Data Preprocessing:

Train image = 251 sample

Test image = 66 sample

Size of each sample = 300 * 300

Normalization:

1. standard normalization with standard scaler in sklearn.preprocessing
2. global contrast normalization (GCN) https://datascience.stackexchange.com/questions/15110/how-to-implement-global-contrast-normalization-in-python

Model:

Logistic regression:

Hyper parameters	normalization	Dimention reduction	time	accuracy
Max iter=300	standardscaler	-	24.815s	0.893
Max iter=300	standardscaler	PCA	0.036s	0.772
Max iter=300	standardscaler	LDA	0.023s	0.984
Max iter=300	GCN	LDA	0.018	0.924

SVM:

Hyper parameters	normalization	Dimention reduction	time	accuracy
Kernel=rbf	standardscaler	-	6.905s	0.878
Kernel=rbf	standardscaler	PCA	0.008s	0.742
Kernel=rbf	standardscaler	LDA	0.005s	0.984
Kernel=rbf	GCN	LDA	0.004s	0.954
Kernel=poly, degree=3	standardscaler	LDA	0.002s	0.969
Kernel=poly, degree=3	GCN	LDA	0.002s	0.954

CNN:

Normalization: GCN

Layer1: conv2d, filter=149, kernel=(3,3), activation=relu, padding=valid

Layer2: maxpooling2d, kernel=(2,2)

Layer3: conv2d, filter=294, kernel=(3,3), activation=relu, padding=valid

Layer4: maxpooling2d, kernel=(2,2)

Layer5: conv2d, filter=299, kernel=(3,3), activation=relu, padding=valid

Layer6: maxpooling2d, kernel=(2,2)

Layer7: conv2d, filter=528, kernel=(3,3), activation=relu, padding=valid

Layer8: maxpooling2d, kernel=(2,2)

Layer9: conv2d, filter=896, kernel=(3,3), activation=relu, padding=valid

Layer10: maxpooling2d, kernel=(2,2)

Layer11: flatten

Layer12: dense, neuron=343, activation=relu

Layer13: dense, neuron=3,activation=softmax

Epoch = 10

Time = 128s

Accuracy = 909

Conclusion:

Svm with lda and standard normalization has had the best performance

In SVM and Logistic regression, the initial values of the models in sickit-learn had the best output.

In PCA, by increasing the components, worse results were obtained

Generally, LDA is suitable for large dataset and multi label data, PCA is suitable for small dataset

Number of component for LDA is 2 because its value must be less than equal to min(n_classes - 1, n_features)