Learning to Compare: Relation Network for Few-Shot Learning

Flood Sung  Yongxin Yang\textsuperscript{1}  Li Zhang\textsuperscript{1,2}  Tao Xiang\textsuperscript{1}  Philip H.S. Torr\textsuperscript{2}  Timothy M. Hospedales\textsuperscript{3}
\textsuperscript{1}Queen Mary University of London  \textsuperscript{2}University of Oxford  \textsuperscript{3}The University of Edinburgh
floodsung@gmail.com  \{yongxin.yang, david.lizhang, t.xiang\}@qmul.ac.uk  philip.torr@eng.ox.ac.uk  t.hospedales@ed.ac.uk

Few-shot learning
Meta-learning

https://www.slideshare.net/SimonJohn21/learning-to-compare-relation-network-for-few-shot-learning
Few-shot learning

Must train a classifier to recognize new classes given few examples from each.

Train set (Relatively large)

Support set (Relatively small)

Test set
Few-shot learning

Must train a classifier to recognize new classes given few examples from each.

- Train set (Relatively large)
- Support set (Relatively small)
- Test set

Disjoint label space

Same label space
Relation network--classifying by comparison

Relation network (RN) compares support images and query images and makes classification according to the returned "relation scores"

Support set
(Relatively small)

Support images

Relation scores

Query images

Test set

Discussion: Converting classification task into retrieval task?
Relation network--classifying by comparison

Relation network (RN) learn to compare with meta-learning: to **mimic** the comparison procedure on the training set and learn the model.

In each episode

\[
S = \{(x_i, y_i)\}_{i=1}^{m} \quad (m = K \times C)
\]

\[
Q = \{(x_j, y_j)\}_{j=1}^{n}
\]
Relation network (RN) learn to compare with meta-learning: to mimic the comparison procedure on the training set and learn the model.

In each episode

\[
S = \{(x_i, y_i)\}_{i=1}^{m} \quad (m = K \times C)
\]

\[
Q = \{(x_j, y_j)\}_{j=1}^{n}
\]

Relation scores

C-way

K-shot
Relation network--structure

Figure 1: Relation Network architecture with a 5-way 1-shot 1-query example.

\[ r_{i,j} = g_\phi(C(f_\phi(x_i), f_\phi(x_j))), \quad i = 1, 2, \ldots, C \]
Relation network—structure

Figure 1: Relation Network architecture with a 5-way 1-shot 1-query example.

\[ r_{i,j} = g_\phi(C(f_\varphi(x_i), f_\varphi(x_j))), \quad i = 1, 2, \ldots, C \]

\( r_{i,j} \) is bounded between (0,1) by sigmoid function
Relation network -- structure

Embedding module

Relation module

Optimization target: \( \varphi, \phi \leftarrow \operatorname{argmin}_{\varphi, \phi} \sum_{i=1}^{m} \sum_{j=1}^{n} (r_{i,j} - 1(y_i = y_j))^2 \)

Figure 1: Relation Network architecture with a 5-way 1-shot 1-query example.

\[ r_{i,j} = g_\phi(C(f_\varphi(x_i), f_\varphi(x_j))), \quad i = 1, 2, \ldots, C \]
A detail for K-shot: K embedded features are pooled by pixel-wise sum operation.
Relation network--structure

Detail structure

Extension to 0-shot learning: different embedding module for sample and query images
<table>
<thead>
<tr>
<th>Model</th>
<th>Fine Tune</th>
<th>5-way Acc.</th>
<th>20-way Acc.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>1-shot</td>
<td>5-shot</td>
</tr>
<tr>
<td>MANN [31]</td>
<td>N</td>
<td>82.8%</td>
<td>94.9%</td>
</tr>
<tr>
<td>CONVOLUTIONAL SIAMESE NETS [18]</td>
<td>N</td>
<td>96.7%</td>
<td>98.4%</td>
</tr>
<tr>
<td>CONVOLUTIONAL SIAMESE NETS [18]</td>
<td>Y</td>
<td>97.3%</td>
<td>98.4%</td>
</tr>
<tr>
<td>MATCHING NETS [38]</td>
<td>N</td>
<td>98.1%</td>
<td>98.9%</td>
</tr>
<tr>
<td>MATCHING NETS [38]</td>
<td>Y</td>
<td>97.9%</td>
<td>98.7%</td>
</tr>
<tr>
<td>SIAMESE NETS WITH MEMORY [16]</td>
<td>N</td>
<td>98.4%</td>
<td>99.6%</td>
</tr>
<tr>
<td>NEURAL STATISTICIAN [8]</td>
<td>N</td>
<td>98.1%</td>
<td>99.5%</td>
</tr>
<tr>
<td>META NETS [26]</td>
<td>N</td>
<td>99.0%</td>
<td>-</td>
</tr>
<tr>
<td>PROTOTYPICAL NETS [35]</td>
<td>N</td>
<td>98.8%</td>
<td>99.7%</td>
</tr>
<tr>
<td>MAML [10]</td>
<td>Y</td>
<td>98.7 ± 0.4%</td>
<td>99.9 ± 0.1%</td>
</tr>
<tr>
<td>RELATION NET</td>
<td>N</td>
<td>99.6 ± 0.2%</td>
<td>99.8 ± 0.1%</td>
</tr>
</tbody>
</table>

Table 1: Omniglot few-shot classification. Results are accuracies averaged over 1000 test episodes and with 95% confidence intervals where reported. The best-performing method is highlighted, along with others whose confidence intervals overlap. ‘-’: not reported.
Why does RN work?

Both deep feature embedding and deep distance metric are learnable.

The concatenation operation is in relatively bottom layer

When training a Siamese network or a triplet network, we apply metric constraint on a specified feature and then use the Euclidean distance (or other fixed metric) for metric for inference.
Why does RN work?

Figure 3: An example relation learnable by Relation Network and not by non-linear embedding + metric learning.
Why does RN work?

The feature embeddings are difficult to separate.

The relation module pair representations are linearly separable.

Figure 4: Example Omniglot few-shot problem visualisations. Left: Matched (cyan) and mismatched (magenta) sample embeddings for a given query (yellow) are not straightforward to differentiate. Right: Matched (yellow) and mismatched (magenta) relation module pair representations are linearly separable.
Why does RN work? My guess

1) Feature concatenation operation in very early stage (bottom layers)

2) K sample images to mimic the K-shot

3) Converting classification to “comparison”, which is a semi-parameter model approach.