Abstractions and Frameworks for Deep Learning: a Discussion – Caffe, Torch, Theano, TensorFlow, et al.

Rémi Emonet

Saintélyon Deep Learning Workshop – 2015-11-26

Disclaimer

Introductory Poll

- Did you ever use?
  - Caffe
  - Theano
  - Lasagne
  - Torch
  - TensorFlow
  - Other?
- Any experience to share?

Overview

- Deep Learning?
- Abstraction in Frameworks
- A Tour of Existing Framework
- More Discussions?

Overview

- Deep Learning?
- Abstraction in Frameworks
- A Tour of Existing Framework
- More Discussions?

Finding Parameters of a Function (supervised)

- Notations
  - Input i
  - Output o
  - Function f given
  - Parameters θ to be learned
  - We suppose: $o = f_\theta(i)$
- How to optimize it: how to find the best θ?
  - need some regularity assumptions
  - usually, at least differentiability
- Remark: a more generic view
  - $o = f_\theta(i) = f(\theta, i)$

Gradient Descent

- We want to find the best parameters
  - we suppose: $o = f_\theta(i)$
  - we have examples of inputs i^n and target output t^n
  - we want to minimize the sum of errors $L(\theta) = \sum_n L(f_\theta(i^n), t^n)$
  - we suppose f and L are differentiable
- Gradient descent (gradient = vector of partial derivatives)
  - start with a random θ^0
  - compute the gradient and update $\theta^{t+1} = \theta^t - \gamma \nabla_\theta L(\theta)$
- Variations
  - stochastic gradient descent (SGD)
  - conjugate gradient descent
  - BFGS
  - L-BFGS
  - ...

Finding Parameters of a "Deep" Function

- Idea
  - f is a composition of functions
  - 2 layers: $o = f_\theta(i) = f^2_{\theta^2}(f^1_{\theta^1}(i))$
  - 3 layers: $o = f_\theta(i) = f^3_{\theta^3}(f^2_{\theta^2}(f^1_{\theta^1}(i)))$
  - K layers: $o = f_\theta(i) = f^K_{\theta^K}(\ldots f^3_{\theta^3}(f^2_{\theta^2}(f^1_{\theta^1}(i)))\ldots)$
  - with all f_l differentiable
- How can we optimize it?
- The chain rule!
- Many versions (with $F = f \circ g$)
  - $(f \circ g)' = (f' \circ g) \cdot g'$
  - $F'(x) = f'(g(x))g'(x)$
  - $\dfrac{df}{dx} = \dfrac{df}{dg} \cdot \dfrac{dg}{dx}$

Finding Parameters of a “Deep” Function

- Reminders: K layers: $o = f_\theta(i) = f^K_{\theta^K}(\ldots f^3_{\theta^3}(f^2_{\theta^2}(f^1_{\theta^1}(i)))\ldots)$
  - minimize the sum of errors $L(\theta) = \sum_n L(f_\theta(i^n), t^n)$
  - chain rule $\frac{df}{dx} = \frac{df}{dg} \cdot \frac{dg}{dx}$
- Goal: compute $\nabla_\theta L$ for gradient descent
  - $\nabla_{\theta^K} L = \frac{dL}{d_{\theta^K}} = \frac{dL}{df^K} \frac{df^K}{d_{\theta^K}}$
  - $\nabla_{\theta^{K-1}} L = \frac{dL}{d_{\theta^{K-1}}} = \frac{dL}{df^K} \frac{df^K}{df^{K-1}} \frac{df^{K-1}}{d_{\theta^{K-1}}}$
  - $\nabla_{\theta^1} L = \frac{dL}{d_{\theta^1}} = \frac{dL}{df^K} \frac{df^K}{df^{K-1}} \cdots \frac{df^2}{df^1} \frac{df^1}{d_{\theta^1}}$
  - $\frac{dL}{df^K}$: gradient of the loss with respect to its input ✔
  - $\frac{df^k}{df^{k-1}}$: gradient of a function with respect to its input ✔
  - $\frac{df^k}{d_{\theta^k}}$: gradient of a function with respect to its parameters ✔

Deep Learning and Composite Functions

- Deep Learning?
  - NN can be deep, CNN can be deep
  - “any” composition of differentiable function can be optimized with gradient descent
  - some other models are also deep… (hierarchical models, etc)
- Evaluating a composition $f_\theta(i) = f^K_{\theta^K}(\ldots f^3_{\theta^3}(f^2_{\theta^2}(f^1_{\theta^1}(i)))\ldots)$
  - “forward pass”
  - evaluate successively each function
- Computing the gradient $\nabla_\theta L$ (for gradient descent)
  - compute the input ($o) gradient (from the output error)
  - for each f_1, f_2, …
  - compute the parameter gradient (from the output gradient)
  - compute the input gradient (from the output gradient)

Back to “seeing parameters as inputs”

- Parameters (θ^k)
- Just another input of f_k
- Can be rewritten, e.g. as $f_k(\theta_k, x)$
- More generic
  - inputs can be constant
  - inputs can be parameters
  - inputs can be produced by another function (e.g. $f(g(x), h(x))$)

Overview

- Deep Learning?
- Abstraction in Frameworks
- A Tour of Existing Framework
- More Discussions?

Function/Operator/Layer

- The functions that we can use for f_k
- Many choices
  - fully connected layers
  - convolutions layers
  - activation functions (element-wise)
  - soft-max
  - pooling
  - ...
- Loss Functions: same with no parameters
- In the wild
  - Torch module
  - Theano operator

Data/Blob/Tensor

- The data: input, intermediate result, parameters, gradient, ...
- Usually a tensor (n-dimensional matrices)
- In the wild
  - Torch tensor
  - Theano tensor, scalars, numpy arrays

Overview

- Deep Learning?
- Abstraction in Frameworks
- A Tour of Existing Framework
- More Discussions?

Contenders

- Caffe
- Torch
- Theano
- Lasagne
- Tensor Flow
- Deeplearning4j
- ...

Overview

- Basics
  - install CUDA/Cublas/OpenBlas
  - blob/tensors, blocks/layers/loss, parameters
  - cuDNN
  - open source
- Control flow
  - define a composite function (graph)
  - choice of an optimizer
  - forward, backward
- Extend
  - write a new operator/module
  - "forward"
  - "backward": gradParam, gradInput

Caffe

- "made with expression, speed, and modularity in mind"
- "developed by the Berkeley Vision and Learning Center (BVLC)"
- "released under the BSD 2-Clause license"
- C++
- layers-oriented http://caffe.berkeleyvision.org/tutorial/layers.html
- plaintext protocol buffer schema (prototxt) to describe models (and so save them too)
- 1,068 / 7,184 / 4,077

Torch7

- By
  - Ronan Collobert (Idiap, now Facebook)
  - Clement Farabet (NYU, now Madbits now Twitter)
  - Koray Kavukcuoglu (Google DeepMind)
- Lua (+ C)
  - need to learn
  - easy to embed
- Layer-oriented
  - easy to use
  - difficult to extend, sometimes (merging sources)
- 418 / 3,267 / 757

Theano

- "is a Python library"
- "allows you to define, optimize, and evaluate mathematical expressions"
- "involving multi-dimensional arrays"
- "efficient symbolic differentiation"
- "transparent use of a GPU"
- "dynamic C code generation"
- Use symbolic expressions: **reasoning on the graph**
    - write numpy-like code
    - no forced “layered” architecture
    - computation graph
- 263 / 2,447 / 878

Lasagne (Keras, etc)

- Overlay to Theano
- Provide layer API close to caffe/torch etc
- Layer-oriented
- 133 / 1,401 / 342

Tensor Flow

- By Google, Nov. 2015
- Selling points
  - easy to move from a cluster to a mobile phone
  - easy to distribute
- Currently slow?
- Not fully open yet?
- 1,303 / 13,232 / 3,375

Deeplearning4j

- “Deep Learning for Java, Scala & Clojure on Hadoop, Spark & GPUs“
- Apache 2.0-licensed
- Java
- High level (layer-oriented)
- Typed API
- 236 / 1,648 / 548

Overview

- Deep Learning?
- Abstraction in Frameworks
- A Tour of Existing Framework
- More Discussions?

Be creative!
anything differentiable
can be tried!

How to choose a framework?

Any experience to share?

Thanks

Abstractions and Frameworks
for Deep Learning:
a Discussion
–
Caffe, Torch, Theano, TensorFlow, et al.

Rémi Emonet

Saintélyon Deep Learning Workshop – 2015-11-26