Task

Given dataset contains following columns
molecule_name - this contains 102 class of molecule
conformation_name - All of which are unique
f1 - f166 - These are the features of every molecule and its conformation
class - this column contains 0 or 1 (1 for musk and 0 for non-musk)

So the classification model can be made either to classify between *musk or no_musk* **or** 102 categories of molecules.

Pre Processing

I have used StandardScaler for scaling the input data.
This is necessary because we don't know on what scale and range of the features are in. StandardScaler makes our data such that its mean value of 0 and standard deviation of 1. So that it helps in efficiency training our neural network model.

Models

I have trained two deep learning algorithms viz, Deep **Neural Network** and **Convolutional Neural Network** for both **musk vs non-musk classification** and also **102 category of molecules classification**.

All these models performed well. But as there are 102 categories of molecules containing musk and non musk molecules. I have applied **transfer learning** as first training 102 category molecules and used the same model and added 2 layers to train for musk vs non musk.
By using transfer learning I achieved **100% accuracy** in musk vs non musk classification.

Details of which are described below.
Link for the jupyter notebooks on github and colab along with the link for the model files of all the models are at the end of this document in the form of table.

Accuracy Table

	musk vs non-musk	102 class of molecules
Deep neural network	**Accuracy -** 0.99924242424	**Accuracy -** 0.971969696969
Convolutional neural network	**Accuracy -** 0.99848484848	**Accuracy -** 0.956818181818
Transfer learning	**Accuracy -** 1.0 or 100%	

Musk vs Non-Musk - Deep Neural Network

Model summary -

```
model = Sequential()
model.add(Dense(200, activation='relu'))
model.add(Dense(100, activation='relu'))
model.add(Dense(1, activation='sigmoid'))
model.compile(optimizer='adam', loss='mse', metrics=['accuracy'])
```

```
epochs = 20
```

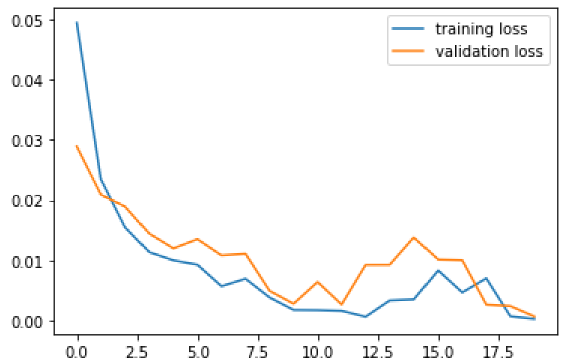


Figure 1 - Training vs validation loss

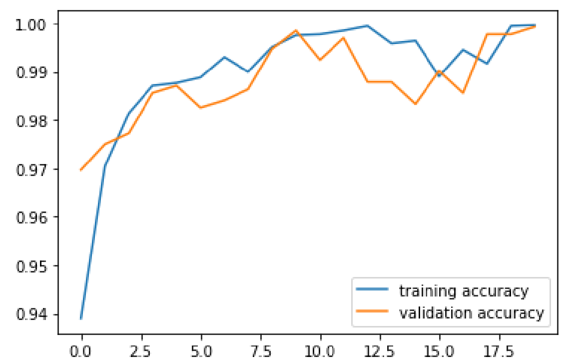


Figure 1 - Training vs validation accuracy

Precision recall f1-score support values

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00      1118
           1       1.00      1.00      1.00       202

    accuracy                           1.00      1320
   macro avg       1.00      1.00      1.00      1320
weighted avg       1.00      1.00      1.00      1320
```

Confusion matrix

```
array([[1117,    1],
       [   0,  202]])
```

Musk vs Non Musk - Convolutional Neural Network

Model summary -

```
model = Sequential()
model.add(Conv1D(filters=64,        kernel_size=3,        activation='relu',
input_shape=(X.shape[1],X.shape[2])))
model.add(Conv1D(filters=32, kernel_size=3, activation='relu'))
model.add(Flatten())
model.add(Dense(50, activation='relu'))
model.add(Dense(1))
model.compile(optimizer='adam', loss='mse', metrics=['accuracy'])

epochs = 20
```

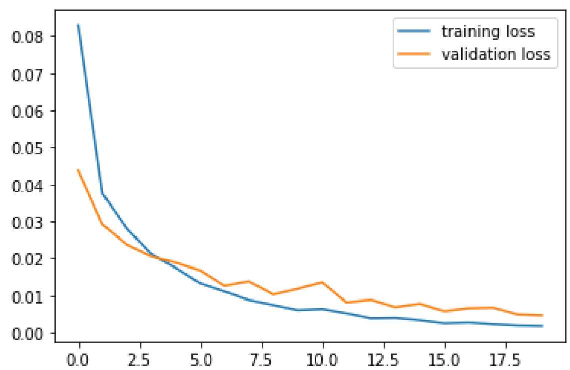


Figure 1 - Training vs validation loss

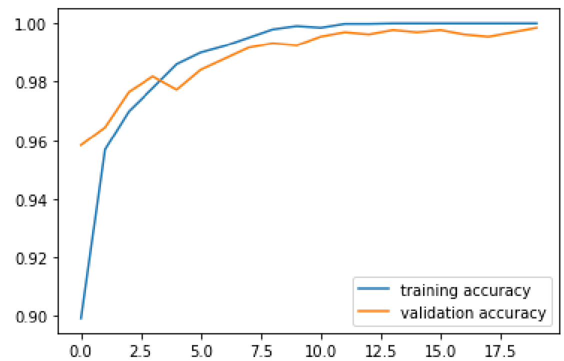


Figure 1 - Training vs validation accuracy

Precision recall f1-score support values

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00      1115
           1       0.99      1.00      1.00       205

    accuracy                           1.00      1320
   macro avg       1.00      1.00      1.00      1320
weighted avg       1.00      1.00      1.00      1320
```

Confusion matrix

```
array([[1113,    2],
       [   0,  205]])
```

102 category of molecule classification - Deep Neural Network

Model summary -

```
model = Sequential()
model.add(Dense(300, activation='softmax'))
model.add(Dense(200, activation='relu'))
model.add(Dense(102, activation='sigmoid'))
model.compile(optimizer='adam', loss='mse', metrics=['accuracy'])

epochs = 100
```

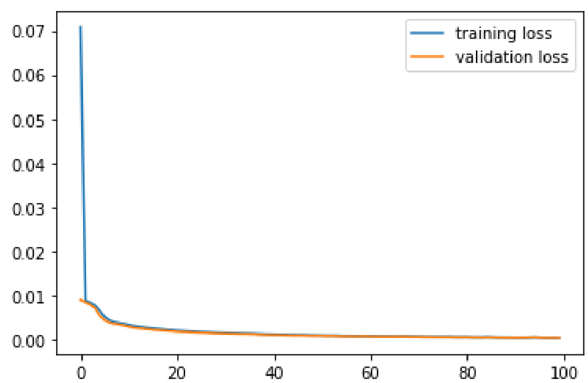


Figure 1 - Training vs validation loss

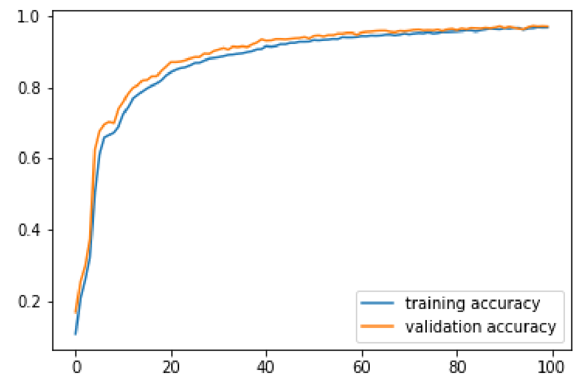
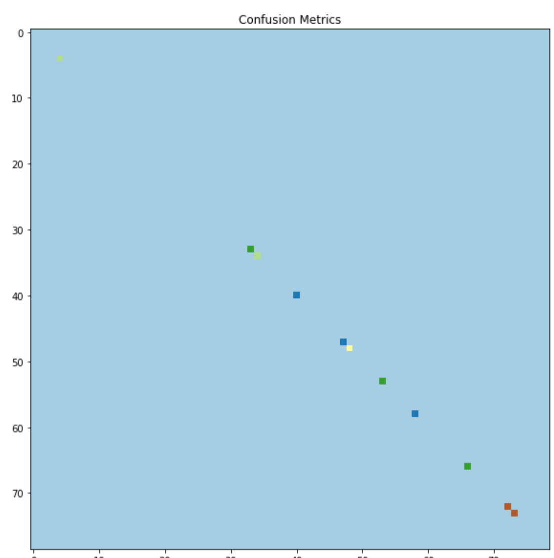


Figure 1 - Training vs validation accuracy

Precision recall f1-score support values

Please prefer the notebook for these data as it generates a long table.

Confusion matrix



102 category of molecule classification - Convolutional Neural Network

Model summary -

```
model = Sequential()
model.add(Conv1D(filters=64, kernel_size=3, activation='softmax',
input_shape=(X.shape[1],X.shape[2])))
model.add(MaxPooling1D())
model.add(Conv1D(filters=32, kernel_size=3, activation='relu'))
model.add(Flatten())
model.add(Dense(y.shape[1], activation='sigmoid'))
model.compile(optimizer='adam', loss='mse', metrics=['accuracy'])

epochs = 100
```

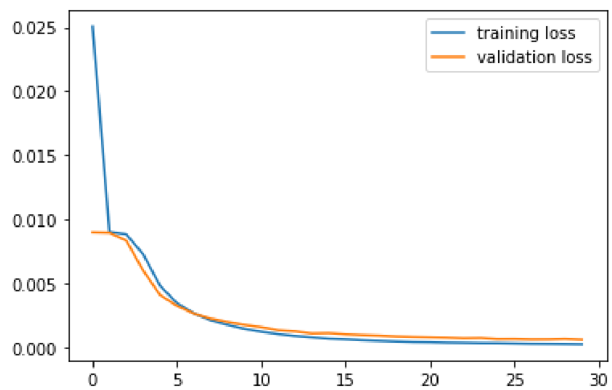


Figure 1 - Training vs validation loss

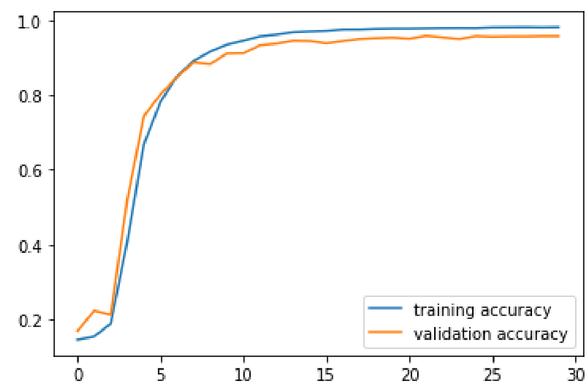
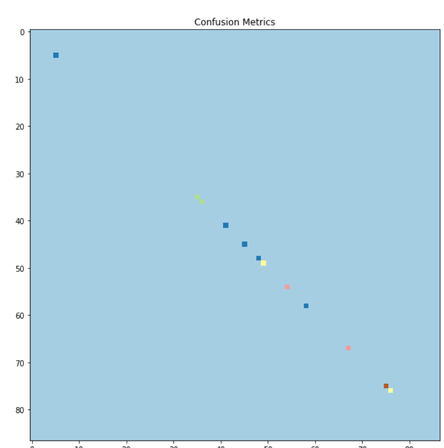


Figure 1 - Training vs validation accuracy

Precision recall f1-score support values

Please prefer the notebook for this data as it generates a long table.

`Confusion matrix`



Musk vs non-musk - Transfer learning

Here our approach is to first train our model with 102 class of molecule and then add another layer on the top of the previously trained model and again train it with binary class of musk vs non-mask with dense layer with 1 perceptron in the final layer

Model summary -

```
model = Sequential()
model.add(Dense(300, activation='softmax'))
model.add(Dense(200, activation='relu'))
model.add(Dense(102, activation='sigmoid'))
model.compile(optimizer='adam', loss='mse', metrics=['accuracy'])
```

```
Epochs = 100
```

After training with the above layers for 102 classes

```
model1.add(Dense(50, activation='relu'))
model1.add(Dense(1, activation='sigmoid'))
model1.compile(optimizer='adam', loss='mse', metrics=['accuracy'])
```

```
epochs = 20
```

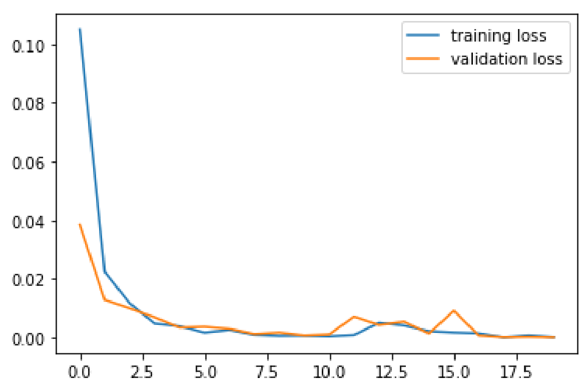


Figure 1 - Training vs validation loss

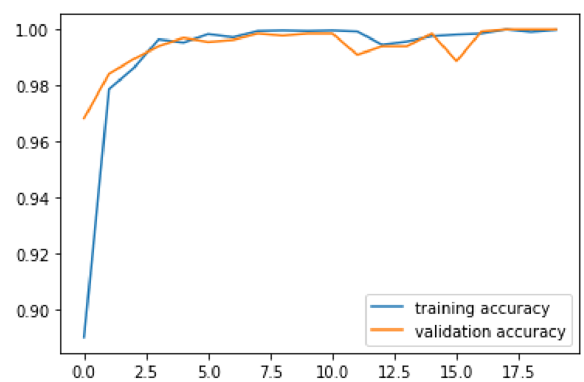


Figure 1 - Training vs validation accuracy

Precision recall f1-score support values

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00      1110
           1       1.00      1.00      1.00       210

    accuracy                           1.00      1320
   macro avg       1.00      1.00      1.00      1320
weighted avg       1.00      1.00      1.00      1320
```

Confusion matrix

```
array([[1110,    0],
       [   0,  210]])
```

Code and model files

	musk vs non-musk	102 class of molecules
Deep neural network	Github Colab model.h5	Github Colab model.h5
Convolutional neural network	Github Colab model.h5	Github Colab model.h5
Transfer learning	Github Colab model.h5	