Review Series of Recent Deep Learning Papers:

Parameter Prediction Paper: Learning Feed-Forward One-Shot Learners

Luca Bertinetto, João F. Henriques, Jack Valmadre, Philip H. S. Torr, Andrea Vedaldi

NIPS 2016

Reviewed by : Arshdeep Sekhon

[1]Department of Computer Science, University of Virginia
https://qdata.github.io/deep2Read/

August 25, 2018

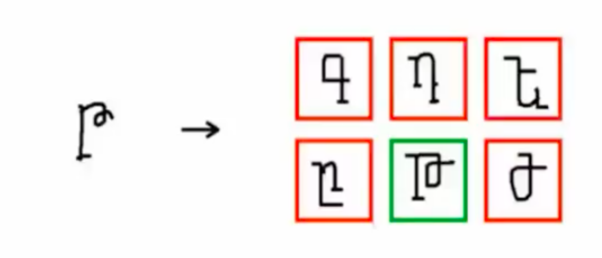
One Shot Learning

Learn a concept (classifier) from a single example

Learning Feed-Forward One-Shot Learners

One Shot Learning

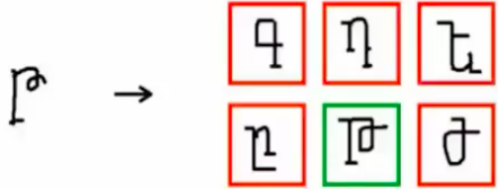
Learn a concept (classifier) from a single example



Task: Identify a character from the Armenian Alphabet

One Shot Learning

Learn a concept (classifier) from a single example



Task: Identify a character from the Armenian Alphabet

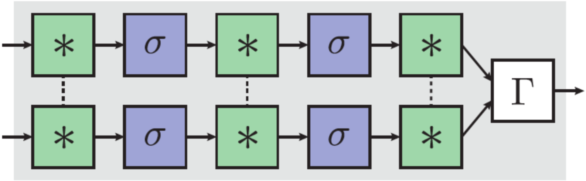
The model hasn't seen the alphabet during training

Learning Feed-Forward One-Shot Learners

One Shot Learning

Learn a concept (classifier) from a single example

Offline Training



$$\min_{W'} \frac{1}{n} \sum_{i=1}^{n} Loss(< \varphi(x_i; W), \varphi(z_i; W) >), \ell_i) \quad (1)$$

Dynamic Parameter Prediction and Learning to Learn Approach

The Task

Given one exemplar, recognize other instances of the same class.

Dynamic Parameter Prediction and Learning to Learn Approach

The Task

Given one exemplar, recognize other instances of the same class.

Dynamic Parameter Prediction: Online Phase

Learnet

predictor

Dynamic Parameter Prediction and Learning to Learn Approach

The Task

Given one exemplar, recognize other instances of the same class.

Dynamic Parameter Prediction: Online Phase

Learnet

Given a single example z predict parameters for predictor

$$W(z) = \omega(z, W^{'}) \tag{2}$$

predictor

$$y = \varphi(x, W(z)) \tag{3}$$

Dynamic Parameter Prediction and Learning to Learn Approach

The Task

Given one exemplar, recognize other instances of the same class.

Dynamic Parameter Prediction: Online Phase

Learnet

Given a single example z predict parameters for predictor

$$W(z) = \omega(z, W^{'}) \tag{2}$$

predictor

$$y = \varphi(x, W(z)) \tag{3}$$

'Learning to Learn' approach

Standard Discriminative Learning vs One shot Discriminative learning

Standard Discriminative Learning

- $$\min_{W} \frac{1}{n} \sum_{i=1}^{n} Loss(\varphi(x_i, W), \ell_i) \quad (4)$$

Standard Discriminative Learning vs One shot Discriminative learning

Standard Discriminative Learning

- $$\min_{W} \frac{1}{n} \sum_{i=1}^{n} Loss(\varphi(x_i, W), \ell_i) \tag{4}$$
- Also add regularization
- However, not enough for one shot learning still.

Standard Discriminative Learning vs One shot Discriminative learning

Standard Discriminative Learning

- $$\min_{W} \frac{1}{n} \sum_{i=1}^{n} Loss(\varphi(x_i, W), \ell_i) \tag{4}$$
- Also add regularization
- However, not enough for one shot learning still.
- Solution: Learning to learn
- Inject prior information in the task

Standard Discriminative Learning vs One shot Discriminative learning

Standard Discriminative Learning

- $$\min_{W} \frac{1}{n} \sum_{i=1}^{n} Loss(\varphi(x_i, W), \ell_i) \tag{4}$$
- Also add regularization
- However, not enough for one shot learning still.
- Solution: Learning to learn
- Inject prior information in the task
- Prior Information about the learning domain is introduced in the offline phase.

Training the Learnet

The Learnet

$$W = \omega(z_i, W^{'}) \tag{5}$$

Training the Learnet

The Learnet

$$W = \omega(z_i, W^{'}) \quad (5)$$

Task: Find optimal parameters $W^{'}$ of the learnet.

Training the Learnet

The Learnet

$$W = \omega(z_i, W') \tag{5}$$

Task: Find optimal parameters W' of the learnet.

old standard discriminative objective function

$$\min_{W} \frac{1}{n} \sum_{i=1}^{n} Loss(\varphi(x_i, W), \ell_i) \tag{6}$$

ℓ_i is true label of x_i

Training the Learnet

The Learnet

$$W = \omega(z_i, W^{'}) \tag{5}$$

Task: Find optimal parameters $W^{'}$ of the learnet.

old standard discriminative objective function

$$\min_{W} \frac{1}{n} \sum_{i=1}^{n} Loss(\varphi(x_i, W), \ell_i) \tag{6}$$

ℓ_i is true label of x_i

New One Shot discriminative objective function

$$\min_{W'} \frac{1}{n} \sum_{i=1}^{n} Loss(\varphi(x_i, \omega(z_i, W^{'})), \ell_i) \tag{7}$$

ℓ_i is positive if (x_i, z_i) are of the same class.

Training the Learnet

New One Shot discriminative objective function

$$\min_{W'} \frac{1}{n} \sum_{i=1}^{n} Loss(\varphi(x_i, \omega(z_i, W^{'})), \ell_i) \tag{8}$$

ℓ_i is positive if (x_i, z_i) are of the same class.
Training Data: labeled sample pairs (x_i, ℓ_i) and (z_i, ℓ_i)
triplets (x_i, z_i, ℓ_i)

The challenge

A fully connected linear layer:

$$y = Wx + b \tag{9}$$

$x\epsilon\ \mathbb{R}^d$, ouptuts $y\epsilon\ \mathbb{R}^k$, weights $W\epsilon\ \mathbb{R}^{d\times k}$ and biases $b\epsilon\ \mathbb{R}^k$

The challenge

A fully connected linear layer:

$$y = Wx + b \tag{9}$$

$x \epsilon \ \mathbb{R}^d$, ouptuts $y \epsilon \ \mathbb{R}^k$, weights $W \epsilon \ \mathbb{R}^{d \times k}$ and biases $b \epsilon \ \mathbb{R}^k$

The Learnet

$$W = \omega(z_i, W') \tag{10}$$

$$y = \omega(z)x + b(z) \tag{11}$$

$$\omega : \mathbb{R}^m \rightarrow \mathbb{R}^{d \times k} \tag{12}$$

A fully connected linear layer:

$$y = Wx + b \tag{9}$$

$x \epsilon \ \mathbb{R}^d$, ouptuts $y \epsilon \ \mathbb{R}^k$, weights $W \epsilon \ \mathbb{R}^{d \times k}$ and biases $b \epsilon \ \mathbb{R}^k$

The Learnet

$$W = \omega(z_i, W^{'}) \tag{10}$$

$$y = \omega(z)x + b(z) \tag{11}$$

$$\omega : \mathbb{R}^m \rightarrow \mathbb{R}^{d \times k} \tag{12}$$

Assuming the Learnet is also a linear layer:

$$\omega(z) = W'z \tag{13}$$

A fully connected linear layer:

$$y = Wx + b \tag{9}$$

$x \epsilon\ \mathbb{R}^d$, ouptuts $y \epsilon\ \mathbb{R}^k$, weights $W \epsilon\ \mathbb{R}^{d \times k}$ and biases $b \epsilon\ \mathbb{R}^k$

The Learnet

$$W = \omega(z_i, W') \tag{10}$$

$$y = \omega(z)x + b(z) \tag{11}$$

$$\omega : \mathbb{R}^m \rightarrow \mathbb{R}^{d \times k} \tag{12}$$

Assuming the Learnet is also a linear layer:

$$\omega(z) = W'z \tag{13}$$

Learnet needs to learn $d \times k \times m$ parameters. d=k=100 and an exemplar with 100 features, total learnet parameters: 1 million.

The solution: Reducing output space of the learnet

Factorized Linear Layers

1. Inspired by SVD
   $Wx = Udiag(s)V^Tx$

The solution: Reducing output space of the learnet

Factorized Linear Layers

1. Inspired by SVD
   $Wx = Udiag(s)V^Tx$
2. $$W(z).x = M'diag(\omega(z))Mx \tag{14}$$
   $M \, \epsilon \mathbb{R}^{d\times d}$ and $M' \, \epsilon \, \mathbb{R}^{d\times k}$
   Offline Phase: Learn constant basis U and V
   Online Phase(Test time): predict weights of diagonal transform

The solution: Reducing output space of the learnet

Factorized Linear Layers

1. Inspired by SVD
   $Wx = Udiag(s)V^T x$
2. $$W(z).x = M' diag(\omega(z)) Mx \tag{14}$$
   $M \, \epsilon \mathbb{R}^{d \times d}$ and $M' \, \epsilon \, \mathbb{R}^{d \times k}$
   Offline Phase: Learn constant basis U and V
   Online Phase(Test time): predict weights of diagonal transform
3. Now, the learnet needs to predict just d parameters. $\omega(z) : \mathbb{R}^m \rightarrow \mathbb{R}^d$

Extending to CNNs

A convolutional layer:

$$y = W * x + b \tag{15}$$

$x \,\epsilon\, \mathbb{R}^{r \times c \times d}$, $W \,\epsilon\, \mathbb{R}^{f \times f \times d \times k}$, $y \,\epsilon\, \mathbb{R}^{r' \times c' \times k}$ d: the number of input channels, f: filter size, k output channels

Extending to CNNs

A convolutional layer:

$$y = W * x + b \tag{15}$$

$x \, \epsilon \, \mathbb{R}^{r \times c \times d}$, $W \, \epsilon \, \mathbb{R}^{f \times f \times d \times k}$, $y \, \epsilon \, \mathbb{R}^{r' \times c' \times k}$ d: the number of input channels, f: filter size, k output channels

The number of parameters to be predicted by learnet are $f^2 dk$.

Extending to CNNs

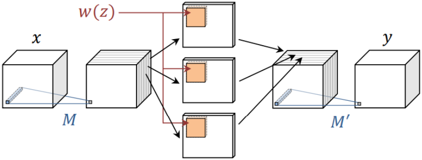
Factorize:

$$y = M' * w(z) *_d M * x + b(z) \tag{16}$$

$M \, \epsilon \mathbb{R}^{1\times1\times d\times d}$, $M' \, \epsilon \mathbb{R}^{1\times1\times d\times k}$, $w(z) \, \epsilon \mathbb{R}^{1\times f\times f\times d}$

$*_d$ does independent filtering of d channels:

$x *_d y$ is the convolution of corresponding channels in x and y.



Factorized Convolutional layer

Extending to CNNs

Factorize:

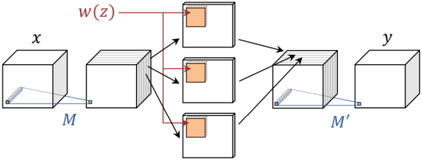
$$y = M' * w(z) *_d M * x + b(z) \tag{16}$$

$M \ \epsilon \mathbb{R}^{1\times1\times d\times d}$, $M' \ \epsilon \mathbb{R}^{1\times1\times d\times k}$,$w(z) \ \epsilon \mathbb{R}^{1\times f\times f\times d}$

$*_d$ does independent filtering of d channels:

$x *_d y$ is the convolution of corresponding channels in x and y.

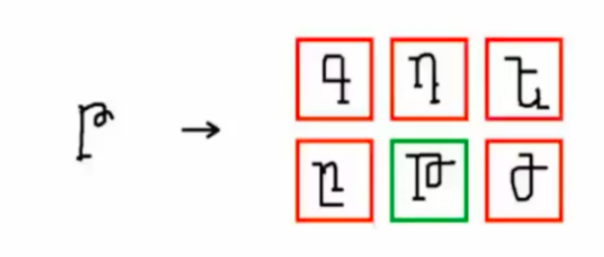
The number of elements to be predicted by learnet now are : $f^2 d$



Factorized Convolutional layer

An example: Character Recognition in Alphabets

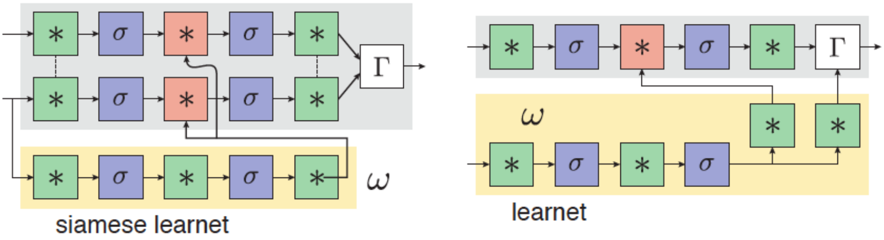
Test Phase(Online):



An example: Character Recognition in Alphabets

Training Phase(Offline):

Architectures



Model	Error Percentage
Siamese (shared)	41.8
Siamese (unshared)	34.6
Siamese (unshared, factorized)	33.6
Siamese Learnet (shared)	31.4
Learnet	28.6