ETC3555 2018 - Lab 7

Deep neural networks

Cameron Roach and Souhaib Ben Taieb

04 September, 2018

Preliminaries

We will be working with the Keras framework to implement our deep neural network (DNN). Keras is a front end for several DNN frameworks (e.g. TensorFlow, Theano, etc.) which aims to simplify the coding process. Install Keras for `R` by running:

```
install.packages("keras")
library(keras)
install_keras(method = c("conda"), conda = "auto", tensorflow = "default",
              extra_packages = c("tensorflow-hub"))
```

TensorFlow should automatically be installed and selected as the default backend.

If at any stage you need to retrain your models from scratch make sure to clear your workspace and refresh your R session. Failing to do so will result in continued training of existing models or unexpected errors.

Exercises

Exercise 1

Work through the introduction to Keras tutorial on the RStudio blog located at https://keras.rstudio.com/. This walk-through will have you implement a feed-forward neural network with:

1. 256 node hidden layer with relu activation function
2. dropout layer with 0.4 dropout rate
3. 128 node hidden layer with relu activation function
4. dropout layer with 0.3 dropout rate
5. fully connected (dense) output layer with softmax output.

After completing the tutorial answer the following questions:

a) Why would you use softmax on the output layer and relu on the input?
b) Why different dropout rates (is there any benefit in varying them)?
c) Is the loss lower on the validation set than the training set for early epochs? Does this agree with your intuition? What happens when you remove the dropout layers and retrain? Explain why.
d) Once you figure out (c), can you justify the design choice?
e) Does the training accuracy and validation accuracy cross? Why? If not, why?

Assignment - Question 1

a) Create another DNN as above, but with half as many units in each hidden layer. Assess the performance of this model against your previous model.
b) Now try L1 or L2 regularization with dropout layers removed.

c) Create another model without dropout or regularization and implement early stopping based on the validation loss. Is the loss improved? Is the accuracy improved? How do the three models compare? Produce a plot comparing both the loss and accuracy of each model.

Assignment - Question 2

Create a new network with the following structure:

1. $d^{(1)}$ node hidden layer with relu activation function
2. dropout layer with q dropout rate
3. fully connected (dense) output layer with softmax output.

Use grid search to select optimal $d^{(1)} \in \{16, 32, 64, 128\}$ and $q \in \{0, 0.25, 0.5, 0.75\}$ values. Create a heatmap or similar showing the loss for each combination of hyperparameter values. Which combination gives the best performance? Comment on the suitability of this grid of values.

Assignment - Question 3

Using your optimal hyperparameters from the previous question, try refitting the network using:

- Momentum
- ADAM
- RMSProp.

Comment on how each affects the model accuracy on the test set. Which converges fastest?

TURN IN

- Your `.Rmd` file (which should knit without errors and without assuming any packages have been pre-loaded)
- Your Word (or pdf) file that results from knitting the Rmd.
- DUE: September 16, 11:55pm (late submissions not allowed), loaded into moodle.