Matching Networks for One Shot Learning

Yunhan Bai
2018.8.24

OUTLINE

1.Introduction

2.Model

3.Training Strategy

4.Experiments

5.Conclusion

1.Introduction

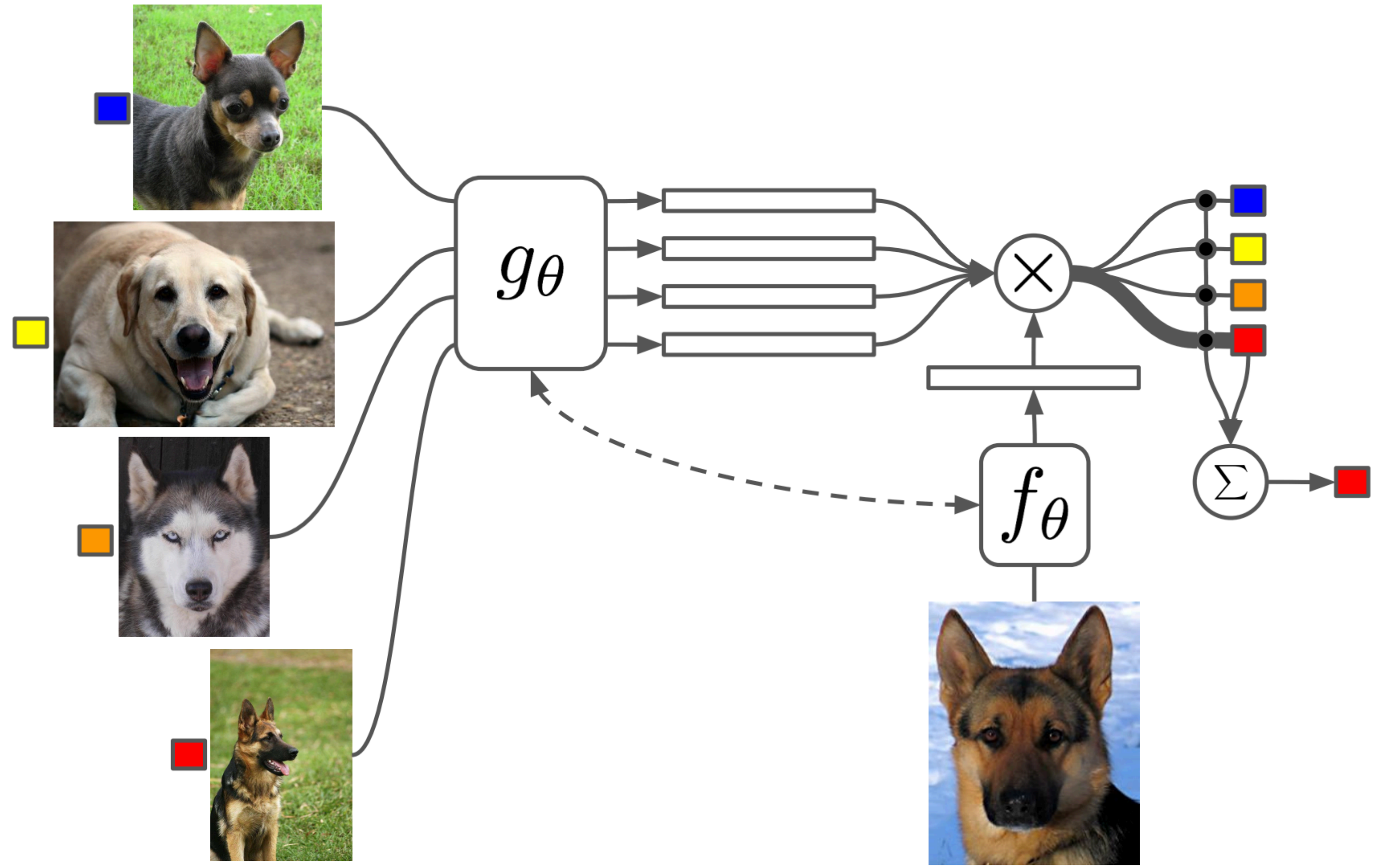
- Learning from a few examples is a key challenge in machine learning.

- Deep learning is powerful, but also notorious for requiring large datasets.

- Non-parametric models allow novel examples to be rapidly assimilated, but the performance depends on the chosen metric.

- How to incorporate parametric deep learning model and non-parametric model, and providing excellent generalization from common examples?

2. Model



2. Model

The basic idea is: build a differentiable nearest neighbor

$$\hat{y} = \sum_{i=1}^{k} a(\hat{x}, x_i) y_i$$

a is a attention kernel,

$$a(\hat{x}, x_i) = e^{c(f(\hat{x}), g(x_i))} / \sum_{j=1}^{k} e^{c(f(\hat{x}), g(x_j))}$$

x_i and y_i comes from **support set S**, f and g are neural networks.

Use **f** and **g** to embed support set examples and test examples, train on it.

2. Model

Full context embeddings

1. Embedding the training examples.

$$
\begin{aligned}
\vec{h}_i, \vec{c}_i &= \mathrm{LSTM}(g'(x_i), \vec{h}_{i-1}, \vec{c}_{i-1}) \\
\overleftarrow{h}_i, \overleftarrow{c}_i &= \mathrm{LSTM}(g'(x_i), \overleftarrow{h}_{i+1}, \overleftarrow{c}_{i+1})
\end{aligned}
$$

$$
g(x_i, S) = \vec{h}_i + \overleftarrow{h}_i + g'(x_i)
$$

It considers not only raw represent of xi, but also relationship with other examples.

2. Model

2. Embedding the test examples.

$$f(\hat{x}, S) = \text{attLSTM}(f'(\hat{x}), g(S), K)$$

$$\begin{aligned}
\hat{h}_k, c_k &= \text{LSTM}(f'(\hat{x}), [h_{k-1}, r_{k-1}], c_{k-1}) \\
h_k &= \hat{h}_k + f'(\hat{x}) \\
r_{k-1} &= \sum_{i=1}^{|S|} a(h_{k-1}, g(x_i)) g(x_i) \\
a(h_{k-1}, g(x_i)) &= \text{softmax}(h_{k-1}^T g(x_i))
\end{aligned}$$

The represent of test example is related to examples in support set.

3. Training Strategy

"one-shot learning is much easier if you train the network to do one-shot learning"

1. Define a task T as distribution over possible label sets L.
2. Sample L from T.
3. Use L to sample support set S and a batch B.
4. Training MN to minimize the error predicting the labels in the B conditioned on S.

$$\theta = \arg\max_{\theta} E_{L\sim T}\left[E_{S\sim L, B\sim L}\left[\sum_{(x,y)\in B} \log P_\theta\left(y|x,S\right)\right]\right].$$

4. Experiments

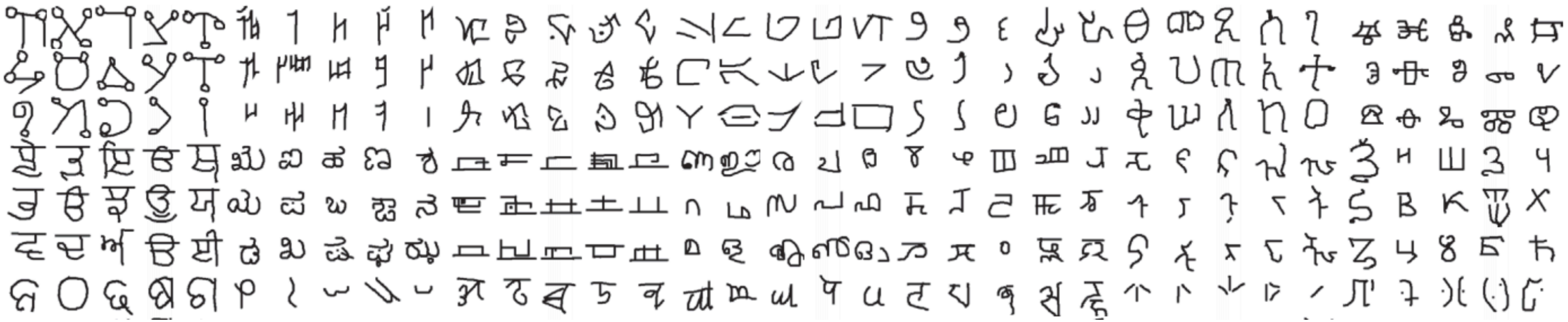
- Task: **N-way k-shot learning task**. i.e. we're given k (e.g. 1 or 5) labelled examples for N classes that we have not previously trained on and asked to classify new instances into he N classes.

- Baseline:
  1. Raw pixels
  2. Base classifier(CNN based)
  3. MANN
  4. Convolutional Siamese Net

Use last layer feature of 2 and 3 to do nearest neighbor.

4. Experiments

Omniglot experiments:



1623 characters of 50 different alphabets.

Stack of 4 cnn modules: 3 * 3 convolution with 64 filter followed by batch normlization, a Relu non-linearity and 2*2 max-pooling.

4. Experiments

Model	Matching Fn	Fine Tune	5-way Acc		20-way Acc	
			1-shot	5-shot	1-shot	5-shot
PIXELS	Cosine	N	41.7%	63.2%	26.7%	42.6%
BASELINE CLASSIFIER	Cosine	N	80.0%	95.0%	69.5%	89.1%
BASELINE CLASSIFIER	Cosine	Y	82.3%	98.4%	70.6%	92.0%
BASELINE CLASSIFIER	Softmax	Y	86.0%	97.6%	72.9%	92.3%
MANN (NO CONV) [21]	Cosine	N	82.8%	94.9%	–	–
CONVOLUTIONAL SIAMESE NET [11]	Cosine	N	96.7%	98.4%	88.0%	96.5%
CONVOLUTIONAL SIAMESE NET [11]	Cosine	Y	97.3%	98.4%	88.1%	97.0%
MATCHING NETS (OURS)	Cosine	N	**98.1%**	**98.9%**	**93.8%**	98.5%
MATCHING NETS (OURS)	Cosine	Y	97.9%	98.7%	93.5%	**98.7%**

4. Experiments

ImageNet experiments:

Model	Matching Fn	Fine Tune	ImageNet 5-way 1-shot Acc			
			L_{rand}	$\neq L_{rand}$	L_{dogs}	$\neq L_{dogs}$
PIXELS	Cosine	N	42.0%	42.8%	41.4%	43.0%
INCEPTION CLASSIFIER	Cosine	N	87.6%	92.6%	**59.8%**	90.0%
MATCHING NETS (OURS)	Cosine (FCE)	N	**93.2%**	**97.0%**	58.8%	**96.4%**
INCEPTION ORACLE	Softmax (Full)	Y (Full)	$\approx$ 99%	$\approx$ 99%	$\approx$ 99%	$\approx$ 99%

4. Experiments

ImageNet experiments:

Table 3: Results on full ImageNet on *rand* and *dogs* one-shot tasks. Note that $\neq L_{rand}$ and $\neq L_{dogs}$ are sets of classes which are seen during training, but are provided for completeness.

Model	Matching Fn	Fine Tune	ImageNet 5-way 1-shot Acc			
			L_{rand}	$\neq L_{rand}$	L_{dogs}	$\neq L_{dogs}$
PIXELS	Cosine	N	42.0%	42.8%	41.4%	43.0%
INCEPTION CLASSIFIER	Cosine	N	87.6%	92.6%	**59.8%**	90.0%
MATCHING NETS (OURS)	Cosine (FCE)	N	**93.2%**	**97.0%**	58.8%	**96.4%**
INCEPTION ORACLE	Softmax (Full)	Y (Full)	$\approx$ 99%	$\approx$ 99%	$\approx$ 99%	$\approx$ 99%

4. Experiments

LM experiments:

1. an experimental vaccine can alter the immune response of people infected with the aids virus a <blank_token> u.s. scientist said.	prominent
2. the show one of five new nbc <blank_token> is the second casualty of the three networks so far this fall.	series
3. however since eastern first filed for chapter N protection march N it has consistently promised to pay creditors N cents on the <blank_token>.	dollar
4. we had a lot of people who threw in the <blank_token> today said <unk> ellis a partner in benjamin jacobson & sons a specialist in trading ual stock on the big board.	towel
5. it's not easy to roll out something that <blank_token> and make it pay mr. jacob says.	comprehensive
Query: in late new york trading yesterday the <blank_token> was quoted at N marks down from N marks late friday and at N yen down from N yen late friday.	dollar

5. Conclusion

Pros:

1. Use memory and neural network in a uniform structure.
2. A creative training strategy.
3. The non-parametric aspect of MN makes it easier for network to remember and adapt to new examples.

Cons:

1. When support set keeps growing, the speed will become a problem.
2. The label distribution has obvious biases.
3. It is not clear whether the order in support set matters?
4. More seriously, use LSTM make k fixed, how to expand k online?

Thank You

Q&A