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

1Department 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

Reviewed by: Arshdeep Sekhon (University of Virginia)
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
Siamese Architecture

\[
\min_{W'} \frac{1}{n} \sum_{i=1}^{n} \text{Loss}(\langle \varphi(x_i; W), \varphi(z_i; W) \rangle, \ell_i)
\]  

(1)
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

\[ \text{Given a single example } z \]
\[ \text{predict parameters for predictor } W(z) = \omega(z, W') \] (2)

predictor \[ y = \phi(x, W(z)) \] (3)

'Learning to Learn' approach

1. Requires sufficient prior information about the learning domain.
2. Offline Phase: Solve many one shot learning tasks and backpropagate errors end-to-end.

Reviewed by: Arshdeep Sekhon (University of Virginia)

Review Series of Recent Deep Learning Papers: Parameter Prediction Paper: Learning Feed-Forward One-Shot Learners

August 25, 2018 5 / 15
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

$$W(z) = \omega(z, W')$$ (2)

**predictor**

$$y = \varphi(x, W(z))$$ (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

Learned
Given a single example $z$ predict parameters for predictor

$$W(z) = \omega(z, W')$$  \(2\)

Predictor

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

'Learning to Learn' approach
Standard Discriminative Learning vs One shot Learning

Discriminative learning

\[ \min_W \frac{1}{n} \sum_{i=1}^{n} \text{Loss}(\varphi(x_i, W), \ell_i) \] (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.
Standard Discriminative Learning vs One shot

Discriminative learning

\[
\min_W \frac{1}{n} \sum_{i=1}^{n} Loss(\phi(x_i, W), \ell_i)
\]  \hspace{1cm} (4)

- Also add regularization
- However, not enough for one shot learning still.
Standard Discriminative Learning vs One shot Discriminative learning

\[ \min_{W} \frac{1}{n} \sum_{i=1}^{n} Loss(\varphi(x_i, W), \ell_i) \] (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

\[
\min_W \frac{1}{n} \sum_{i=1}^{n} \text{Loss}(\phi(x_i, W), \ell_i)
\] (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

\[ W = \omega(z_i, W') \]  

(5)

Task: Find optimal parameters \( W' \) of the Learnet.

Old standard discriminative objective function

\[
\min_{W} \sum_{i=1}^{n} \text{Loss}(\phi(x_i, W), \ell_i) \]  

(6)

\( \ell_i \) is the true label of \( x_i \).

New One Shot discriminative objective function

\[
\min_{W'} \sum_{i=1}^{n} \text{Loss}(\phi(x_i, \omega(z_i, W')), \ell_i) \]  

(7)

\( \ell_i \) is positive if \((x_i, z_i)\) are of the same class.
Training the Learnet

The Learnet

\[ W = \omega(z_i, W') \] (5)

Task: Find optimal parameters \( W' \) of the learnet.
Training the Learnet

The Learnet

\[ W = \omega(z_i, W') \]  

Task: Find optimal parameters \( W' \) of the learnet.

old standard discriminative objective function

\[
\min_{W} \frac{1}{n} \sum_{i=1}^{n} \text{Loss}(\varphi(x_i, W), \ell_i)
\]  

\( \ell_i \) is true label of \( x_i \)
Training the Learnet

The Learnet

\[ W = \omega(z_i, W') \]  \hspace{1cm} (5)

Task: Find optimal parameters \( W' \) of the learnet.

---

**old standard discriminative objective function**

\[
\min_{W} \frac{1}{n} \sum_{i=1}^{n} Loss(\varphi(x_i, W), \ell_i)
\]  \hspace{1cm} (6)

\( \ell_i \) is true label of \( x_i \)

---

**New One Shot discriminative objective function**

\[
\min_{W'} \frac{1}{n} \sum_{i=1}^{n} Loss(\varphi(x_i, \omega(z_i, W')), \ell_i)
\]  \hspace{1cm} (7)

\( \ell_i \) is positive if \((x_i, z_i)\) are of the same class.
Training the Learnnet

New One Shot discriminative objective function

\[
\min_{W'} \frac{1}{n} \sum_{i=1}^{n} \text{Loss}(\varphi(x_i, \omega(z_i, W'))), \ell_i)
\]  

(8)

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

Training Data: labeled sample pairs \((x_i, \ell_i)\) and \((z_i, \ell_i)\) 

triplets \((x_i, z_i, \ell_i)\)
A fully connected linear layer:

\[ y = Wx + b \]  

\( x \in \mathbb{R}^d \), outputs \( y \in \mathbb{R}^k \), weights \( W \in \mathbb{R}^{d \times k} \) and biases \( b \in \mathbb{R}^k \)
The challenge

A fully connected linear layer:

\[ y = Wx + b \]  \hspace{1cm} (9)

\( x \in \mathbb{R}^d \), outputs \( y \in \mathbb{R}^k \), weights \( W \in \mathbb{R}^{d \times k} \) and biases \( b \in \mathbb{R}^k \)

The Learnet

\[ W = \omega(z_i, W') \]  \hspace{1cm} (10)

\[ y = \omega(z)x + b(z) \]  \hspace{1cm} (11)

\( \omega : \mathbb{R}^m \rightarrow \mathbb{R}^{d \times k} \)  \hspace{1cm} (12)
The challenge

A fully connected linear layer:

\[ y = Wx + b \]  \hspace{1cm} (9)

\( x \in \mathbb{R}^d \), outputs \( y \in \mathbb{R}^k \), weights \( W \in \mathbb{R}^{d \times k} \) and biases \( b \in \mathbb{R}^k \)

The Learnet

\[ W = \omega(z_i, W') \]  \hspace{1cm} (10)

\[ y = \omega(z)x + b(z) \]  \hspace{1cm} (11)

\( \omega : \mathbb{R}^m \rightarrow \mathbb{R}^{d \times k} \)  \hspace{1cm} (12)

Assuming the Learnet is also a linear layer:

\[ \omega(z) = W'z \]  \hspace{1cm} (13)
The challenge

A fully connected linear layer:

\[ y = Wx + b \]  \hspace{1cm} (9)

\( x \in \mathbb{R}^d \), output \( y \in \mathbb{R}^k \), weights \( W \in \mathbb{R}^{d \times k} \) and biases \( b \in \mathbb{R}^k \)

The Learnert

\[ W = \omega(z_i, W') \]  \hspace{1cm} (10)

\[ y = \omega(z)x + b(z) \]  \hspace{1cm} (11)

\( \omega : \mathbb{R}^m \rightarrow \mathbb{R}^{d \times k} \)  \hspace{1cm} (12)

Assuming the Learnert is also a linear layer:

\[ \omega(z) = W'z \]  \hspace{1cm} (13)

Learnert needs to learn \( d \times k \times m \) parameters. \( d=k=100 \) and an exemplar with 100 features, total learnert parameters: \( 1 \) million.
The solution: Reducing output space of the learnet

Factorized Linear Layers

Inspired by SVD

\[ Wx = U \text{diag}(s)V^T x \]
The solution: Reducing output space of the learnet

Factorized Linear Layers

1. Inspired by SVD
   \[ Wx = U \text{diag}(s) V^T x \]

2. \[
   W(z)x = M' \text{diag}(\omega(z)) Mx \tag{14}
   \]

   \[ M \in \mathbb{R}^{d \times d} \text{ and } M' \in \mathbb{R}^{d \times k} \]

   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
   \[ Wx = U \text{diag}(s)V^T x \]

2. \[ W(z).x = M' \text{diag}(\omega(z))Mx \] (14)

   \[ M \in \mathbb{R}^{d \times d} \text{ and } M' \in \mathbb{R}^{d \times k} \]

   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. \( \omega(z) : \mathbb{R}^m \rightarrow \mathbb{R}^d \)
A convolutional layer:

\[ y = W * x + b \]  \hspace{1cm} (15)

\( x \in \mathbb{R}^{r \times c \times d}, \ W \in \mathbb{R}^{f \times f \times d \times k}, \ y \in \mathbb{R}^{r' \times c' \times k} \)

d: the number of input channels, f: filter size, k output channels
A convolutional layer:

\[ y = W \ast x + b \]  

\( x \in \mathbb{R}^{r \times c \times d}, \ W \in \mathbb{R}^{f \times f \times d \times k}, \ y \in \mathbb{R}^{r' \times c' \times k} \)

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:

\[ y = M' \ast w(z) \ast_d M \ast x + b(z) \] (16)

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

\( \ast_d \) does independent filtering of \( d \) channels:

\( x \ast_d y \) is the convolution of corresponding channels in \( x \) and \( y \).
Extending to CNNs

Factorize:

\[
y = M' \ast w(z) \ast_d M \ast x + b(z)
\]

\(M \in \mathbb{R}^{1 \times 1 \times d \times d}, \; M' \in \mathbb{R}^{1 \times 1 \times d \times k}, \; w(z) \in \mathbb{R}^{1 \times f \times f \times d}\)

\(\ast_d\) does independent filtering of \(d\) channels:

\(x \ast_d y\) is the convolution of corresponding channels in \(x\) and \(y\).

The number of elements to be predicted by learnet now are: \(f^2 d\)
An example: Character Recognition in Alphabets

Test Phase (Online):
An example: Character Recognition in Alphabets

Training Phase (Offline):
Architectures

**Siamese Learnet**

**Learnet**

<table>
<thead>
<tr>
<th>Model</th>
<th>Error Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Siamese (shared)</td>
<td>41.8</td>
</tr>
<tr>
<td>Siamese (unshared)</td>
<td>34.6</td>
</tr>
<tr>
<td>Siamese (unshared, factorized)</td>
<td>33.6</td>
</tr>
<tr>
<td>Siamese Learnet (shared)</td>
<td>31.4</td>
</tr>
<tr>
<td>Learnet</td>
<td>28.6</td>
</tr>
</tbody>
</table>