CS540 Introduction to Artificial Intelligence

Lecture 4

Young Wu

Based on lecture slides by Jerry Zhu, Yingyu Liang, and Charles Dyer

July 1, 2022

Perceptron Algorithm vs Logistic Regression

Motivation

- For LTU Perceptrons, w is updated for each instance x_i sequentially.

$$w = w - \alpha (a_i - y_i) x_i$$

- For Logistic Perceptrons, w is updated using the gradient that involves all instances in the training data.

$$w = w - \alpha \sum_{i=1}^{n} (a_i - y_i) x_i$$

Stochastic Gradient Descent

Motivation

- Each gradient descent step requires the computation of gradients for all training instances $i = 1, 2, ..., n$. It is very costly.
- Stochastic gradient descent picks one instance x_i randomly, compute the gradient, and update the weights and biases.
- When a small subset of instances is selected randomly each time, it is called mini-batch gradient descent.

Stochastic Gradient

- Full gradient descent computes the gradient with respect to all instances.

$$w = w - \alpha \frac{\partial C}{\partial w} = w - \alpha \sum_{i=1}^{n} \frac{\partial C_i}{\partial w}$$

- Stochastic gradient descent repeatedly select a random instance i and computes the gradient with respect to that instance.

$$w = w - \alpha \frac{\partial C_i}{\partial w}$$

- Usually, random sampling is done without replacement by shuffling the training set instead of sampling with replacement, so that all instances are included in each iteration (called an epoch).

Stochastic Gradient Descent, Part 1

Algorithm

- Inputs, Outputs: same as backpropagation.
- Initialize the weights.
- Randomly permute (shuffle) the training set. Evaluate the activation functions at one instance at a time.
- Compute the gradient using the chain rule.

$$\frac{\partial C}{\partial w_{j'j}^{(l)}} = \delta_{ij}^{(l)} a_{ij'}^{(l-1)}$$

$$\frac{\partial C}{\partial b_j^{(l)}} = \delta_{ij}^{(l)}$$

Stochastic Gradient Descent, Part 2

Algorithm

- Update the weights and biases using gradient descent.

  For $l = 1, 2, ..., L$

$$w_{j'j}^{(l)} \leftarrow w_{j'j}^{(l)} - \alpha \frac{\partial C}{\partial w_{j'j}^{(l)}}, j' = 1, 2, ..., m^{(l-1)}, j = 1, 2, ..., m^{(l)}$$

$$b_j^{(l)} \leftarrow b_j^{(l)} - \alpha \frac{\partial C}{\partial b_j^{(l)}}, j = 1, 2, ..., m^{(l)}$$

- Repeat the process until convergent.

$$|C - C^{\text{prev}}| < \varepsilon$$

Choice of Learning Rate

Discussion

- Changing the learning rate α as the weights get closer to the optimal weights could speed up convergence.
- Popular choices of learning rate include $\frac{\alpha}{\sqrt{t}}$ and $\frac{\alpha}{t}$, where t is the current number of iterations.
- Other methods of choosing step size include using the second derivative (Hessian) information, such as Newton's method and BFGS, or using information about the gradient in previous steps, such as adaptive gradient methods like AdaGrad and Adam.

Multi-Class Classification

Motivation

- When there are K categories to classify, the labels can take K different values, $y_i \in \{1, 2, ..., K\}$.
- Logistic regression and neural network cannot be directly applied to these problems.

Method 1, One VS All

Discussion

- Train a binary classification model with labels $y_i' = \mathbb{1}_{\{y_i = j\}}$ for each $j = 1, 2, ..., K$.
- Given a new test instance x_i, evaluate the activation function $a_i^{(j)}$ from model j.

$$\hat{y}_i = \underset{j}{\operatorname{argmax}}\, a_i^{(j)}$$

- One problem is that the scale of $a_i^{(j)}$ may be different for different j.

Method 2, One VS One

Discussion

- Train a binary classification model for each of the $\frac{K(K-1)}{2}$ pairs of labels.
- Given a new test instance x_i, apply all $\frac{K(K-1)}{2}$ models and output the class that receives the largest number of votes.

$$\hat{y}_i = \underset{j}{\operatorname{argmax}} \sum_{j' \neq j} \hat{y}_i^{(j \text{ vs } j')}$$

- One problem is that it is not clear what to do if multiple classes receive the same number of votes.

One Hot Encoding

Discussion

- If y is not binary, use one-hot encoding for y.
- For example, if y has three categories, then

$$y_i \in \left\{ \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \right\}$$

Method 3, Softmax Function

Discussion

- For both logistic regression and neural network, the last layer will have K units, a_{ij}, for $j = 1, 2, ..., K$, and the softmax function is used instead of the sigmoid function.

$$a_{ij} = g\left(w_j^T x_i + b_j\right) = \frac{\exp\left(-w_j^T x_i - b_j\right)}{\sum_{j'=1}^{K} \exp\left(-w_{j'}^T x_i - b_{j'}\right)}, j = 1, 2, ..., K$$

Softmax Derivatives

Discussion

- Cross entropy loss is also commonly used with a softmax activation function.
- The gradient of cross-entropy loss with respect to a_{ij}, component j of the output layer activation for instance i has the same form as the one for logistic regression.

$$\frac{\partial C}{\partial a_{ij}} = a_{ij} - y_{ij} \Rightarrow \nabla_{a_i} C = a_i - y_i$$

- The gradient with respect to the weights can be found using the chain rule.

Generalization Error

Motivation

- With a large number of hidden units and small enough learning rate α, a multi-layer neural network can fit every finite training set perfectly.
- It does not imply the performance on the test set will be good.
- This problem is called overfitting.

Method 1, Validation Set

Discussion

- Set aside a subset of the training set as the validation set.
- During training, the cost (or accuracy) on the training set will always be decreasing until it hits 0.
- Train the network until the cost (or accuracy) on the validation set begins to increase.

Method 2, Drop Out

Discussion

- At each hidden layer, a random set of units from that layer is set to 0.
- For example, each unit is retained with probability $p = 0.5$. During the test, the activations are reduced by $p = 0.5$ (or 50 percent).
- The intuition is that if a hidden unit works well with different combinations of other units, it does not rely on other units and it is likely to be individually useful.

Method 3, $L1$ and $L2$ Regularization

Discussion

- The idea is to include an additional cost for non-zero weights.
- The models are simpler if many weights are zero.
- For example, if logistic regression has only a few non-zero weights, it means only a few features are relevant, so only these features are used for prediction.

Method 3, $L1$ Regularization

Discussion

- For $L1$ regularization, add the 1-norm of the weights to the cost.

$$C = \sum_{i=1}^{n} (a_i - y_i)^2 + \lambda \left\| \begin{bmatrix} w \\ b \end{bmatrix} \right\|_1$$
$$= \sum_{i=1}^{n} (a_i - y_i)^2 + \lambda \left(\sum_{i=1}^{m} |w_i| + |b| \right)$$

- Linear regression with $L1$ regularization is called LASSO (least absolute shrinkage and selection operator).

Method 3, $L2$ Regularization

Discussion

- For $L2$ regularization, add the 2-norm of the weights to the cost.

$$C = \sum_{i=1}^{n} (a_i - y_i)^2 + \lambda \left\| \begin{bmatrix} w \\ b \end{bmatrix} \right\|_2^2$$

$$= \sum_{i=1}^{n} (a_i - y_i)^2 + \lambda \left(\sum_{i=1}^{m} w_i^2 + b^2 \right)$$

$L1$ and $L2$ Regularization Comparison

Discussion

- $L1$ regularization leads to more weights that are exactly 0. It is useful for feature selection.
- $L2$ regularization leads to more weights that are close to 0. It is easier to do gradient descent because 1-norm is not differentiable.

Method 4, Data Augmentation

Discussion

- More training data can be created from the existing ones, for example, by translating or rotating the handwritten digits.

Hyperparameters

Discussion

- It is not clear how to choose the learning rate α, the stopping criterion ε, and the regularization parameters.
- For neural networks, it is also not clear how to choose the number of hidden layers and the number of hidden units in each layer.
- The parameters that are not parameters of the functions in the hypothesis space are called hyperparameters.

K Fold Cross Validation

Discussion

- Partition the training set into K groups.
- Pick one group as the validation set.
- Train the model on the remaining training set.
- Repeat the process for each of the K groups.
- Compare accuracy (or cost) for models with different hyperparameters and select the best one.

5 Fold Cross Validation Example

Discussion

- Partition the training set S into 5 subsets S_1, S_2, S_3, S_4, S_5

$$S_i \cap S_j = \emptyset \text{ and } \bigcup_{i=1}^{5} S_i = S$$

Iteration	Training	Validation
1	$S_2 \cup S_3 \cup S_4 \cup S_5$	S_1
2	$S_1 \cup S_3 \cup S_4 \cup S_5$	S_2
3	$S_1 \cup S_2 \cup S_4 \cup S_5$	S_3
4	$S_1 \cup S_2 \cup S_3 \cup S_5$	S_4
5	$S_1 \cup S_2 \cup S_3 \cup S_4$	S_5

Leave One Out Cross Validation

Discussion

- If $K = n$, each time exactly one training instance is left out as the validation set. This special case is called Leave One Out Cross Validation (LOOCV).