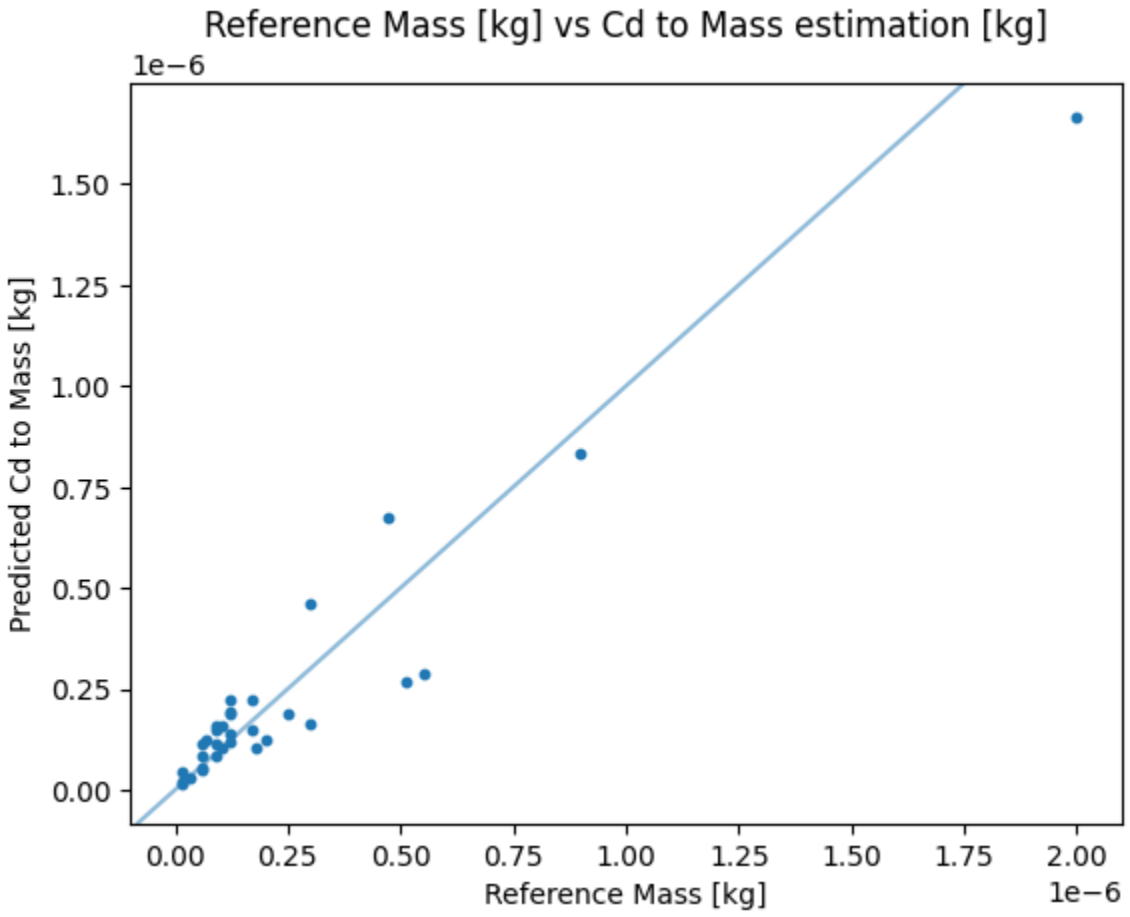

Figure 3.10: Locatelli and Hobbs [2] Drag coefficient estimation converted to mass and compared against mass recorded by Locatelli and Hobbs.

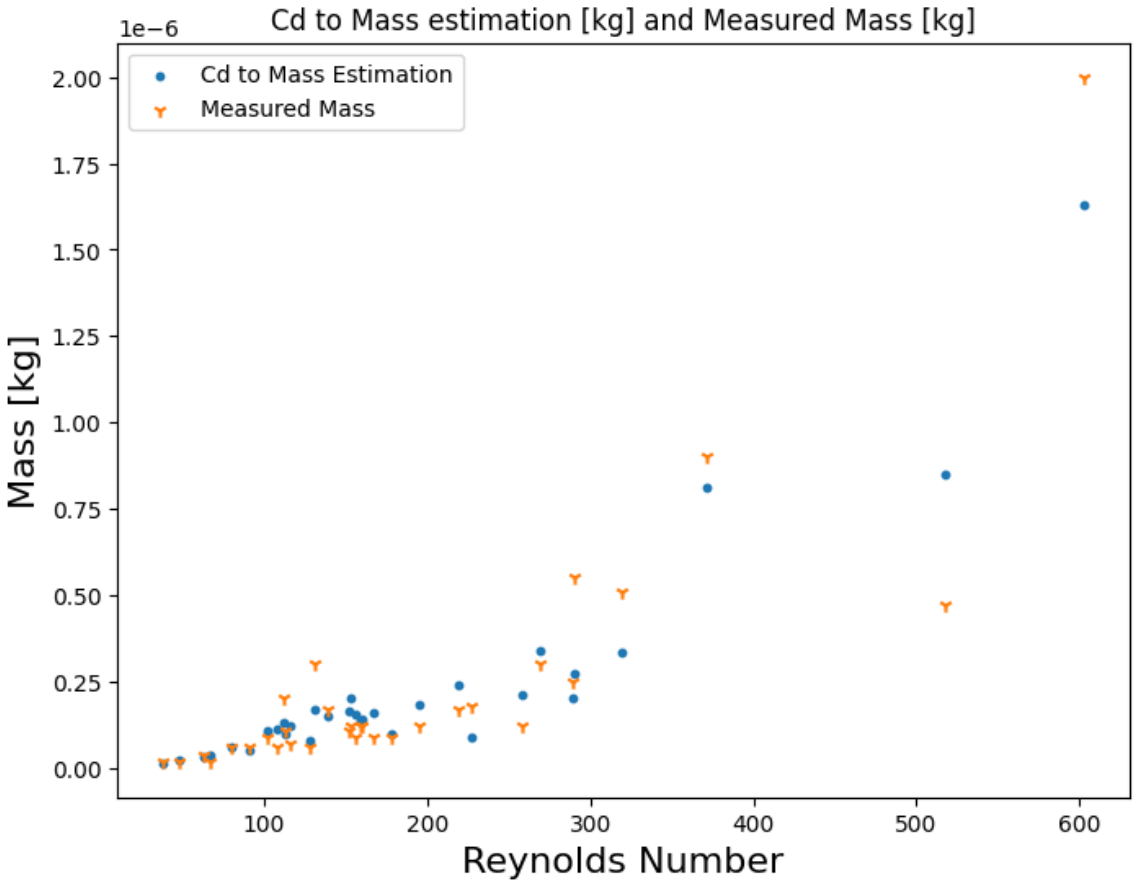

Figure 3.11: Locatelli and Hobbs [2] drag coefficient to mass and mass recorded by Locatelli and Hobbs against Reynolds Number

Chapter 4

Discussion of Results and Methods

It was found that combining EfficientNetB7 [9] with KPCA feature reduction [17] on 2D snowflake images from Locatelli and Hobbs [2] to obtain extracted features was an effective and accurate method of obtaining the drag coefficient of snowflakes. This method is cost-effective and quick, making it very easy to format data and use it with machine learning models. Specifically, EfficientNetB7 [9] proved to be a very useful tool for teams to use to remove the background on their own snowflake data.

4.1 Empirical Correlations

In Figures 3.5-3.9, we show our models predictions against various empirical correlations and the actual drag coefficient calculated with data from Locatelli and Hobbs [2]. As shown in Figure 3.5, Heymsfield and Westbrook [3] correlation is compared against the model prediction and the actual drag coefficient. Heymsfield and Westbrook [3] follows the curve where the drag coefficient decreases as Reynolds number increase, with little spread on that curve. The actual drag data, represented as green crosses, don't follow tightly along this curve. Most of the correlation estimation falls in between drag coefficient values 0.5 to 1.0, only predicting two values above 1.0, thus failing to capture a section of the testing data. The real drag is much more scattered, but generally follows the same curve, with the drag coefficient decreasing as the Reynolds number increases. Our model follows the same curve but has the scattered nature that the real-world drag data captures. Moving to Hölzer and Sommerfeld [4] correlation in Figure 3.6, their estimations of drag follow a similar curve, where the drag coefficient decreases as the Reynolds number increases, but does not predict well for drag coefficient values below 1.0. The curve is more shallow than the Heymsfield

and Westbrook [3] curve, that predicts drag coefficient values below 1.0 and predicts correctly the smaller drag coefficient values around 0.5. The correlation predicts values mostly between 0.8 to 1.5, a larger range than the Heymsfield and Westbrook [3] correlation. More accurate drag coefficient values exist in this range, giving Hölzer and Sommerfeld [4] a slightly better MSE value than Heymsfield and Westbrook [3]. Next, the Ganser [5] correlation in Figure 3.7 does have the scattered nature, not noticeably following a curve, unlike Heymsfield and Westbrook [3] and Hölzer and Sommerfeld [4]. This nature did end up negatively impacting the Ganser [5] correlation's MSE. This is because the scattered trend is mostly between drag values of 0.6 to 1.0. Failing to capture larger and lower drag coefficient values. We see a very similar result with Ganser with the Leith shape factor correction [6]. The only difference between the two correlations is a small change in k_S calculation, using the square root of length-wise sphericity multiplied by the first term in the shape factor calculation. As seen in the results in Table 3.5 and comparing Figure 3.7 and Figure 3.8, the shape factor correction did not to the estimation quality of the correlation, in fact, it made the MSE slightly larger than without the shape correction. Lastly, the Haider and Levenspiel [7] correlation in Figure 3.9, performed the best out of all the empirical correlations. Figure 3.9 follows a similar downward trend curve that we have seen in Figure 3.5 and Figure 3.6, where the drag coefficient decreases with the increase in Reynolds number. We see a more loose fit to this curve with moderate scatter along the curve, with most of the drag coefficient values falling between the range 0.8 to 1.5, which results in the Haider and Levenspiel [7] correlation obtaining the best test MSE among the empirical correlations.

Looking more specifically at our model prediction versus the real drag coefficient

data, by looking at any Figure 3.5-3.9, we see that our model does follow a general trend that the empirical correlations follow. As the drag coefficient decreases, the Reynolds number increases. The model has some scatter along this curve. The model is struggling in the prediction of snowflakes with lower Reynolds number and lower drag coefficient ($C_D < 1$). With the model being split at Re = 158, this can be accounted for by the lower Reynolds number regime model. There is some crowding between 1.0 to 1.5 that occurs in the lower regime model. The upper regime model is performing very well, capturing the spread of the real drag coefficient. Both models would perform better given there was more data from the Locatelli and Hobbs [2] available, but unfortunately there is not.

4.2 Mass Estimation

We also experimented with converting the drag coefficient to mass. Compared with the reference mass recorded by Locatelli and Hobbs [2], we achieved a MSE of $1.218 * 10^{-14}$ and NRMSE of 46.714, illustrated in Figure 3.10 and 3.11. Unfortunately, Leinonen et al [8] predicted mass using the multi-angled snowflake camera, capturing three images of the falling snowflake at three different angles on artificial snowflakes. Because of this, we are unable to test our results directly against their 3DGAN. Therefore, comparisons against NRMSE from two different datasets, both predicting mass are the closest we can get to comparing the two models. The 3DGAN mass estimation on artificial snowflakes obtained an NRSME of 42. There is only a difference of 4 between the 3DGAN and our model, showing that our models are very similar in predicting mass, but with our model having a much simpler architecture

over the 3DGAN [8], which uses three 2D views of the same snowflake and convolutions to obtain mass, while our model uses one 2D image of the snowflake with Reynolds number and shape class. Leinonen et al [8] presented their model against other models predicting mass, including Matrosov et al [28] and Baker and Lawson [29]. Both articles and methods were applied to individual 2D MASC images of artificially created snowflakes, calculating mass for each 2D image and then averaging the three image views to get the mass. Matrosov et al [28] obtained a NRMSE of 103 and Baker and Lawson [29] obtained a NRMSE of 68. Our model outperforms both methods that use single 2D images to extract mass, showing that our method, to our knowledge, is the most accurate method of extracting the mass from snowflakes using single-view 2D snowflake images. However because they validated their results on artificially created snowflakes and not real snowflakes, the comparison should be taken with some caution.

4.3 Testing on MASCDB and Transfer Learning

In machine learning, the more data you have, the better your model will be. Using Locatelli and Hobbs [2], we were limited to only 128 data points. Snowflakes outside of the fall speed range described by Fitch et al 2021 [36] were removed from the database, but snowflakes in the fall speed range still have a small inherent error from the presence of wind.

The same model architecture was used and applied to the MASCDB dataset. 1372 data points from MASCDB were pulled from the MASCDB and were trained similarly

to the Locatelli and Hobbs [2] dataset, but without a learning rate scheduler, which should not greatly effect the results given. In Figure 4.1, we notice two lines with a constant slope of 0. The lower and upper Reynolds number regime models both got stuck, resulting in both predicting the same value, of around 8 and 4. There was a great effort to optimize the MASCDB model, with just a single MLP model, and the ensemble of two MLP models, but there was always an issue with the models being stuck or overfitting the data. This was another reason why we used the Locatelli and Hobbs [2] over the MASCDB [1]. The best model trained on Locatelli and Hobbs [2] was applied and tested on MASCDB data [1], resulting in a test MSE of 47.490. Figures 4.2 and 4.3 show the prediction compared to the actual drag and the error produced, respectively. The results are very scattered with a lot of error present. Looking at Figure 4.4, it shows the actual drag coefficient compared to the predicted drag on a graph comparing the drag coefficient and Reynolds number. It is clear in this graph that the model does not accurately capture the nature of the MASCDB [1], thus resulting in a lot of error. The data is very scattered, following no obvious trend as we see in Figures 3.5-3.9, with the Locatelli and Hobbs [2]. Until we zoom out in Figure **??** and see that there is a similar trend to Figure 2.3, but with a lot more scatter, which could be accounted for by the 3DGAN [8] only validating their model on artificial snowflake data. Our model was tested and validated on real snowflake data and images, making our results on MASCDB more plausible than the drag coefficient data converted from mass estimations with 3DGAN [8]. It is important to note that the MASCDB [1] has a greater range of drag coefficient and Reynolds number. Locatelli and Hobbs [2] maximum drag coefficient and Reynolds number is 2.5 and 600, respectively. While MASCDB [1] has a maximum drag coefficient and Reynolds

number of 40 and 1000, respectively. This does make the problem of drag coefficient prediction much harder given the larger range of values, this could account for some of the challenges encountered in the training and testing process with MASCDB [1]. Given the issues with 3DGAN mass estimation and fall velocity measurements, which both in turn effect our transformation from mass of the snowflake to drag coefficient, it was concluded that using the MASCDB with mass estimations from [8] Leinonen et al. (2021) for training and validation data would introduce a lot of noise and error into our model. This was also found to be true when running our model with training and validation data from MASCDB seen in Figure 4.1. Transfer learning also did not show to be successful as shown in Figures 4.2, 4.3, and 4.4.

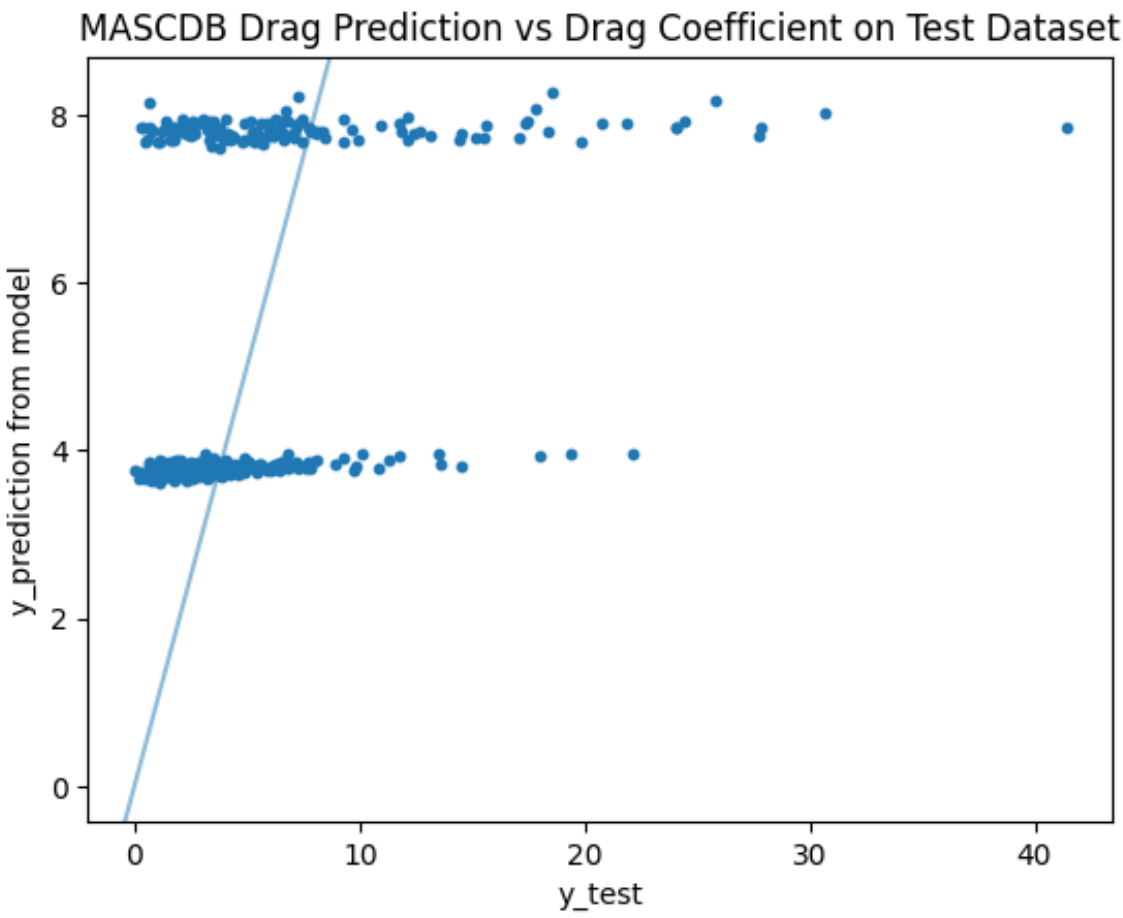

Figure 4.1: MASCDB [1] Drag Coefficient estimation compared to the 3DGAN [8] predictions of mass converted to drag coefficient.

4.4 Feature Importance

A feature importance analysis can be done with the extracted KPCA features with Reynolds number and shape class via random forests by using random forest mean decrease impurity (MDI) [19, 20]. In random forest models, all the trees are created at once, while constructing them, the mean decrease in impurity is found for each feature in each tree. The calculation for the decrease in impurity is done by calcu-

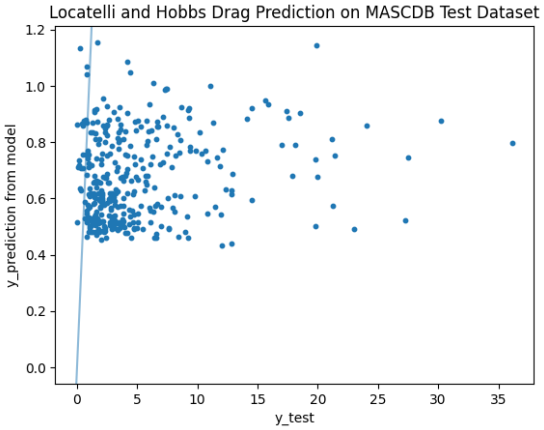

Figure 4.2: Locatelli and Hobbs [2] Model Transfer Learning on MASCDB [1]

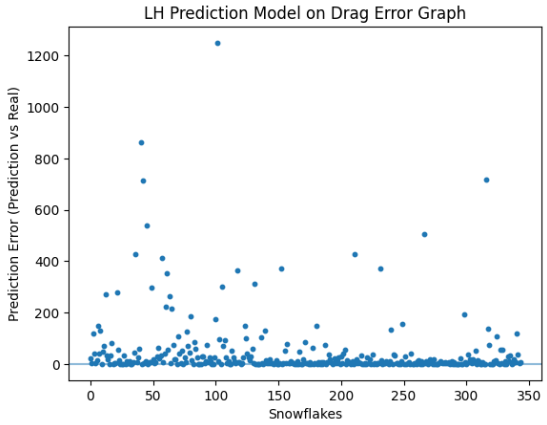

Figure 4.3: Error Plot of Transfer Learning on MASCDB [1]

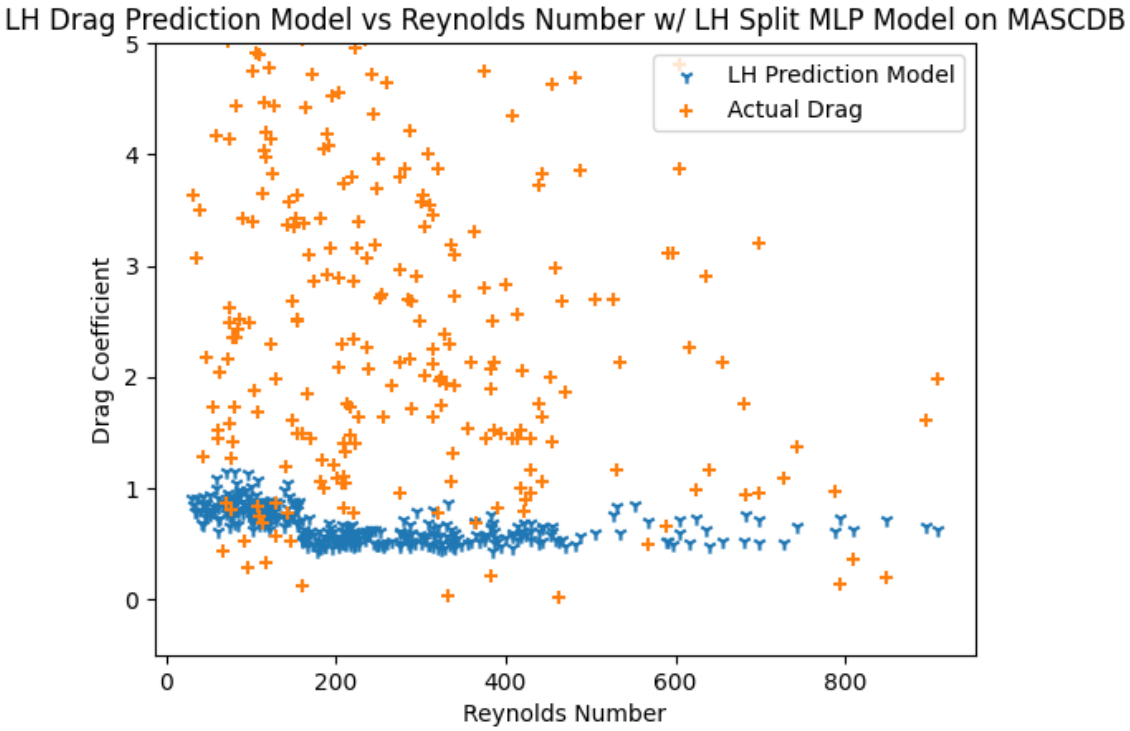

Figure 4.4: Locatelli and Hobbs [2] prediction model tested on MASCDB data.

lating Gini Impurity [19, 20], which counts each time that feature was used to split the tree. We can average the mean decrease in impurity for each feature in all the

trees, just as the random forest does for its final regression prediction, to get the average feature importance of each feature. Figure 4.5 shows this ranking. Specifically, the Reynolds number is the most important feature, followed by KPCA1, the first extracted feature from the five extracted features, followed by KPCA4, KPCA3, KPCA5, shapes, and KPCA2. This ranking does make sense as the Reynolds number is extremely important in the prediction of drag coefficient, as we see in Eq. 2.1-2.5, all use the Reynolds number. KPCA1 being the 2nd most important rank also does make sense, since in the KPCA feature extraction, the first feature extracted is the most important feature. It was expected that KPCA5 would be the lowest rank as it is the least important feature extracted from the KPCA feature extraction, but we find that KPCA2 is the lowest ranked importance. Shape classification also did not receive a high rank, being 6th ranked for importance, behind KPCA2. This could be because the KPCA feature extraction and MLP model are learning the shape of the snowflake, making the actual shape classification a shallow feature, not giving as much information as KPCA 1-5, thus making it unimportant.

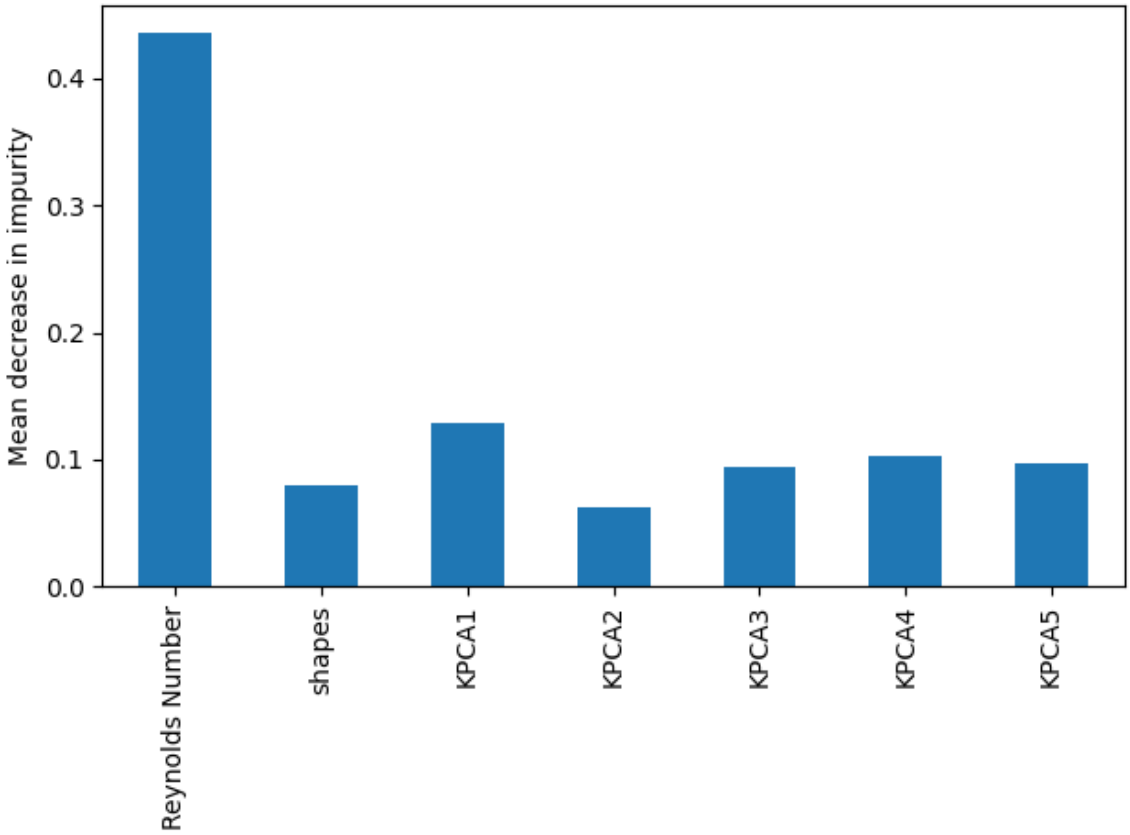

Figure 4.5: Feature importance graph of five extracted features from KPCA, with Reynolds number and Shape class, using random forest Mean Decrease Impurity (MDI)

Chapter 5

Conclusions

5.1 Summary

To address the challenges with drag coefficient prediction with 2D images, we propose a method to remove the background and border of 2D images with EfficientNetB7 [9], use KPCA [17] to extract features from the images, to be used with the Reynolds number and shape class on an ensemble method of two MLP models, each trained on different Reynolds number flow regimes to predict the drag coefficient on the Locatelli and Hobbs dataset [2]. This method is cost-effective and time-efficient as it uses purely tabular data. Researchers can capture single-view 2D images of falling snowflakes, where Reynolds number and shape class can be determined from visual analysis of each snowflakes, to accurately predict the drag coefficient, avoiding expensive equipment and methods to produce 3D images and calculate different sphericity features. Our method was compared directly against the following empirical correlations: (2.1) Heymsfield and Westbrook (2010) [3], (2.2) Hölzer and Sommerfeld (2008) [4], (2.3) Ganser (1993) [5] and (2.4) with Leith shape factor correction [6], and (2.5) Haider and Levenspiel (1988) [7] tested on the Locatelli and Hobbs dataset [2]. It was found that our model outperformed all empirical correlations presented, with a test MSE of 0.195, with the best empirical correlation being (2.5) Haider and Levenspiel (1988) [7] with a test MSE of 0.220. It was shown that the KPCA extracted features added valuable information to our model from feature importance analysis using mean decrease impurity (MDI), and is recommended for analysis of feature importance in future work.

5.2 Future Work

The architecture used in the MLP model is not deep, with only five hidden layers, this can be accounted for by the lack of data, with only 128 data points. Presently, deeper architecture has not been explored, past nine hidden layers, as the model starts to overfit. In the future, with more high-quality 2D image data being captured, this could increase the accuracy of the model by a significant amount. With more data, deeper architectures should be explored to improve the benchmarks set by our ensemble MLP method.

Bibliography

[1] Jacopo Grazioli, Gionata Ghiggi, Anne-Claire Billault-Roux, and Alexis Berne. Mascdb, a database of images, descriptors and microphysical properties of individual snowflakes in free fall. *Scientific Data*, 9(1):186, 2022.

[2] John D. Locatelli and Peter V. Hobbs. Fall speeds and masses of solid precipitation particles. *Journal of Geophysical Research (1896-1977)*, 79(15):2185–2197, 1974.

[3] A. J. Heymsfield and C. D. Westbrook. Advances in the estimation of ice particle fall speeds using laboratory and field measurements. *Journal of the Atmospheric Sciences*, 67(8):2469 – 2482, 2010.

[4] Andreas HĂślzer and Martin Sommerfeld. New simple correlation formula for the drag coefficient of non-spherical particles. *Powder Technology*, 184(3):361–365, 2008.

[5] Gary H. Ganser. A rational approach to drag prediction of spherical and nonspherical particles. *Powder Technology*, 77(2):143–152, 1993.

[6] David Leith. Drag on nonspherical objects. *Aerosol Science and Technology*, 6(2):153–161, 1987.

[7] A. Haider and O. Levenspiel. Drag coefficient and terminal velocity of spherical and nonspherical particles. *Powder Technology*, 58(1):63–70, 1989.

[8] J. Leinonen, J. Grazioli, and A. Berne. Reconstruction of the mass and geometry of snowfall particles from multi-angle snowflake camera (masc) images. *Atmospheric Measurement Techniques*, 14(10):6851–6866, 2021.

[9] Mingxing Tan and Quoc V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks. *International Conference on Machine Learning 36*, 2019.

[10] Tin Kam Ho. Random decision forests. 1:278–282, 1995.

[11] Tianqi Chen and Carlos Guestrin. XGBoost: A scalable tree boosting system. pages 785–794, 2016.

[12] Simon Haykin. *Neural networks: a comprehensive foundation*. Prentice Hall PTR, 1994.

[13] J. Leinonen and A. Berne. Unsupervised classification of snowflake images using a generative adversarial network and k-medoids classification. *Atmospheric Measurement Techniques*, 13(6):2949–2964, 2020.

[14] Kilian Köbschall, Jan Breitenbach, Ilia V Roisman, Cameron Tropea, and Jeanette Hussong. Geometric descriptors for the prediction of snowflake drag. *Experiments in Fluids*, 64(1):4, 2023.

[15] J. Leinonen, D. Moisseev, and T. Nousiainen. Linking snowflake microstructure to multi-frequency radar observations. *Journal of Geophysical Research: Atmospheres*, 118(8):3259–3270, 2013.

[16] Karl Pearson F.R.S. Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11):559–572, 1901.

[17] Bernhard Schölkopf, Alexander Smola, and Klaus-Robert Müller. Kernel principal component analysis. pages 583–588, 1997.

[18] Choji Magono and Chung Woo Lee. Meteorological classification of natural snow crystals. *Journal of the Faculty of Science, Hokkaido University. Series 7, Geophysics*, 2(4):321–335, 1966.

[19] Leo Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

[20] Leo Breiman. *University of California Berkeley*, 1, 2002.

[21] Giorgia Tagliavini, Mark McCorquodale, Chris Westbrook, Pascal Corso, Quirine Krol, and Markus Holzner. Drag coefficient prediction of complex-shaped snow particles falling in air beyond the stokes regime. *International Journal of Multiphase Flow*, 140:103652, 2021.

[22] Francisco J. Valdes-Parada, J. Alberto Ochoa-Tapia, and Jose Alvarez-Ramirez. Validity of the permeability carmanâkozeny equation: A volume averaging approach. *Physica A: Statistical Mechanics and its Applications*, 388(6):789–798, 2009.

[23] K. Steiros and M. Hultmark. Drag on flat plates of arbitrary porosity. *Journal of Fluid Mechanics*, 853:R3, 2018.

[24] Jai Prakash and G.P. Raja Sekhar. Dynamic permeability of an assemblage of soft spherical particles. *Mathematical Methods in the Applied Sciences*, 36(16):2174–2186, 2013.

[25] Gholamhossein Bagheri and Costanza Bonadonna. On the drag of freely falling non-spherical particles. *Powder Technology*, 301:526–544, 2016.

[26] F. Dioguardi, D. Mele, and P. Dellino. A new one-equation model of fluid drag for irregularly shaped particles valid over a wide range of reynolds number. *Journal of Geophysical Research: Solid Earth*, 123(1):144–156, 2018.

[27] Sabine Tran-Cong, Michael Gay, and Efstathios E Michaelides. Drag coefficients of irregularly shaped particles. *Powder Technology*, 139(1):21–32, 2004.

[28] Sergey Y. Matrosov, Kurt A. Clark, and David E. Kingsmill. A polarimetric radar approach to identify rain, melting-layer, and snow regions for applying corrections to vertical profiles of reflectivity. *Journal of Applied Meteorology and Climatology*, 46(2):154 – 166, 2007.

[29] Brad Baker and R. Paul Lawson. Improvement in determination of ice water content from two-dimensional particle imagery. part i: Image-to-mass relationships. *Journal of Applied Meteorology and Climatology*, 45(9):1282 – 1290, 2006.

[30] A. Hicks and B. M. NotaroĹĄ. Method for classification of snowflakes based on images by a multi-angle snowflake camera using convolutional neural networks. *Journal of Atmospheric and Oceanic Technology*, 36(12):2267 – 2282, 2019.

[31] C. Praz, Y.-A. Roulet, and A. Berne. Solid hydrometeor classification and riming degree estimation from pictures collected with a multi-angle snowflake camera. *Atmospheric Measurement Techniques*, 10(4):1335–1357, 2017.

[32] M. Schaer, C. Praz, and A. Berne. Identification of blowing snow particles in images from a multi-angle snowflake camera. *The Cryosphere*, 14(1):367–384, 2020.

[33] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. pages 2672–2680, 2014.

[34] Martin Arjovsky, Soumith Chintala, and LĂŠon Bottou. Wasserstein gan. 2017.

[35] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron Courville. Improved training of wasserstein gans. 2017.

[36] Kyle Fitch, Chaoxun Hang, Ahmad Talaei, and Timothy Garrett. Arctic observations and numerical simulations of surface wind effects on multi-angle snowflake camera measurements. *Atmospheric Measurement Techniques*, 14:1127–1142, 02 2021.

[37] A. K. Jain, M. N. Murty, and P. J. Flynn. Data clustering: A review. *ACM Comput. Surv.*, 31(3):264â323, sep 1999.

[38] Giorgia Tagliavini, Majid H. Khan, Mark McCorquodale, Chris Westbrook, and Markus Holzner. Wake characteristics of complex-shaped snow particles: Comparison of numerical simulations with fixed snowflakes to time-resolved particle tracking velocimetry experiments with free-falling analogs. *Physics of Fluids*, 34(5):055112, 05 2022.

[39] Qibin Hou, Ming-Ming Cheng, Xiaowei Hu, Ali Borji, Zhuowen Tu, and Philip H. S. Torr. Deeply supervised salient object detection with short connections. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(4):815–828, apr 2019.

[40] Mark A. Kramer. Nonlinear principal component analysis using autoassociative neural networks. *AIChE Journal*, 37(2):233–243, 1991.

[41] L.J. Cao, K.S. Chua, W.K. Chong, H.P. Lee, and Q.M. Gu. A comparison of pca, kpca and ica for dimensionality reduction in support vector machine. *Neurocomputing*, 55(1):321–336, 2003. Support Vector Machines.

[42] Simon Cramer, Daniel Buschmann, and Robert H. Schmitt. Comparison of feature extraction algorithms for prediction of quality characteristics. *Procedia CIRP*, 112:579–584, 2022. 15th CIRP Conference on Intelligent Computation in ManufacturingEngineering, 14-16 July 2021.

[43] J. Ghosh and A. Nag. *An Overview of Radial Basis Function Networks*, pages 1–36. Physica-Verlag HD, Heidelberg, 2001.

[44] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[45] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.

[46] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. 2017.