



UNIVERSITY OF
OXFORD

Learning feed-forward one-shot learners

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



One-shot learning




UNIVERSITY OF
OXFORD

One-shot learning:

Learning a new concept from 1 or few samples.

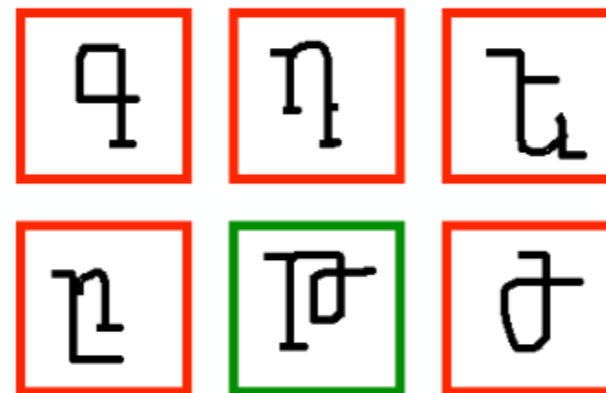
Examples:

- Specializing OCR to new writers or new alphabets.
- Single-object tracking.

Exemplar 



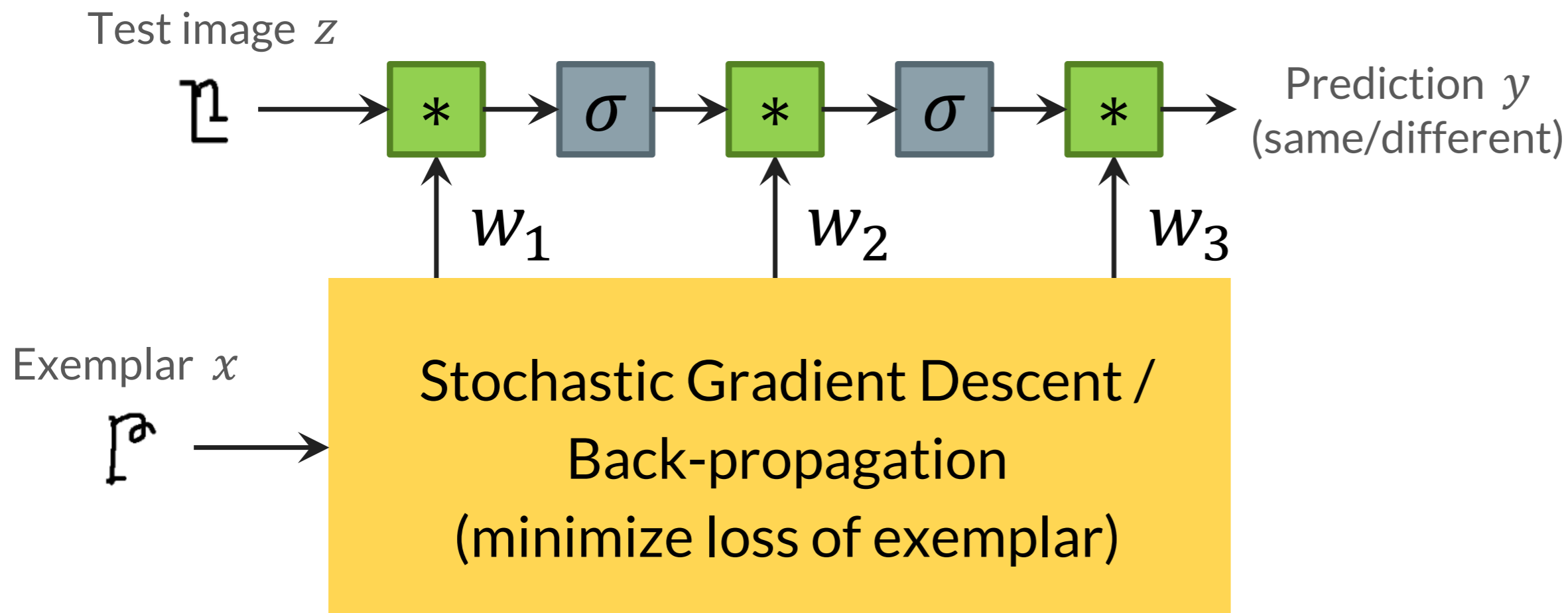
Test
images



Standard discriminative learning



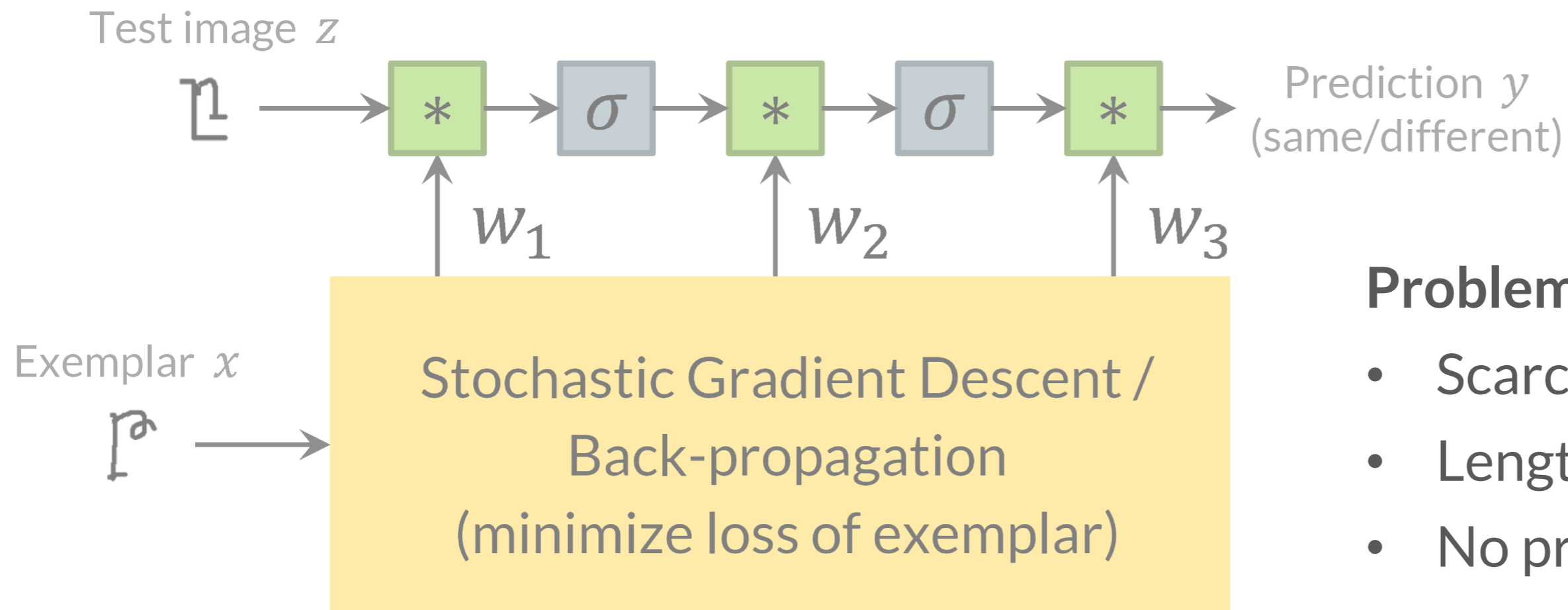
Starting point: Standard SGD/back-propagation learning



Standard discriminative learning



Starting point: Standard SGD/back-propagation learning



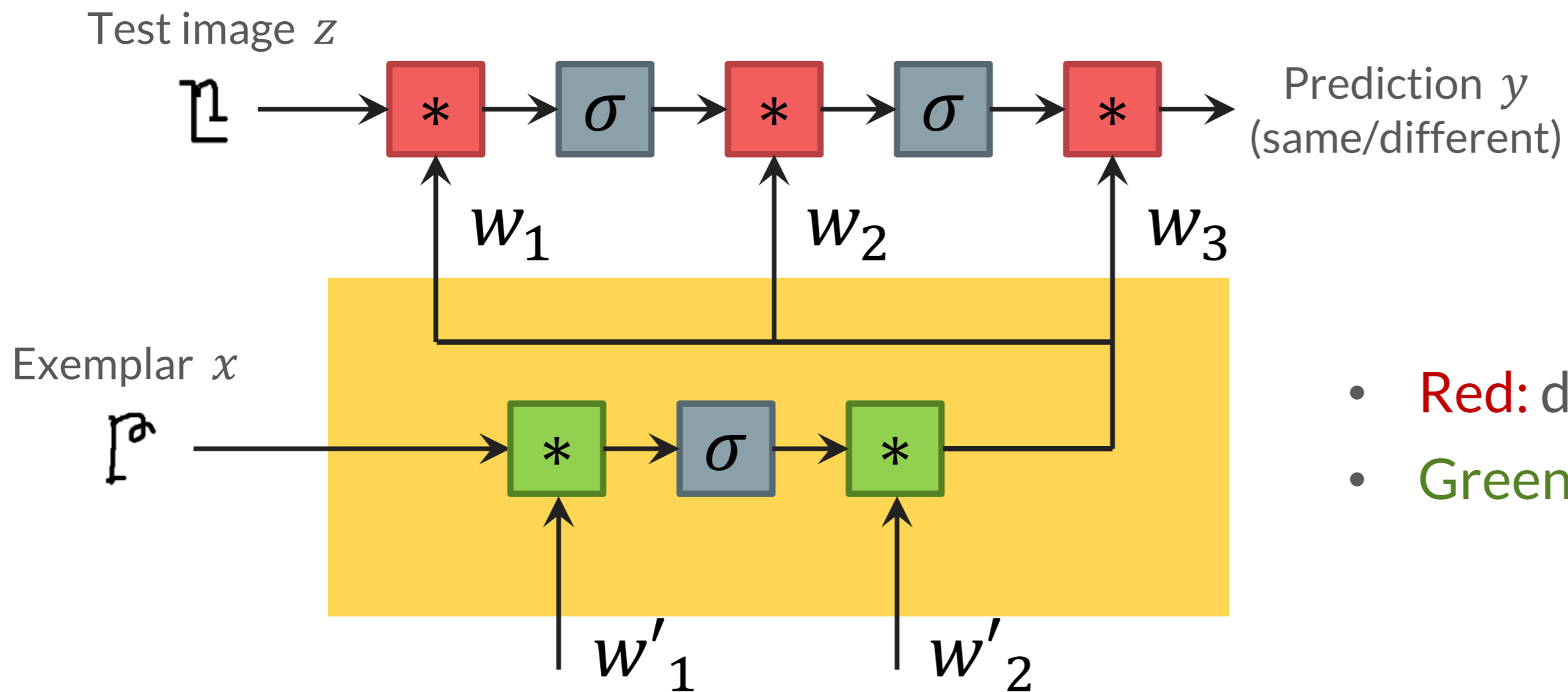
Problems:

- Scarce data/overfitting
- Lengthy optimization process
- No priors (“learning to learn”)

Parameter prediction



Idea: Re-interpret “training” as “parameter prediction”

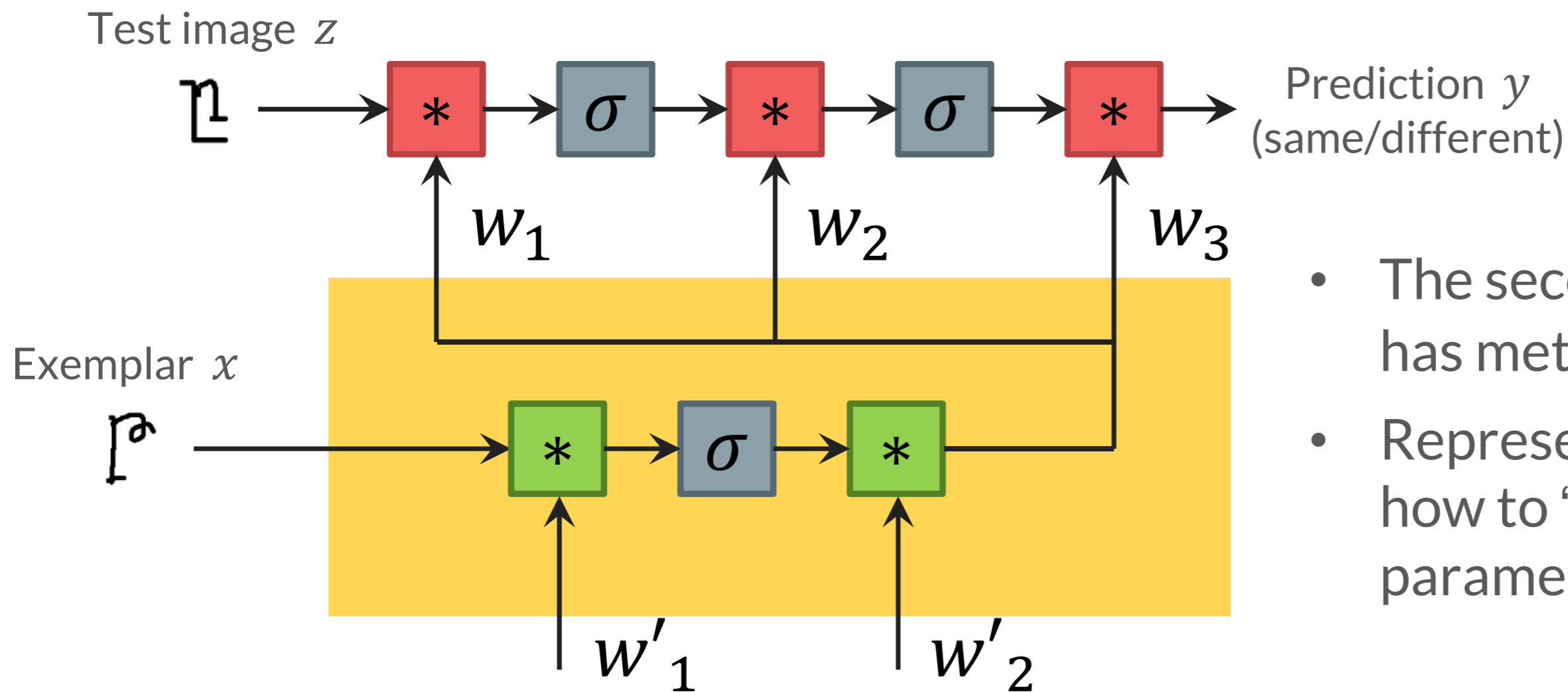


- **Red:** dynamic convolution
- **Green:** standard convolution

Parameter prediction



Idea: Re-interpret “training” as “parameter prediction”



- The second network (*learnet*) has meta-parameters w'_i .
- Represent prior knowledge about how to “learn” (predict) parameters, from one exemplar.

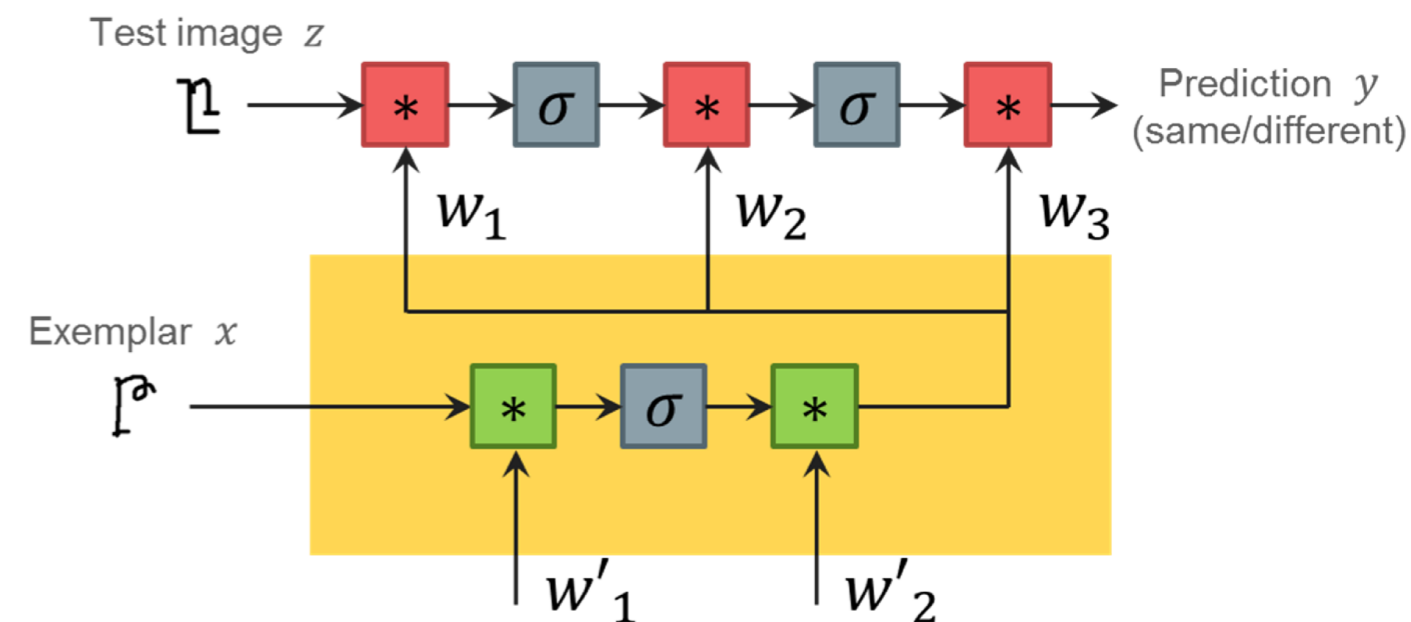
Feed-forward one-shot learning:

- Learns a classifier (predicts w_i from x) and evaluates it (on z) in one pass.

Because this is a standard computational graph, it can be differentiated with back-prop.

Training “meta-parameters” w'_i :

- Draw exemplar/test-image/label triplets (x, z, y) .
- Back-propagate through graph, and update w'_i with SGD.



Technical challenges

Typical number of parameters:

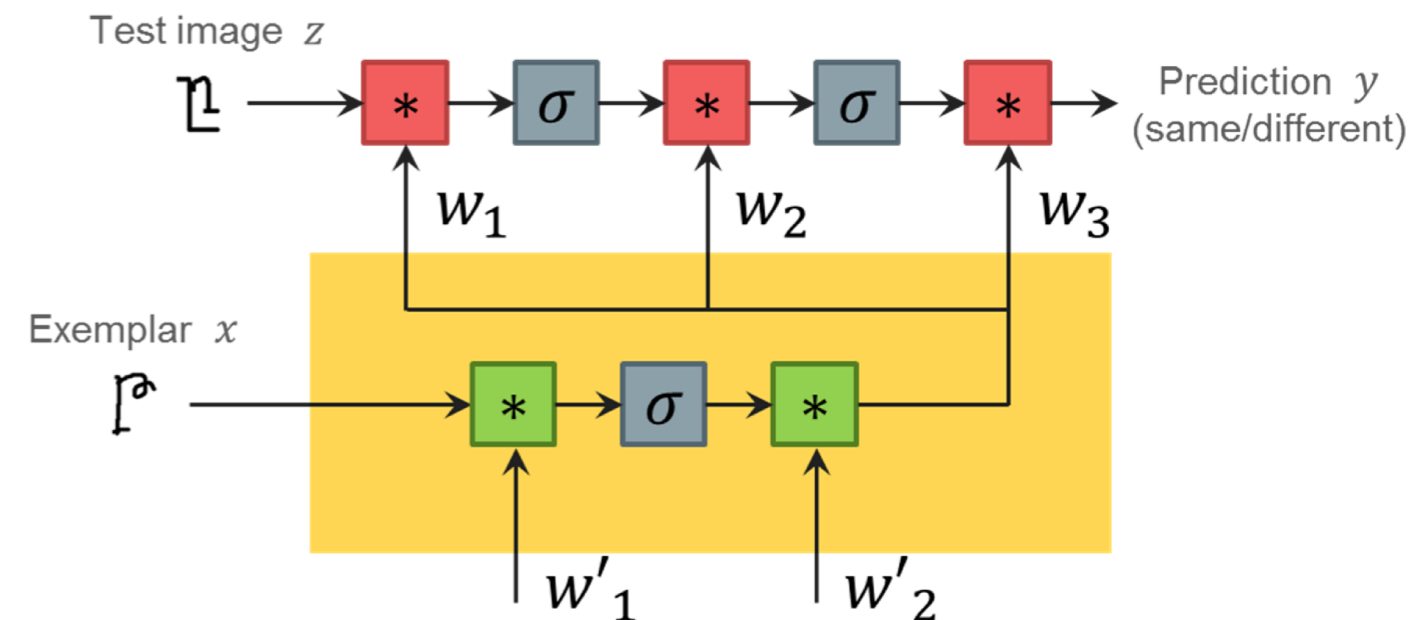
- Fully-connected: $4096 \times 4096 \cong 2 \times 10^7$
- Convolutional: $3 \times 3 \times 192 \times 256 \cong 4 \times 10^6$

The output-space for parameter prediction can be very large.

To predict this many outputs from a 4096-dim. vector:

$$4096 \times 4 \times 10^6 \cong 1 \times 10^{10} \text{ parameters (15.2 GB)}$$

- Storage issues
- Overfitting



- We start with the **fully-connected** case (easier).

$$y = W(z)x + b(z)$$

- The number of dynamic weights $W(z)$ scales **quadratically** with the size of x .
- Inspired by SVD, factorize:

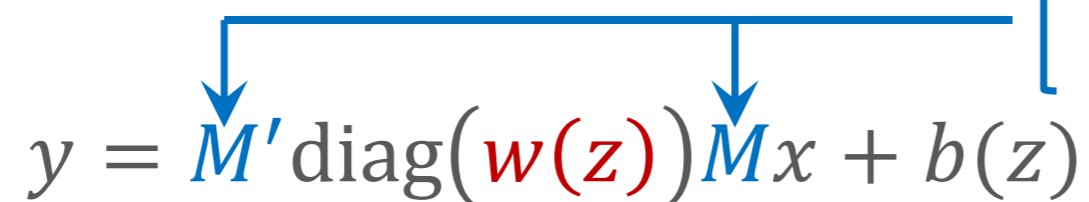
$$y = M' \text{diag}(w(z)) Mx + b(z)$$

- We start with the **fully-connected** case (easier).

$$y = W(z)x + b(z)$$

- The number of dynamic weights $W(z)$ scales **quadratically** with the size of x .

- Inspired by SVD, factorize:

$$y = M' \text{diag}(w(z)) M x + b(z)$$


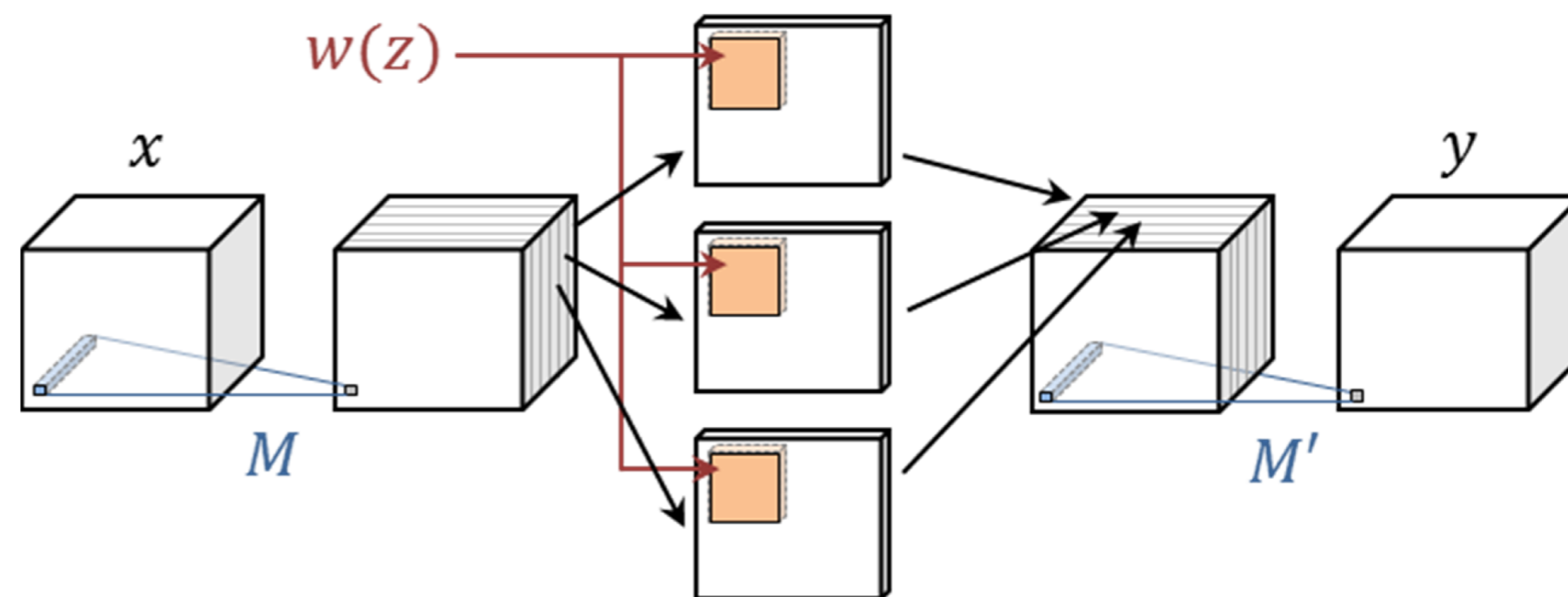
- Learned offline and fixed.
- Projection into space with independent factors of variation.

- Predicted dynamically.
- Scales linearly with size of x .

To be broadly useful, we need to **generalize** for convolutional layers.

Factorized convolution:

- 1×1 convolution (M)
- Diagonal convolution with $w(z)$
- 1×1 convolution (M')



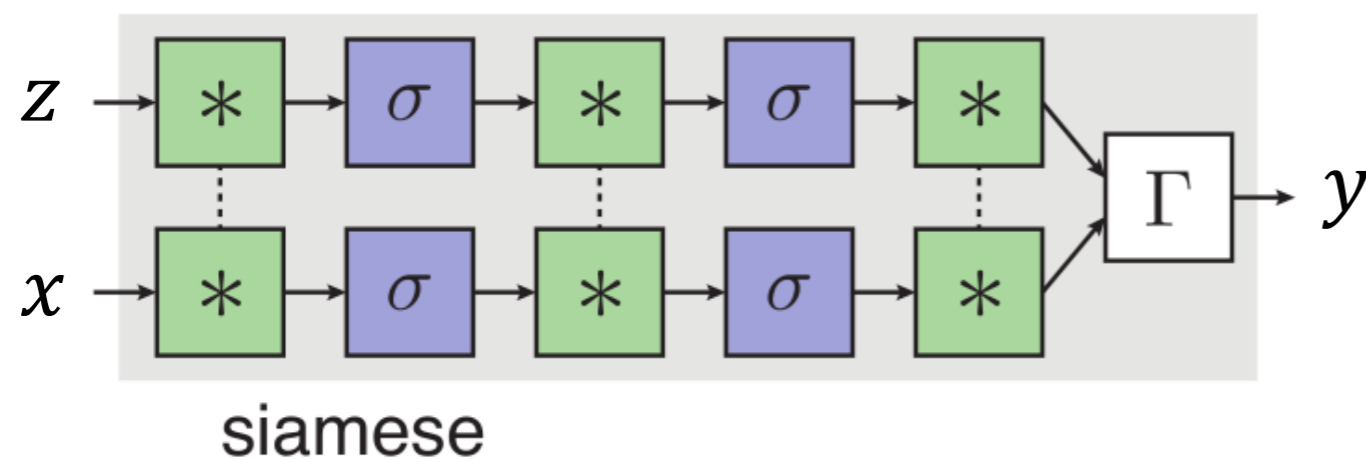
Diagonal convolution applies k independent filters to k input channels.

In the fully-connected case (1×1 output) reduces to $\text{diag}(w(z))$.

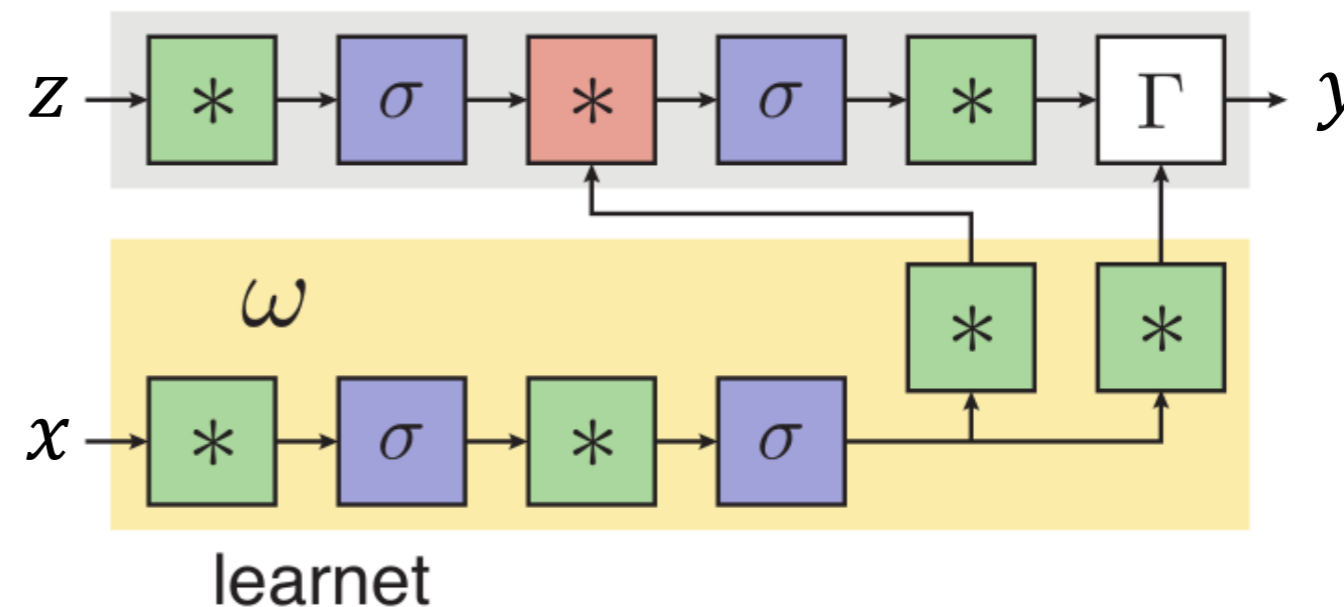
Siamese networks



The proposed architecture is reminiscent of siamese networks.



$$f(z, x) = \Gamma(\varphi(x; W), \varphi(z; W))$$

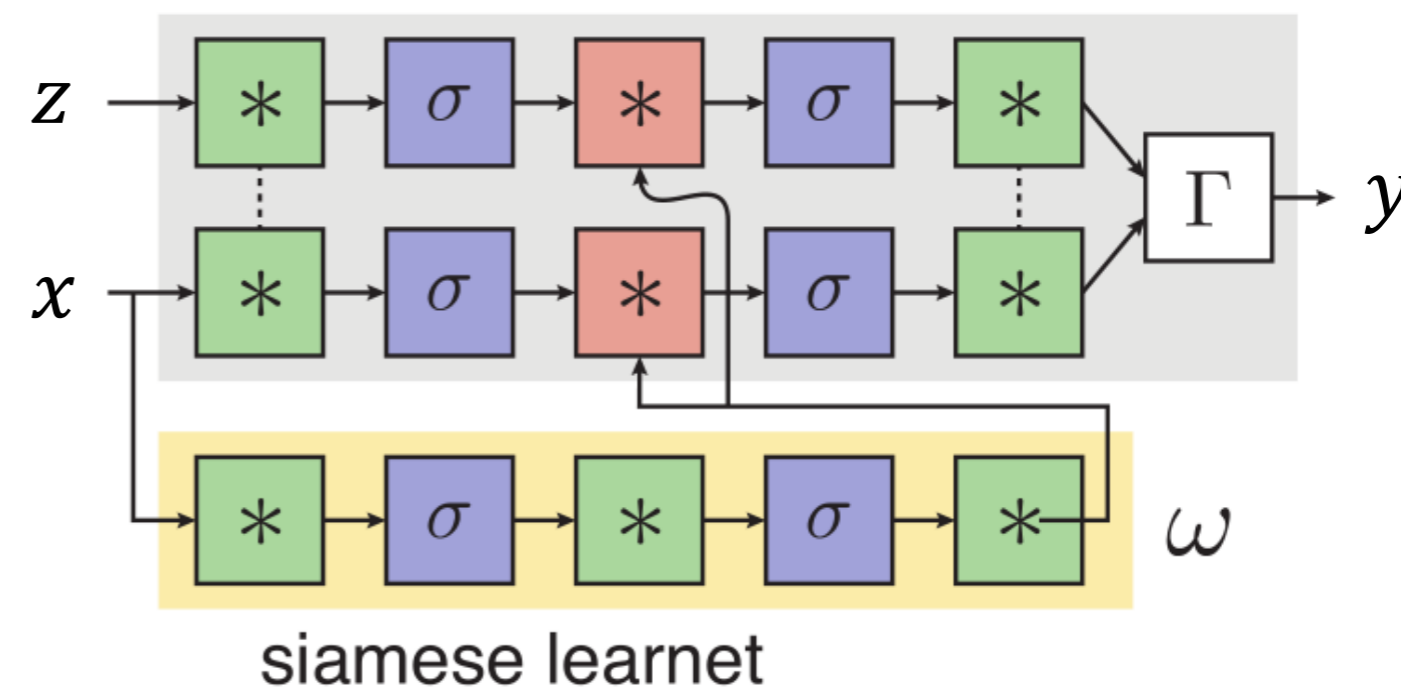
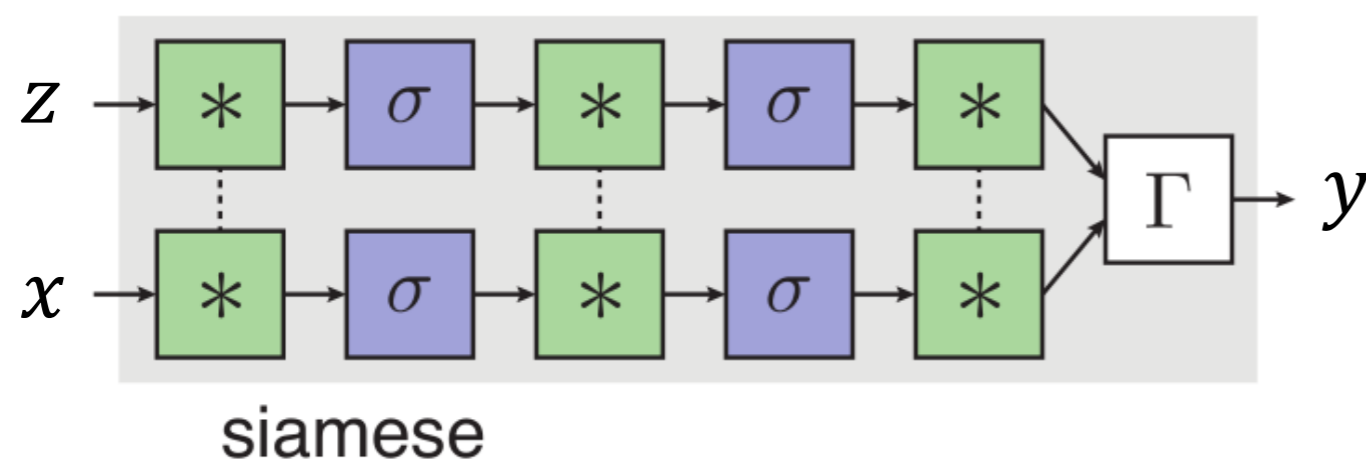


$$f(z, x) = \Gamma(\varphi(x; \omega(z; W')), \varphi(z; W))$$

Key differences:

- Siamese net applies **same model with shared weights** to x and z .
- The proposed “learnnet” changes **intermediate representations** of another net (**red**).

Siamese networks



- To highlight that they are not mutually exclusive, a learnet can be used to **dynamically change** the parameters of a siamese net.

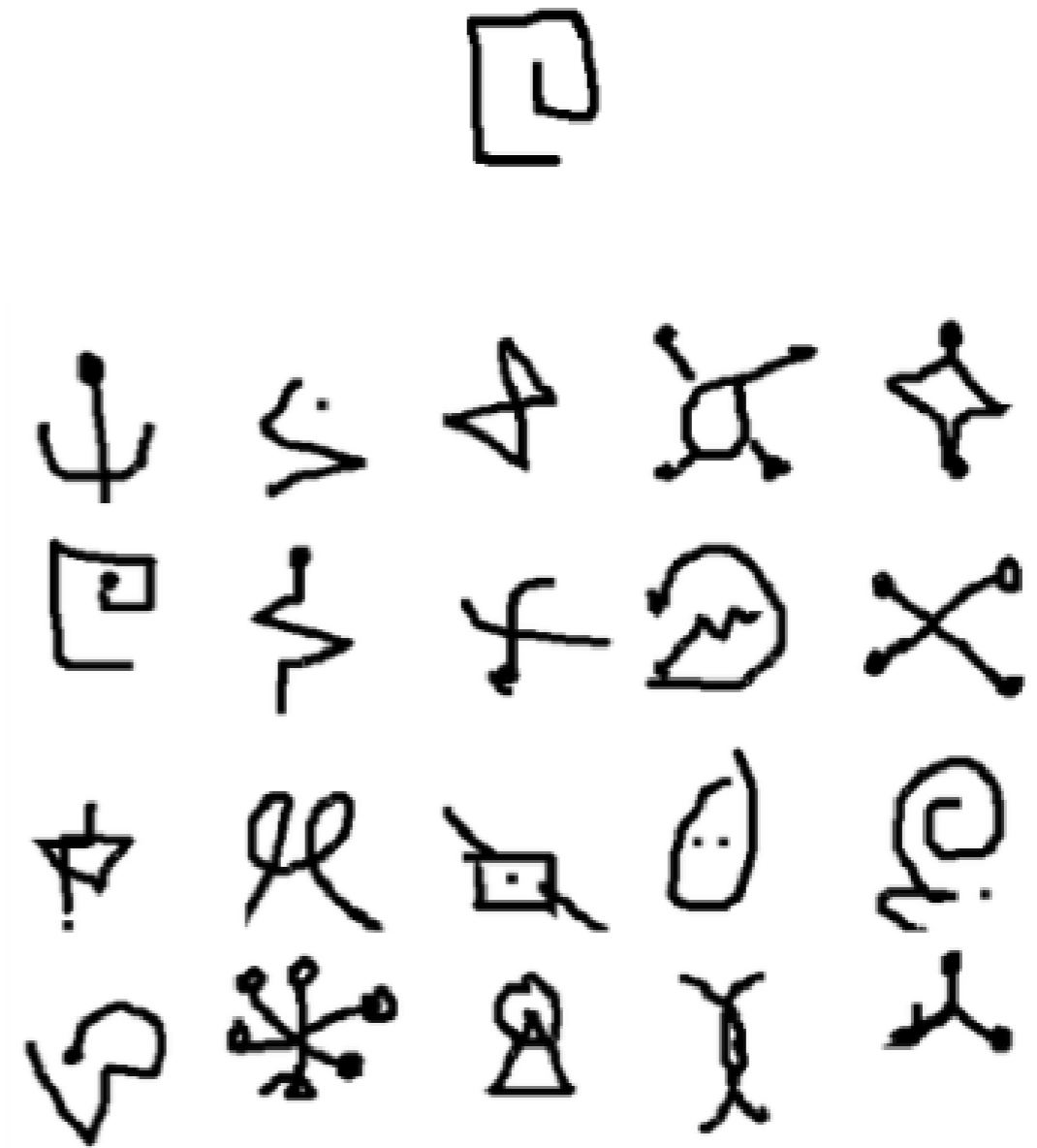
Omniglot dataset

- 30 training alphabets, 20 testing alphabets.
- Resized to 28x28 pixels.
- Find match among 20 characters from same alphabet (chance is 95% error).
- Architecture: 3 conv layers, final layer Γ is a weighted L1 distance.
- Learnnet predicts parameters of 2nd conv.



Omniglot dataset – results

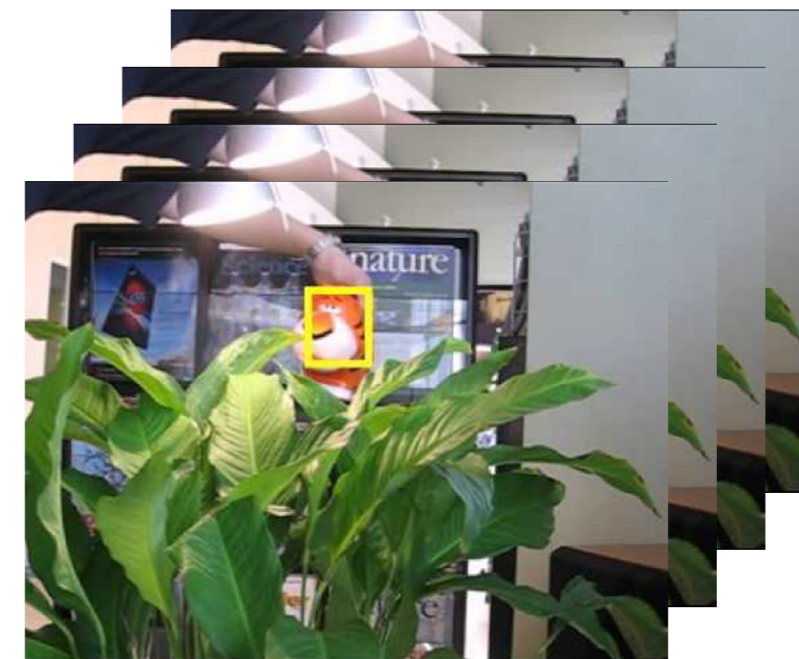
	Error (%)
Siamese	41.8
Siamese (unshared)	34.6
Learnnet	28.6
Siamese learnnet	31.4



Single-object tracking

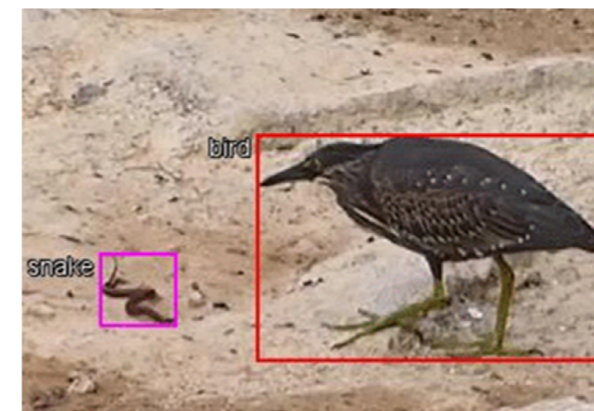
- Can be naturally posed as a **one-shot learning** problem:
 1. **Learn** classifier, with the initial object patch as the exemplar.
 2. **Classify** patches over remaining video into object/background.

- Possible to update online, but in our experiments this was not needed.



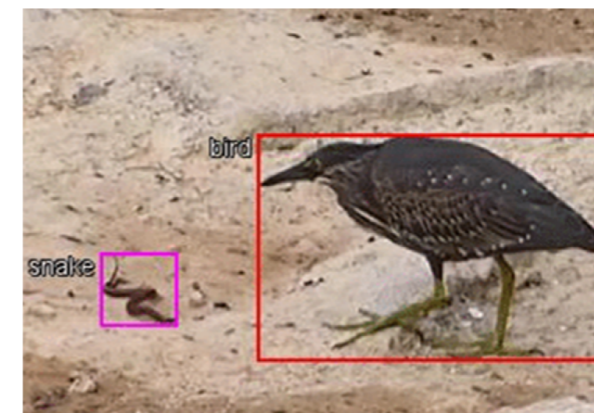
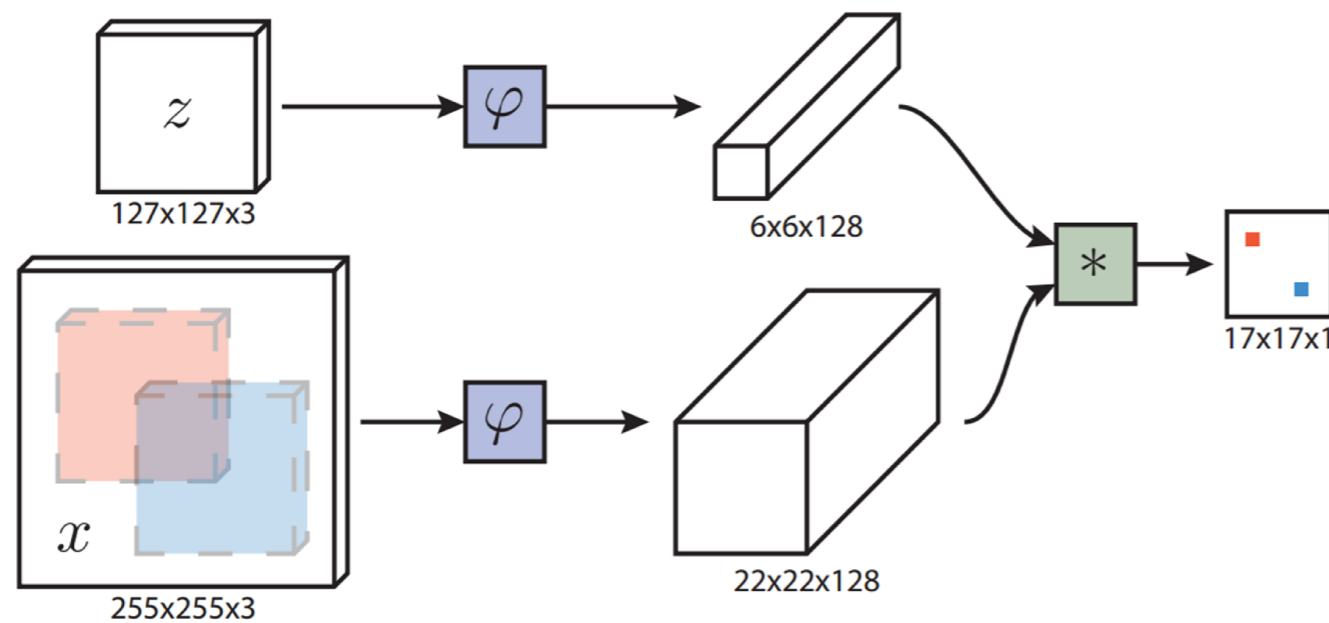
Training

- Used ImageNet Video dataset.
 - 4,500 videos / 1,200,000 bounding boxes.
 - 30 classes: mostly animals (~75%) and some vehicles (~25%). Class data ignored.
- Pos. pairs: patches near in time (same video).
- Neg. pairs: patches far away/different videos.
- Architecture: slim AlexNet (less channels, for speed).



Fully convolutional architecture

- To efficiently classify many patches, the net is applied convolutionally to a larger image.
- Standard trick from detection; produces a heat map of possible object locations.



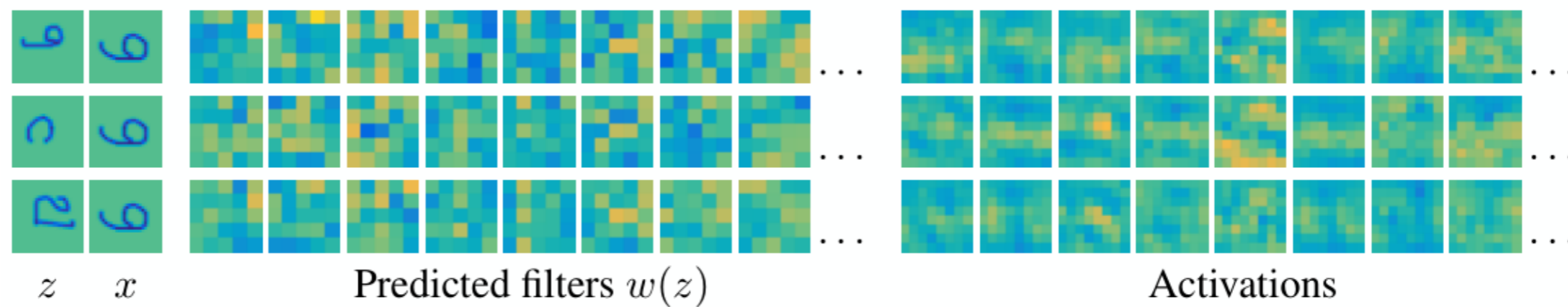
Visual Object Tracking (VOT) 2015 benchmark – results

	Accuracy (IoU)	Num failures
Siamese	0.465	105
Siamese (unshared)	0.447	131
Siamese learnet	0.500	87

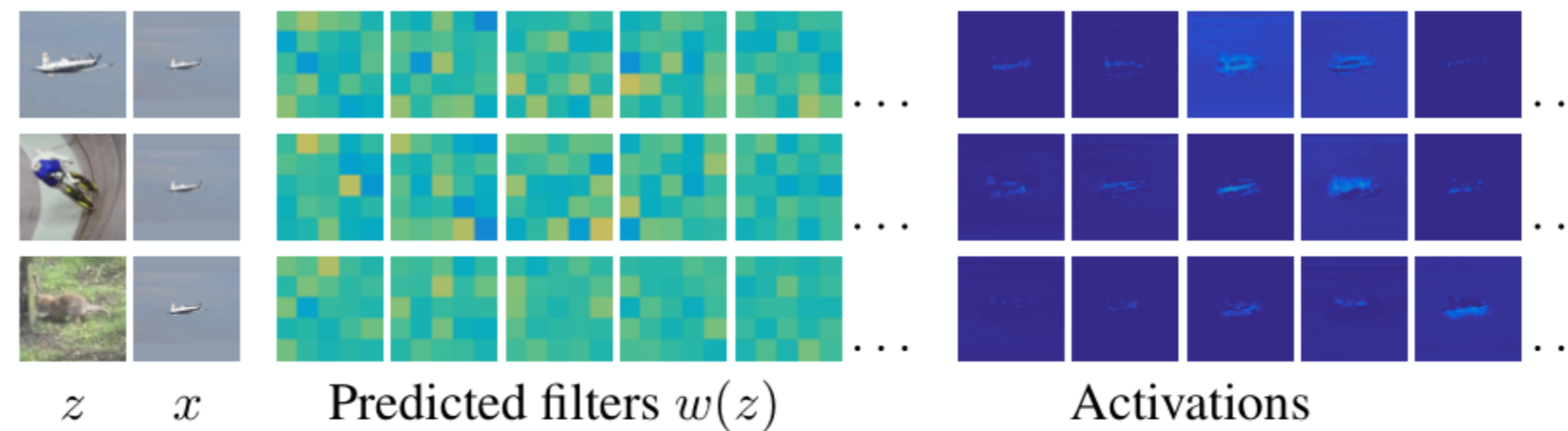
	Accuracy (IoU)	Num failures
DSST	0.483	163
MEEM	0.458	107
MUSTer	0.471	132
DAT	0.442	113
SO-DLT	0.540	108

Visualization – predicted filters and activations

Omniglot:



ImageNet
Video:



Conclusions

- It is possible to obtain the parameters of a deep network by a single feed-forward prediction.
- Related to siamese nets, more general.
- “Learning-to-learn” direction:
Train meta-parameters by solving millions of small learning tasks offline, as feed-forward computations.

