Convolution Neural Nets meet PDE's

Eldad Haber
Lars Ruthotto

SIAM CS&E 2017

- Convolution Neural Networks (CNN)
- Meet PDE's
- Optimization
- Multiscale
- Example
- Future work

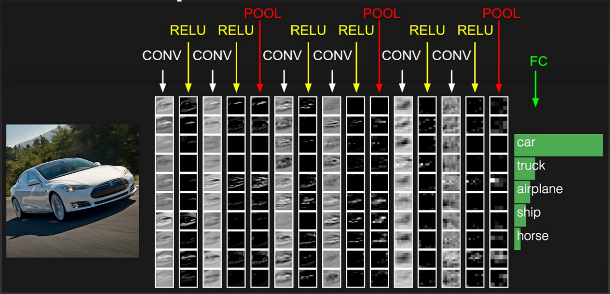
CNN - A quick overview

- Neural Networks with a particular architecture
- Exist for a long time (90's), (Lecun, Hinton)
- For large amounts of data can perform very well
- Applications
  - Image classification
  - Face recognition
  - Segmentation
  - Driverless cars
  - ...

CNN - A quick overview

- Neural Networks with a particular architecture
- Exist for a long time (90's), (Lecun, Hinton)
- For large amounts of data can perform very well
- Applications
  - Image classification
  - Face recognition
  - Segmentation
  - Driverless cars
  - ...
- A few recent quotes:
  - Apple Is Bringing the AI Revolution to Your iPhone, WIRED 2016
  - Why Deep Learning Is Suddenly Changing Your Life, FORTUNE 2016

CNN - A quick overview



ResNet architecture

- Propagation: for $j = 1, \ldots, N$

$$\mathbf{Y}_{j+1} = \mathbf{Y}_j + h\sigma(\mathbf{Y}_j \mathbf{K}_j + b_j)$$

- Classification

$$\mathbf{c} \approx \mathbf{W}\,\mathbf{Y}_N$$

CNN - A quick overview

$$\mathbf{Y}_{j+1} = \mathbf{Y}_j + h\sigma(\mathbf{Y}_j\mathbf{K}_j + b_j) \qquad \mathbf{c} \approx \mathbf{W}\,\mathbf{Y}_N$$

- $\mathbf{Y}_1$ - data $\mathbf{c}$ - class
- $\mathbf{K}_j$ - convolution kernel
- b_j - bias
- $\mathbf{W}$ - Classifier

Forward: Given an image $\mathbf{y}_1$ find its class c

Inverse (training): Given data and classes $\{\mathbf{Y}_1, \mathbf{c}\}$ obtain $\mathbf{K}_j, b_j$ and $\mathbf{W}$ that approximately classify the given data

CNN - The optimization problem

Define a similarity measure $\mathcal{S}$

$$\min_{\mathbf{K}_j, b_j, \mathbf{W}} \quad \mathcal{S}(\mathbf{c}, \mathbf{W}\,\mathbf{Y}_N)$$

$$\text{s.t} \quad \mathbf{Y}_{j+1} = \mathbf{Y}_j + h\sigma(\mathbf{Y}_j\mathbf{K}_j + b_j)$$

- How to solve the problem efficiently
- How to change scales
- Add robustness

CNN as ODE

$$\mathbf{Y}_{j+1} = \mathbf{Y}_j + h\sigma(\mathbf{Y}_j\mathbf{K}_j + b_j) \quad \leftrightarrow \quad \dot{\mathbf{Y}} = \sigma(\mathbf{Y}\mathbf{K}(t) + b(t))$$

Path planning: Find different path's for different classes

Convolution and PDE's

1D convolution

$$\mathbf{Ky} = [\mathbf{K}_1, \mathbf{K}_2, \mathbf{K}_3] * [\mathbf{y}_1, \ldots, \mathbf{y}_n]$$

Change the basis

$$\mathbf{Ky} = \mathbf{s}_1 * [0, 1, 0] + \frac{\mathbf{s}_2}{2h} * [-1, 0, 1] + \frac{\mathbf{s}_3}{h^2}[1, -2, 1]$$

$\mathbf{y}$ - a discretization (grid function) of $y(x)$
At the limit $h \to 0$

$$\mathbf{Ky} \approx \mathbf{s}_1 y + \mathbf{s}_2 \frac{dy}{dx} + \mathbf{s}_3 \frac{d^2 y}{dx^2}$$

CNN - continuous formulation

$$
\begin{aligned}
\min_{s_j(t), b(t), \mathbf{W}} \quad & \mathcal{S}(\mathbf{c}, \mathbf{W}\,\mathbf{Y}_N) \\
\text{s.t} \quad & \dot{\mathbf{Y}} = \sigma\left(\sum_j s_j(t) D_j \mathbf{Y} + b(t)\right)
\end{aligned}
$$

D_j - differential operators

Continuous formulation - independent of resolution

Guide to move between scales and add layers

CNN - optimization

- Common - stochastic gradient descent and its variance
- Recent work on stochastic Newton (Nocedal's talk on Fri)
- We use
  - stochastic GN with very large sample size - SAA
  - Variable projection for convex variables

Computational bottleneck

Computing $\mathbf{YK}$

$$\begin{pmatrix} - & \mathbf{y}_1^\top & - \\ - & \vdots & - \\ - & \vdots & - \\ - & \vdots & - \\ - & \mathbf{y}_s^\top & - \end{pmatrix} \begin{pmatrix} & & \\ & \mathbf{K} & \\ & & \end{pmatrix}$$

$\mathbf{y}_i^\top$ training image
For large images or 3D computationally expensive

Idea: Train on coarse mesh

Multiscale learning

Restrict the images n times
Initialize the convolution kernels and classifiers
for $k = n : -1 : 1$ **do**
  Solve the optimization problem on mesh k
  from its initial point
  Prolong the convolution kernel to level $k-1$
  Update the classifier weights
end for

How to prolong the kernels?

Moving Kernels between scales

Use multigrid methodology

Approach 1 rediscretization

The Kernel represents a differential operator. Find the operator and rediscretize.
Example in 1D

- Convolution kernel $[-1 \quad -2 \quad 1]$ $\qquad h = 1$
- Operator: $-2 + 2\frac{d}{dx} - 0.5\frac{d^2}{dx^2}$
- On a mesh size $h = 2$ the kernel is $[-1 \quad 1 \quad -0.25]$

Moving Kernels between scales

Use multigrid methodology

Approach 2 Galerkin

- In MG we are usually given the fine mesh operator $\mathbf{K}_h$
- The coarse mesh operator is defined as

$$\mathbf{K}_H = \mathbf{R}\mathbf{K}_h\mathbf{P}$$

R - restriction **P** - prolongation

Moving Kernels between scales - Galerkin

$$\mathbf{K}_H = \mathbf{R}\mathbf{K}_h\mathbf{P}$$

Fine $\rightarrow$ Coarse, but not coarse to fine

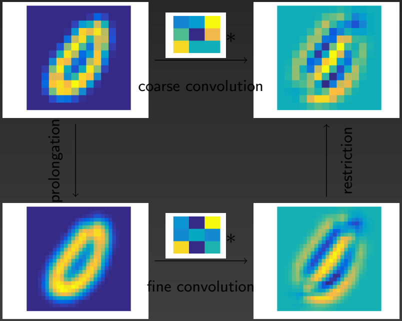
Coarse $\rightarrow$ Fine - in general, non unique

Assume the same sparsity on fine grid leads to a unique solution

Can be computed solving a small linear system (size of stencil)

2D example

Use MNIST library of hand writing



$$\mathbf{K}_h = \begin{pmatrix} -0.89 & -2.03 & 4.30 \\ -2.07 & 0.00 & -2.07 \\ 4.39 & -2.03 & 1.28 \end{pmatrix} \qquad \mathbf{K}_H = \begin{pmatrix} -0.48 & -0.17 & 0.82 \\ -0.15 & -0.80 & 0.37 \\ 0.84 & 0.40 & 0.07 \end{pmatrix}.$$

Training MNIST

MNIST is a data set of labeled images of hand written digits

Goal - automatic classification

Small images 28×28 toy data set

Training MNIST - Experiment 1

Train on fine - classify on coarse

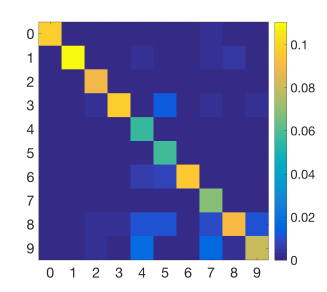
In some cases train using large computational resources

In the "field" poor resources - classify on coarse mesh

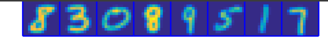
82% success using fine mesh kernels on coarse mesh

Training MNIST - Experiment 1

confusion matrix

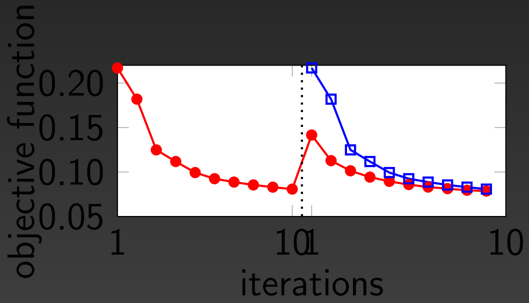


correctly classified as "83089517"



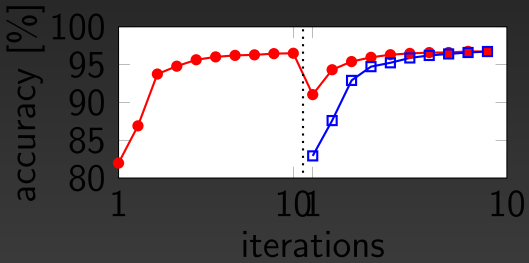
incorrectly classified as "69599339"

Experiment 2: multiscale



Multilevel convergence

Experiment 2: multiscale



Classification accuracy for fine-level and multi-level learning.

Summary

- Proposed a new interpretation to CNN as a PDE optimization problem
- Can move between scales
- Optimization algorithms based on Newton
- Preliminary results on MNIST

To come

- More about optimization
- Depth of network and time
- Adaptive mesh refinement

An add

Call for US students and postdocs from Iran, Iraq, Syria, Yemen, Somalia, Libya and Sudan

If you are a student or a postdoc

1. Admitted or studying at a major US graduate school
2. Working in the field of scientific computing/computational science and engineering with interest in numerical optimization, partial differential equations or machine learning
3. Would like to pursue a PhD degree at the University of British Columbia

I would love to hear from you.