CS365: Deep Learning

Practical Methodology

Arijit Mondal
Dept. of Computer Science & Engineering
Indian Institute of Technology Patna
arijit@iitp.ac.in

Introduction

- Successful application of deep learning
  - Require knowledge of different techniques available
  - Need to know the principle how it works
- Common issues faced
  - Require more data
  - Increase or decrease model complexity
  - Choice of regularizer
  - Optimization model
  - Debug procedure
- All are time consuming

Recommended design process

- Determine the goals
  - Choice of error metric
  - Target value for error metric
    - It depends on the problem at hand
- Setup a working end-to-end pipeline
- Find out bottlenecks and components having poor performance
  - Overfitting, underfitting, defect in data or software etc.
- Repeatedly make incremental changes
  - Gather new data
  - Adjust hyperparameter
  - Try with different algorithms

Performance metric

- Determine your goal and error metric
- Achieving absolute zero error is nearly impossible
- Limited by finite data
  - More data can be collected after it is in operation
  - Data collection is a tedious process and requires money, time, human suffering
  - For benchmark, no extra data should be collected
- Performance level
  - Academic setting — use previously published results
  - Real world — We need to have some information for it to be safe, cost effective, appealing to users

Performance metric

- Performance metric and cost function are different
  - **Precision** — Fraction of detection reported by the model that are correct
  - **Recall** — Fraction of true events that are detected
- **PR-curve** — Threshold required
  - Precision in y-axis and recall in x-axis
- To have single number for comparison **F-score** is used ($F = \dfrac{2pr}{p+r}$)
- Coverage — Fraction of examples for which the machine learning system able to produce response
  - Accuracy vs Coverage trade-off

Selection of baseline

- Depending on the complexity of the problem, deep learning may be required
- If the problem is "AI-Complete" such as object identification then deep learning may be a good choice
- Initially general category model is selected
- Supervised learning with fixed input
  - Feed forward network with fully connected layers
- Input has known topological structure like image
  - CNN can be chosen
- Input or output is a sequence
  - Gated recurrent network is preferred

Choice of optimization

- SGD with momentum with decay rate
- Batch normalization can help (specially for CNN or network with sigmoidal non-linearities)
- For small batch size it is better to have regularization at the start
- Early stopping is good
- Dropout is a good regularizer
- Start with already existing model

Data

- Check for performance on training data
  - If it is not acceptable (and no more data is required)
    - Increase the model size
    - Add more layers, more units
    - Tune learning rate
    - Check optimization algorithm
      - Probably problem with training data!
  - Acceptable in training data, check performance in test data
    - Not acceptable in test data
      - May require more data, reduce model size

Selection of hyperparameters

- Has significant effect on the performance
  - Time
  - Memory
  - Quality
- To choose it manually, understanding of the hyperparameter is required
- For automatic selection, more computation are required
- For some hyperparameters, generalization error follow U shape curve

Manual hyperparameter tuning

- Need to understand the relationship between hyperparameter and
  - Training error
  - Generalization error
  - Computational resource
    - Need to understand effective capacity
- Target of hyperparameter is to minimize generalization error
- Effective capacity
  - Representational capacity of model
  - Learning algorithm to minimize cost function
  - Degree to which the cost function and training procedure regularize the model

Learning rate

- Controls the effective capacity in a very complicated manner
  - Effective capacity is highest when learning rate is correct
- When learning rate is very high, training error may increase
- When learning rate us small, training will be slower and may prematurely stuck with high training error
- Tuning other parameters requires monitoring of training and test error
- If training error is higher than the desired target error
  - Increase capacity

Hyperparameters

Hyperparameters	Capacity	Reason	Caveats
Number of hidden units	Increase	More representational capacity	Time and memory will increase
Learning rate	Need to tune optimally	Improper learning rate may result in poor performance	
Convolution kernel width	Increase	Increases the number of parameters	May require 0 padding. Memory and time will increase
Implicit 0 padding	Increase	Keeps representation size large	Memory and time will increase
Weight decay	Decrease	Model parameters can become large	
Dropout rate	Decrease	Ensemble	

Automatic hyperparameter optimization

- Neural network is good when lot of hyperparameters are available
- Manual tuning of hyperparameters is good but requires experience
- Start point may be known for some cases (manual tuning is possible)
- Hyperparameter optimization
  - Hyperparameter will have their own hyperparameters
  - Easier to choose secondary hyperparameters

Grid search

- Common practice is to perform grid search when number of hyperparameter is three or less
- Smallest and largest values are chosen conservatively
- Picks the value in log scale
- Performs well when applied repeatedly
  - Refinement of ranges
- Computation cost is very high

Debugging strategies

- Visualize the model in action
- Visualize the worst mistakes
- Reason about software using training and test error
- Fit a tiny data set
- Compare back-propagated derivatives
- Monitor histogram of activations and gradients