Modern Systems for Neural Networks

Valentin Dalibard
This talk

1. Practicalities of training Neural Networks
2. Leveraging heterogeneous hardware

Source: wikipedia
Modern Neural Networks Applications: Image classification

airplane
automobile
bird
cat
deer
dog
frog
horse
ship
truck
Modern Neural Networks Applications: NLP

Paul Graham generator:

“The surprised in investors weren’t going to raise money. I’m not the company with the time there are all interesting quickly, don’t have to get off the same programmers. There’s a super-angel round fundraising, why do you can do. If you have a different physical investment are become in people who reduced in a startup with the way to argument the acquirer could see them just that you’re also the founders will part of users’ affords that and an alternation to the idea. [2] Don’t work at first member to see the way kids will seem in advance of a bad successful startup. And if you have to act the big company too.”

Source: Andrej Karpathy: The Unreasonable Effectiveness of Recurrent Neural Networks
Modern Neural Networks Applications: Reinforcement Learning
Training Procedure: Stochastic Gradient Descent

Optimize the weights of the neurons to yield good predictions

Use “minibatches” of inputs to estimate the gradient

Source: wikipedia
Software platforms

- Caffe (C++)
- Torch (Lua)
- Theano (Python)
- Tensorflow (Python/C++)
- Lasagne
- Keras
Single Machine Setup:

One or a couple beefy GPUs
Distribution: Parameter Server Architecture

Parameter Server: $w' = w - \eta \Delta w$

Source: Dean et al.: Large Scale Distributed Deep Networks
Trends in software architecture

- Fewer bits per floating point
- Integers rather than floating points
Optimizing the scheduling on a heterogeneous cluster

Which machines to use as workers? As parameter servers?

↑ workers ⇒ ↑ computational power & ↑ communication

How much work to schedule

Must load balance
Ways to do an Optimization

<table>
<thead>
<tr>
<th>Random Search</th>
<th>Genetic algorithm / Simulated annealing</th>
<th>Bayesian Optimization</th>
</tr>
</thead>
<tbody>
<tr>
<td>No overhead</td>
<td>Slight overhead</td>
<td>High overhead</td>
</tr>
<tr>
<td>High #evaluation</td>
<td>Medium-high #evaluation</td>
<td>Low #evaluation</td>
</tr>
</tbody>
</table>
Bayesian Optimization

Find parameter values with high performance in the model

Evaluate the objective function at that point

Update the model with this measurement
Bayesian Optimization

- Parameter Space
- Utility Function
- Probabilistic Model
- Predicted Performance
- Performance
Structured Bayesian Optimization

- Parameter Space
- Utility Function
- Performance & Runtime properties
- Probabilistic Parameters
- Probabilistic Program
- Predicted Performance
Optimizing the scheduling of Neural Networks

Two separate models:

- Individual machine model: How fast can a machine process $k$ inputs
- Network model: How long does it take to transfer the parameters from parameter servers to workers

Iteratively learn the behavior
Optimizing the scheduling of Neural Networks

![Graph showing best iteration time vs function evaluations for Traditional BO and Structured BO.]
More CPU cores aren’t always better

![Graph](image)
Exposing Tradeoff
Conclusion

Growing demand for Neural networks platforms

Can leverage heterogeneous hardware but requires tuning

Bayesian Optimization can find good scheduling in a relatively short time