Machine Learning for Economic Analysis
Problem Set 9

Jonas Lieber[*]

Due: 11:59pm Friday, April 12, 2024

Problem 1. *More on Backprop*

In the lecture, we have derived the derivatives of the loss with respect to the outputs of the hidden layers and the weights but not the values. Let's fill this gap!

1. *Show that* $\frac{\partial h^l}{\partial b^l} = (f^l)'(W^l h^{l-1} + b^l)$*, where, as in the lecture,* $(f^l)'(\cdot)$ *is the derivative of* f^l*.*
2. *How can we compute* $\frac{\partial L}{\partial b^l}$ *in the spirit of backpropagation? Hint: Use the chain rule in a clever way.*

Problem 2. *Training Neural Networks*

1. *Loss Functions*
   - (a) *Implement MSE and cross-entropy loss with* L^2 *penalty on all parameters as a function of the data and the parameters of the network.*
   - (b) *Derive and implement the derivatives (with respect to the output of the neural network) of MSE and cross-entropy loss with* L^2 *penalty as a function of the label in the data and the output of a neural network.*
2. *Derivative of Activation functions*
   - (a) *Implement the derivative of sigmoid and ReLU using vectorization.*
3. *Backpropagation*
   - (a) *Implement the backpropagation algorithm to compute the gradient of a neural network with respect to all weights and biases.*

[*]Department of Economics, Yale University. jonas.lieber@yale.edu

(b) *Consider the network architecture and feature vector x from problem set 8. Suppose that the y associated with x is $y = 3$. For both MSE and cross-entropy loss, use back-propagation to compute the gradient of the loss with respect to all weights and biases. Report the derivatives with respect to the weights and bias of the first hidden layer.*

4. *Stochastic Gradient Descent with Early Stopping*

   (a) *Explain the idea behind "early stopping" and why it is sometimes considered a "free lunch" in machine learning.*[1]

   (b) *Implement a function for stochastic gradient descent with early stopping.*

      i. *Inputs should be*
         - *the function (of parameters and data) to be optimized*
         - *its gradient with respect to parameters*
         - *the data*
         - *the batch size*
         - *initial parameters*
         - *the patience for early stopping (see algorithm 7.1 in the "Deep Learning" book), evaluate the training error whenever you have passed through all (shuffled) training data.*

      ii. *The function should*

         A. *split the data into a training, a validation set and a test set (use the same split for every λ) with the usual partitioning sizes (50%,25%,25%),*

         B. *shuffle the training data,*

         C. *perform SGD updates for batches of the shuffled training data,*

         D. *evaluate the validation error after passing through all of the shuffled training data,*

         E. *saves the best validation error together with the associated parameters,*

         F. *stops when the best validation error is not reduced for p consecutive times.*

5. *Starting values*

   (a) *We consider the same network structure as on problem set 8. Explain considerations that influence how we choose starting values for neural networks. Choose a set of starting values for all weights and biases that you consider to be good.*

6. *Penalization (transfer from Ridge)*

   - *Would you normalize the data for neural networks? Why or why not?*

[1] We covered this in the discussion of Ridge Regression.

- *Would you penalize weights and biases? Or would you prefer to penalize just the biases? Or would you prefer to penalize just the weights? Explain your reasoning.*

7. *Train the neural network with the architecture from problem set 8. Consider the following parameters*
   - *p=5,*
   - *MSE and cross-entropy,*
   - $\lambda = 1, 10, 100, 1000, 10000$ *(use validation error to select the optimal parameter for λ),*
   - *and the data supplied on canvas.*

   (a) *With which λ do you start and why?*

   (b) *For the selected value of λ, report the training, validation and test error.*

   (c) *Imagine the following scenario: You work in your first job and trained this neural network to solve a business problem. The loss you have obtained with this architecture and training parameters is 25% higher than expected. In a meeting with your team, the following next steps are considered.*

      i. *Collect more data.*

      ii. *Add additional layers and/or neurons to the network.*

      iii. *Remove layers and/or neurons from the network.*

      Which trade-off could inform your choices? To evaluate these options, retrain the model with 5% of the data, 10% of the data, . . . , 95% of the data, and plot the training, validation and test loss (always start at the same initial value) in a single figure. Does this diagnostic indicate that a lack of data might be the bottleneck? Should you consider a more or less complicated network structure?

   (d) *Suppose you made the decision to collect 50% more data and train the same network. You merge the old and the new dataset and retrain the network. If you randomly split the data into training, validation and test, do you expect the test error estimate to be an unbiased estimate of the expected test error? How could you split the data to obtain an unbiased estimate of the expected test error?*

Problem 3. ***(For graduate students only)*** *SGD with Momentum*

1. *Read Section 8.3.2 in the "Deep Learning" book.*

2. *Adapt your SGD algorithm above by adding momentum α as an input parameter.*

3. *Which α value corresponds to SGD without momentum?*

4. *Consider the values* $\alpha = 0, 0.25, 0.5, 0.75, 0.9, 0.0.95, 0.99$. *Report the computation time, training error, validation error and test error of the network for each of these values.*