Set2Model Networks: Learning Discriminatively To Learn Generative Visual Models

Alexander Vakhitov
Andrey Kuzmin
Victor Lempitsky

Skoltech
Christmas Colloquium 2017
Set2Model Networks

- Web-initialized image search
- Meta-learning
  - Approaches overview
  - Notable works
- Set2Model
  - Learning within a task
  - Learning across tasks
- Experiments
- Live demo
- Conclusions
Web-initialized image search

- Problem: to find a certain visual concept in the image collection
- Solution:
  - get ‘training’ images using Internet image search
  - build a concept model in the latent space
  - use the model to compute relevance of images
One (few) shot learning

- A child generalizes a new concept from a single picture
  but
- Deep learning systems need vast datasets to train

=> Need for new learning mechanisms
Meta-learning

“Learning to learn” loop:

Learn within a task

Task solution (e.g., classifier)

Learn across tasks, $\theta$

Task $T$
Meta-learning cost function

\[ \theta_* = \arg\min_{\theta} \mathbb{E}_{D \sim p(D)} L(D; \theta) \]

- Dataset \( D = \{d_t\} = \{(x_t, y_t)\} \)
- \( L(\theta, D) \) is a usual learning cost
- Both regression/classification possible
- \( p(D) \) can combine two distributions:
  - on classes (labels)
  - on samples
- Ideally, we wish to output \( p(y_t | X_t, D_{1:t-1}; \theta) \) with input \( X_t \)
Omniglot: MNIST “transposed”

- Each character is a class
- “MNIST transposed” 1623 classes, 20 images
- Training/testing classes are different
- To the right: 20-way 5-shot task
(Lake et al., 2011)
Meta-learning architecture requirements

According to (Santoro et al., 2016), meta-learning requires:

1) Elementwise accessible stable memory $M$
2) No. of parameters $\theta$ not tied to the size of the memory $M$
Some approaches

Denote embedding function as $g$...

- Metric learning + nearest neighbor
  - $M = g(D)$, $\theta$ describes the metric
e.g. Siamese nets (Koch et al., 2015)

- Aggregation models
  - NCM: $M = \text{mean descriptors}$
    (Mensink et al., 2012)
  - Set2Model: $M = \text{Gaussian mixture} (g(D))$

- Memory-augmented networks
  - Neural Turing Machine: $M$ in the machine
    (Santoro et al., 2016)
  - Matching nets: $M = g(D) \cup f(g(D), \theta)$
    (Vinyals et al., 2016)
Siamese net solution

- Siamese net trained with verification loss (0 – same, 1 – different class)
- Nearest neighbor with Siamese net as a metric
- Evaluation: M is the training set, θ describes the net
Nearest class mean
(Mensink et al., 2012)

- SIFT-based features
- Generalization to unseen classes with and without prior

\[ i^*_i = \arg\min_i \text{softmax} \{\rho(m_i, x)\} , \]
\[ m_i \] - mean descriptor for class i
Meta-learning with Memory Augmented NN

1st phase: store data-label association in M
2nd phase: retrieve label for data, back-propagate

(Santoro et al., ICML 2016)
Matching Networks for One Shot Learning

- Use attention kernel $a(x, x_i)$ where $x$ is a test example, $(x_i, y_i) \in D$
  $$y = \sum_i a(x, x_i) y_i$$
- Omniglot: “Soft” nearest neighbor
  $$a(x, x_i) = \text{softmax}(x^T x_i)$$
- ImageNet/Language: Attention constructed by LSTMs
- Training loops:
  - choose labels $y_i$
    - choose samples $x_i$
  - learn for $D$

(Vinyals et al., NIPS 2016)
Set2Model Network: learning within a task

X - class examples
Y - test examples
(* - we use AlexNet, can be any Deep Net
(+) - we use Gaussian mixture as a model
Set2Model Layer

- Normalize a descriptor in $l_2$
- Build a Gaussian mixture model with $k_* = 1..4$ components
  - $k_*$ fixed a priori, or
  - $k_*$ chosen using BIC (Bayesian Information Criterion):
    
    $$k_* = \text{argmin}_k 2k n \log(N) - 2 \log p(X|X),$$
    
    where $n$ is a descriptor dimension, $N = |X|$
Set2Model Network: learning across tasks

\[ S2M^+ \]

\[ p(Y_{+/-}|X) \]

\[ \text{Hist Loss} \]

\[ \text{dHLoss/dθ} \]

\[ X - \text{class examples} \]

\[ Y_+, Y_- - \text{positive/negative test examples} \]

\[ (*) - \text{we use AlexNet, can be any Deep Net} \]

\[ (\dagger) - \text{we use Gaussian mixture as a model} \]
Histogram Loss

\[ d_{+/-} = p(Y_{+/-} | X) \]
\[ h_{+/-} = \text{histogram} \ (d_{+/-}) \]

\[ L(d_+, d_-) = \sum_i h_+[i] \sum_{j>i} h_-[j] \]
Backpropagation through the S2M Layer: implicit diff

- $\nabla_{Y \pm}$ Loss: closed form expression
- Focus on $\nabla_X$ Loss
- Denote GMM parameters as $q^*$
- $\nabla_q$ Loss: closed form expression; $\nabla_X q = ?$
  \[ q^* = \text{argmax } p_{GMM}(X; q) \]

\[ \nabla_q p_{GMM}(X; q) = 0 \]

\[ \nabla_{qq}^2 p_{GMM}(X; q) \nabla_X q + \nabla_{qX}^2 p_{GMM}(X; q) = 0 \]

Solve a linear system w.r.t. $\nabla_X q$. 
Experiments

- Baselines (pre-learned & fine-tuned)
  - Nearest neighbor
  - SVM 1-vs-all
  - Nearest class mean
- Datasets
  - Oxford flowers
  - RGBD Object
  - ImageNet
  - Omniglot
- Test & train classes do not intersect
Implementation

- Caffe/Python
- Fine-tuning of AlexNet provided with Caffe (except Omniglot)
- Live demo
Oxford Flowers

- Train 80 / Test 22 classes
- (Nilsback & Zisserman, 2008)
RGBD Object

- Train 40 / Test 11 classes
- (Lai et al., 2011)
- Train 608 / Test 91 classes
- Random classes, not from ILSVRC’12
Query = ‘crucian carp’ (left), ‘silverbush’ (right)
'supernova' (left), 'seal' (right)

- Relevance by the mixture components
Query = ‘pinto’

green: web search images,
blue: ImageNet class images
Omniglot

S2M: 1-vs-all training
Others: n-way cross entropy training

<table>
<thead>
<tr>
<th>Model</th>
<th>5-way</th>
<th>20-way</th>
</tr>
</thead>
<tbody>
<tr>
<td>Matching Nets</td>
<td>0.989</td>
<td>0.985</td>
</tr>
<tr>
<td>MANN (no Conv)</td>
<td>0.949</td>
<td>-</td>
</tr>
<tr>
<td>Convolutional Siamese net</td>
<td>0.984</td>
<td>0.965</td>
</tr>
<tr>
<td>S2M-Gauss</td>
<td>0.985</td>
<td>0.956</td>
</tr>
</tbody>
</table>
Conclusions

- Set2Model is a method for discriminative training of generative visual models
- Backprop using implicit differentiation
- Generalizes well to unseen classes
- Tested on 4 datasets
- CVIU’2017, ICCV Workshop’2017

a.vakhitov @ skoltech.ru
Alexander Vakhitov
Senior researcher
Skoltech, CV group