E1 213 : Pattern Recognition and Neural Networks
Assignment 2 Report

Bhartendu Kumar, Jeevithiesh Duggani, Nishanth Shetty, Rahul Raju Pogu

I. Support Vector Machines for Classification

The Code has the implementation for SVM with different kernels (with grid serach on the hyper-parameters of the kernels). We implemented our own **SMO**.
The Dataset we implemented SVM are:

- **Pneumonia MNIST** (Binary Classification)
- **Blood MNIST** (Multi Classification)
- **Road-Sign-Detection** (Regression)
- **TIMIT** (Binary Classification)

The kernels that we implemented are:

- Linear kernel will be defined as

$$K(x_i, x_j) = x_i^T x_j + b \tag{1}$$

  where x_i and x_j are arrays of input feature vectors, and b is an optional. This kernel calculates a pairwise linear combination of the points listed in x_i and x_j. Using this kernel will result in the generation of a linear decision boundary. The hyper-parameter is b.
- Gaussian kernel will be defined as

$$K(x_i, x_j) = e^{\frac{-||x_i - x_j||^2}{2\sigma^2}} \tag{2}$$

  where x_i and x_j are arrays of input feature vectors and σ is a width parameter describing how wide the kernel is (this can be set based on the spacing between data points). This kernel calculates a Gaussian similarity between the training examples listed in x_i and x_j, with a value of 1 indicating that the points have exactly the same feature vector and 0 indicating dissimilar vectors. Using this kernel allows for the construction of more complex, non-linear decision boundaries.
- Polynomial
- Sigmoid

A. Important Observations

- **Feature Scaling**
  - The Data should be normalized, i.e. all the features should be at same scale and normalized.
  - Accuracy generally increases with Feature Normalization.
  - Makes SVM fast to run.
- Comparing the performance of Neural Nets also **SVM** is better in Pneomonia MNIST (NN is shallow one). The accuracy of SVM increases as
  Slack SVM $\leq$ Feature Normalized SVM $\leq$ Proper kernel SVM $\leq$ Optimal Hyper-Parameter of Proper Kenal SVM
- **Metrics for Selecting hyper-Parameters** : In Grid Search for hyper-parameter search, we had following metrics to be used as a criterion to select the hyper-parameter from the Cross-validation Set:
  - Weighted Combination of AUC score and F1 score
  - AUC score
  - F1 score

  We selected F1 score to select the hyper-parameters because:
  - Real data will tend to have an imbalance between positive and negative samples. This imbalance has large effect on F1 score but not ROC/AUC.
  - So in the real world, the F1 score is used more since positive and negative samples are very uneven. The ROC/AUC curve does not reflect the performance of the classifier, but the PR curve can.

B. Intuition of Hyper-parameters

- $\mathcal{C}$: A large C gives low bias and high variance. Low bias because you penalize the cost of miss-clasification a lot. A small C gives higher bias and lower variance.
- Gaussian Kernel (γ): To "raise" the points we use the RBF kernel, gamma controls the shape of the "peaks" where we raise the points. A small gamma gives a pointed bump in the higher dimensions, a large gamma gives a softer, broader bump. So a small gamma will give **low bias and high variance while a large gamma will give higher bias and low variance**.
  The RBF Gamma parameter influences the distance of impact of a single training point. Low gamma values mean a broad similarity radius which results in more points being clustered together. In the case of high gamma values, points must be very close to each other in order to be included in the same category (or class). Models with very high gamma values appear to be over-fitting, thus. If gamma is high, the impact of c will become negligible. If gamma is weak, $\mathcal{C}$ affects the model just as it affects the linear model.

C. Multi-Class Classification

A single SVM does binary classification and can differentiate between two classes. So that, according to the two breakdown approaches, to classify data points from m classes data set:

- In the **One-to-Rest** approach, the classifier can use m SVMs. Each SVM would predict membership in one of the m classes.

- In the **One-to-One** approach, the classifier can use $\frac{m(m-1)}{2}$ SVMs.
  For final prediction for any input use the concept of majority voting along with the distance from the margin as its confidence criterion.

D. Major Adversities and Solutions

- **Grid Search** This aspect of SVM is the slowest and make SVM to train taking a lot of time. **Possible Solution:** We should combine **Random Search** and **Grid Search** in a mutually complementing manner.
  We should do Random Search to discoverable favourable parameters and perform Grid Search for spot-checking combinations that are known to perform by Random Search Discovery.
- **SLOW:** Practically we have seen that it is possible for SVM to **train forever** at a single value of hyperparameter for a kernel.
  Solution: We can as a first resort increase the **tolerance** factor or **Max-Iterations** that governs our stopping criterion and might help **when training is not converging**.
- **Multi-Class** classification problem have too many SVMs to train.
  One-vs-All :
  - **Too much Computation:** To implement the OVA strategy, we require more training points which increases our computation.
  - **Problems becomes Unbalanced:** In BloodMNIST dataset, in which there are 8 classes from 0 to 7 and if we have 1000 points per class, then for any one of the SVM having two classes, one class will have 9000 points and other will have only 1000 data points, so our problem becomes unbalanced.
    SOLUTION :
    * Use the 3-sigma rule of the normal distribution: Fit data to a normal distribution and then subsampled accordingly so that class distribution is maintained.
    * Pick some data points randomly from the majority class.
    * Use a popular subsampling technique named SMOTE.
- **One-vs-All & One-vs-One SOLUTION:**
  Directed Acyclic Graph (DAG)
  - This approach is more hierarchical in nature and it tries to addresses the problems of the One vs One and One vs All approach.
  - This is a graphical approach in which we group the classes based on some logical grouping.
  - **Benefits:** Benefits of this approach includes a fewer number of SVM trains with respect to the OVA approach and it reduces the diversity from the majority class which is a problem of the OVA approach.
  - **Problem:** If we have given the dataset itself in the form of different groups (e.g, BloodMNIST image classification dataset) then we can directly apply this approach but if we don't give the groups, then the problem with this approach is of finding the logical grouping in the dataset i.e, we have to manually pick the logical grouping.
- **Breaking Ties :** The tie-breaking mechanism will create a non-convex decision boundary in the area where there the classes are tied and then then all inputs in that area won't be classified in one class. Having **tie breaking rule is costly as per classification time is concerned.**

E. SVR (Support Vector Regression) :

A penalty-free area is captured around the maximum-margin decision boundary, called the error tube, where errors are accepted; this is a consequence of the fact that it must learn to compute continuous outputs. The goal of SVR is to find a tube that is as small as possible, without compromising much in model complexity and training time. when performing a regression task, you want the regressed function to be somewhere in the middle of the samples. This makes Support Vector Machines a good fit for (linear, and if not linear using some kernel function with the kernel trick) regression problems: **using support vectors near the middle of our dataset**, it will regress a function that maps those inputs to outputs.

1) Epsilon-SVR and nu-SVR: There are in fact two types of Support Vector Regression: epsilon-based SVR (ϵ-SVR) and nu-SVR (ν-SVR). They differ by means of the control that they offer over the regression problem:
When using nu-SVR, we have control over the total number of support vectors used but not necessarily over the error that is acceptable (often yielding smaller but possibly worse models).
When using epsilon-SVR, we have control over the error that is acceptable but not necessarily over the number of support vectors used (often yielding better but large models).
As a rule of thumb, choose ν-SVR when we want **model to be trained quickly with possibly less accuracy** and ϵ-SVR when we want **best performance**.
Solving Multi-output Regression Regression is ideally formulated for **uni-output** regression model. To implement **multi-output** regression model, we will regress over each output and then combine the output to make a multi-output regression.

II. Experimentation

A. pneumoniamnist

Binary Classification

- Sigmoid Kernel
  Best Classifier Metrics:
  - Optimal C: 1
  - Optimal gamma: 0.01
  - Optimal f1: 0.7748091603053435
  - Optimal accuracy: 0.7748091603053435
    Observations:

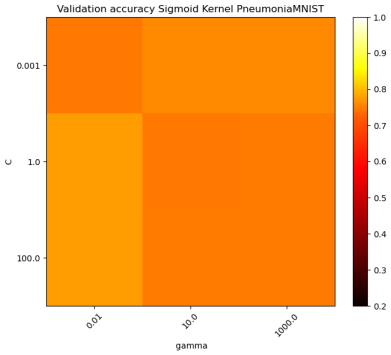


Fig. 1. Grid Search Gamma vs C

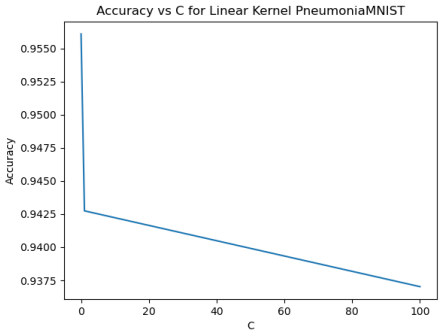


Fig. 3. Grid Search C

  - The accuracy of the classifier after C=1 is attained **does not** change further however we increase C value.
- Gaussian Kernel
  - Optimal C: 100
  - Optimal gamma: 0.01
  - Optimal f1: 0.8759541984732825
  - Optimal accuracy: 0.8759541984732825

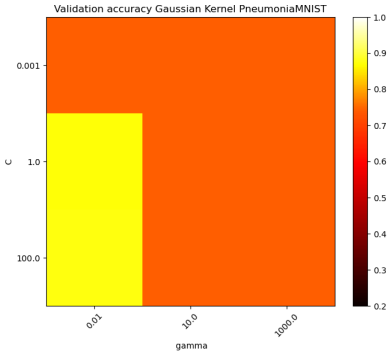


Fig. 2. Grid Search Gamma vs C

Observations

  - small gamma implies the class of this support vector will have influence on deciding the class of the vector x_i even if the distance between them is large.
  - C is big, SVM aims to reduce the number of misclassified examples due to a high penalty resulting in a decision boundary with a smaller margin.
  - So, the **optimal SVM has SMALLER margin and also the influence if the support vectors is at less distance.**
- Linear Kernel
  - Optimal C: 0.001
  - Optimal f1: 0.9561068702290076
  - Optimal accuracy: 0.9561068702290076

Observations

  - Accuracy is decreasing with increasing C.
- Polynomial Kernel
  - Optimal C: 0.001
  - Optimal f1: 0.7538167938931297
  - Optimal accuracy: 0.7538167938931297

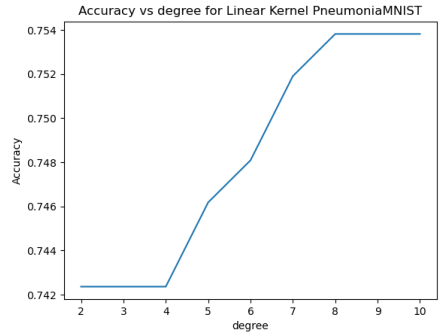


Fig. 4. Grid Search degree

  - Accuracy increases with **higher degree** but it saturates at degree 8.
  - Thus Using **Polynomial kernel, the best separation is in 8 degree transformation** and further that we can't get more separation on increasing degree.

BEST SVM CLASSIFIER

Best classifier is **Linear Classifier** with **C = 0.001**.

- AUC value of the best classifier: 0.9236357659434582
- Accuracy of the best classifier: 0.8509615384615384

1) Observations on Binary SVM:

- **Accuracy , precision, recall and f1 score are equal** across all hyperparameters. If we solve the system of equations, we find another solution: FP = FN. So, if the number of false positives is the same as the number of false negatives, all three metrics have identical values.

B. TIMIT Dataset

Getting bad performance on this dataset. **Gaussian kernel** gives 76 percentage accuracy. Using Sigmoid as kernel its giving **48** percentage accuracy. **Reason for less accuracy:** One

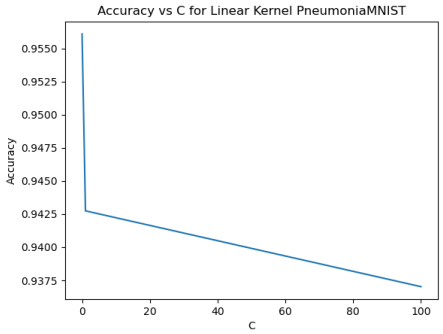


Fig. 5. Grid Search C

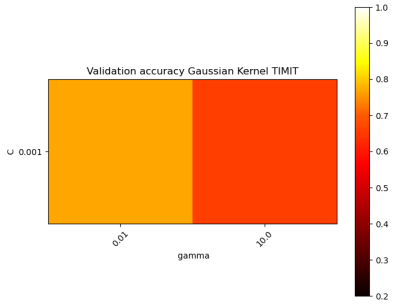


Fig. 7. Grid Search C vs gamma Gaussian Kernel

possible reason would be mutation of original in preprocessing audio data, padding and variable length make it difficult for classifiers to predict accurately.

- Sigmoid kernel
  Best Classifier Metrics
  - C = 1
  - gamma = 1000
  - Accuracy = 0.48701143946615827
    It seems Large Gamma is favoured means influence of a Support vector should be less and it saturates at very low C (less than 1). Smaller C, SVM optimizer is allowed at least some degree of freedom so as to meet the best hyperplane

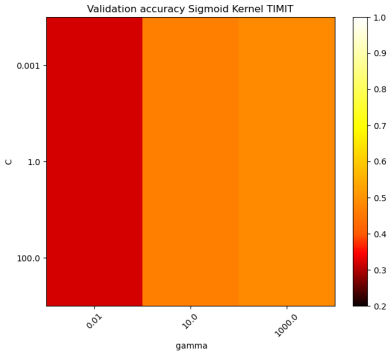


Fig. 6. Grid Search C vs Gamma Sigmoid Kernel

- Gaussian kernel
  - C = 0.001
  - gamma = 0.01
  - Accuracy = 0.7694530505243089
    Small gamma is required meaning large area influence of Support Vectors. And less C, means points can come inside margin or be in other side and still good learning.

C. bloodmnist

Multi-Class Classification

- Sigmoid Kernel
  Best Classifier Metrics:
  - Optimal C: 0.001
  - Optimal gamma: 0.01
  - Optimal f1: 0.4369158878504673
  - Optimal accuracy: 0.4369158878504673

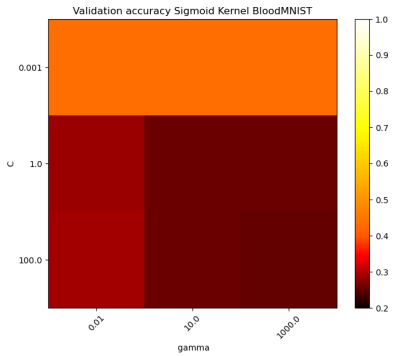


Fig. 8. Grid Search C vs gamma

- Gaussian Kernel
  - Optimal C: 1
  - Optimal gamma: 0.1
  - Optimal f1: 0.19626168224299062
  - Optimal accuracy: 0.19626168224299065
- Linear Kernel
  - Optimal C: 0.001
  - Optimal f1: 0.8399532710280374
  - Optimal accuracy: 0.8399532710280374
- Polynomial Kernel
  - Optimal C: 0.001
  - Optimal f1: 0.7538167938931297
  - Optimal accuracy: 0.7538167938931297

1) Observations on SVM Multi-class:

- In general the accuracy of SVM on this Multi-Class Classification problem is very low. Linear and Polynomial kernel have good accuracy (more than 0.75) in this dataset. Thus it can be **inferred that the MULTI-CLASS**

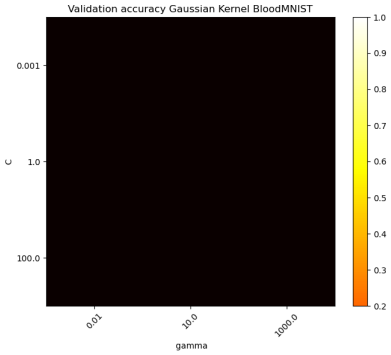


Fig. 9. Grid Search C vs gamma

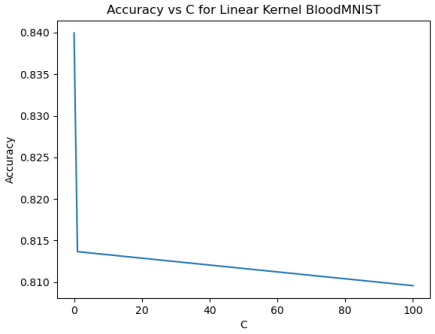


Fig. 10. Grid Search C

data is LINEARLY SEPERABLE ,i.e. there can exist m hyperplanes (m = 8) in our case that can classify our dataset with accuracy. Because accuracy of linear kernel is much grater than

D. Road-Sign-Detection

Multi-Value Regression

- Sigmoid Kernel Metrics
  - MSE :795.2466677539762
  - MAE :16.30288937638975
  - mIoU :0.4534773767125764
- Gaussian Kernel Metrics
  - MSE :54.10324854174014
  - MAE :4.733583922096336
  - mIoU : 0.7117262474452777
- Linear Kernel Metrics
  - MSE :19.517316150996685
  - MAE :2.9535892681016094
  - mIoU :0.8019789118814175
- Polynomial kernel Metrics
  - MSE :68.45260977655784
  - MAE :6.076780466969544
  - mIoU :0.6415305305037629

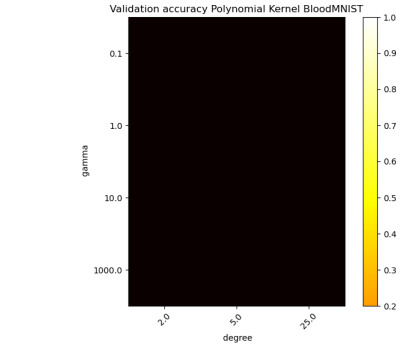


Fig. 11. Grid Search gamma vs degree

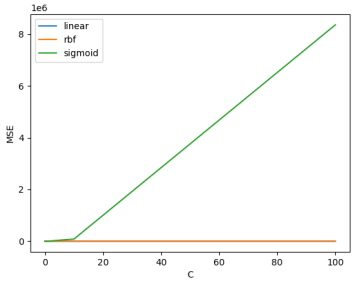


Fig. 12. MSE vs C different kernels

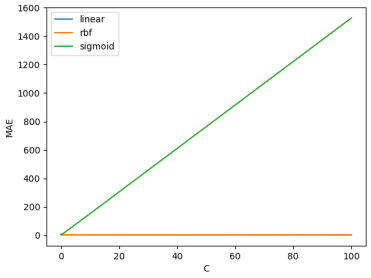


Fig. 13. MAE vs C different kernels

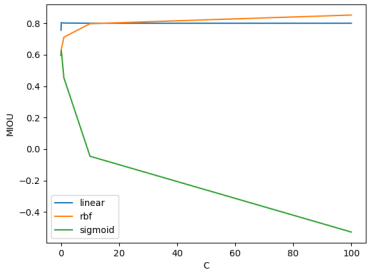


Fig. 14. MIOU vs C different kernels

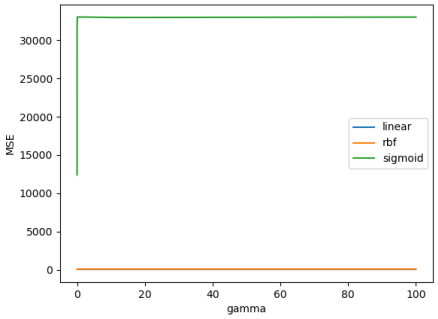


Fig. 15. MSE vs gamma different kernels

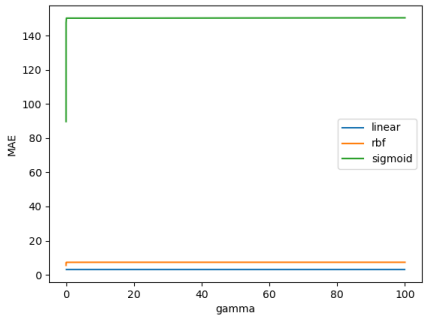


Fig. 16. MAE vs gamma different kernels

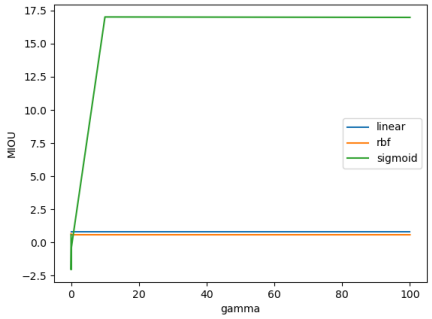


Fig. 17. MIOU vs gamma different kernels

Observations

- Linear Kernel is best performing here meaning again that there is much more chance of data to be linearly seperable than the non-linear transforms that we applied.
- **Gamma** change has effect only on **sigmoid kernel** and **linear** and **gaussian** kernel are **performing much same** in regression.
- Changing **C** has effect only on **sigmoid kernel** and **linear kernel** almost seems to have no effect **meaning at a very small value of C, it has saturated means however penalty we impose it's indifferent.**

E. General Observations

- In general **increasing** the degree of **polynomial kernel** will improve the accuracy of the model.
- The training time for **Polynomial kernel** is very large, its so large that for a grid search to a small grid too took **days to train**. Using a **12GB GPU** too in **24 hours** polynomial kernel could only iterate for **9 points** in the grid.
  Possible Cause: Might the convergence is not attained and the training is going on and on, though the same **tolerance and epsilon** were used for other kernels. Even though the accuracy of polynomial kernel is **better than sigmoid , gaussian in general** but still the convergence seems to be an issue. **Solution:** One solution might be to use **MAX-ITERATIONS** as stopping criteria along with tolerance.

III. Fisher Linear Discriminant

Fisher Linear Discriminant performs slightly worse than the best Support Vector Machine for classification on the PneumoniaMNIST dataset for binary classification. This is possibly due to how Fisher Linear Discriminant projects the data onto a lower dimension which might not fully capture the differences between some of the high-dimensional datapoints.

Accuracy	0.864
F-1 Score	0.900

IV. Backpropagation in Multilayer Perceptrons with various regularization schemes

Binary Classification for PneumoniaMNIST dataset with single hidden layer neural network gives the following results with different regularization schemes.

	Vanilla	L2	L1	Dropout
Test Accuracy	0.625	0.742	0.765	0.785
F1 Score	0.769	0.852	0.872	0.874
AUC	0.776	0.768	0.774	0.785

From the table, we observe that regularization helps improve test accuracy as expected by reducing overfitting. If the dropout rate is too high, this can slow the convergence rate of the model.
Multiclass classification by training the single hidden layer neural network on the BloodMNIST dataset gives a test accuracy of 0.724.

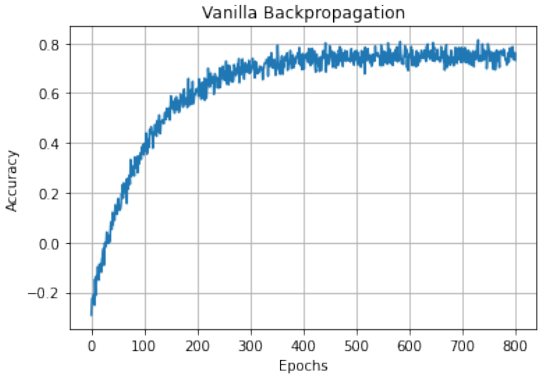
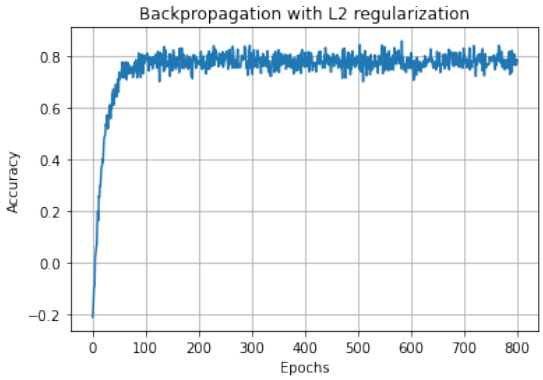
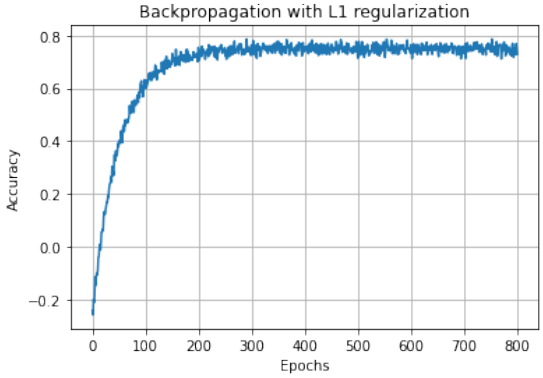


Fig. 18. Accuracy vs Epochs with vanilla backpropagation



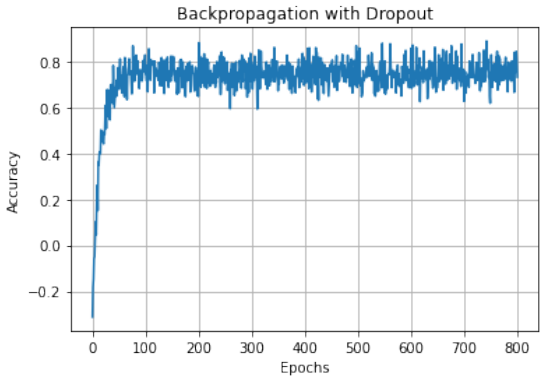
[h]

Fig. 19. Accuracy vs Epochs with backpropagation and L2 regularization



[h]

Fig. 20. Accuracy vs Epochs with backpropagation and L1 regularization



[h]

Fig. 21. Accuracy vs Epochs with backpropagation and Dropout

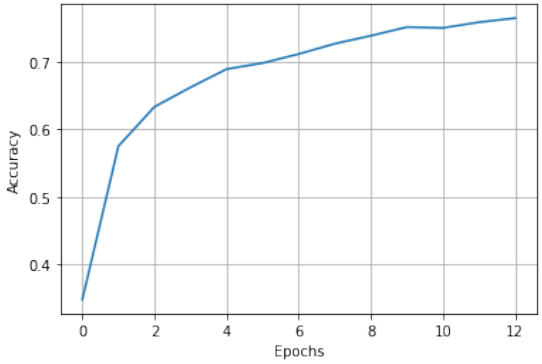


Fig. 22. Accuracy vs Epochs plot

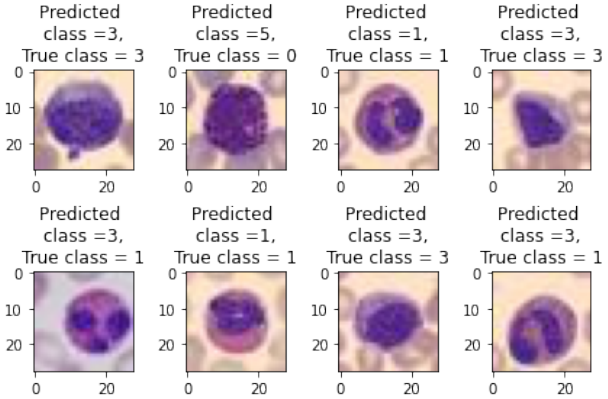
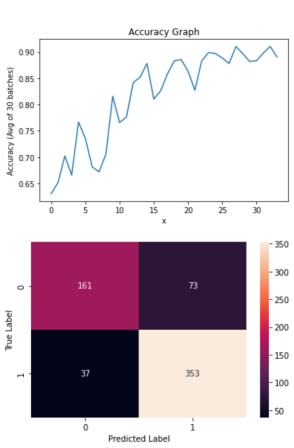
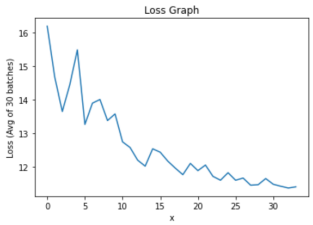


Fig. 23. Test Predictions

V. Convolutional Neural Network from scratch

We implemented CNN with 3 convolutional layers with sigmoid activations and one fully connected layer with stride = 2, depth = 4.



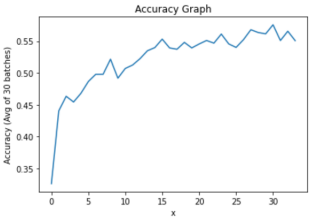


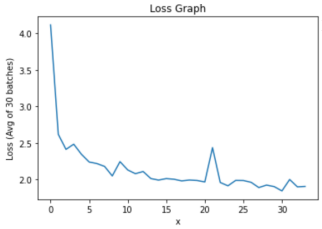
	f1-score	precision	recall	support
0	0.745370	0.813131	0.688034	234.000000
1	0.865196	0.828638	0.905128	390.000000
accuracy	0.823718	0.823718	0.823718	0.823718
macro avg	0.805283	0.820885	0.796581	624.000000
weighted avg	0.820261	0.822823	0.823718	624.000000

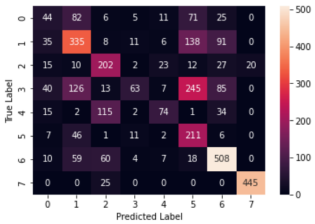
TABLE I
CNN ACCURACY,LOSS,F1 SCORE ON PNEUMONIAMNIST

pneumoniamnist	Train	Valid	Test
Accuracy	90.85	90.08	82.37

The above results are for 1000 iterations. The accuracy increase saturated at around 90% on training dataset.







	f1-score	precision	recall	support
0	0.214634	0.265060	0.180328	244.000000
1	0.521807	0.507576	0.536859	624.000000
2	0.545209	0.469767	0.649518	311.000000
3	0.186115	0.642857	0.108808	579.000000
4	0.396783	0.569231	0.304527	243.000000
5	0.430612	0.303161	0.742958	284.000000
6	0.704577	0.654639	0.762763	666.000000
7	0.951872	0.956989	0.946809	470.000000
accuracy	0.550132	0.550132	0.550132	0.550132
macro avg	0.493951	0.546160	0.529071	3421.000000
weighted avg	0.523426	0.587521	0.550132	3421.000000

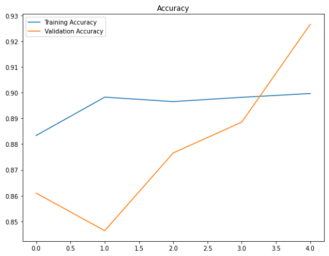
TABLE II
CNN ACCURACY,LOSS,F1 SCORE ON BLOODMNIST

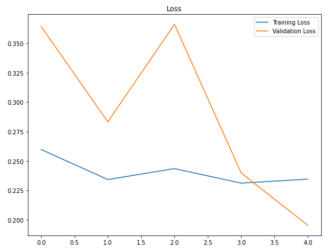
bloodmnist	Train	Valid	Test
Accuracy	55.32	55.78	55.01

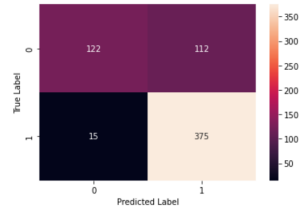
one can observe classes 0,3,4 are predicted poorly. For several classification algorithms same pattern was observed. Increasing the number of neurons or depth of CNN layers did not improve accuracy much. The above results are also for 1000 iterations.

VI. Pretrained CNN: Resnet and Vgg16

The last classification layer of resnet and vgg16 are modified. Here transfer learning is used. All the convolutional layers are frozen.







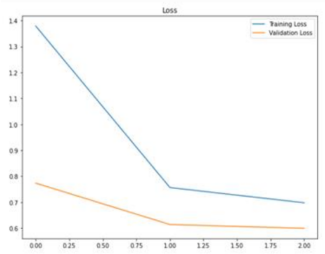
	f1-score	precision	recall	support
0	0.657682	0.890511	0.521368	234.000000
1	0.855188	0.770021	0.961538	390.000000
accuracy	0.796474	0.796474	0.796474	0.796474
macro avg	0.756435	0.830266	0.741453	624.000000
weighted avg	0.781123	0.815204	0.796474	624.000000

TABLE III
RESNET50 ON PNEUMONIAMNIST

pneumoniamnist	Train	Valid	Test
Accuracy	92.8	90	79.51

The accuracies with resnet50 are similar to CNN we implemented from scratch. But the number of iterations used here is just 3.

```
Test Loss:0.630
Test Accuracy:76.50%
Train Loss: 0.698
Val Loss: 0.599
Train Accuracy: 75.17%
Val Accuracy: 78.16%
```

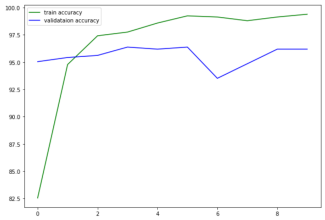


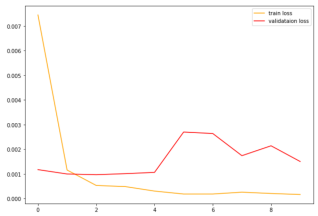
	f1-score	precision	recall	support
0	0.335526	0.850000	0.209016	244.000000
1	0.857661	0.987474	0.758013	624.000000
2	0.800705	0.886719	0.729904	311.000000
3	0.647658	0.533557	0.823834	579.000000
4	0.733179	0.840426	0.650206	243.000000
5	0.582222	0.789157	0.461268	284.000000
6	0.823301	0.723549	0.954955	666.000000
7	0.957688	0.929860	0.987234	470.000000
accuracy	0.764981	0.764981	0.764981	0.764981
macro avg	0.717243	0.817593	0.696804	3421.000000
weighted avg	0.755045	0.805479	0.764981	3421.000000

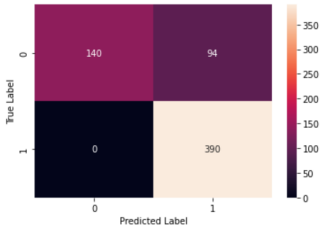
TABLE IV
RESNET50 ON BLOODMNIST

bloodmnist	Train	Valid	Test
Accuracy	75.17	78.16	76.5

Resnet50 performed better than the CNN we implemented from scratch.





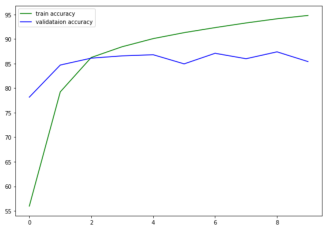


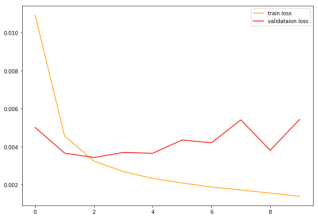
	f1-score	precision	recall	support
0	0.748663	1.000000	0.598291	234.000000
1	0.892449	0.805785	1.000000	390.000000
accuracy	0.849359	0.849359	0.849359	0.849359
macro avg	0.820556	0.902893	0.799145	624.000000
weighted avg	0.838529	0.878616	0.849359	624.000000

TABLE V
VGG16 ON PNEUMONIAMNIST

pneumoniamnist	Train	Valid	Test
Accuracy	99.17	96.56	86.14

The number of iterations is 10. Again very good accuracy on train,poor on test. we added L2 reg and got 92% on test data.





	f1-score	precision	recall	support
0	0.781250	0.857843	0.717213	244.000000
1	0.877408	0.967181	0.802885	624.000000
2	0.828986	0.754617	0.919614	311.000000
3	0.763432	0.712575	0.822107	579.000000
4	0.848948	0.792857	0.913580	243.000000
5	0.758621	0.894737	0.658451	284.000000
6	0.902941	0.884726	0.921922	666.000000
7	0.979766	0.980810	0.978723	470.000000
accuracy	0.853844	0.853844	0.853844	0.853844
macro avg	0.842669	0.855668	0.841812	3421.000000
weighted avg	0.854008	0.864390	0.853844	3421.000000

TABLE VI
VGG16 ON BLOODMNIST

bloodmnist	Train	Valid	Test
Accuracy	85.2	83	82.26

The number of iterations is 10. Here increasing the number of iterations did not lead to improved accuracy on train and valid data.

VII. LSTM ON DARPA-TIMIT DATA

The audio data is preprocessed successfully. we were able to extract the vectors for phonemes and label them corresponding to presence of vowel or not. Due to memory requirement error, we were not able to run the code on any of our machines. But we did write the code for running it.