Machine learning models for prediction of droplet collision outcomes

Arpit Agarwal
arpit.r.agarwal@gmail.com

Abstract

Predicting the outcome of liquid droplet collisions is an extensively studied phenomenon but the current physics based models for predicting the outcomes are poor (accuracy $\approx 43\%$). The key weakness of these models is their limited complexity. They only account for 3 features while there are many more relevant features that go unaccounted for. This limitation of traditional models can be easily overcome through machine learning modeling of the problem. In an ML setting this problem directly translates to a classification problem with 4 classes. Here we compile a large labelled dataset and tune different ML classifiers over this dataset. We evaluate the accuracy and robustness of the classifiers. ML classifiers, with accuracies over 90%, significantly outperform the physics based models. Another key question we try to answer in this paper is whether existing knowledge of the physics based models can be exploited to boost the accuracy of the ML classifiers. We find that while this knowledge improves the accuracy slightly for small datasets, it is useless with larger datasets being available.

1 Introduction

1.1 Motivation

Droplet collisions are important to the dynamics of a liquid spray. Sprays of engineering relevance have a very high droplet density and therefore a lot of droplet collisions. For example, diesel sprays have been reported to have 10^8 collisions per cm^3 per microsecond [1]. Collisions have a direct effect on the liquid drops as they change the droplet sizes and velocities. Due to the high frequency of droplet collisions, and their cascading effect on the system dynamics, it is important to accurately model the droplet collision phenomena.

The problem of modeling the outcome of droplet collisions has been worked on extensively by fluid dynamicists. The models proposed in literature for predicting the outcomes of droplet collisions are phenomenological. They are based simplified analyses of the governing fluid dynamics equations. Agarwal et al. [2] evaluate the current state of the art in physics based models and report an accuracy of around 64% on the data they considered. On considering additional data reported by Sommerfeld and Kuschel [3] this accuracy drops to around 43%. The complexity of these models is limited by our understanding of fluid mechanics, and our cognitive limitations. Models of higher complexity are needed to improve the accuracy.

The problem of droplet collision prediction can be directly framed as a 4-class classification problem. This problem is ideal for a machine learning solution because of two reasons:

1. Ground truth complexity is high - beyond physical modeling
2. The problem is well studied, hence a lot of experimental data has been published over the years

Preprint. Work in progress.

Here I compile labelled datasets from literature (7898 points in all, see table 1) and train machine learning classification models over this data. The entire dataset, source code for the different models and tests that appear in the paper, and all the figures are available on github [4].

1.2 Fluid Mechanics Background

Four main types of outcomes are expected from collisions of two droplets, these are illustrated in figure 1. As this is a deterministic phenomenon, a priori prediction of collision outcomes is possible with enough information about the droplet and flow characteristics. Due to limitations of the techniques used, in current fluid mechanics literature, only three main non-dimensional quantities, Δ, We and B are considered in the outcome of a binary collision. It is well known that more features are relevant, but no models exist that account for all relevant features.

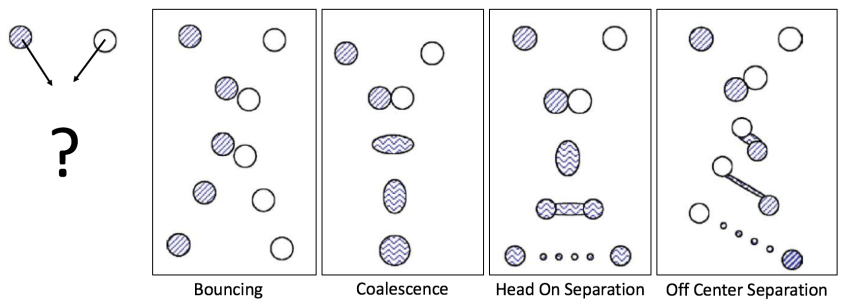
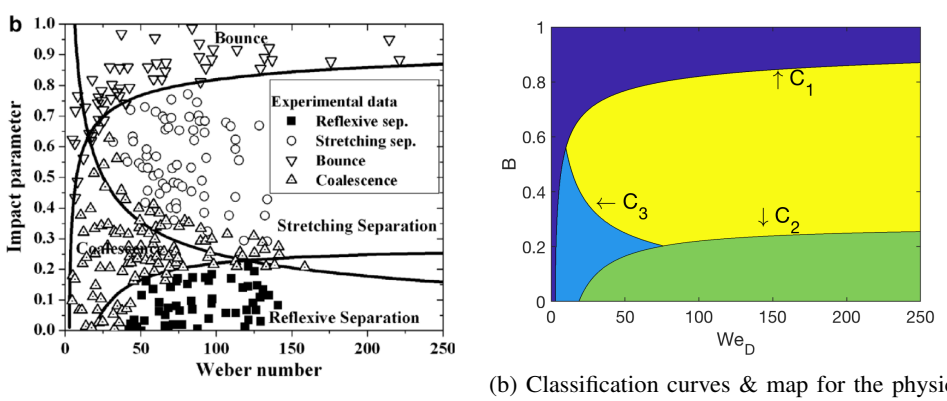


Figure 1: An illustration of the 4 types of outcomes expected. These are the classification labels for the present study.

As highlighed in fig. 1 the four class labels (collision outcomes) are:

1. Coalescence
2. Bouncing
3. Stretching separation (or off-center separation)
4. Reflexive separation (or head-on separation)

As per the traditional modeling approach, the 3-dimensional feature space is divided into 4 regions corresponding to the respective outcomes. Figure 2 shows a slice of this 3-dimensional space (Δ is set to 1), showing the four different outcomes. This figure also gives an example of a typical labelled experimental dataset.



(a) A sample dataset with the 4 labels and the physics based classification curves

(b) Classification curves & map for the physics based models. The four different colors correspond to the four different labels.

Figure 2: A sample dataset by Munnannur and Reitz [5] and the physics based classification map by Agarwal et al. [2].

2 Dataset

Data has been collected from four published articles. Details of the references are provided in Table 1. I manually extracted 7898 data points from the sources listed in Table 1 using a web-based tool [6]. Each data point corresponds to one collision experiment.

Table 1: Source datasets

Date	Source reference	No. of data points	Comments
1990	Ashgriz and Poo [7]	986	Early work - only basic features considered
1999	Estrade et al. [8]	905	
1997	Qian and Law [9]	217	
2016	Sommerfeld and Kuschel [3]	5790	Newer work - emphasis on features that are not represented in traditional models
		Total: 7898	

All the input features are numerical. The 5 base features and 3 additional features are described in table 2. A slice of the overall dataset showing variation along We, and B is presented in Figure 3. Spray droplet collisions are most studied in the context of fuel sprays, and in typical physical operating ranges, We and B are the most relevant features in the outcome of the droplet collision. The distribution of the datapoints along the We and B axis are presented alongside the scatterplot.

Table 2: Descriptions of the features

	Features	Range	Comments
Base Features	1. We (Weber number)	$\mathbb{R}^+$	See fig. 3
	2. B (Impact param.)	[0, 1]	See fig. 3
	3. Δ (Drop size ratio)	[0, 1]	Most data has $\Delta = 1$; a few points with $\Delta = \{0.5, 0.75\}$ (see fig. 4a)
	4. P (Gas pressure)	$\mathbb{R}^+$	Only discrete training data available (see fig. 4b)
	5. μ (Liquid viscosity)	$\mathbb{R}^+$	Only discrete training data available (see fig. 4c)
Additional Features (combinations of base features)	Curve C_1	$\mathbb{R}$	see eq. 1
	Curve C_2	$\mathbb{R}$	see eq. 2
	Curve C_3	$\mathbb{R}$	see eq. 3

2.1 Incorporating domain knowledge

The 3 decision curves that form the physics based models [5] are given below:

$$C_1 = We - \frac{\Delta(1+\Delta^2)(4\Phi - 12)}{\chi_1 \left[\cos\left(\sin^{-1} B\right)\right]} \tag{1}$$

$$C_2 = We - \frac{3[7(1+\Delta^3)^{2/3} - 4(1+\Delta^2)]\Delta(1+\Delta^3)^2}{\Delta^6\eta_1 + \eta_2} \tag{2}$$

$$C_3 : B - \sqrt{\frac{2.4(\Delta^3 - 2.4\Delta^2 + 2.7\Delta)}{We}} \tag{3}$$

These curves divide the feature space or the physical input space into separate regions, and depending on where the datapoint lies with respect to these curves, the physics based models assign the outcome (see Figure 2b). Here we incorporate these 3 decision curves as features into our model. C_1, C_2 and C_3 are the additional features formed through combinations of the base features.

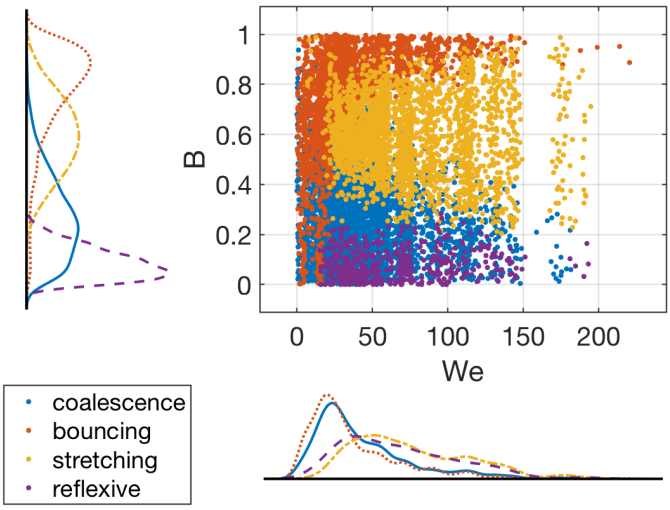
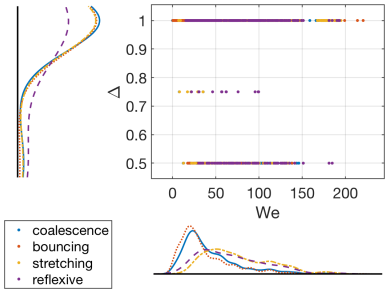
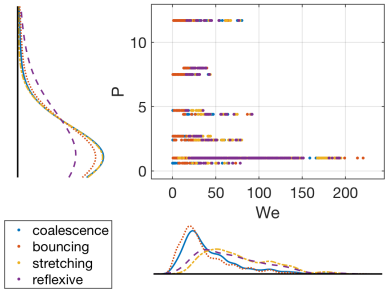


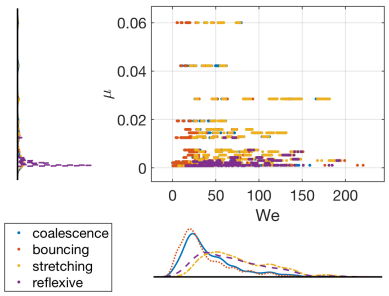
Figure 3: Overall droplet collision dataset (N = 7898, details in Table 1), showing variation in collision outcomes with respect to We and B (features 1 and 2).



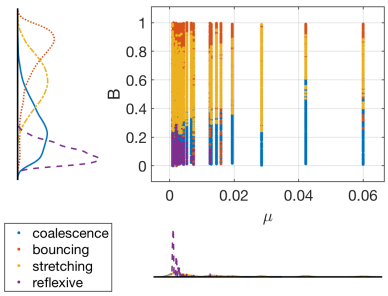
(a) Collision outcomes with respect to Δ and We (features 1 and 3)



(b) Collision outcomes with respect to P and We (features 1 and 4)



(c) Collision outcomes with respect to μ and We (features 1 and 5)



(d) Collision outcomes with respect to B and μ (features 5 and 2)

Figure 4: Overall droplet collision dataset (N = 7898, details in Table 1), showing variation in collision outcomes with respect to the different features listed in Table 2.

2.2 Gaps in the data set

The traditional physics based models only consider 3 features (Δ, We, B), therefore, in reporting the data some authors do not report values for the other two features (P, μ) considered here. To fill the gaps I have made educated guesses (based on the description of the setup and the fluid used) of those

features. Furthermore, Qian and Law [9] do not distinguish between classes 3 and 4, and report them as the same. Here, again, I have made educated guesses for their labels based on where the datapoints lie in the regime map (Figure 2b).

3 Machine Learning Modeling

Scikit-learn [10] and numpy are the two main libraries used through this work. All the machine learning model implementations are used directly from scikit-learn. As part of preprocessing, the dataset is shuffled randomly and the feature values are standardized (scaled to a mean of 0 and standard deviation of 1). Throughout the paper, ten-fold cross validation results have been reported.

3.1 Hyperparameter tuning

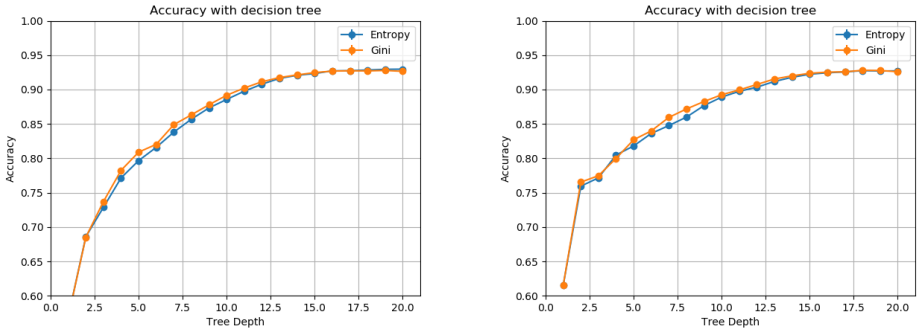
In this section we tune hyperparameters for a few non-linear classifiers. For each of the classifiers, we consider two settings:

1. Simply use the 5 base features, i.e., no domain knowledge used
2. Use a total of 8 features (5 base + 3 extra), i.e., leverage domain knowledge

3.1.1 Decision Trees

Here we test two different splitting criterions over different values of maximum tree depth. A few observations can be made:

- When tree depth is short (< 5), the 3 additional features significantly improve the prediction accuracy. The extra features play a diminishing role in accuracy as the trees get deeper. The accuracy values for both the cases (fig. 5a and fig. 5b) saturate around 0.93.
- Choice of splitting criterion (gini impurity vs. entropy) has little effect on accuracy across all tests. The accuracy from both criteria remains roughly the same across tree depths and in 5 features vs. 8.
- Accuracy saturates beyond max tree depth of around 20. This value remains similar with 5 input features and 8 input features. This is confirmed in experiments with random forests as well (see figure 6).



(a) Without domain knowledge (5 features) (b) With domain knowledge (8 features)

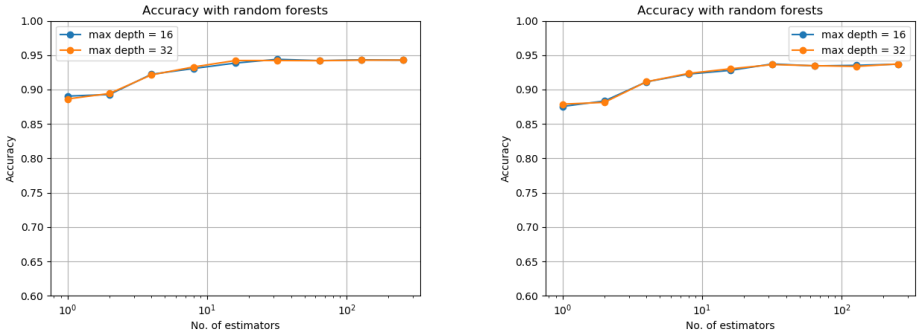
Figure 5: Model prediction accuracy (10-fold cross-validation tests) with decision trees

3.1.2 Random Forests

Here we test accuracy vs. the number of estimators used. We do this for two different values of max tree depth. This is repeated with both cases, one with 5 features and one with the 3 extra features. A few observations can be made from these tests:

- The accuracy values are fairly high (close to 0.95 for high number of estimators).

- The extra features have no impact on the accuracy. Even for a small number of estimators. This makes sense considering the results seen in figure 5. The accuracy of decision trees was not affected by the 3 additional features beyond a tree depth of 16. Here we consider tree depths of 16 and 32, therefore, even with 1 estimator, the accuracy values are not affected by the additional features.
- The maximum tree depth values (beyond 16) has no impact on accuracy of the random forests.
- The accuracy value is practically saturated beyond 30 estimators for all cases.

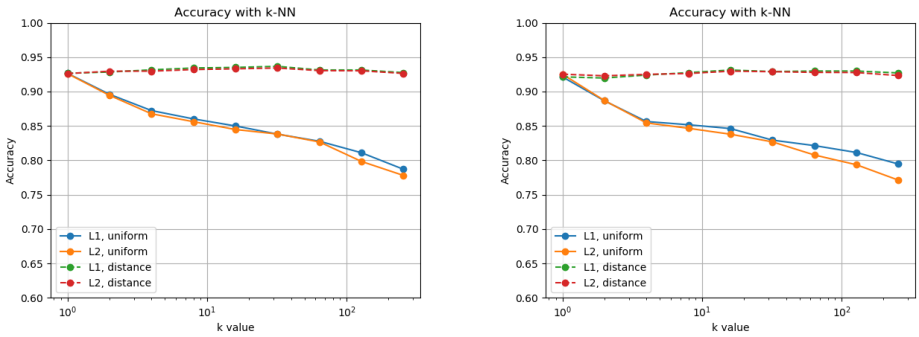


(a) Without domain knowledge (5 features) (b) With domain knowledge (8 features)

Figure 6: Model prediction accuracy (10-fold cross-validation tests) with random forests

3.1.3 K-Nearest Neighbors

Here we experiment with different distance metrics (L_1 and L_2) and different label computation options (uniform and distance weighted averaging). These experiments are repeated for both cases, with the 5 base features and with all the 8 features. Results are presented in Figure 7.



(a) Without domain knowledge (5 features) (b) With domain knowledge (8 features)

Figure 7: Model prediction accuracy (10-fold cross-validation tests) with k-nn

Here are a few observations based on results in Figure 7:

- The 3 additional features provide an insignificant improvement in the prediction accuracy.
- Label calculation method directly impacts accuracy.

  With $k = 1$, we have a single neighbor, therefore the distance weighted label coincides with the direct computation.

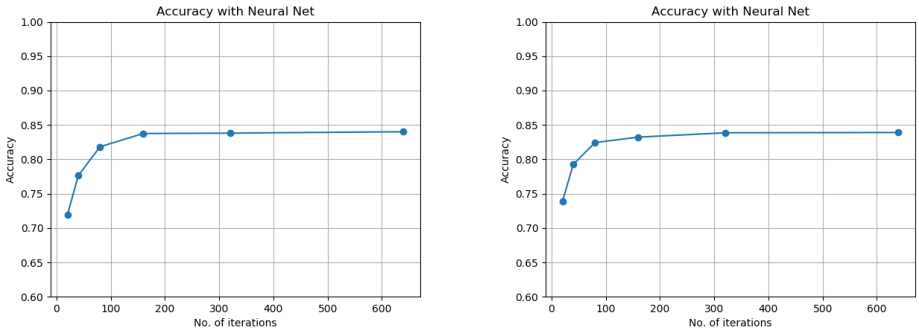
  As the k value increases the uniformly weighted label accuracy drops quickly. $k = 1$ is the optimal value for uniform weighing. This makes sense when we consider the data in Figure 3. There is no gap between the different classes, moreover, there is some mixing of the classes as well. Therefore higher k values pollute the labels.

This issue is remedied in the distance weighted labels, where closer points are weighted higher. Here we observe a small increase in accuracy with increasing k values up to $k = 30$.

- Finally, the the L_1 and L_2 distance metrics have little impact on the accuracy.

3.1.4 Neural Networks

Here we consider the accuracy of neural network classifiers as a function of number of training epochs/iterations. We tested different neural network architectures (different number of hidden layers and neurons per layer) but they performed similarly with 2 or more hidden layers with more than 10 neurons each. In Figure 8 we present results with a 5 hidden layers, first four have 40 neurons and the last one has 8 neurons. Once again we see that the additional features have little impact on the accuracy. Also, the accuracy saturates beyond 300 iterations.

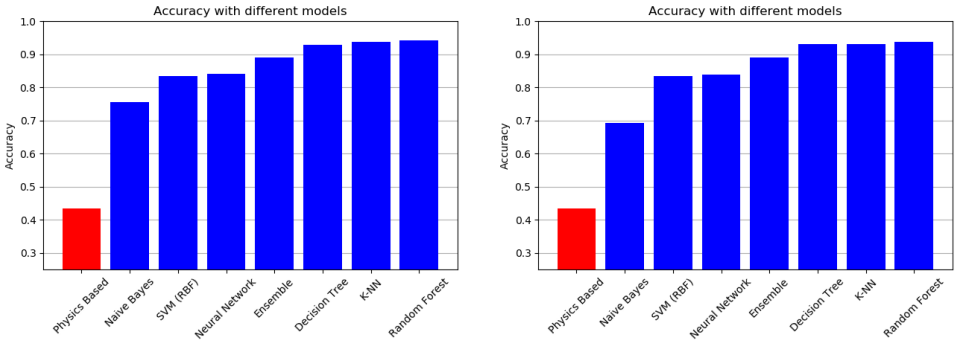


(a) Without domain knowledge (5 features) (b) With domain knowledge (8 features)

Figure 8: Model prediction accuracy (10-fold cross-validation tests) with Neural Networks

3.2 Accuracy with tuned models

In this section we compare the accuracy of different machine learning models (with tuned hyperparameters based on prior sections) for the two cases, one with the 5 base features and the other with the additional features. The results are presented in Figure 9. The accuracy of the state of the art physics based models [2] is also presented (in red) as a baseline. The 'ensemble' method refers to a majority vote of the random forest, neural network, SVM and k-NN predictions.



(a) Without domain knowledge (5 features) (b) With domain knowledge (8 features)

Figure 9: Model prediction accuracy (10-fold cross-validation tests) across different models

A few comments and observations regarding the results:

- Even the simplest ML models provide a significant improvement over the current state of the art physics based models. This is mainly due to the low model complexity of the physics based models.

- Little difference in the accuracy in the two cases, with and without domain knowledge. This is consistent with our observations in Figures 5–8. The naive bayes classifier actually suffers with the additional features.
- The random forest classifier performs the best on this dataset. Although it is only marginally better than the much simpler decision tree and k-NN classifiers.
- This ensemble classifier does not boost the accuracy. An ensemble is expected to improve accuracy when the different classifiers exploit different aspects of the dataset.

The normalized confusion matrix, showing the predicted labels vs. the true lables, for the random forest classifier is presented in Figure 10 as a sample. The random forest classifier is picked because it has the highest accuracy. The accuracy is fairly high for the first 3 classes. The highest error is when the true label is 'reflexive' and it gets classified as 'coalescence'. More training data near this boundary can help boost the model accuracy further. This can possibly guide further experiments.

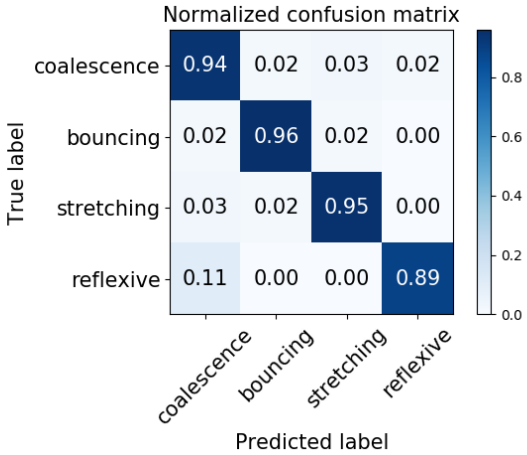


Figure 10: Confusion matrix for the random forest classifier.

3.3 Why did domain knowledge not help?

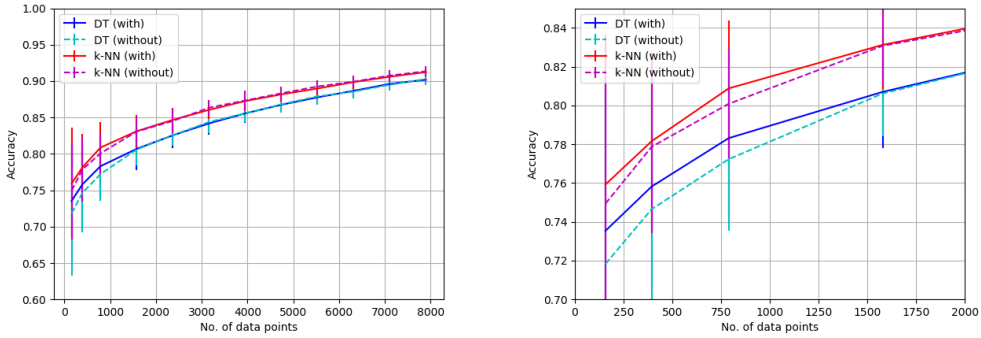
A key question that arises from the experiments presented so far is why the domain knowledge in the form of the additional features did not help the performance of any of the models. To answer this we consider an experiment with increasing size of the dataset. The two simple classifiers that perform well (see Figure 9), decision tree and k-NN, are considered for this test. For both the classifiers we consider training with 5 features (dotted lines) and 8 features (solid lines).

A few observations can be made:

- The accuracy has not saturated with respect to increasing number of data points. More data is expected to boost accuracy of our models further.
- Domain knowledge does not help boost accuracy when the training dataset is large (2000+). However, for a smaller number of training data points, there is an improvement when domain knowledge is used. This is highlighted in Figure 11b, where we zoom in on the relevant region.

  The improvement (albeit small) in performance by addition of the extra features suggests that when there is a small amount of training data the models are unable to search the entire hypothesis space. The addition of the relevant features allows a better accuracy even when the hypothesis space is not completely spanned.

  This improvement vanishes as the number of training data points increases because the models are able to better search the hypothesis space with additional data.



(a) Over the whole range of dataset sizes (b) For small dataset sizes (zoomed into Figure 11a)

Figure 11: Model accuracy (10-fold cross-validation tests) with different models as a function of number of training data points.

3.4 Robustness to noise

A key metric to consider for the current application is the robustness to noise. As described in Table 1, the training data has been extracted from different sources going as far back as 1990. Certain experimental errors (up to 10%) have been reported in literature and some variations can be expected across the different datasets. Therefore, we must account for noise in our input data. To do this we run experiments where we add a varying degree of noise to the data.

Gaussian noise is added to the scaled feature vectors. The noisy data is given as

$$X(i,j) = X(i,j) + N(i,j), \tag{4}$$

where $N \sim \mathcal{N}(0,n)$, i is the instance index and j is the feature number. The value of n is represented on the x-axis in Figure 12. Four different classifiers are compared here, decision tree (DT), k-NN, neural network (NN) and random forests (RF).

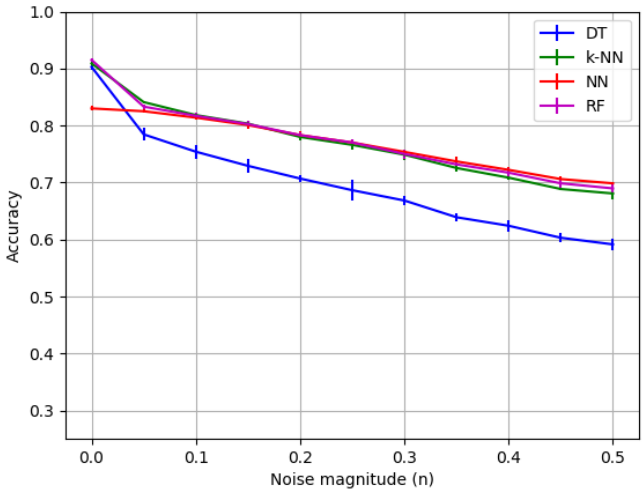


Figure 12: Accuracy of different classifiers as a function of noise in the data. The n value denotes the standard deviation in the Gaussian noise.

Two key things emerge from this test:

- The decision tree classifier is the least robust to noise. There is a sharp drop in accuracy initially up to 5% noise. The other 3 models are more robust to input noise.

- The neural network classifier, while starting from a lowest accuracy is the most robust to input noise. The neural network performs even better than the random forest for high levels of noise.

4 Conclusions

A major improvement is seen over the current state of the art physics based models ($\approx 43\%$) with all ML classifiers studied here ($> 90\%$). The accuracy is more than doubled. The best classifiers for the current dataset based on three key metrics are:

- Accuracy: Random forests
- Interpretability: Decision tree
- Robustness: Neural Networks

The k-NN classifier performs decently well on all three criteria.

Furthermore, the model accuracy has not saturated with the current size of the training dataset (see Figure 11). In the confusion matrix (Figure 10) the highest error is seen when the true label is 'reflexive' and it gets classified as 'coalescence'. More training data near this boundary can help boost the model accuracy further.

Droplet collision remains a relevant fluid-dynamics problem for several industries (e.g., automotive, food processing). This paper demonstrates that ML based classifiers can significantly improve the prediction of droplet collision outcomes. Moreover, other than simply improving the model accuracy, this work can be leveraged to provide insights into the fluid mechanics of the problem itself. Interpreting the present ML based models, mainly the decision tree, can lead to further insights about the dependence of the droplet collision phenomena on the different features.

References

[1] JM MacInnes and FV Bracco. Comparisons of deterministic and stochastic computations of drop collisions in dense sprays. *Numerical Approaches to Combustion Modeling*, pages 615–642, 1991.

[2] Arpit Agarwal, Yue Wang, Long Liang, Chitralkumar Naik, and Ellen Meeks. The computational cost and accuracy of spray droplet collision models. Technical report, SAE Technical Paper, 2019.

[3] Martin Sommerfeld and Matthias Kuschel. Modelling droplet collision outcomes for different substances and viscosities. *Experiments in Fluids*, 57(12):187, 2016.

[4] Arpit Agarwal. Github repository titled machine-learning-droplet-collision-modeling. `https://github.com/arpitragarwal/machine-learning-droplet-collision-modeling`, 2021.

[5] Achuth Munnannur and Rolf D Reitz. A new predictive model for fragmenting and non-fragmenting binary droplet collisions. *International journal of multiphase flow*, 33(8):873–896, 2007.

[6] Ankit Rohatgi. Webplotdigitizer (https://automeris.io/webplotdigitizer/), 2011.

[7] N Ashgriz and JY Poo. Coalescence and separation in binary collisions of liquid drops. *Journal of Fluid Mechanics*, 221:183–204, 1990.

[8] J-P Estrade, Hervé Carentz, G Lavergne, and Y Biscos. Experimental investigation of dynamic binary collision of ethanol droplets–a model for droplet coalescence and bouncing. *International Journal of Heat and Fluid Flow*, 20(5):486–491, 1999.

[9] J Qian and CK Law. Regimes of coalescence and separation in droplet collision. *Journal of Fluid Mechanics*, 331:59–80, 1997.

[10] 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.