* (The **learning rate decayed** by a factor of 1.0000002 every batch update until it reached a minimum of 10^−6,)
* (An additional boost in performance is obtained by using the **dropout** training algorithm, in which we stochastically drop neurons in the top hidden layer with 50% probability during training.)
* (Done) We selected a five-layer neural network with 300 hidden units in each layer,
* (Done) a **learning rate** of 0.05, and a **weight decay** coefficient of 1 × 10−5.
* (Done)Hidden layers have **tanh activation function**
* (Done)Gradient computations were made on **mini-batches** of size 100
* An additional boost in performance is obtained by using the dropout training algorithm, in which we stochastically drop neurons in the top hidden layer with 50% probability during training.
* Weights were initialized from a normal distribution with zero mean and standard deviation 0.1 in the first layer, 0.001 in the output layer, and 0.05 all other hidden layers.
* A **momentum** term increased linearly over the first 200 epochs from 0.9 to 0.99, at which point it remained constant.
* The learning rate decayed by a factor of 1.0000002 every batch update until it reached a minimum of 10^−6,

Learning Rate – controls quickly/slowly NN learns a problem, configurable hyperparameter, usually small positive value range 0.0 and 1.0, controls apportioned error that weights of model are updated with each time they are updated (i.e. at the end of batch training), large learning rate – model learns faster at cost of arriving on sub-optimal final set of weights/smaller learning rate may allow better model to learn optimal but longer to train; traditional default 0.1 or 0.01; scalar used to train model via gradient descent; during each iteration, gradient descent algorithm multiplies learning rate by gradient – result is gradient step. Controls how much to change the weight to correct for error (i.e. value of 0.1 will update the weight 10% of amount it could be updated), small rates preferred that cause slower learning over larger number of iterations; amount of change to model during each step of searching for optimal solution (global vs local), (aka step size), positive scalar determining the size of step, learning rate less than 1.0 and greater than 10^-6

Smaller learning rates require more training epochs

Smaller batch sizes better suited to smaller learning rates

Learning Rate Decay – the way in which learning rate changes over time / training epochs,

Drop Out – form of regularization, removes a random selection of fixed number of units in a network layer for a single gradient step; more dropped, the stronger the regularization;

Weight Decay -

Activation Functions/Transfer Functions – takes weighted sum of all inputs from previous layer and generates/passes an output value to next layer

* Tanh activation
* Sigmoid activation – S shape, logistic function, take any input and produce result between 0 and 1
* Softmax activation -

Backpropagation Algorithm – method for training weights, supervised learning method for multilayer feed-forward networks, calculate error for each output neuron to get error signal (input) to propagate backward through network

Momentum – does not make it easier to configure learning rate, step size is independent of momentum, improve speed of optimization in concert with step size value greater 0.0 and less than 1.0, common values 0.9, 0.99, 0.5; change to stochastic gradient descent when exponentially weighted average of prior updates to weight can be included when weights are updated, adds inertia to update procedure to continue to move in one direction, accelerate learning especially with high curvature, small/consistent gradients, noisy gradients .. aka velocity, smoothing optimization process

Batch – set of examples used in one iteration (one gradient update) of model training

Batch size – number of examples in batch, batch size of SGD is 1; batch size of mini-batch is usually between 10 and 1000.

Epoch – a full training pass over the entire dataset such that each example has been seen once; epoch represents N/batch size training iterations, where N is the total number of examples. Loop through fix number of epochs and within each, update network for reach row in training data

AUC / ROC CURVE – evaluation metric that considers all possible classification thresholds; areas of ROC curve is probability that a classifier will be more confident that a randomly chosen positive example is actually positive than that a randomly chosen negative example is positive

Neuron – has set of weights, one weight for each input connection and an additional weight for bias.

NN – input layer is row from training set; real first layer is hidden layer, then output layer that has one neuron for each class value

Forward-propagation – calculate output from neural network by propagating an input signal through each layer until the output layer outputs its values, generate predictions during training that will need to be correct and also used to make predictions on new data

References:

<https://developers.google.com/machine-learning/glossary>

<https://machinelearningmastery.com/implement-backpropagation-algorithm-scratch-python/>

<https://machinelearningmastery.com/what-is-deep-learning/>

<https://machinelearningmastery.com/learning-rate-for-deep-learning-neural-networks/>