Small Data Deep Learning: AI Applied to Domain Datasets

Michael Majurski
NIST | ITL | SSD | ISG
Outline

- AI Overview
  - Motivation: Why AI on domain datasets?
  - The Deep Learning Revolution
  - Model Training

- Small Data Mitigation Techniques
  - Data Augmentation
  - Transfer Learning
    - Research Datasets
    - Representation Learning

- Ongoing Results
  - RPE Stem Cell Segmentation from 1000 annotations
Motivation

- **Motivation:** enable scientists to use AI based models to derive measurements

- **Significance:** image-based measurements can become more accurate by introducing supervised AI-based models instead of using the traditional machine learning (ML) based models.
Deep Learning: Why do we care?
Deep Learning: Why do we care?

- Has improved modeling accuracy
  - Image classification now has super human performance
  - 25% ImageNet error rate reduced to 2%

- Learns intermediate representations of the data

- Revolutionized how machine translation is done
  - Google translate might be the largest NN in the world right now

- End to end deep learning is out performing human tuned features in almost every application tested
Deep Learning: Why do we care?

“It turns out that a large portion of real-world problems have the property that it is significantly easier to collect the data (or more generally, identify a desirable behavior) than to explicitly write the program.”

– Andrej Karpathy
The Deep Learning Revolution

- **Key Components**
  - Data size
    - Both Annotated and Unannotated
  - Model Capacity
    - How large is the Neural Network
  - Hardware Acceleration
    - Enables Model Training

- **End Goal:**
  - Deep Learning has improved modeling accuracy
Dataset Size

Figure: Research Dataset size plotted against the year released. Source: “Deep Learning” by Ian Goodfellow
**Model Capacity**

**Number of Neurons**

- Perceptron (Rosenblatt, 1958, 1962)
- Adaptive linear element (Widrow and Hoff, 1960)
- Neocognitron (Fukushima, 1980)
- Early back-propagation network (Rumelhart et al., 1986b)
- Recurrent neural network for speech recognition (Robinson and Fallside, 1991)
- Multilayer perceptron for speech recognition (Bengio et al., 1991)
- Mean field sigmoid belief network (Saul et al., 1996)
- LeNet-5 (LeCun et al., 1998b)
- Echo state network (Jaeger and Haas, 2004)
- Deep belief network (Hinton et al., 2006)
- GPU-accelerated convolutional network (Chellapilla et al., 2006)
- Deep Boltzmann machine (Salakhutdinov and Hinton, 2009a)
- GPU-accelerated deep belief network (Raina et al., 2009)
- Unsupervised convolutional network (Jarrett et al., 2009)
- GPU-accelerated multilayer perceptron (Ciresan et al., 2010)
- OMP-I network (Coates and Ng, 2011)
- Distributed autoencoder (Le et al., 2012)
- Multi-GPU convolutional network (Krizhevsky et al., 2012)
- COTS HPC unsupervised convolutional network (Coates et al., 2013)
- GoogLeNet (Szegedy et al., 2014a)

Figure: Research AI model size plotted against the year released. Source: “Deep Learning” by Ian Goodfellow
Model Capacity

Figure: ConvNet model size plotted against forward pass computation cost (x-axis) and ImageNet accuracy.

Hardware Acceleration

- GPU acceleration is an enabler for Deep Learning

- Training Deep Learning models involves lots of linear algebra
  - GPUs are good at linear algebra

- Increased GPU GFlops for training larger models

Figure: Correlation between model size and GPU compute power. Source: https://ai.googleblog.com/2017/07/revisiting-unreasonable-effectiveness.html
Research Datasets vs Domain Datasets

- Gathering annotations is:
  - Tenedious (error prone)
  - Time consuming
  - Expensive

- ImageNet has 1M+ annotations.
  - Result of considerable effort over multiple years
  - Recent NIST domain dataset: 1000 annotations
    - [https://isg.nist.gov/deepzoomweb/data/RPEimplants](https://isg.nist.gov/deepzoomweb/data/RPEimplants)

- We cannot put forth that type of labeling effort for every new domain problem we encounter
  - Not a practical cost to benefit
Model Training
Model Fitting

- Machine Learning is fitting a function to data

- Performance metric needed to judge quality of fit
  - Metric is actively optimized over the training data
  - Model accuracy is evaluated using the metric on unseen test data
  - Cannot use data the model has seen to create an unbiased estimate of the accuracy

- Split limited annotations into
  - Training group (80%)
  - Testing group (20%)

Figure: example regression model fit
Source: “Deep Learning” by Ian Goodfellow
Model Optimization - SGD

- **Model Training/Optimization Steps**
  1. Initialize all model parameters with random values (zero mean, small variance)
  2. Compute loss/error for a batch of the training data
  3. Compute the gradient of that loss surface
  4. Use the gradient to update all parameters to reduce the loss value
  5. Repeat 2-4 until converged

- Each iteration improves the model slightly

**Gradient Descent Path**

**Local Gradient**

Figure: example SGD path through loss surface.

Source: “Deep Learning” by Ian Goodfellow
Model Optimization

- Loss is a function of every parameter in the model
  - Very high dimensional (millions of dimensions)

- Stochastic Gradient Descent (SGD) algorithm
  - Walks downhill on the loss surface finding sets of parameters with lower loss values
  - Uses gradient information to descend the loss surface
  - Minimizes loss, but no guarantee of global minima
  - Empirical evidence suggests that most local minima are equivalent
A machine learning practitioner has two goals for every model:
- Make the training error small
- Make the gap between training and test error small

Underfitting: when a model cannot reach an acceptable training error

Overfitting: when a model has too large a gap between train and test error

Figure: training accuracy convergence curves for UNet semantic segmentation CNN. Accuracy as a function of training step.
Generalization

“The central challenge in machine learning is that we must perform well on new, previously unseen inputs – not just those on which our model was trained. The ability to perform well on previously unseen inputs is called generalization.”

– Ian Goodfellow
Small Data Mitigation

annotated
Small Data Mitigation Techniques

1. **Data Augmentation**
   - Create label preserving transformations of your data
   - Builds invariances into your model

2. **Transfer Learning**
   - Build a model on a large dataset before refining on your domain specific data
   - Research Datasets
     - Annotations from different domain
   - Generative Adversarial Networks (GANs)
     - Use your unlabeled data to learn a good representation
Label Preserving Transformations

- **Data augmentation**: popular technique for generating additional labeled training examples through class-preserving transformations
- Critical to almost every current state of the art result

<table>
<thead>
<tr>
<th>Model Objective</th>
<th>Augmentation Model</th>
<th>Parameterization</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Invariance</strong></td>
<td>Rotation</td>
<td>Uniform (random angle)</td>
</tr>
<tr>
<td></td>
<td>Reflection (x,y)</td>
<td>Bernoulli</td>
</tr>
<tr>
<td></td>
<td>Jitter (x,y)</td>
<td>% of image size</td>
</tr>
<tr>
<td><strong>Robustness</strong></td>
<td>Noise</td>
<td>% change</td>
</tr>
<tr>
<td></td>
<td>SNR</td>
<td>% change</td>
</tr>
<tr>
<td><strong>Reproducibility</strong></td>
<td>Scale (x,y)</td>
<td>% change</td>
</tr>
<tr>
<td></td>
<td>Shear (x,y)</td>
<td>% change</td>
</tr>
</tbody>
</table>

Table: set of commonly used data augmentation models.
Label Preserving Transformations

Figure: Label preserving data augmentation transformations applied to ‘6’ from MNIST dataset.
Source: https://dawn.cs.stanford.edu/2017/08/30/tanda/
## Literature Survey of Label Preserving Transformations

<table>
<thead>
<tr>
<th>Augmentation Method</th>
<th>Papers Using these Methods</th>
</tr>
</thead>
<tbody>
<tr>
<td>cutout</td>
<td>ImageNet Classification with Deep Convolutional Neural Networks</td>
</tr>
<tr>
<td>mixup</td>
<td>Applying Data Augmentation to Handwritten Arabic Numeral Recognition Using Deep Learning Neural Networks</td>
</tr>
<tr>
<td>cutmix</td>
<td>Understanding data augmentation for classification: when to warp?</td>
</tr>
<tr>
<td>sample paring</td>
<td>Return of the Devil in the Details: Delving Deep into Convolutional Nets</td>
</tr>
<tr>
<td>Jitter</td>
<td>Very Deep Convolutional Networks for Large-Scale Image Recognition</td>
</tr>
<tr>
<td>scale</td>
<td>Some Improvements on Deep Convolutional Neural Network Based Image Classification</td>
</tr>
<tr>
<td>shear</td>
<td>Improved Regularization of Convolutional Neural Networks with Cutout</td>
</tr>
<tr>
<td>sharpness</td>
<td>Improving the Robustness of Deep Neural Networks via Stability Training</td>
</tr>
<tr>
<td>blur</td>
<td>Data Augmentation by Pairing Samples for Images Classification</td>
</tr>
<tr>
<td>contrast</td>
<td>mixup: Beyond Empirical Risk Minimization</td>
</tr>
<tr>
<td>blur</td>
<td>The Effectiveness of Data Augmentation in Image Classification using Deep Learning</td>
</tr>
<tr>
<td>color shift</td>
<td>Population Based Augmentation: Efficient Learning of Augmentation Policy Schedules</td>
</tr>
<tr>
<td>Rotation</td>
<td>CutMix: Regularization Strategy to Train Strong Classifiers with Localizable Features</td>
</tr>
<tr>
<td>reflection</td>
<td></td>
</tr>
<tr>
<td>invert</td>
<td></td>
</tr>
<tr>
<td>auto-contrast</td>
<td></td>
</tr>
<tr>
<td>jpeg compression</td>
<td></td>
</tr>
<tr>
<td>elastic deformation</td>
<td></td>
</tr>
</tbody>
</table>
Transfer Learning: General Approach

- Leverage a large research dataset
  - ImageNet/COCO
- Pretrain your model using the large dataset
- Save the model weights
- Load pre-trained weights
- Refine (continue training) on your domain data

Figure: Overview of training a network on COCO before transferring those weights to the target application.
Typical Source Datasets

- **COCO - Common Objects in Context**
  - Semantic image segmentation
  - 200K images over 80 categories

- **COCO-Stuff**
  - Semantic image segmentation
  - Extension of COCO with stuff classes
  - 176K images over 172 categories

- **ImageNet – ILSVRC**
  - Image classification
  - 1.2M images over 1000 categories
Unsupervised Representation Learning

- GANs operate on unannotated data
- Setup two networks in competition
  - Discriminator: tries to determine if an images is real or fake
  - Generator: tries to construct a realistic image from latent noise
- Networks compete until they find an equilibrium.
  - Neither can improve without reducing the accuracy of the opponent
- Website lets you play with a GAN to see how they work/converge
  - https://poloclub.github.io/ganlab/
GAN Representation Learning

Figure: Simplified outline of GAN architecture using MNIST data.
Source: https://towardsdatascience.com/understanding-generative-adversarial-networks-4dacf963f2ef
Leveraging the Learned Representation

- How are GANs useful for small data?

- Trained with unannotated data!

- Build an internal representation useful for fooling the discriminator

- We can leverage the learned representation for transfer learning

Figure: Overview of training a network on unannotated domain data before transferring those weights to the target application.
Leveraging the Learned Representation

- The fundamental usefulness of unsupervised representation learning is to start the network with features that will be useful for its task, instead of random weights.

- Weight Initialization Methods
  - Transfer Learning
  - Semi/Self-Supervised Learning
    - GANs
    - Auto-Encoders
Example Application

RPE Stem Cell Segmentation (CVMI @ CVPR)

Code:
https://github.com/usnistgov/small-data-cnns

Paper:
Motivation – Non Destructive QA/QC

- Age Related Macular Degeneration
- Caused by loss of rod and cone cells due to Retinal Pigment Epithelial (RPE) cell death
- New Induced Pluripotent Stem Cell (iPSC) implant treatments
- Cell Implants require quality control
  - Destructive testing
    - Trans-Epithelial Resistance (TER)
    - Vascular Endothelial Growth Factor (VEGF)
  - Non-Destructing testing
    - Imaging based assays
Problem - QA/QC Via Image Segmentation

- AMD iPSC implant quality control image assays
  - Segment boundaries between cells to determine junction quality
- Small/Limited Domain Datasets

![Example of brightfield modality Absorbance image (left), ground truth mask (center), and reference fluorescent image (right) used to create the ground truth.](image)

1000 Annotated Images
80,400 Unannotated images
Data Available: [isg.nist.gov](http://isg.nist.gov)
Experimental Configuration

- Subset Training Annotations: \{50, 100, 200, 300, 400, 500\}
- Test Annotations: \{500\}
- 6 Model Configurations
  - \{Baseline, TL-COCO, TL-GAN\} × \{With Aug, Without Aug\}
  - 1 Model: UNet
  - 1 Set of hyperparameters

Data Augmentation Models

<table>
<thead>
<tr>
<th>Augmentation Model</th>
<th>Parameterization</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rotation</td>
<td>Uniform</td>
</tr>
<tr>
<td>Reflection</td>
<td>Bernoulli</td>
</tr>
<tr>
<td>Translation</td>
<td>Uniform ±10% Image Size</td>
</tr>
<tr>
<td>Scale</td>
<td>Uniform ±10% Image Size</td>
</tr>
</tbody>
</table>
Baseline Configuration

- Train UNet directly on the varying number of annotations

Transfer Learning Configurations

- **TL-COCO**
  - Train UNet to convergence on out of domain COCO dataset
    - 200K images over 80 categories
  - Initialize weights with parameters learned from COCO
  - Refine model on \( N \) domain annotations

- **TL-GAN**
  - Train UNet GAN to convergence on unannotated domain data
    - 80,400 RPE Absorbance Images
  - Initialize encoder model weights with the discriminator from the GAN
  - Refine on \( N \) domain annotations
TL-GAN

- Trains UNet weights to produce realistic fake images
- Architecture motivated by DCGAN and adapted to UNet
Example GAN Images

Gan Fake: Epoch 0

Real:

Gan Fake: Epoch 400

Fake:
Results – Without Augmentation

Contour Dice - Without Augmentation

Adjusted Rand Index - Without Augmentation
Results – With Augmentation

Contour Dice - With Augmentation

Adjusted Rand Index - With Augmentation
Segmentation Results

<table>
<thead>
<tr>
<th>Absorbance Image</th>
<th>Ground Truth Segmentation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline No-Augmentation</td>
<td>Baseline With-Augmentation</td>
</tr>
<tr>
<td>TL-COCO No-Augmentation</td>
<td>TL-COCO With-Augmentation</td>
</tr>
<tr>
<td>TL-GAN No-Augmentation</td>
<td>TL-GAN With-Augmentation</td>
</tr>
</tbody>
</table>
Result Summary

- TL-COCO outperforms TL-GAN representation learning
  - This matches trends in big data ConvNets
- DICE metric: domain knowledge driven data augmentation is optimal
- ARI metric: TL-COCO is optimal
  - Hypothesis: structure learned from COCO benefits cell edge segmentation
- GPU Costs of performing transfer learning:

<table>
<thead>
<tr>
<th>Training Configuration</th>
<th>GPU Time*</th>
</tr>
</thead>
<tbody>
<tr>
<td>TL-COCO (pretrain + refine)</td>
<td>4036 + 78 min</td>
</tr>
<tr>
<td>TL-GAN (pretrain + refine)</td>
<td>3120 + 78 min</td>
</tr>
<tr>
<td>Baseline (refine)</td>
<td>78 min</td>
</tr>
</tbody>
</table>

*These times were generated on a single IBM “Witherspoon” node containing two 20-core Power9 CPUs and four Nvidia V100 GPUs with NVLink2 interconnection fabric. Data augmentation has no impact on runtimes.
Summary: Small Data Mitigation Techniques

- **Data Augmentation**
  - Create label preserving transformations

- **Transfer Learning**
  - Leverage a model trained for a different task
    - Research Datasets
    - Unannotated Data
  - Refine the model on the limited domain data
Compute/Code Resources

- NIST GPU cluster: “Enki”
  - https://gitlab.nist.gov/gitlab/aihpc/pages/wikis/home

- ConvNet (CNN) Code ready for Enki
  - Single-Node Multi-GPU
    - Tensorflow 1.12 and 2.0
  - Semantic Segmentation: https://github.com/usnistgov/semantic-segmentation-unet
  - Classification: https://gitlab.nist.gov/gitlab/mmajursk/Classification
  - Regression: https://gitlab.nist.gov/gitlab/mmajursk/Regression
  - Object Detection: https://gitlab.nist.gov/gitlab/mmajursk/Object-Detection
Thank you

Questions?