Ming-hwei Carol Sun
DATASCI 207 Dec, 2023

Snowflake Classification

Brief Description/ Title

If you look closely at falling snow, you can see a great many different crystal shapes. There's a lot more to see than you might think![1] Through search engines, over 25,000 human-classified images of snowflakes were found by Colorado State University. The images were sorted into the following categories: AG (aggregate), CC (columnar crystal), GR (graupel), PC (planar crystal), SP (small particle). This dataset is perfect for machine learning tests to see which algorithm is the most accurate detecting the categories.[2]

Team members

Ming-hwei Carol Sun

Problem Statement

My original idea was to find images with non-organic metal particles taken under microscope since the company I am working for wants to have a way of categorizing the particles in images taken under microscope. The problem was that this type of images were not found on the internet. The closest one only had less than 300 images which is not enough to do different machine learning tests. By searching image datasets, many interesting image datasets were found and the snowflakes dataset ironically looked even closer to the images taken in the company than the one found with insufficient numbers. The only problem with the snowflake images is that they are all with black backgrounds and white subjects. The particle images taken are sometimes with white background and black subjects to indicate if the subject is an indentation or a raised particle. This problem can be resolved by altering the images during EDA.

Objective

The task of this project is to use the snowflakes images to see which machine learning algorithm is the most accurate in detecting its categories. If it is successful, this process may be able to detect similar images that contain particles under a microscope in the future.

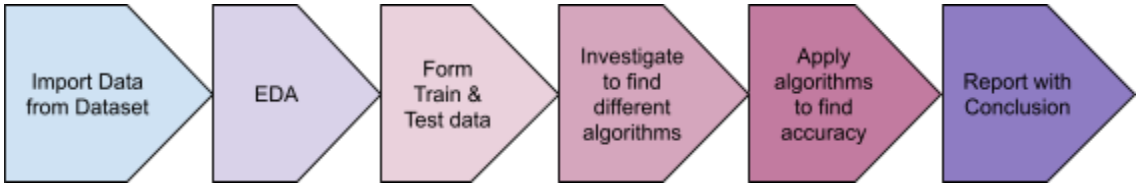
[1] Libbrecht, Kenneth G. “A Guide to Snowflakes.” *Guide to Snowflakes*, Caltech, 1 Feb. 1999, www.its.caltech.edu/~atomic/snowcrystals/class/class-old.htm.

[2] Key, C. et al. (2021) Advanced Deep Learning-Based Supervised Classification of Multi-Angle Snowflake Camera Images, JTECH

Approach/ Methodology

Using snowflake images in the dataset to make training data and test data for supervised machine learning. Use different algorithms to see how long it takes and how accurate the result would be.

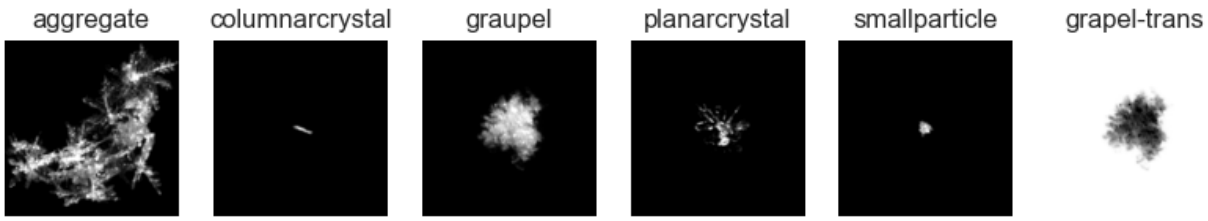
Block Diagram



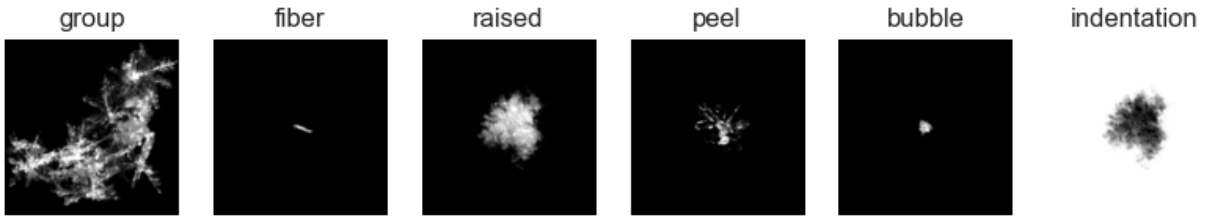
Datasets

Snowflakes images collected by Colorado State University: https://zenodo.org/record/4584200

Example of the snowflake types



The labels translated into particle types



What is considered success/ failure?

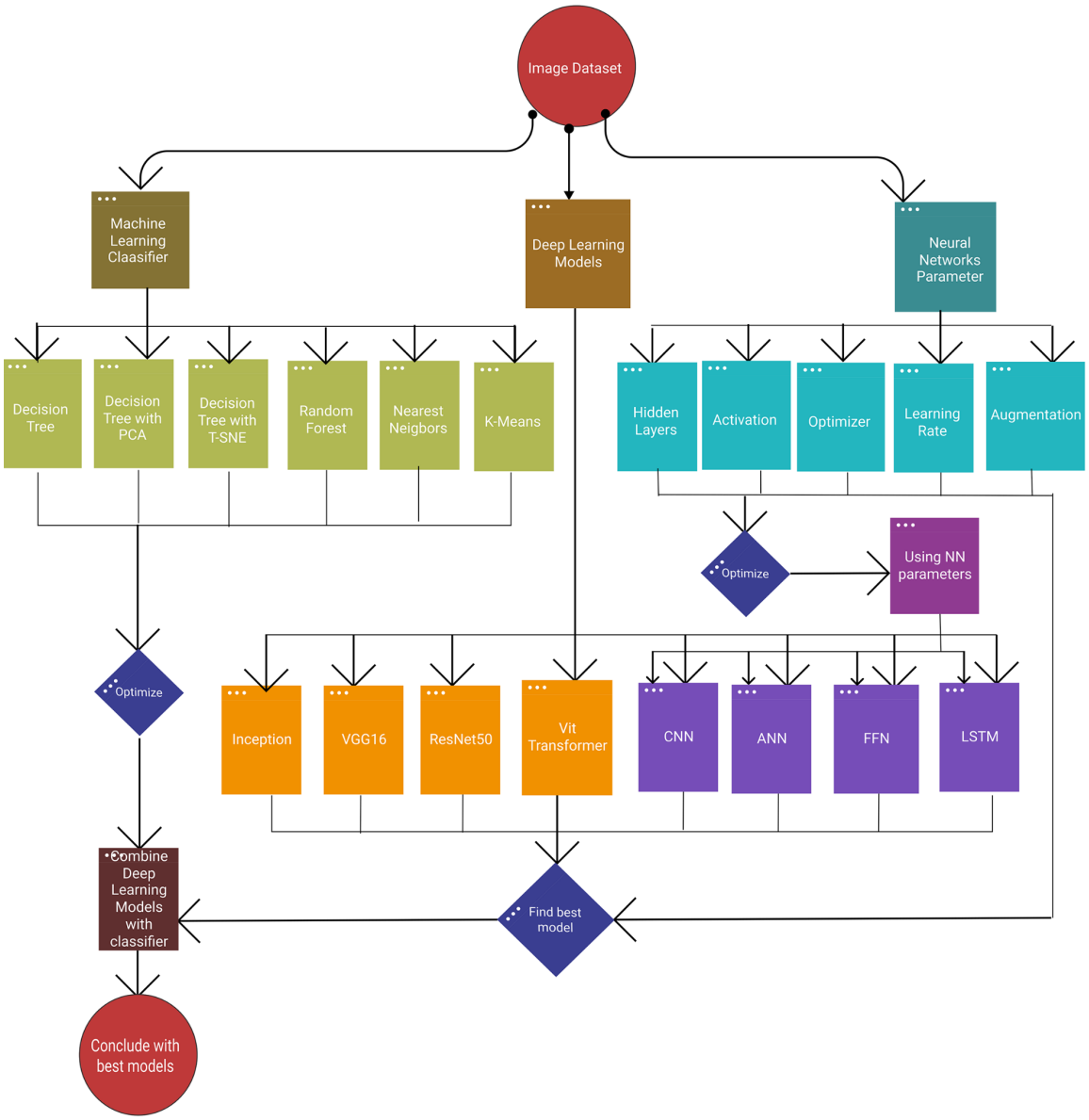
Find the best algorithm detecting the categories of the snowflakes. If the accuracy rate is above 90%, it is considered a success.

Evaluation Parameters

Accuracy - Final result of evaluation of test data after the model was trained with training data.

Experiments

1. The data is experimented with three types of process
    a. Machine learning classifiers
    b. Neural network parameters
    c. Deep Learning models
2. With optimized models of (b) neural network parameters and (c) deep learning models stack them to create new models.
3. With the top of the neural network models and stacked models combine with the top machine learning classifiers to see if it improves the performance.



Results

Machine Learning Classifier Results

The result is the accuracy from running the model for image classification. PCA and T-SNE are data that was altered. Four different models that were built and fit with the training data and predict the result of the test data. All models are supervised learning except K-means. It took more work to try to get the right classification categories with K-means

Name	Alteration	Accuracy
Decision Tree		0.785
Decision Tree	PCA	0.32
Decision Tree	T-SNE	0.49
Random Forest		0.88
Nearest Neighbors		0.85
K-means		0.67

Decision Tree has the simplest algorithm which produces medium good results. The altered data produced the worst result since the images may have less information that was necessary for the calculation. The best result is random forest.

Neural Network model with different parameters

The baseline NN model with tanh/SGD and no hidden layer and the rest are with relu/Adam and different hidden layers.

DATA	HIDDEN SIZES	ACTIVATION	OPTIMIZER	LEARNING RATE	#PARAMETERS	TEST ACCURACY
Snow Flakes	[]	tanh	SGD	0.01	114695	0.5800
Snow Flakes	[]	relu	Adam	0.01	114695	0.6767
Snow Flakes	[128]	relu	Adam	0.01	2097280	0.7067
Snow Flakes	[128,128]	relu	Adam	0.01	2097280	0.6933
Snow Flakes	[64, 128, 256]	relu	Adam	0.01	1048640	0.6233

has worse accuracy than some of the machine learning classifiers. With relu/Adam got the medium result like the classifiers. With hidden layers, the accuracy increased a bit but still stayed in the range of 65% to 70% accuracy.

CNN parameters

Convolutional Neural Network with different parameters including strides, pool size, learning rate, optimizer, Augmentation, epochs and batch size.

kernel_size	strides	pool_size	learning_rate	opt	augmentation	epochs	batch_size	Accuracy
(3,3)	(1,1)	(2,2)	0.001	Adam	False	1	64	0.8300
(5,5)	(2,2)	(2,2)	0.001	Adam	False	1	64	0.7850
(5,5)	(1,1)	(3,3)	0.001	Adam	False	1	64	0.8300
(5,5)	(1,1)	(2,2)	0.01	Adam	False	1	64	0.6150
(5,5)	(1,1)	(2,2)	0.001	SGD	False	1	64	0.8050
(3,3)	(2,2)	(2,2)	0.001	Adam	False	1	64	0.8250
(3,3)	(2,2)	(3,3)	0.001	Adam	False	1	64	0.7500
(3,3)	(2,2)	(2,2)	0.001	Adam	False	10	128	0.8667
(3,3)	(2,2)	(2,2)	0.001	Adam	True	20	128	0.8567
(3,3)	(2,2)	(2,2)	0.001	Adam	False	18	128	0.8633

The results of CNN are basically higher than just neural networks. The baseline 61.5% accuracy rate is with (5,5) kernel size, (1,1) strides, (2,2) pool size, 0.01 learning rate with 1 epoch and 64 batch size. 83% accuracy is the best result with 1 epoch and 64 batch size baseline. The parameters of the models are either (3,3) kernel size plus (2,2) pool size or (5,5) kernel size plus (3, 3) pool size. Fine tuning of the model used (3,3) kernel size and (2, 2) pull size as the baseline and ran with different epochs. 10 epochs got the best result with 86.67% accuracy.

Different Neural Networks Models

The three different neural networks model was experimented including ANN, FFN and LSTM. Each of the models was built with baseline (5,5) and (3,3) kernel size, (1,1) and (2,2) strides and no CNN hidden layers. The models were then experimented with different CNN hidden layers and fine tuned with different epochs.

NN Name	hidden_layer	CNN_Hidden_Layers	Kernal_size	strides	epochs	batch_size	Accuracy
ANN			(5,5)	(1,1)	6	128	0.7700
ANN		[128, 128]	(5,5)	(1,1)	6	128	0.8400
ANN		[128, 128]	(5,5)	(1,1)	8	128	0.8600
ANN		[32, 64, 128]	(3,3)	(2,2)	20	128	0.8550
ANN		[64, 64, 128]	(3,3)	(2,2)	8	128	0.8650
ANN		[64, 64, 128]	(3,3)	(2,2)	8	128	0.8750
ANN		[64, 128, 256]	(3,3)	(2,2)	8	128	0.8650
ANN		[64, 128, 256]	(3,3)	(2,2)	8	128	0.9000
FFN	[128, 128]		(5,5)	(1,1)	6	128	0.6200
FFN	[32, 64, 128]		(5,5)	(1,1)	6	128	0.4650
FFN	[128, 128]	[128, 128]	(5,5)	(1,1)	6	128	0.8550
FFN	[128, 128]	[128, 128]	(3,3)	(2,2)	6	128	0.8900
FFN	[128, 128]	[64, 128, 256]	(3,3)	(2,2)	8	128	0.8550
FFN	[64, 128, 256]	[64, 128, 256]	(3,3)	(2,2)	5	128	0.90
LSTM			(3,3)	(2,2)	7	64	0.6050
LSTM		[128, 128]	(3,3)	(2,2)	7	64	0.8500
LSTM		[128, 128]	(3,3)	(2,2)	7	128	0.8733
LSTM		[32, 64, 128]	(3,3)	(2,2)	7	64	0.8250
LSTM		[128, 128]	(3,3)	(2,2)	6	128	0.8650
LSTM		[128, 128]	(3,3)	(2,2)	10	128	0.8750
LSTM		[64, 128, 256]	(3,3)	(2,2)	10	128	0.885
LSTM		[64, 128, 256]	(3,3)	(2,2)	15	128	0.89
LSTM		[64, 128, 256]	(3,3)	(2,2)	14	128	0.88
LSTM		[64, 128, 256]	(3,3)	(2,2)	6	128	0.87

The baseline of all models without CNN layers result in the lowest accuracy rate of the model. The highest accuracy ANN model had was built with [64, 128, 256] CNN layers. The highest accuracy FFN model was built with [64, 128, 256] layers and with [64, 128, 256] CNN layers.The highest accuracy LSTM model was built with [64, 128, 256] CNN layers. ANN and FFN got the same top accuracy of 90% and LSTM got 89%.

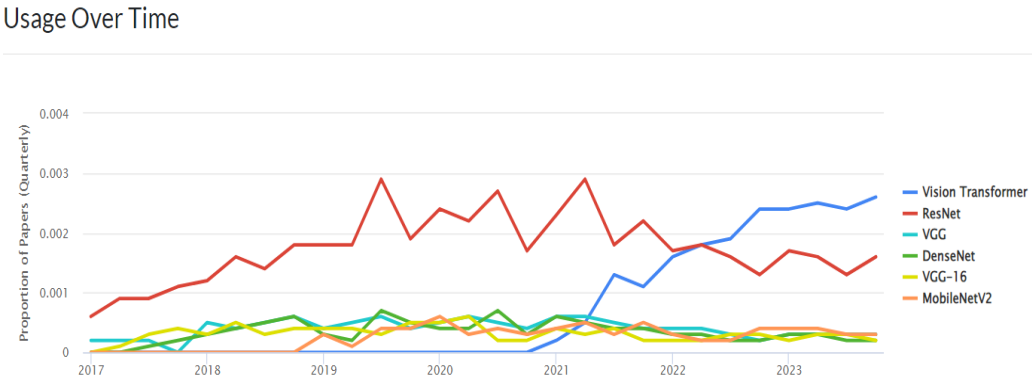
Transfer Learning and Transformer

With research of the transfer learning models, article found with the results some models[3]:

Comparison					
Network	Year	Salient Feature	top5 accuracy	Parameters	FLOP
AlexNet	2012	Deeper	84.70%	62M	1.5B
VGGNet	2014	Fixed-size kernels	92.30%	138M	19.6B
Inception	2014	Wider - Parallel kernels	93.30%	6.4M	2B
ResNet-152	2015	Shortcut connections	95.51%	60.3M	11B

ResNet-152 was said to be the best with the result as high as 95% accuracy. Therefore ResNet-152 was first experimented. It took around five hours of fitting the data and came out to be 16% accuracy. It proves that ResNet-152 is not useful for the dataset. With the reference of keras, three transfer learning models were experimented including InceptionV3, VGG16 and ResNet50V2.

Vit Transformer was added for the experiment due to its outstanding performance from research document[4]:



The problem with the plot is that it is used with lots of time which is not permitted with this project. Since these transfer learning models and the transformer model took around 10 minutes to 30 minutes to fit the training data due to the large amount of layers and parameters in each model, only 1 epoch in each model was implemented. The result of the transfer learning model and vit transformer is as the follow:

Model Name	epochs	batch_size	Parameters	Train Accuracy	Test Accuracy
InceptionV3	1	256	21815078	0.737	0.166
InceptionV3	1	128	21815078	0.759	0.166
VGG16	1	128	14717766	0.259	0.419
ResNet50V2	1	128	23577094	0.786	0.167
Vit Transformer	1	128	21329869	0.345	0.48

All the models have very low accuracies compared to the neural networks and VGG16 has the highest accuracy among them with the least parameters. Vit Transformer has higher accuracy than the transfer learning models.

[3] Anwar, Aqeel. “Difference between Alexnet, Vggnet, ResNet and Inception.” *Medium*, Towards Data Science, 22 Jan. 2022, towardsdatascience.com/the-w3h-of-alexnet-vggnet-resnet-and-inception-7baaaecccc96.
[4] “Papers with Code - Vision Transformer Explained.” *Explained | Papers With Code*, paperswithcode.com/method/vision-transformer. Accessed 13 Dec. 2023.

From looking the the transfer learning models, all of them except VGG16 suffered from over-fitting problems since the training accuracy is much higher than the test accuracy.

Model stacking using Random Forest Classification and Deep Learning models

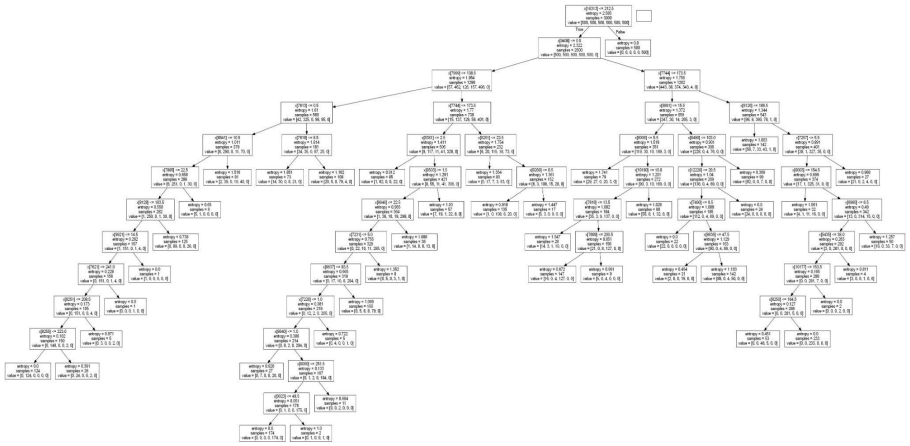
This experiment is to use the final layer before classification of the best deep learning models to generate data and put it into the best classifier to see if it will increase the accuracy. This is more of a fine tuning of the accuracy using the stacking method.

NN Name	hidden_layer	CNN_Hidden	Kernal	strides	epochs	batch	features	parameters	Accuracy	RF Accuracy
CNN			(3,3)	(2,2)	30	128	1024	9293830	0.87	0.87
CNN			(3,3)	(2,2)	20	128	1024	9293830	0.87	0.90
CNN + Aug			(3,3)	(2,2)	18	128	1024	4220294	0.86	0.87
CNN + LSTM		[64, 128, 256]	(3,3)	(2,2)	6	128	1024	9293830	0.88	0.91
CNN + LSTM		[64, 128, 256]	(3,3)	(2,2)	20	128	1024	9293830	0.89	0.90
CNN + ANN		[128, 128]	(3,3)	(2,2)	6	128	100	39765662	0.84	0.88
CNN + ANN		[64, 128, 256]	(3,3)	(2,2)	5	128	100	707870	0.90	0.91
CNN + FFN	[128, 128]	[128, 128]	(3,3)	(2,2)	6	128	8192	1214854	0.89	0.89
CNN + FFN	[64, 128, 256]	[64, 128, 256]	(3,3)	(2,2)	5	128	256	478150	0.90	0.90
CNN + FFN	[64, 128, 256]	[64, 128, 256]	(3,3)	(2,2)	5	128	1024	478150	0.90	0.90

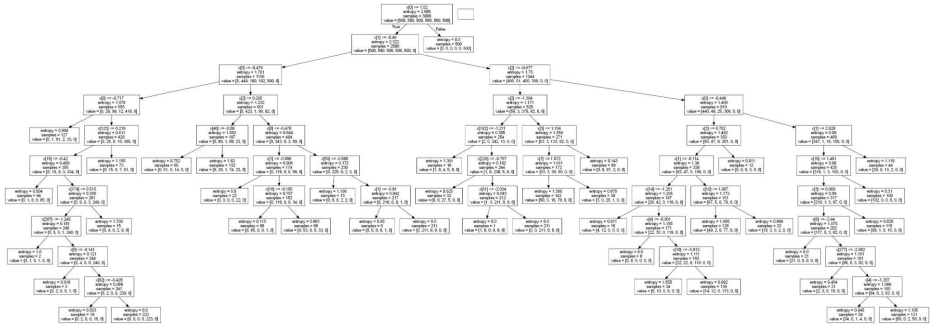
The best models that reached 91% accuracy include CNN + LSTM model and CNN + ANN model. The CNN model by itself increased 3% accuracy and reached 90%. The other models that reached 90% accuracy are CNN + LSTM and CNN + FFN. The best models are using hidden layers of [64, 128, 256] both on its own hidden layers and CNN hidden layers.

Tests/ Graphs/ Discussions

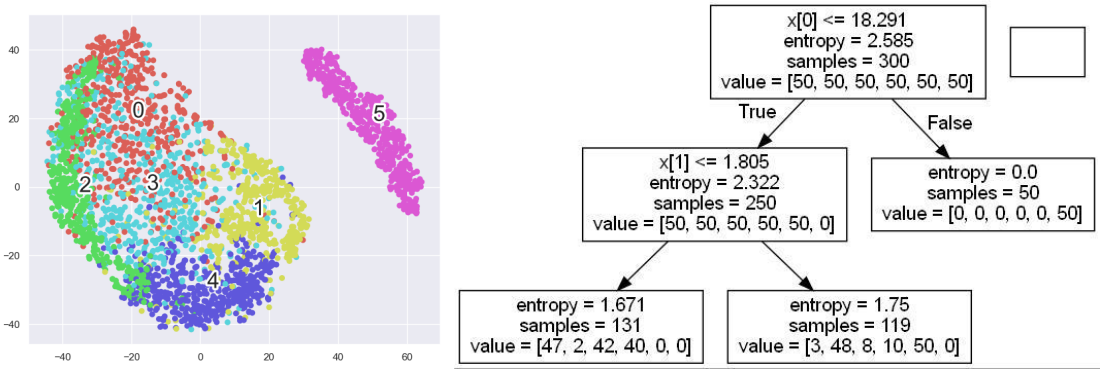
Decision Tree - `Max Depth: 13  Tree Nodes: 85`



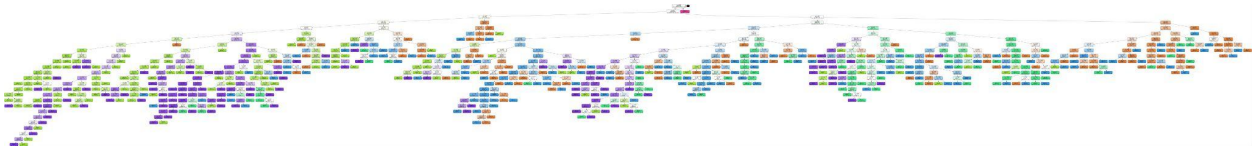
Decision Tree with PCA - `Max Depth: 10  Tree Nodes: 75`



Decision Tree with T-SNE - `Max Depth: 2  Tree Nodes: 5`



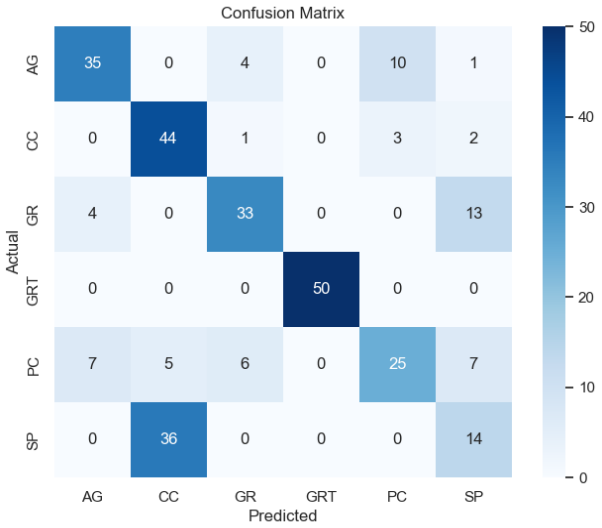
Random Forest - max depth: `Max Depth: 42129  Tree Nodes: 35`



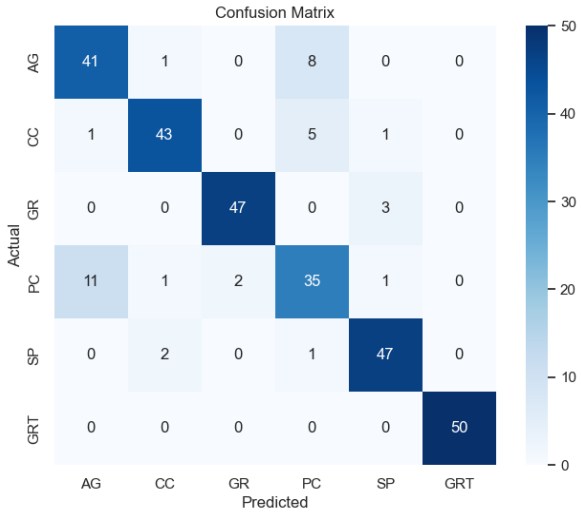
From looking at the graphs, PCA image is simplified with less accurate prediction ability and T-SNE is oversimplified. Random forest on the other hand has the most nodes and depth hence with the best result.

Confusion Matrix

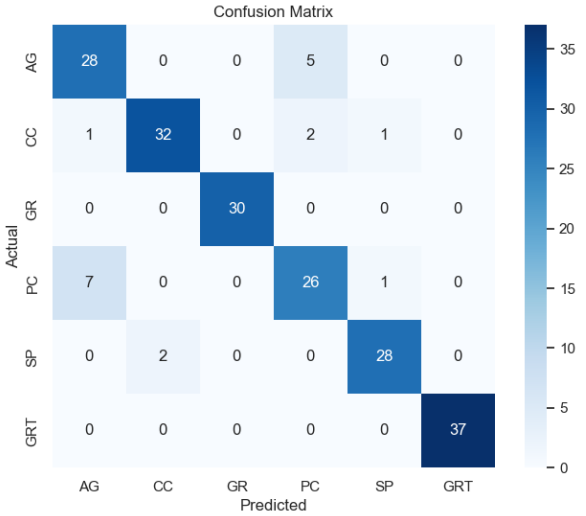
K-Means : Accuracy 67%



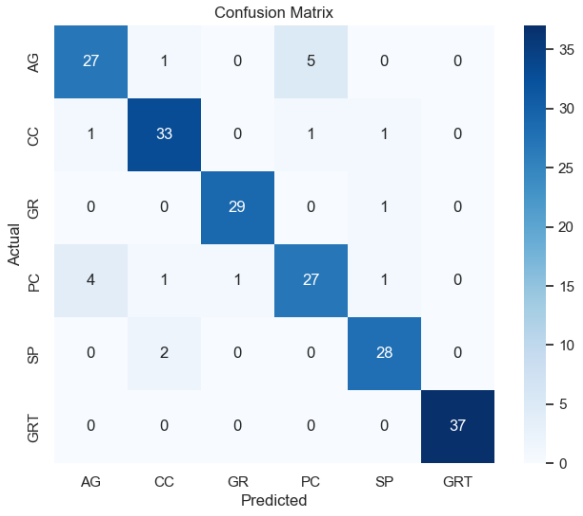
Random Forest: Accuracy 88%



CNN + LSTM Random Forest: 91%Accuracy



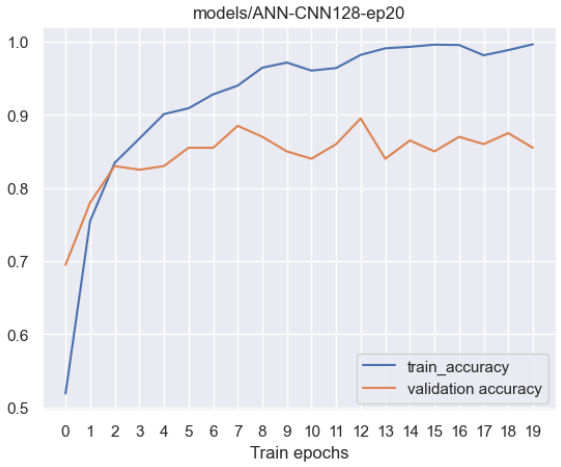
CNN + ANN Random Forest: 91% Accuracy

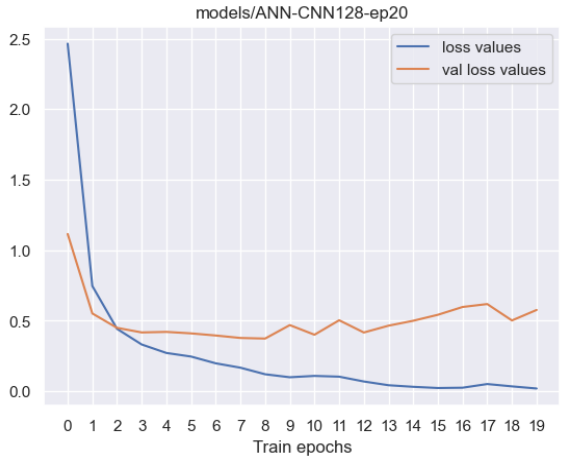


With confusion matrix, improvement shows with stacked models and the best model is CNN+ANN with Random forest.

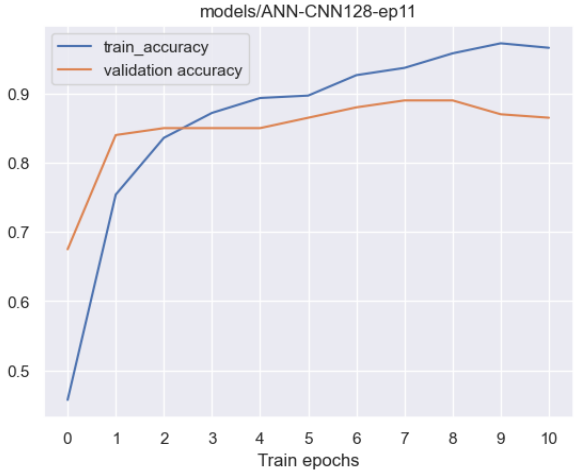
Accuracy and Loss with epochs fine tuning

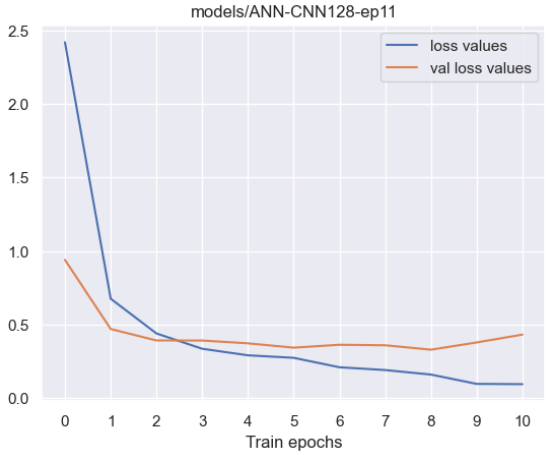
CNN + ANN with 20 epochs



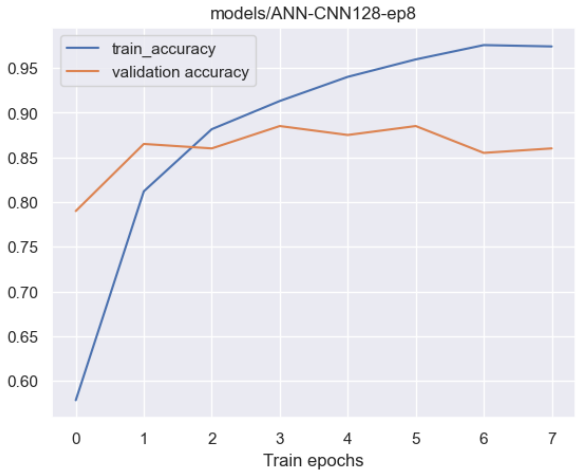


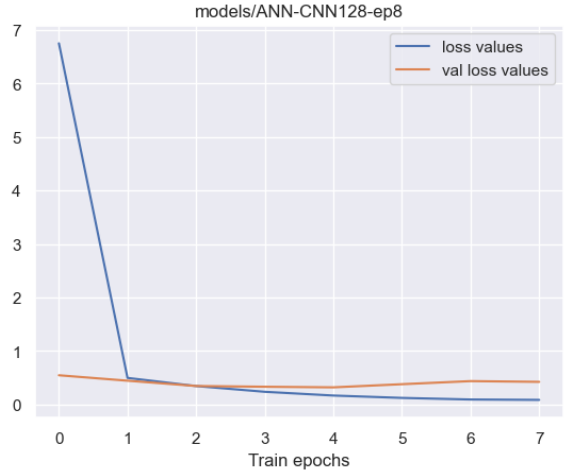
CNN+ANN with 11 epochs



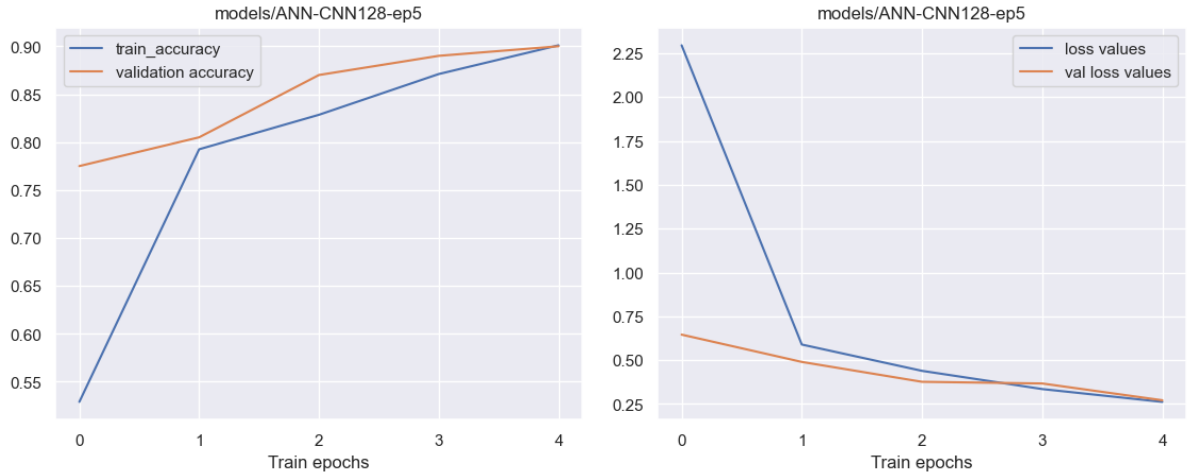


CNN+ANN with 8 epochs





CNN+ANN with 5 epochs



CNN+ANN fine tuning with before stack with random forest. The model converged with accuracies and losses with 5 epochs of training.

Constraints

Training time is the biggest constraint for many models, especially the model that is prebuilt with more than ten layers. The models that need to generate new sets of images with training also require lots of time even though it is not as bad as the ones with more than ten layers. Many models may be able to improve with deeper investigation and variation of tests.

The dataset which was originally black and white got less information from the beginning may not be reduced to get better training accuracy since it may be at the minimum already. Most pre-built models also would not take images with grayscale images and need to twist the images to include RGBs.

Many other models could be translated to be used in the project but are not used due to lack of knowledge and research. For example RNN may be implemented to evaluate image dataset but there is limited information and examples to build RNN models for image classification. Some information may be deceiving indicating that RNN is only suitable for imitating images and not suitable for image classification.

Standards

- Python: the project is running on jupyter notebook alone with .py python file
- Plotform: Tried to use colab and since the data is too big, the code runs mainly locally with visual studio code on python version 3.10.
- Modeling: The packages that are provided to do the modeling are mainly TensorFlow, Keras and Sklearn. Plot: Pyplot from matplotlib is for plotting.
- Data manipulation and storage: Pandas dataframe is often used for data storing alone with os, csv and pickle when keras does work in storing and retrieving data.

Comparison

Comparing all the machine learning models, random forest has the best performance and neural network models perform better with multiple layers. Stacking the models together gives the best performance. If there is time, the pre-built models may have a chance to run with more than one epoch to see if it has a better performance. To improve the models, a variation of parameters can be further applied such as epochs, batch size and layers. There is no end to how many parameters can be done as long as there is the need and time provided.

Limitations of the Study

The study of the model depends on the examples and homeworks provided by the class and researches that had been done and posted online. Without these resources, it is difficult to understand the concepts of the models and implement them according to what people have done.

Future Work

The knowledge and experience of research and using the models with image data is important for future research and development. The next step would be to apply the models with the particle data and run to manipulate the image with the right image modifying tool and eventually develop the classifier to automatically identify the particles on the batteries and update the database. With the system implemented, quality control could save lots of time by running the program instead of manually checking individual batteries under microscope.