Deep Learning Study Lecture Note

Lecture 4 : How to experiment

Kim Tae Young

January 13, 2020

4.1 Cross Validation

Cross validation is well used method for regularization in machine learning. The basic idea is changing validation set.

	Dataset 1	Dataset 2	Dataset 3	Dataset 4	Dataset 5
Train 1	Validation	Train	Train	Train	Train
Train 2	Train	Validation	Train	Train	Train
Train 3	Train	Train	Validation	Train	Train
Train 4	Train	Train	Train	Validation	Train
Train 5	Train	Train	Train	Train	Validation

Figure 4.1: 5-fold cross validation

This k-fold cross validation, where we split data set into k same size subsets, then train as the table. The total k steps becomes one epoch. There are other types of cross validation, like leave-p-out cross validation which uses p data for validation and training for every $\binom{N}{p}$ training - validation set pair. Also well used one is called Monte Carlo cross validation, which train for randomly splitted data sets.

4.2 Ensemble Method

Ensemble Method resolves random seed issue. As we remember, choosing different random seed may yields different optimum, due to non convexity of loss function. Actually, this is well used in machine learning, things like bagging, random forest, boosting. In – cases, if problem is classification we choose maximum voted class, and if problem is regression we use average value of each models.

4.2.1 Voting Classifier

First, we can create multiple models, then learn dataset with each of these models. You can change initial points for each nets, or you can change hyperparameters for each nets. Then after learning, the prediction of voting classifier is maximum voted class of each nets.
There are also other method called stacking, which replaces maximum/average voter to another model.

4.2.2 Bagging

Bagging is abbreviation of bootstrap aggregating, where we does create multiple identical models, however we learn them to different subsets of dataset.

4.2.3 Boosting

Unlike former two, boosting is not parallel learning but sequential learning. Simply, we create sequences of model, that is not that strong. The most major two boosting is Adaboosting and Gradient Boosting.
In adaboosting, we train first model to whole data. Then for mispredicted data, we give them higher weight. We adapted weights, we train next model, and repeat this.

Data: (x_i, y_i) for $i = 1, \ldots, N$
Initialize : $w_1(i) = \dfrac{1}{N}$;
for $t = 1 \ldots T$ **do**
  Train M_t with weight w_i;
  $\epsilon_t = P_{w_t}(M_t(x_i) \neq y_i)$;
  $\alpha_t = \frac{1}{2} \log(\dfrac{1 - \epsilon_t}{\epsilon_t})$;
  $w_{t+1}(i) = \dfrac{w_t(i)}{Z_t} \times \begin{cases} e^{\alpha_t} & M_t(x_i) \neq y_i \\ e^{-\alpha_t} & M_t(x_i) = y_i \end{cases}$
end
return $H(x) = \sum_{t=1}^{T} \alpha_t h_t(x)$

Algorithm 1: Adaboosting

In gradient boosting, similarly we train first model to whole data. Then second model trained to residual error of first model. And next model trained to residual error of former model, and it is repeated.

Data: (x_i, y_i) for $i = 1, \ldots, N$
for $t = 1 \ldots T$ **do**
  Train M_t with $(x_i, y_i - \sum_{i=1}^{t-1} M_i(x_i))$
end
return $H(x) = \sum_{i=1}^{T} M_i(x)$

4.3 Hyperparameter Tuning

Here, I will explain two main methods used on hyperparameter tuning. First one is grid tuning, we run experiment for grid candidates. Suppose we are doing hyperparemter tuning for number of layer, hidden dimension in MLP. Then your candidates for n_layer would be $1 \ldots 5$ and hidden dimension is $100 \ldots 500$. Then we run experiment for following pairs, [(1, 100), (1, 300), (1, 500), (3, 100), (3, 300), (3, 500), (5, 100), (5, 300), (5, 500)].
Random Tuning is as the name says, you just sample some random examples for your candidates range, then run experiment for those.

4.4 Bayesian Optimization

You can automate hyperparameter tuning process by well formulating selection for your grid tuning and random tuning however it may not efficient and there are chances of grid tuning and random tuning does not find out maximum of performance. Also, both grid tuning and random tuning do not create candidate based on known informations. Bayesian optimization is one way of solving this. Simply, it is assuming some model for parameter-performance relation. This model is called surrogate model. Also we need function to create new candidate, called acquisition function. Given these two functions, bayesian optimization is simple process.

Data: Acquisition function f, Surrogate model M, target function h
for $t = 1 \dots T$ **do**
  $x_{t+1} = f((x_1, y_1), \dots (x_t, y_t))$;
  $y_{t+1} = h(x_{t+1})$;
end

Usually, Gaussian process is major for surrogate model. I will not explain details about Bayesian optimization, since it is more ML related. If you are interested, see [2].

Bibliography

[1] Bayesian Optimization 개요, 김길호 http://research.sualab.com/introduction/practice/2019/02/19/bayesian-optimization-overview-1.html

[2] 앙상블 학습 및 랜덤 포레스트, Excelsior-JH https://excelsior-cjh.tistory.com/166