6.867: Homework 3

Anonymous authors

***Abstract*— This is Machine Learning homework 3. Topics include neural networks, conv nets, and LDA. All work has been done in Python.**

I. Neural Networks

In this problem, we implement a simple neural network in Python. The network is trained with backpropogation and stochastic gradient descent.

A. Part 1

In this problem, we are mainly interested in classification tasks. Thus, we pass the output of the final hidden layer through a softmax layer to generate a probability (outputs are positive and sum to 1) vector where each element k is the probability of the input being in class k.

In general, the training objective is to minimize loss, so at each training step, we evaluate the loss function and compute its gradient, and then adjust the weights and biases in a direction to decrease loss according to the learning rate. According to the backpropogation algorithm, the derivative of loss with respect to final activation is $\delta^L = Diag[f'(z)]\nabla_a loss$. For this network with softmax output activation and cross-entropy loss function, we can compute $\delta^L = p(x) - y$, where p(x) is the predicted output for a particular training data point, x in class y (where p is also a function of the network weights, and other parameters).

B. Part 2

We initialize biases to zero, but we should not initialize weights to zero as well. This is because with weights of zero, every element of the hidden layer would be the same, since the hidden layer is based on input layer and weights. If every element of the hidden layer is the same, there is no way to break the symmetry, because backpropogation would consider every weight equally responsible for the network's loss.

Therefore, according to the Xavier Initialization, we can initialize weights to be a zero-mean Gaussian, where the variance is a key parameter. If variance is too low, all the weights will be close to zero and we will be faced with the same problem described above. If variance is too high, training the network will take a long time to undo the bias imposed during initialization. Even worse, it's possible the network will get stuck in a local minimum that is far from the global optimum. This tradeoff of initializing enough randomness to avoid symmetry while not imposing too much bias, is an important piece of neural network training.

C. Part 3

To add weight regularization, the objective function would look like:

$$J(w) = l(w) + \lambda(||w^{(1)}||_F^2 + ||w^{(2)}||_F^2) \quad (1)$$

The only change to the pseudo-code from the lecture notes would be an updated gradient term during the output layer's backpropogation step.

The effect of regularization on the network would be the same as regularization in general. The λ term penalizes weight matrix size, so increasing λ causes weight matrices to shrink in size (lower model complexity), which decreases training accuracy (less likely to overfit), but should increase test accuracy, up to a point where λ is large enough that the weight term dominates the loss term in the objective function.

D. Part 4

Next, we use the network for binary classification on the 2D datasets from the previous homework assignment.

Dataset	1 HL (sm)		1 HL (lg)		2 HL (sm)		2 HL (lg)	
	Train	Test	Train	Test	Train	Test	Train	Test
1	100.0	99.5	99.8	99.5	100.0	100.0	100.0	100.0
2	94.5	91.5	95.25	93.0	94.5	91.5	95.0	94.5
3	98.5	95.0	98.25	96.5	98.25	96.0	98.0	96.0
4	96.25	95.25	97.0	94.0	95.25	93.75	93.75	92.50

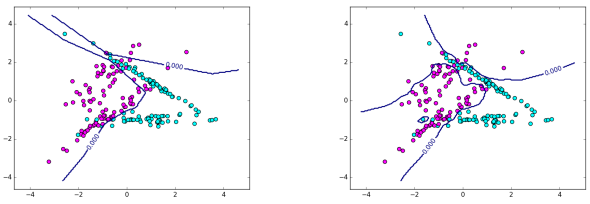
TABLE I: Accuracy of NN on 2D datasets.

The results in Table I show the training and test accuracy for four 2D datasets, with four different network architectures (1 or 2 hidden layers (HL) and small or large number of hidden nodes). The accuracies in general are quite high ($> 90\%$). As expected, test accuracy is worse than training accuracy for almost all cases. Training was terminated after the change in validation loss from the previous epoch was small, which is one method to protect against overfitting.

An example of how the decision boundary changes with architecture is shown in Fig. 1. The simpler model Fig. 1a leads to a simpler decision boundary (somewhat more linear) than the complex model Fig. 1b

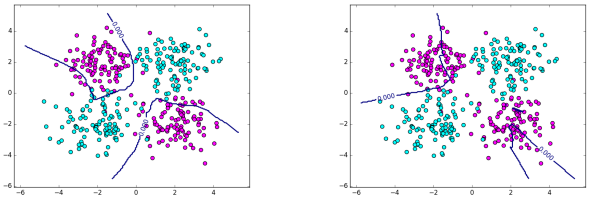
A thorough hyperparameter search could be done for this problem to find optimal learning rate, but a value around $10^{-3} - 10^{-4}$ worked well for most scenarios.

It is worth noting the effect of initialization, as depicted in Fig. 2. Since the weights are initialized randomly, sometimes the network would not converge to a good decision boundary (Fig. 2a vs. Fig. 2b), and the boundaries varied significantly from run-to-run. This makes it less meaningful to compare architectures with results from single runs.



(a) 1 Hidden Layer (small) (b) 2 Hidden Layers (large)

Fig. 1: The low-complexity model (left) with one small hidden layer (10 nodes) has a simpler decision boundary than the high-complexity model (right) with two large hidden layers (100 nodes each).



(a) Good initialization (b) Bad initialization

Fig. 2: The random initialization has a large effect on final decision boundary. On the left, the boundary is pretty good. The boundary on the right is much worse even with all the same hyperparameters and model structure.

In general, the NN classifier works best on the (easy) linearly separable dataset ($\approx 100\%$ on 1), and still does very well on the other datasets with non-linear decision boundaries.

Compared to the classifiers from the previous homework assignment, the NN of course outperforms Linear SVM and LR on the non-linearly separable data, and does approximately as well on the linearly separable ones. The performance is also comparable/better than Guassian RBF SVM and training time seems much faster (though speed depends on quality of implementation).

E. Part 5

Finally, we use the network for digit classification on the MNIST dataset. Results are shown in Fig. 3, where the training accuracy is 98% and validation accuracy is 86%. This performance is very good, and training takes < 1 min on a CPU.

The model used was a NN with a single hidden layer with 100 elements. The accuracy was lower with a smaller number of nodes.

Learning rate was 10^{-2}, which was chosen by trying multiple values and choosing the one with best validation accuracy. When learning rate was too high (> 0.1), learning diverged and some weights became NaN. Accuracy was similar with learning rate $10^{-2} - 10^{-4}$ When learning rate was too low ($< 10^{-6}$), learning was extremely slow ($< 30\%$ accuracy after 100 epochs).

Normalization is imporant for this dataset, as our network was unable to converge without first normalizing the data to be within [-1,1].

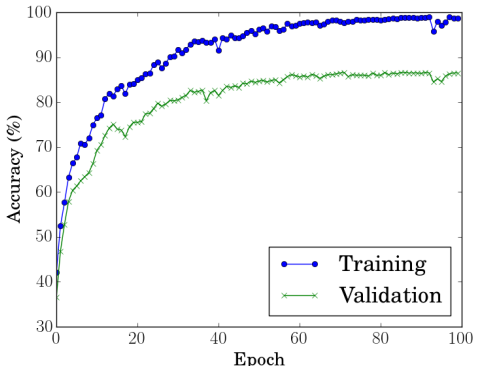


Fig. 3: For digit classification on MNIST dataset, training and validation accuracy are plotted over 100 epochs. Training accuracy is better than validation accuracy (as expected), and they level out around 98% and 86%, respectively.

II. Convolutional Neural Network (CNN)

In this section, we consider the Convolutional Neural Network (CNN) to perform artist identification on paintings.

A. Part 1

In a convolutional filter, if the first layer applies a 5x5 patch to the image to generate feature Z_1, and the second layer applies a 3x3 patch to feature Z_1 to generate feature Z_2, the receptive field of Z_2 (or dimensions of image that affect the node) is 7x7. That is, a window of 49 neighboring pixels in the original image affects a single node at the output of the filter. This allows the network to learn spatial features from the original image. If the conv net becomes deeper (more layers), the network can use larger and more complex combinations of features/regions of the image.

B. Part 2

We are provided with a conv net (conv.py). The layers in this network are: 2 convolutional layers, 1 fully-connected layer with ReLU activation, and 1 fully-connected layer with linear activation. The output is the maximum logit of the final fully-connected layer. The hidden layer activation function is ReLU. The loss function is softmax cross entropy with logits. Loss is minimized with Gradient Descent (with a tunable learning rate parameter).

The provided network took about 45 seconds to train on a Macbook CPU. After 1500 steps, the training accuracy is 87.4%, and the validation accuracy is 57.5%. These numbers suggest overfitting, because the model does not generalize to unseen data very well.

C. Part 3

Next, we try two common techniques to improve CNN performance. The first is early stopping. Early stopping is when training is stopped after validation accuracy levels out, before it starts to decrease, to avoid overfitting. In this scenario, Fig. 4 shows that early stopping could be applied around step 1,000, because at this point, validation accuracy levels off but training accuracy continues to increase. That

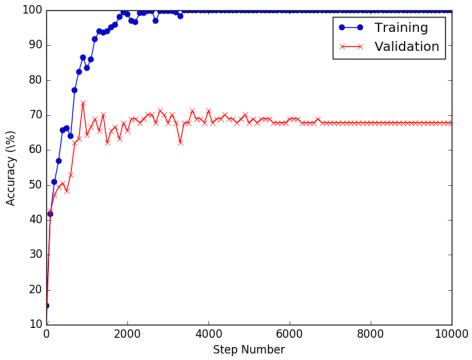


Fig. 4: Training and validation accuracy are plotted over 10,000 training steps. Validation performance levels off after 1,000 training steps, but training accuracy continues to increase. This behavior looks like overfitting, so training should be stopped early around step 1,000.

behavior is a good indication of overfitting, because the model is improving on data it has seen, but not improving on data it has never seen. In addition to reducing overfitting and reducing model complexity, early stopping also has the benefit of shorter training time.

The pooling layers seem to make no difference on performance.

D. Part 4

Finally, we use the network on a transformed version of the dataset. Results in Table II show that the performance is not the same for every transormation type. For example, the original CNN only gets 10% accuracy on inverted images, compared to 66.7% on low contrast images (all relative to a 70.1% accuracy on normal images).

Table II's columns show the original CNN, a CNN with Early Stopping and one with both Early Stopping and Max Pooling. The results are not significantly different. This aligns with the earlier observation that max pooling does not affect this problem much. Early Stopping surprisingly didn't have much effect either.

Transformation	Regular Val Acc (%)	ES Val Acc (%)	ES & Pooling Val Acc (%)
Normal	65.5	70.1	69.0
Translated	33.3	29.9	35.6
Brightened	46.0	47.1	43.7
Darkened	47.1	49.4	46.0
High Contrast	62.1	63.2	58.6
Low Contrast	66.7	66.7	67.8
Flipped	42.5	41.4	44.8
Inverted	8.0	10.3	12.6

TABLE II: Accuracy of CNNs on transformed dataset, with Early Stopping (ES) and Max pooling.

III. LDA

In this section, we use latent Dirichlet allocation (LDA) to perform topic modeling on a corpus of text.

A. Part 1

The first part is very straightforward. We installed the Mallet implementation of LDA and ran it on the corpus of text of 19,997 news articles in 20 categories from the Usenet newsgroups dataset.

B. Part 2

During training, we asked the model to discover 100 topics, which include several keywords related to the topic. Two example topics are:

- health cancer disease medical aids hiv number research patients page april volume newsletter study hicnet drug children risk years age
- image data graphics package images program processing line format ray code objects analysis points center it's postscript sgi formats ftp

The first topic seems to be related to medicine/health and the second is related to image processing/computers. The topics aren't extremely specific, but each component seems roughly connected. However, if a human were to develop a list of keywords related to health, it probably would not include words like "newsletter" or "april", so there are some strange components of the topic lists.

Of the 20 article categories, we chose to analyze two in more detail: "sci.med" and "rec.sport.hockey", which include articles about Medicine and Hockey, respectively. The top topics related to Medicine were:

1) article pain doctor writes gordon banks patients good surgery treatment disease patient medicine blood intellect chronic skepticism surrender shameful chastity
2) food msg eat diet article people fat writes foods chinese body eating effects taste reaction i'm effect apr steve meat
3) candida yeast medical patients medicine symptoms treatment infection quack infections body steve doctors article clinical disease writes studies nutrition jon

and the top topics related to Hockey were:

1) period game play goal pts power shots puck blues goals flyers leafs team detroit win wings scoring penalty season goalie
2) game games team win espn won year lost baseball fans boston division series hockey teams chicago pittsburgh fan season cubs
3) team game players hockey writes play player article roger games apr league baseball good time year teams bob nhl ice

At first glance, these topics seem appropriate for the categories. Medicine's top topics roughly relate to medicine, nutrition, and disease, whereas hockey's top topics include rules of the game and team names. Upon closer look, there are some bizarre elements of the chosen topics, like "i'm" and "jon" for medicine, and "bob" and "apr" for hockey. But, of the 60 keywords for hockey, only about 3 of them are ridiculous whereas the other 57 are very "on-topic."

C. Part 3

Then, we lowered the number of topics down to 20. We hypothesized that this would lead to broader topics, because the same number of articles and words now would need to be binned into a smaller number of categories. Interestingly, the number of article classes is now equal to the number of topics.

Two of the resulting topics were:

- writes article people don't objective moral truth true question evidence god point good i'm morality it's religion exist claim reason
- israel president jews israeli writes jewish war article people arab state peace stephanopoulos arabs world policy land years country american

These topics seem slightly more broad than before, but it's hard to measure/quantify this concept.

The top topics most related to Medicine were:

1) article medical writes disease study health patients food cancer number treatment men msg medicine research doctor sex drug aids studies
2) don't it's people i'm good make time writes article that's i've things thing can't doesn't lot point you're find work
3) writes article theory science fact universe evolution bill book university black water time don't apr michael physical scientific hole books

and the top topics most related to Hockey were:

1) game team year games writes players article play hockey season baseball win good league player apr teams hit fans time
2) don't it's people i'm good make time writes article that's i've things thing can't doesn't lot point you're find work
3) san period convention pit party institute pts political adl det van april bos cal los committee college state vote karl

The first topic for each article category still seems appropriate, but the second and third are not very related. This result matches with our original intuition that that there would be fewer specific topics to pair with each article type as the number of topics decreases.