למידה עמוקה – תרגיל בית 1

מגישים:

תמר מנו 206295917
אורגד שלישמן 303142897

החלק התיכנותי מצורף בתיקיית הקולאב המשותפת:

https://drive.google.com/drive/u/1/folders/1_zZbowfJLw36po3paxEO3kp10LfsSYOf

חלק תיאורטי

1) Suppose you have an MLP composed of an input of size 10, followed by one hidden layer with output of size 50, and finally one output layer with 3 output neurons. All artificial neurons use the ReLU activation function. The batch size used is m.
   a. The shape of X is: $m \times 10$
   b. The shape of W_h is: 10×50, the shape of b_h is: 50
   c. The shape of W_o is: 50×3, the shape of b_o is: 3
   d. The shape of Y is: $m \times 3$
   e. The output equation is: $Y = ReLu(ReLu(XW_h + b_h)W_o + b_o)$

2) Consider a CNN composed of three convolutional layers, each with 3 × 3 kernels, a stride of 2, and SAME padding. The lowest layer outputs 100 feature maps, the middle one outputs 200, and the top one outputs 400. The input images are RGB images of 200 × 300 pixels. What is the total number of parameters in the CNN? Explain your answer.

בשביל לחשב את מספר הפרמטרים, נסתכל עבור כל שכבה מה מספר הפרמטרים בכניסה לשכבה זו, נכפיל בגרעין ונוסיף bias.
כל גרעיני הקונבולוציה יהיו בגודל 3×3 .

- כניסה ושכבה ראשונה:

נתון שהקלט לרשת הוא תמונה של RGB, לכן, יהיה עומק של 3. נוסיף bias ונכפיל במספר המוצאים של השכבה הבאה:

$3 * 3 * 3 + 1 = 28$
28*100 = 2,800

- שכבה שנייה:

$100 * 3 * 3 + 1 = 901$

901*200 = 180,200

- שכבה שלישית:

$200 * 3 * 3 + 1 = 1{,}801$

1,801*400 = 720,400

- סה"כ:

2,800 + 180,200 + 720,400 = 903,400

3)

a. $\frac{df}{d\gamma} = \sum_{i=1}^{m} \frac{df}{dy_i}\frac{dy_i}{d\gamma} = \sum_{i=1}^{m} \frac{df}{dy_i}\hat{x}_i$

b. $\frac{df}{d\beta} = \sum_{i=1}^{m} \frac{df}{dy_i}\frac{dy_i}{d\beta} = \sum_{i=1}^{m} \frac{df}{dy_i}$

c. $\frac{df}{d\hat{x}_i} = \frac{df}{dy_i}\frac{dy_i}{d\hat{x}_i} = \frac{df}{dy_i}\gamma$

d. $\frac{df}{d\sigma^2} = \sum_{i=1}^{m} \frac{df}{d\hat{x}_i}\frac{d\hat{x}_i}{d\sigma^2} = -0.5\sum_{i=1}^{m} \frac{df}{d\hat{x}_i}\frac{(x_i-\mu)}{(\sigma^2+\epsilon)^{1.5}}$

e. $\frac{df}{d\mu} = \frac{df}{d\hat{x}_i}\frac{d\hat{x}_i}{d\mu} + \frac{df}{d\sigma^2}\frac{d\sigma^2}{d\mu} = -\frac{df}{d\hat{x}_i}\frac{1}{\sqrt{\sigma^2+\epsilon}} - \frac{2}{m}\sum_{i=1}^{m}(x_i-\mu) = \sum_{i=1}^{m} \frac{df}{d\hat{x}_i}\frac{-1}{\sqrt{\sigma^2+\epsilon}}$

f. $\frac{df}{dx_i} = \frac{df}{d\hat{x}_i}\frac{d\hat{x}_i}{dx_i} + \frac{df}{d\sigma^2}\frac{d\sigma^2}{dx_i} + \frac{df}{d\mu}\frac{d\mu}{dx_i} = \frac{df}{d\hat{x}_i}\frac{1}{\sqrt{\sigma^2+\epsilon}} + \frac{-1}{m\sqrt{\sigma^2+\epsilon}}\sum_{j=1}^{m}\frac{df}{d\hat{x}_j} -$

$\sum_{j=1}^{m} \frac{df}{d\hat{x}_j}\frac{(x_j-\mu)}{(\sigma^2+\epsilon)^{1.5}}\frac{(x_i-\mu)}{m} = \frac{1}{m\sqrt{\sigma^2+\epsilon}}\left(m\frac{df}{d\hat{x}_i} - \sum_{j=1}^{m}\frac{df}{d\hat{x}_j} - \hat{x}_i\sum_{j=1}^{m}\frac{df}{d\hat{x}_j}\hat{x}_j\right)$

Q4 – Practical

Implement the Lenet5 network over the FashionMNIST data set.

Compare the usage of the following techniques with Lenet5:

- Dropout (at the hidden layer)
- Weight Decay (also known as l2 loss)
- Batch Normalization

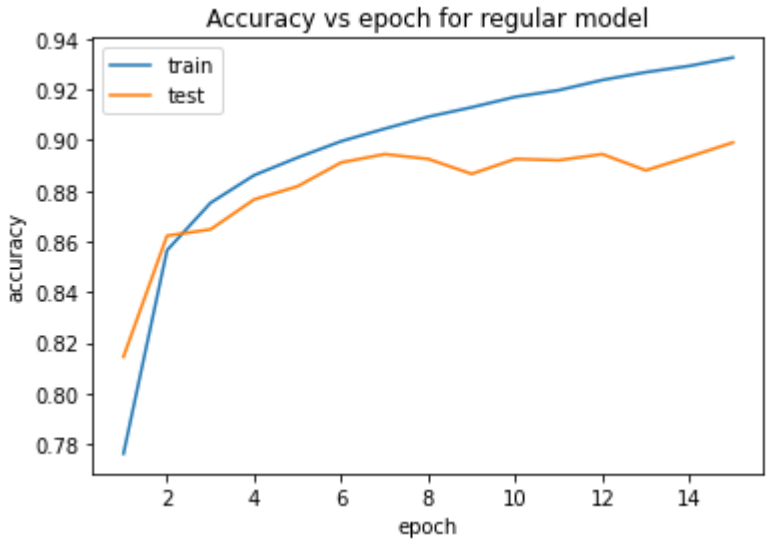
First, we ran with 50 epochs and we could see we're getting overfitting (See results and Table 1).

Table 1 – overfitting when running with 50 epochs:

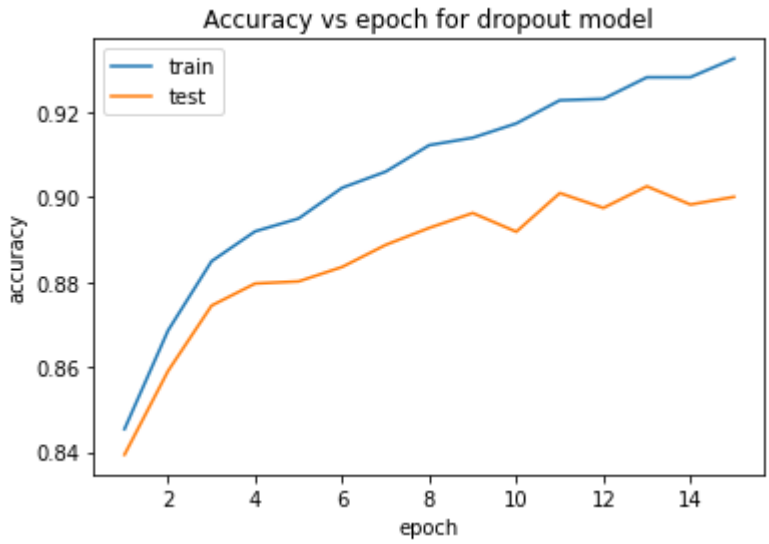
MODEL	TRAIN	TEST
Regular Model	0.9785666666666667	0.9024
Dropout Model	0.9685166666666667	0.906
Weight Decay Model	0.9375333333333333	0.9071
Batch-norm Model	0.9777166666666667	0.9019

So we've decided to work with 15 epochs, and here is what we got:

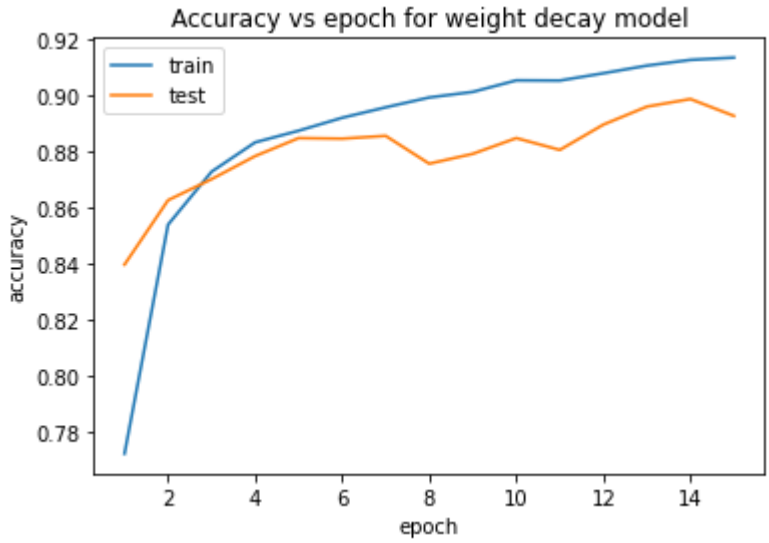
a) Convergence graph (15 epochs):
   a. Lenet regular model



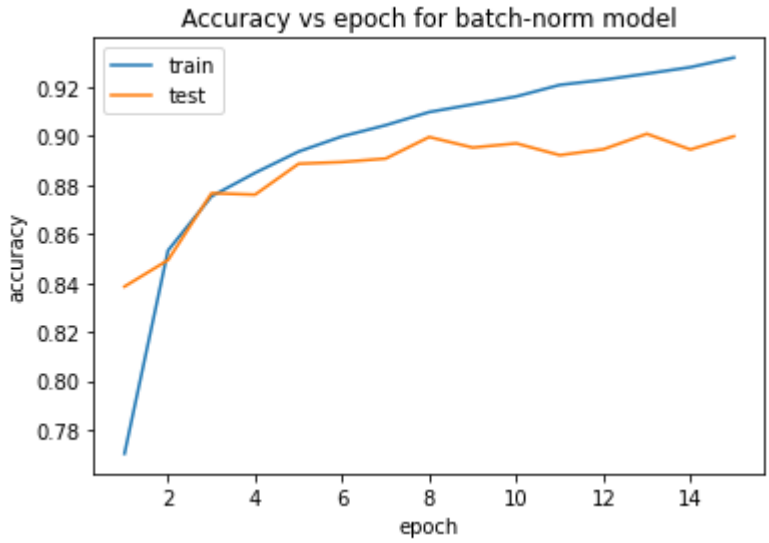
b. Dropout model



c. Weight decay model



d. Batch norm model



b) We made sure that train accuracy, for "dropout" model, was measured without dropout.

c) Accuracy table (15 epochs):

MODEL	TRAIN	TEST
Regular Model	0.9172	0.8966
Dropout Model	0.91575	0.8963
Weight Decay Model	0.9056666666666666	0.8923
Batch-norm Model	0.9186	0.8992

d) Conclusions:

We've seen that by changing number of epochs from 50 to 15, we got pretty much the same results, which means, more epochs, does not contribute to improving the net and our assumption regarding overfitting when using 50 epochs was correct.

We were working with relatively small net, hence, we don't have to use a big number of epochs. We assume that if the net was bigger, the different variations of it could have larger impact.

We can also see that all versions of the net are showing similar results, when BN is showing slightly better than the other ones.