Dataiku Academy

Low-shots with deep learning

Disclaimer

- The field is very new
- I’m super neophyte
- I may tell you wrong things

Era of “big data”

Era of “big data”

Huge datasets:

- ImageNet [1]:
  - 14M images, 1000 classes
  - 300 Gb
- Open Image [2]:
  - 30M images, 20K classes
- Berkeley Deep Drive [3]:
  - 100K videos of 40s with 30 fps (120M images)

[1]: http://www.image-net.org [2]: https://storage.googleapis.com/openimages/web/index.html [3]: https://deepdrive.berkeley.edu

When datasets are too small:

Transfer Learning:

- Use model pre-trained on ImageNet
- Replace the last fully connected layers of the model

- Train the last layer on the new dataset
- Fine-tune *very slowly* the pretrained model weights

- With >= 1000 images per class, it's very good [3] [4]

[3]: https://blog.keras.io/building-powerful-image-classification-models-using-very-little-data.html [4]: https://arxiv.org/abs/1403.6382

When datasets are huge but not enough labeled:

Semi-supervised learning:

- Train model on known data
- Predict labels on non-labeled data
- Fine-tune model using labels + predictions

Weakly supervised learning:

- Train models on imperfect labels:
  - Facebook on Instagram hashtags [5]
  - Heuritech on clothes description [6]

[5]: https://arxiv.org/abs/1805.00932 [6]: https://arxiv.org/abs/1709.09426

Very imperfect labels...

Not everything is “big data”

dataiku

When datasets are *way* too small:

Datasets with very few labeled data:

Omniglot [7]:

- 50 alphabets
- 1623 different characters, each written per 20 people

Humpback Whales [8]:

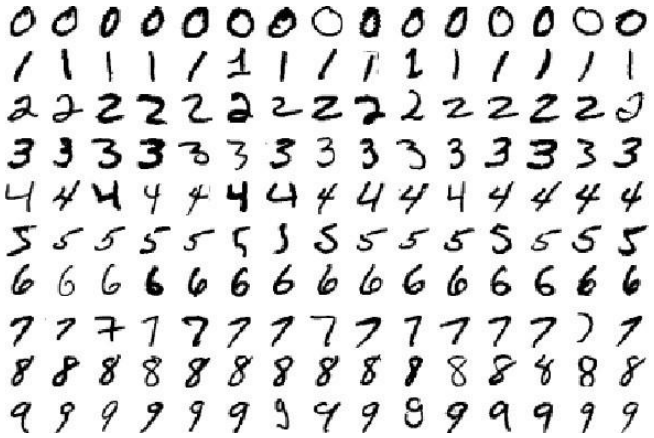
- 3000 images
- Between 1 and 3 images per unique whales!

[7]: https://github.com/brendenlake/omniglot [8]: https://www.kaggle.com/c/whale-categorization-playground

By comparison with *toy datasets*

MNIST [9]:

- 70.000 images, 10 classes
- **7.000 images per class**



Omniglot:

- 32.460 images, 1623 classes
- **20 images per class**

[9]: http://yann.lecun.com/exdb/mnist/

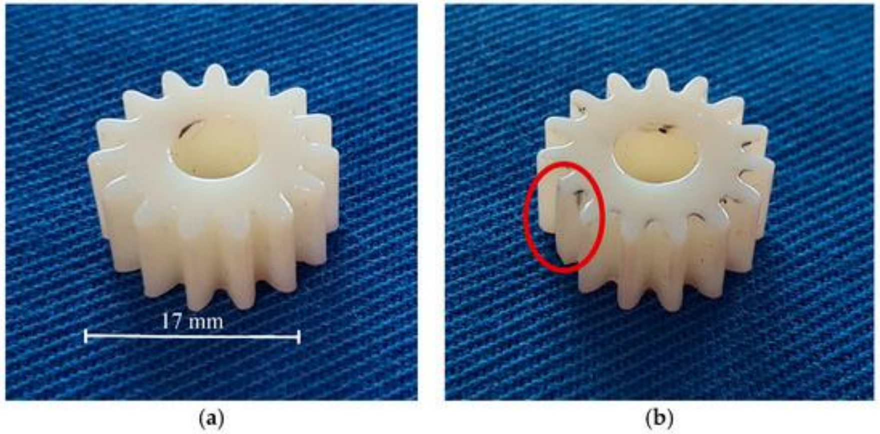
Potential real use cases for Computer Vision

Fashion:

- Plenty of data for "*shirt*", "*dress*", etc.
- Very few data (less than 5 images) for a particular clothe model:
  - "*Kenzo shirt edition 2018 spring collector*", "*balenciaga high heel model 2016 black*", etc.

Factory:

- Plenty of data for "*gear*"
- Very few data for "*gear with this particular defect that arise once in a million*"



Potential real use cases for Computer Vision

Face identification:

- Detect identity of a person
- Your IPhone won't ask you 1,000 photo of your face to use FaceID

Not only in Computer Vision

Speech:

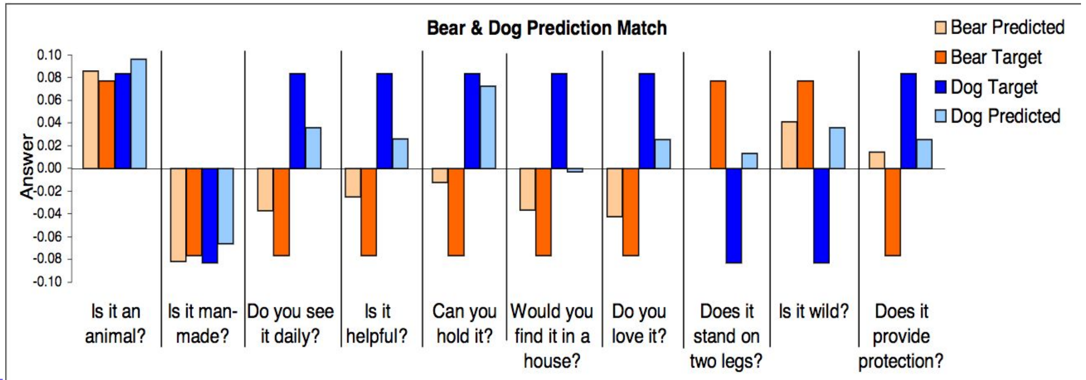
- Recognize a word heard only once
- Clone a voice with only few spoken sentences [10][11]

NLP:

- Understand the sense of a word seen only once [12]

Zero-Shot:

- No training sample but a description of what it should be [13]

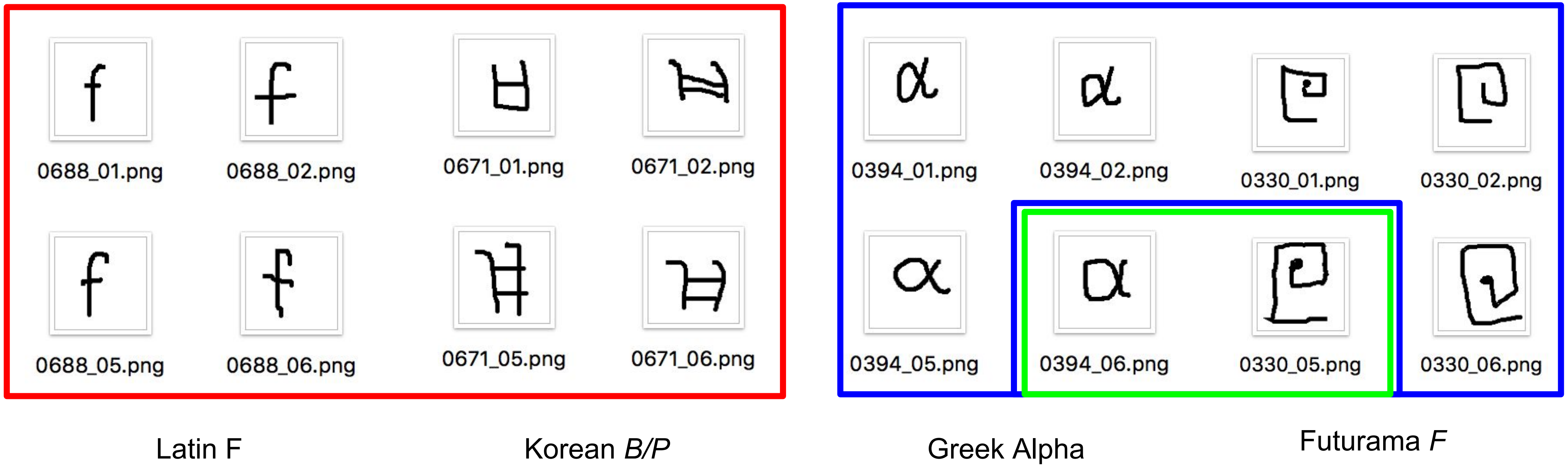


[10]: https://arxiv.org/abs/1802.06006 [11]: https://lyrebird.ai [12]: https://arxiv.org/abs/1710.10280
[13]: http://www.cs.cmu.edu/afs/cs/project/theo-73/www/papers/zero-shot-learning.pdf

Defining the task

Vocabulary | example of one-shot on Omniglot

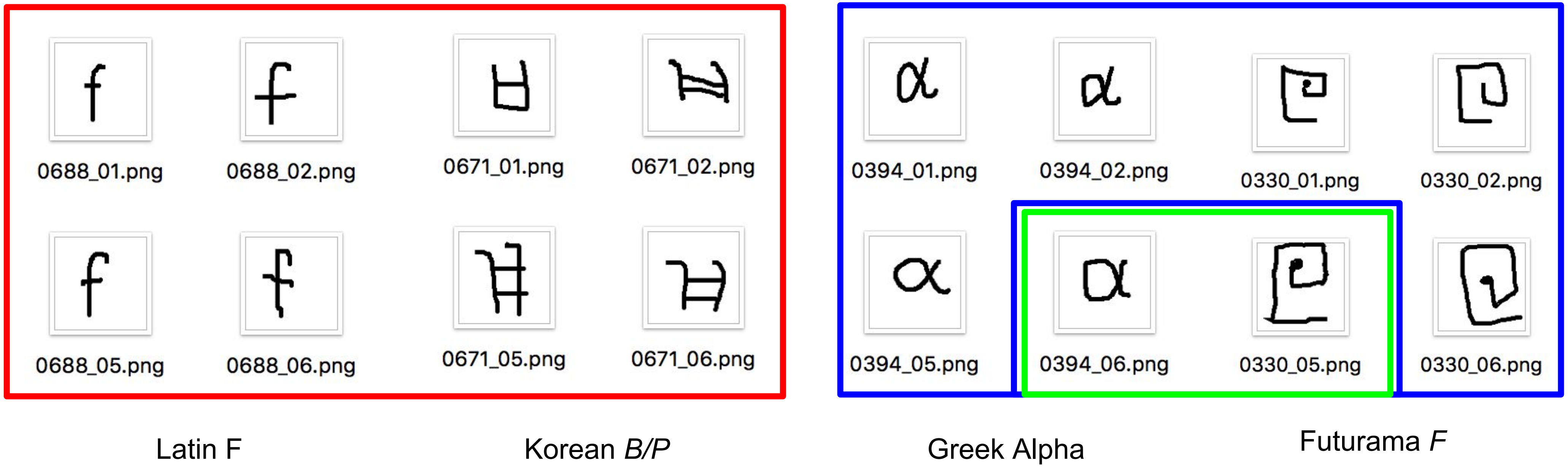
1. Learn to distinguish between characters with the **background set**.
2. We want to distinguish characters in a unseen alphabets (Greek & Futurama):
   a. but we only have **one** labeled image for each character.
   b. It's the **support set**.
3. We predict character class on the **query set**.



Vocabulary | example of one-shot on Omniglot

- Two novel classes (Greek Alpha, Futurama *F*), one labeled image per class

2-way, 1-shot

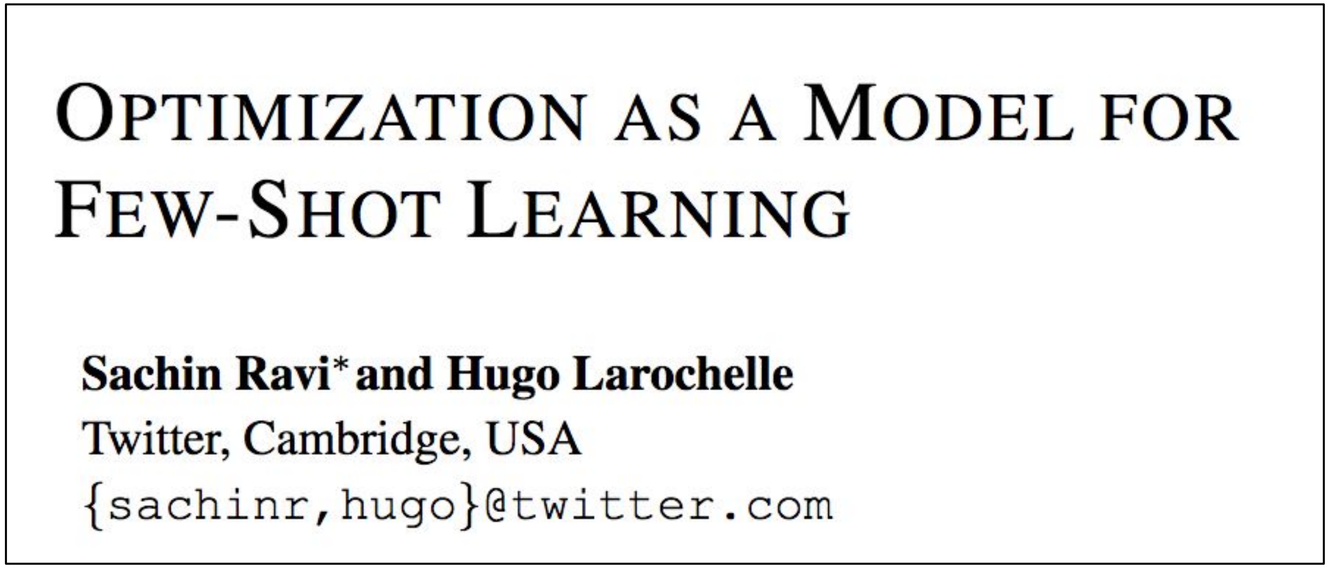


Different approaches to Low-shot learning

Meta-learning

- Learning a classifier

- *"Meta-learning suggests framing the learning problem at two levels. The first is quick acquisition of knowledge within each separate task presented. This process is guided by the second, which involves slower extraction of information learned across all the tasks."* [14]


OPTIMIZATION AS A MODEL FOR FEW-SHOT LEARNING

Sachin Ravi* and Hugo Larochelle
Twitter, Cambridge, USA
{sachinr,hugo}@twitter.com

[14]: https://openreview.net/pdf?id=rJY0-Kcll

Memory Augmented Network

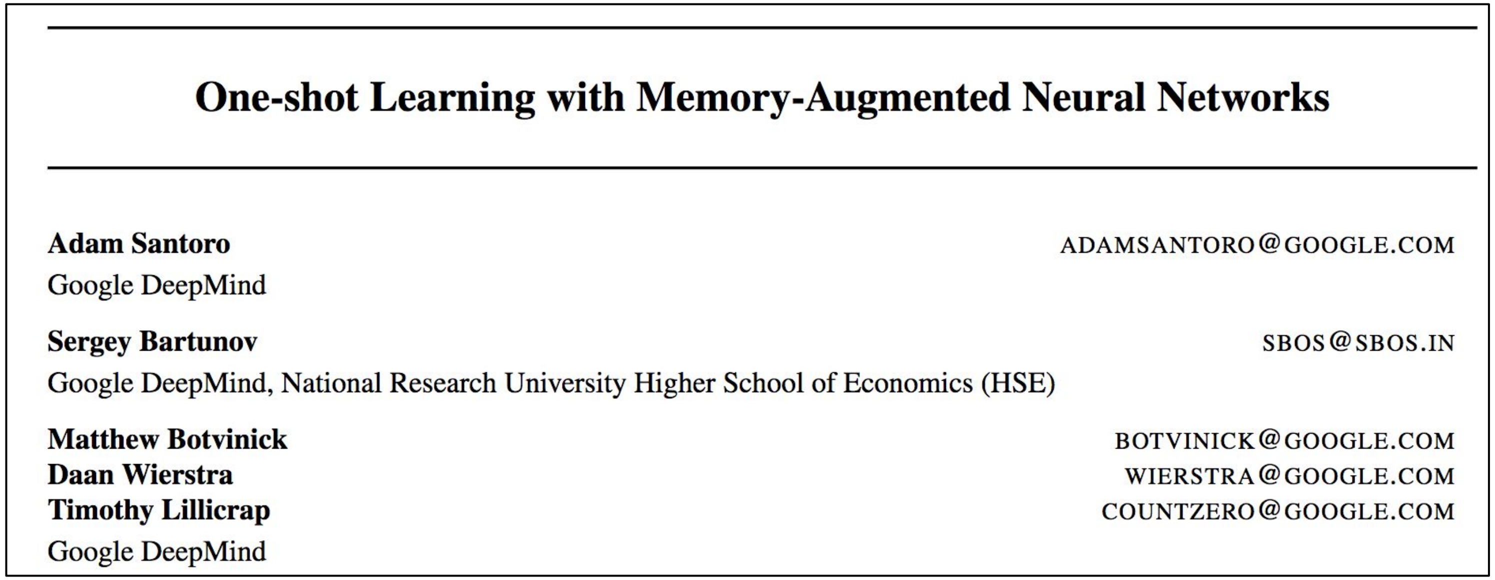
- Want to avoid "*catastrophic forgetting*"

- Model must not relearn everything from scratch at each new task

- Use of "*Memory Augmented Network*" like the Neural Turing Machine [15] [16]
- Encode & retrieve efficiently new information

[15]: https://arxiv.org/abs/1410.5401

[16]: https://arxiv.org/abs/1605.06065



One-shot Learning with Memory-Augmented Neural Networks

Adam Santoro ADAMSANTORO@GOOGLE.COM
Google DeepMind

Sergey Bartunov SBOS@SBOS.IN
Google DeepMind, National Research University Higher School of Economics (HSE)

Matthew Botvinick BOTVINICK@GOOGLE.COM
Daan Wierstra WIERSTRA@GOOGLE.COM
Timothy Lillicrap COUNTZERO@GOOGLE.COM
Google DeepMind

Metric learning

- Learn a distance between images

- Often followed by a K Nearest Neighbors to determine the *closest* image

Siamese Neural Networks for One-shot Image Recognition

Gregory Koch GKOCH@CS.TORONTO.EDU
Richard Zemel ZEMEL@CS.TORONTO.EDU
Ruslan Salakhutdinov RSALAKHU@CS.TORONTO.EDU
Department of Computer Science, University of Toronto. Toronto, Ontario, Canada.

Siamese Networks

Siamese Networks for low-shot learning

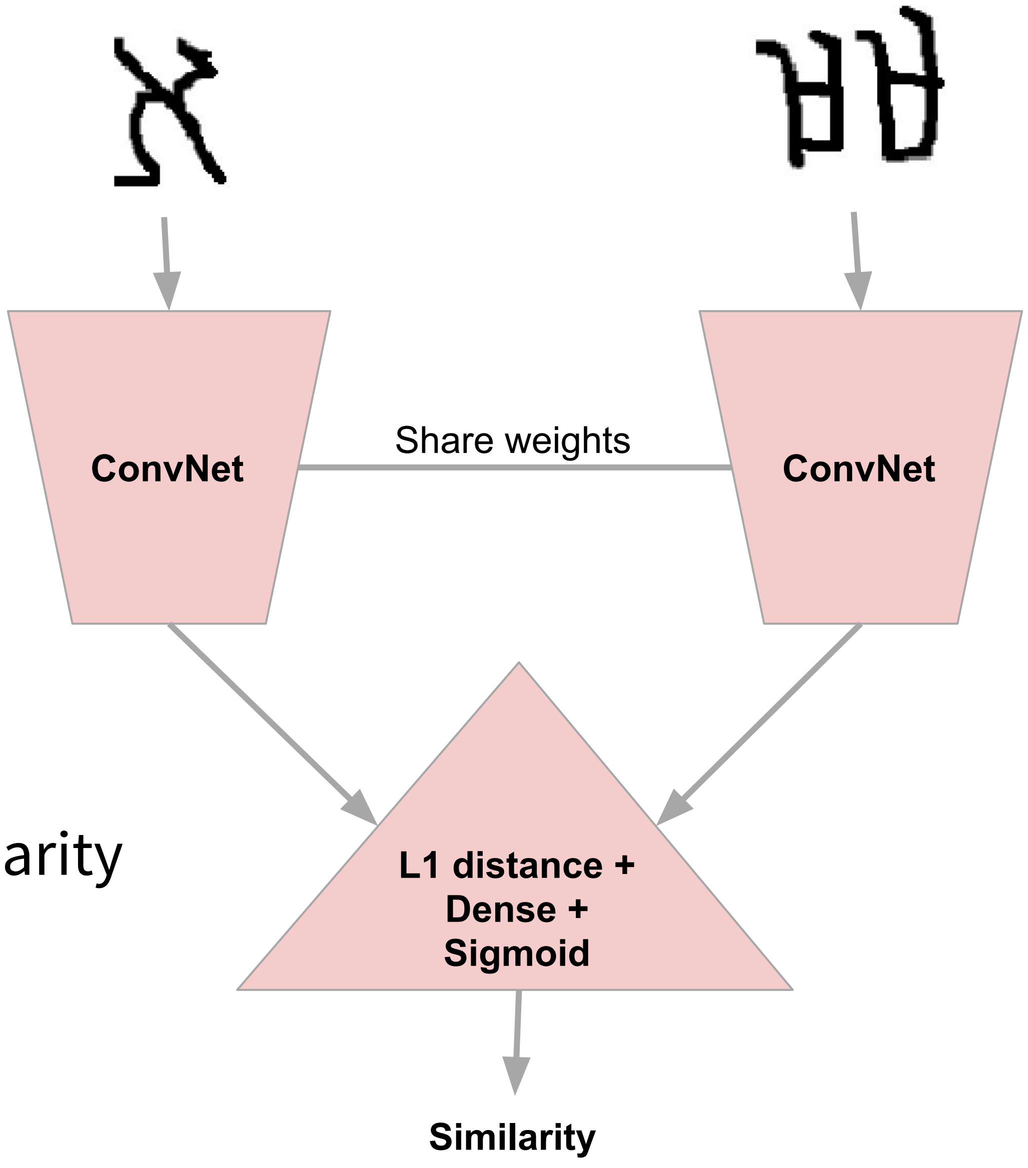
Siamese Networks [17]:

- Learn similarity between pairs of images

- Shared ConvNets encoding the images
- L1 distance (abs difference) between the two feature maps.
- Similarity between 0 and 1.

- Test time:
  - Nearest-neighbours based on this similarity

[17]: https://www.cs.cmu.edu/~rsalakhu/papers/oneshot1.pdf

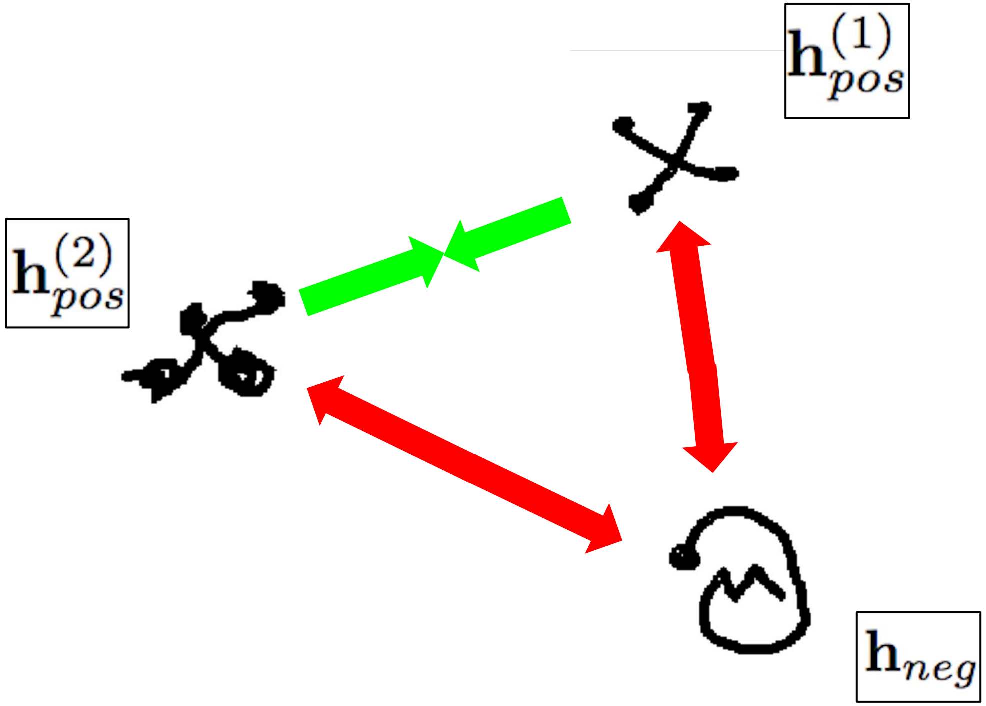


Triplet Network

Triplet Networks for low-shot learning

Triplet Networks [18][19]:

- Learn *relative* distance between images
- Shared ConvNets encoding the images
- Three input images:
    - Two of the same class
    - One different
- Triplet Loss:



$$\mathcal{L}_{triplet}\left(\mathbf{h}_{pos}^{(1)}, \mathbf{h}_{pos}^{(2)}, \mathbf{h}_{neg}\right) = \left[m + d\left(\mathbf{h}_{pos}^{(1)}, \mathbf{h}_{pos}^{(2)}\right) - d\left(\mathbf{h}_{pos}^{(1)}, \mathbf{h}_{neg}\right)\right]_{+} + \left[m + d\left(\mathbf{h}_{pos}^{(1)}, \mathbf{h}_{pos}^{(2)}\right) - d\left(\mathbf{h}_{pos}^{(2)}, \mathbf{h}_{neg}\right)\right]_{+}$$

[18]: https://arxiv.org/abs/1412.6622 [19]: https://arxiv.org/abs/1804.07275v1

Matching Networks

Matching Networks for One Shot Learning

Oriol Vinyals
Google DeepMind
vinyals@google.com

Charles Blundell
Google DeepMind
cblundell@google.com

Timothy Lillicrap
Google DeepMind
countzero@google.com

Koray Kavukcuoglu
Google DeepMind
korayk@google.com

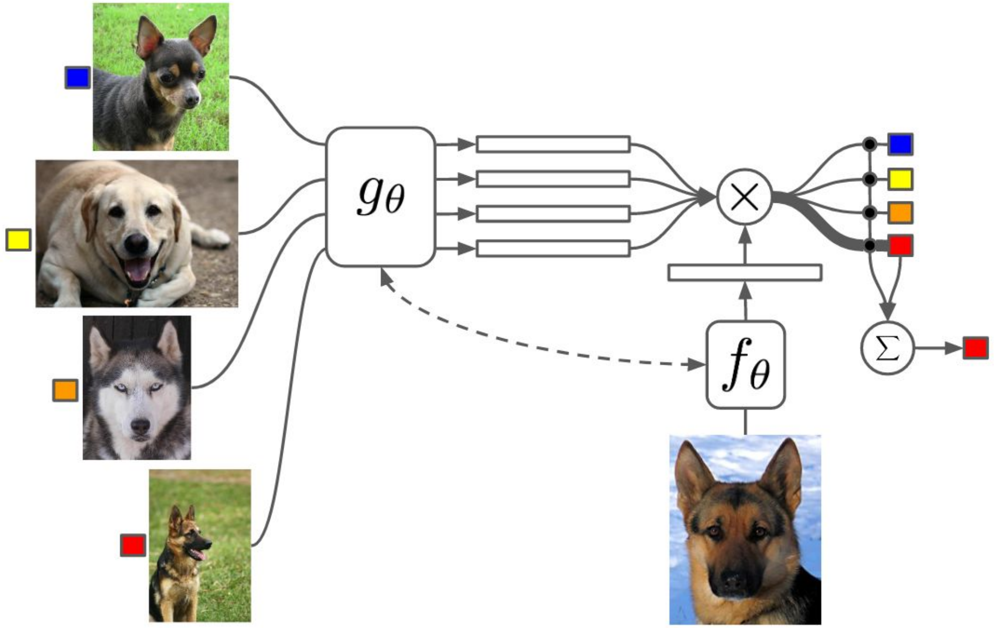
Daan Wierstra
Google DeepMind
wierstra@google.com

Matching Network [20]

- Previous models treated each shots of the support set independently

- MN embeds the support set with bi-LSTM

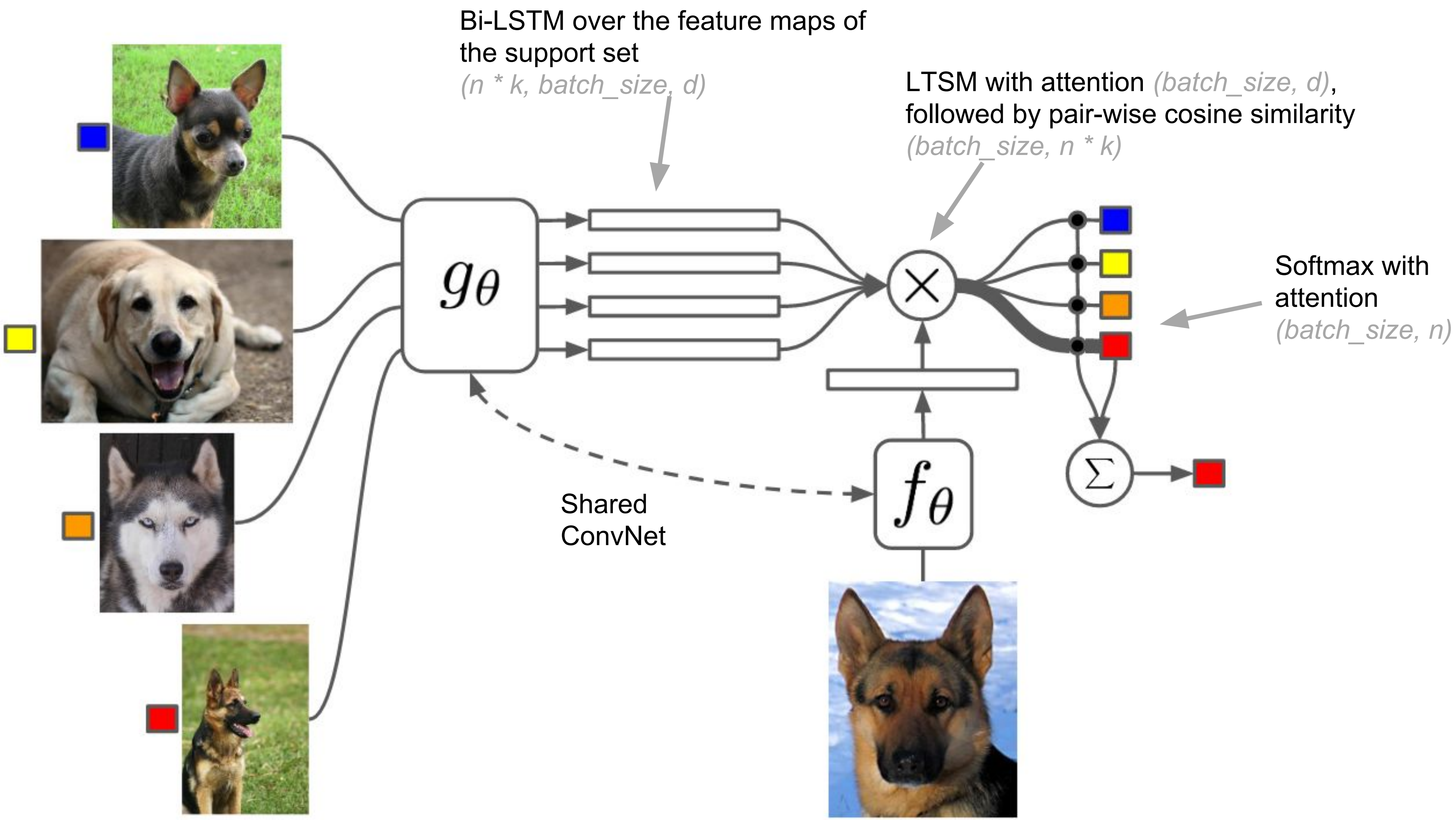
- Training task is the same as the test task



[20]: https://arxiv.org/abs/1606.04080

Matching Network

BE AWARE!



Going further with metric learning

Going further with metric learning

- Try different distance
  - Mahalanobis, Cosine, etc.

- Learn own distance
  - Concatenate the features and feed it to a FC with sigmoid

- Try loss with fancier names:
  - Histogram loss [21], Quadruplet loss [22]

- Tweak heavily regularization to avoid overfitting

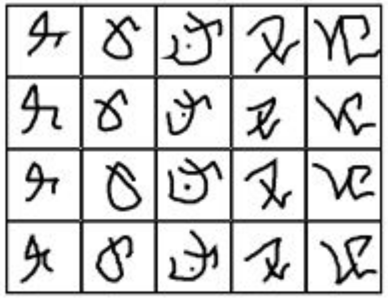
[21]: https://arxiv.org/abs/1611.00822 [22]: https://arxiv.org/abs/1704.01719

In production

In production

- Pre-compute the features of the support set

- Limit the possibility of pair-wise testing:
  - To detect the “Kenzo shirt edition 2018 spring collector”
  - Use a classic classifier to know if it is a:
    - shirt
    - spring style (flowers, hawaiian, whatever)
  - Compare with only those that have similar features



Thank you for attending!

data
iku