Neural Networks QA

Q1

In page 390

> The PPR model (11.1) is very general, since the operation of forming nonlinear functions of linear combinations generates a surprisingly large class of models. For example, the product $X_1 \cdot X_2$ can be written as $[(X_1 + X_2)^2 - (X_1 - X_2)^2]/4$, and higher-order products can be represented similarly.

we write $\frac{(X_1+X_2)^2-(X_1-X_2)^2}{4}$ as the form of $f(X) = \sum_{m=1}^{M} g_m\left(\omega_m^T X\right)$ because $(X_1 + X_2)^2$ and $(X_1 - X_2)^2$ are ridge function.

I want to know how to prove they are ridge function and how to know whether a function is ridge function.

A: Ridge finctions are functions that can be written in form of $g\left(\omega^T X\right)$, and $\frac{(X_1+X_2)^2-(X_1-X_2)^2}{4}$ can be written in $f(X) = \sum_{m=1}^{2} g_m\left(\omega_m^T X\right)$, where $g_1(V) = \frac{V^2}{4},\ V = \omega_1^T X,\ \omega_1 = (1,1)^T$,
$g_2(V) = -\frac{V^2}{4},\ V = \omega_2^T X,\ \omega_2 = (1,-1)^T$.

Q2

> In Figure 11.7 we repeated the experiment for the sum of sigmoids model, with no weight decay in the left panel, and stronger weight decay ($\lambda = 0.1$) in the right panel. With no weight decay, overfitting becomes even more severe for larger numbers of hidden units. The weight decay value $\lambda = 0.1$ produces good results for all numbers of hidden units, and there does not appear to be overfitting as the number of units increase. Finally, Figure 11.8 shows the test error for a ten hidden unit network, varying the weight decay parameter over a wide range. The value 0.1 is approximately optimal.

Q1. Why we could choose lambda = 0.1 here despite the fact that the learning rate does not satisfy the convergence condition of the learning result. Is this kind of learning rate often being chosen or this is the special case here.

A: This λ is not a learning rate. It's the parameter in the penalized error function:

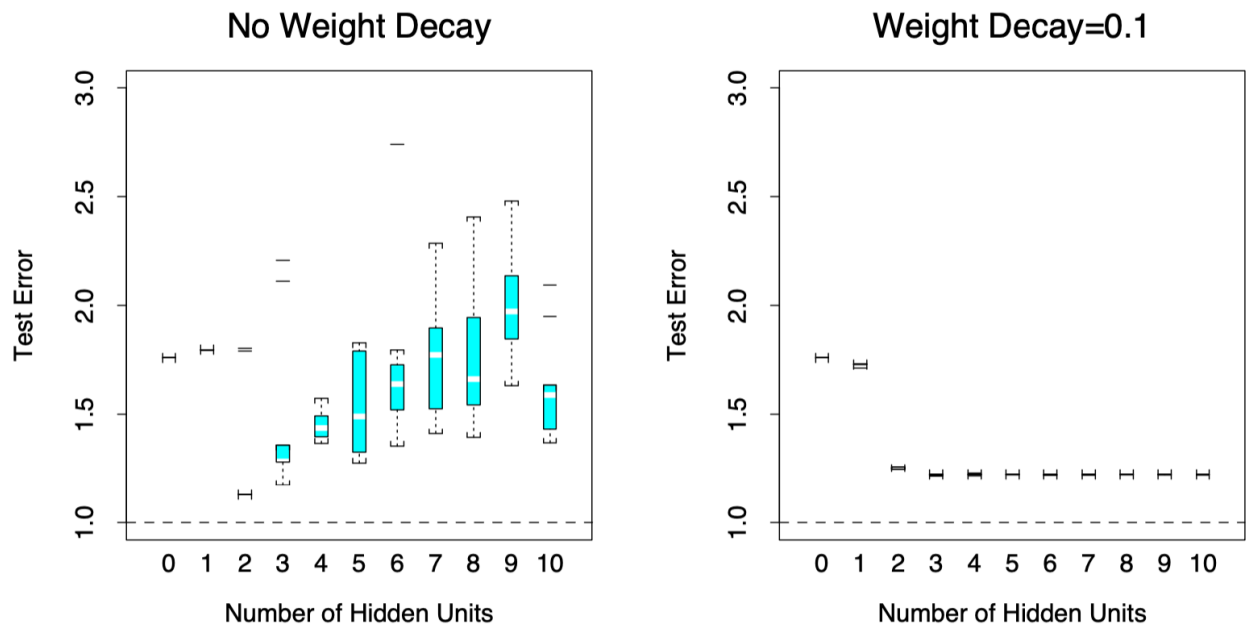
$$\text{function } R(\theta) + \lambda J(\theta), \text{ where}$$

$$J(\theta) = \sum_{km} \beta_{km}^2 + \sum_{m\ell} \alpha_{m\ell}^2$$

Q3

Q2. In Figures 11.7 and 11.8, why should we cross-validate the parameter that is less sensitive? What is the purpose?

A:



It's sensitive.

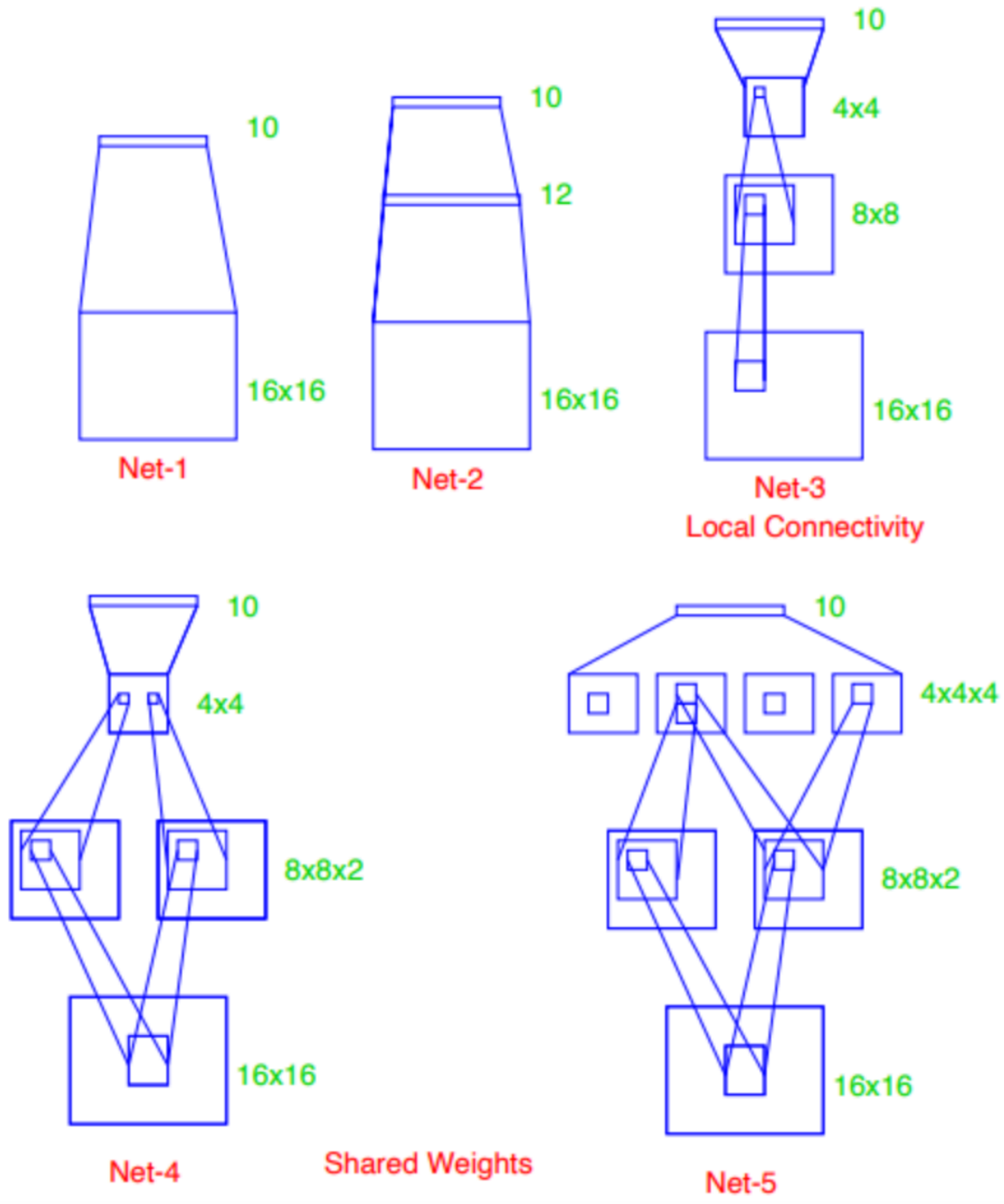
Q4

For Neural Network, there is a statement for `example 11.7` that:

> Net-3 uses local connectivity: this means that each hidden unit is connected to only a small patch of units in the layer below. In the first hidden layer (an 8×8 array), each unit takes inputs from a 3×3 patch of the input layer; for units in the first hidden layer that are one unit apart, their receptive fields overlap by one row or column, and hence are two pixels apart. In

the second hidden layer, inputs are from a 5 × 5 patch, and again units that are one unit apart have receptive fields that are two units apart. The weights for all other connections are set to zero. Local connectivity makes each unit responsible for extracting local features from the layer below, and reduces considerably the total number of weights. With many more hidden units than Net-2, Net-3 has fewer links and hence weights (1226 vs. 3214), and achieves similar performance.

Net-4 and Net-5 have local connectivity with shared weights. All units in a local feature map perform the same operation on different parts of the image, achieved by sharing the same weights. The first hidden layer of Net4 has two 8×8 arrays, and each unit takes input from a 3×3 patch just like in Net-3. However, each of the units in a single 8×8 feature map share the same set of nine weights (but have their own bias parameter).



Q1: For `Net-3` I'm curious, why it can get nearly the same performance as `Net-2` while it only get input from a local area. And it particular, why it is a 3x3 grid and why the upper layer(the second hidden layer) is getting input from a 5x5 grid? What difference does it make?

A: Like the text said:

> Local connectivity makes each unit responsible for extracting local features from the layer below, and reduces considerably the total number of weights.

It's kind of a way of forcing the first layer to focus on local features, though a fully connected layer might able to do so as well, it's not as handy to get a good result as this approach.

For second layer, 5x5 makes it extract high level features from a 10x10 area of the original image, larger, but still a focus.

Q5

Q3. In NN reading, how to calculate Bayes error rate in figure 11.4? Could you please show the calculations step by step?

A:

$$R(\theta) = -\sum_{i=1}^{N}\sum_{k=1}^{K} y_{ik} \log f_k(x_i),$$

Q6

Q1: For NN, the text said "The PPR model (11.1) is very general, since the operation of forming nonlinear functions of linear combinations generates a surprisingly large class of models. For example, the product $X_1 \cdot X_2$ can be written as $\left[(X_1+ X_2)^2 - (X_1 - X_2)^2\right]/4$, and higher-order products can be represented similarly." I'm wondering how linear transformation can get a result of $X_2)^2 - (X_1 - X_2)^2\right]/4$?

A: Similar to Q1, it is linear combination and ridge function together that generates $\left[(X_1 + X_2)^2 - (X_1 - X_2)^2\right]/4$, not just linear transformation.

Q7

1. On page 394, when the book is describing the sigmoid activation function, it states that, "the rate of the sigmoid activation depends on the norm of alpha_m, and if the magnitude of alpha_m is very small, the unit will indeed be operating on the linear part of its activation." I understand this, but I want to know why this is an important feature of the activation function. For alpha_m with large magnitude, this function has a saturating effect so is the point of this function to act non-linearity and normalize the values?

A: To my understanding, this is not an important feature, but a fact of the sigmoid function.

All activation functions are to add non-linearity to the model, but normalization is not a must-have.

Q8

2. When referring to learning rate, the book states, "with online learning, learning rate should decrease to zero". Why is this?

A: Fixed learning rate might miss the optimal point, so as the training goes on, learning rate should be decreasing to better approximate the optimal point.