Deep Learning - Practical Methodology

Cosmin G. Alexandru

BucharestCV

March 19, 2017

Layout

Introduction

Recomended design process:

- Determine your goal - choose performance(error) metric and a target for it.
- Establish working end-to-end pipeline as soon as possible.
- Instrument the system well.
- Do incremental changes.

Layout

Peformance Metric

- Choose a performance metric specific to the problem.
- How to determine the target performance:
  - Academic setting - Usually, an estimate exists.
  - Real-world setting - Cost effective, appeal to the customer, usage safety.
- Different from the cost function used to train the model.

Peformance Metric - Examples

Performance/error metric examples:

- accuracy
- precision and recall
- coverage
- precision-recall curve
- $F - score = \frac{2pr}{p+r}$
- $logloss = \frac{-1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M}y_{ij}log(p_{ij})$

Layout

Baseline Models - Algorithms

Baseline algorithms recomandation:

- Simple problem - simple algorithm (e.g. logistic regerssion)
- Supervized learning + fixed size input vectors = a feed forword network with fully connected layers.
- Input with topological structure = convolutional neural network + piecewise linear units e.g.: ReLU, Leaky ReLU, PreLus, maxout.
- Input or output is a squence = gated recurrent network (LSTM or GRU).

Baseline Models - Optimization

Optimization algorithms:

- Stochastic Gradient Descent(SGD) with momentum.
- Adam.[5]

Baseline Models - Tips

Tips for improving optimization:

- Popular learning rate decay schemes for SGD:
  - Linear decay until fixed minimum.
  - Exponential decay.
  - Decrease learning rate by a factor of 2-10 each time validation error plateaus.
- Batch normalization can be omitted at first but it should be introduced when optimization becomes problematic. It allows for higher learning rates.
- Dropout is an excellent regularizer.
- "Early stoping should be used almost universally." [2]

Baseline Models - Supervised vs. Unsupervised

Supervised vs. unsupervised learning:

- Start with supervised learning. If the model overfits you can try unsupervised learning.
- Use unsupervised for applications in a context that is known to benefit from unsupervised learning (e.g: natural language processing) or the problem you try to solve is unsupervised.

Layout

1. Introduction
2. Performance Metric
3. Baseline Models
4. More Data
5. Hyperparameters
6. Debugging

More Data(0)

It is usually better to gather more data than to improve the algorithm.

First check the training set error. High error probably means that the model is not using the data. Things to try:

- Increase the model size(e.g: number of neurons per layer, number of layers, etc.)
- Tune the learning rate.
- Check the data quality.

More Data(1)

Check error on a test set.

- Small error - you are set.
- High error - gather more data.

If the cost for gathering more data is high, try:

- Adding drop out.
- Adjusting hyperparameters.

Plot performance vs. training data size to determine how much data to gather in order to obtain the desired performance.

Layout

Hyperparameters - Search

- Manual search:
  - Start with the learning rate.
- Automatic search:
  - Gird
  - Random
- Model based search:
  - Spearmint[4]
  - TPE[3]
  - SMAC[1]

Hyperparameters

The primary goal of hyperparameter search is to adjust the effective capacity of the model.

Effective capacity of the model depends on three factors:

- Representation capacity of the model (more hidden layers / more units per hidden layer = greater representational capacity).
- Optimization algortihm to successfully minimize the cost function.
- Degree to which the cost function and training procedure regularizes the model.

Learning rate vs training error

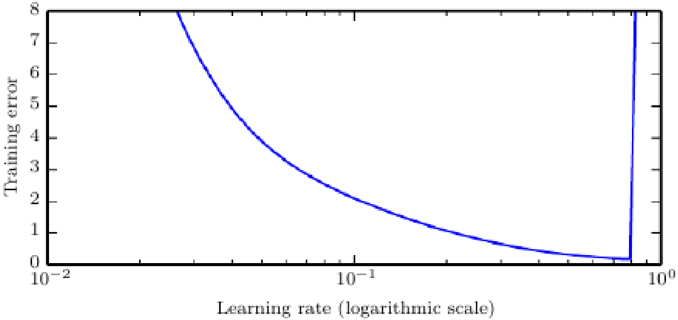

Figure 1: Typical relation between learning rate and training error (source:[2])

What are we searching for

Hyperparameter curve:

- Extreme 1: Low capcaity - generalization error high because training error is high - underfitting regime.
- Extreme 2: High capacity - generalization error is high because the gap between the trainging and test error is high - overfitting regime

Monitor both train and test error:

- train error higher than target $\rightarrow$ increase capacity.
- tetst error $=$ train error $+$ gap between train error and test error (such insight, much wow :P). Goal $=$ reduce gap at a higher rate than than the rate at which the training eror increases.

Grid vs. random search

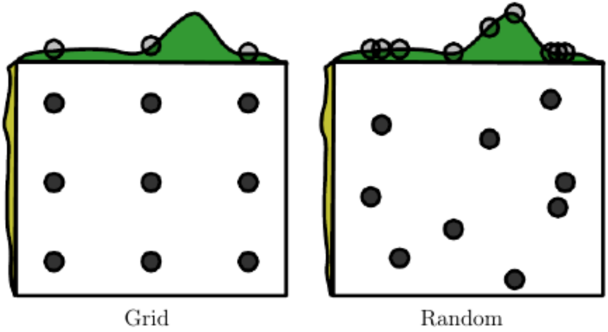

Figure 2: Comparison between grid and random search (source:[2])

Layout

Debugging

Methods to debug software problems:

- Visualize the model in action.
- Visualize the worst mistake.
- Reason about software using train and test error.
- Fit a small dataset.
- Compare back-propagation derivatives to numerical derivatives.
- Monitor historgrams of activations and gradient.

Hutter F., Hoos H., and Layton-Brown K.
Sequential model-based optimization for general algorithm configuration.
Lion-5 Extended version as UBC Tech report TR-2010-10., 2011.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville.
Deep Learning.
MIT Press, 2016.
http://www.deeplearningbook.org.

Bergstra J., Bardenet R., Bengio Y., and Kegl B.
Algorithms for hyper-parameter optimization.
NIPS 2011, 2011.

Snoek J., Larochelle H., and Adams R.P.

Practical bayesian optimization of machine learning algorithms.
NIPS 2012, 2012.

Diederik P. Kingma and Jimmy Ba.
Adam: A method for stochastic optimization.
CoRR, abs/1412.6980, 2014.