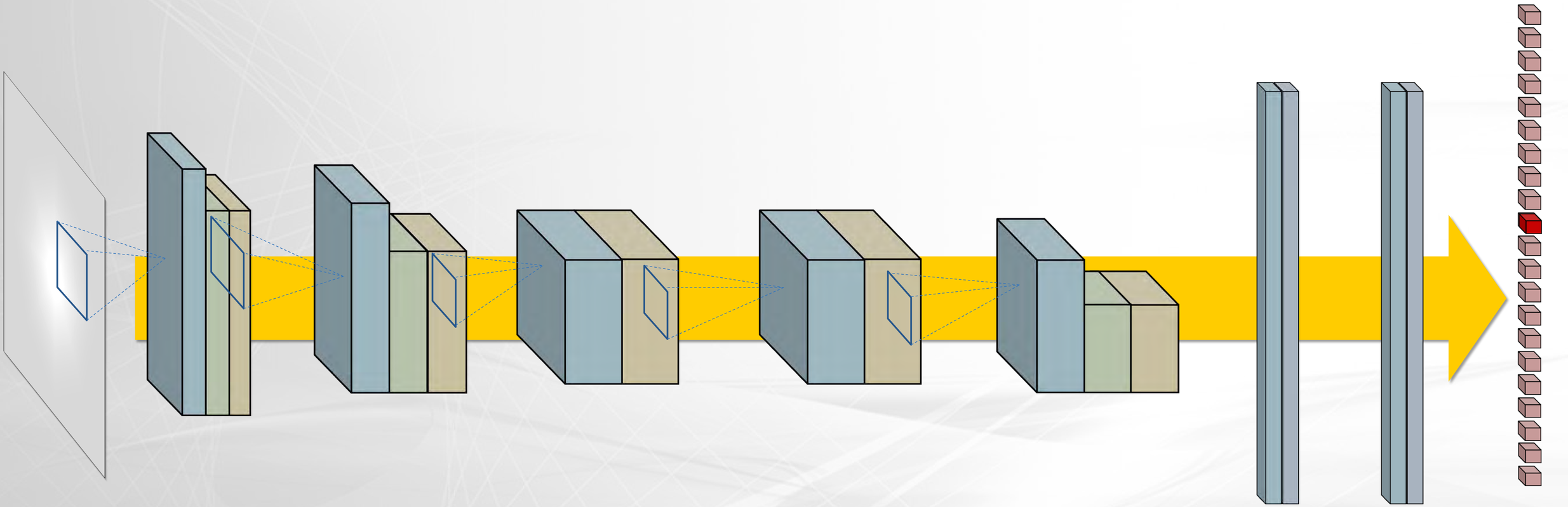
Deep Model Adaptation using Domain Adversarial Training

Victor Lempitsky,
joint work with Yaroslav Ganin

Skoltech

Skolkovo Institute of Science and Technology (Skoltech)
Moscow region, Russia

Deep supervised neural networks



- are a “big thing” in computer vision and beyond
- **are hungry for labeled data**

Where to get the data?

Lots of modalities do not have large labeled data sets:

- Biomedical
- Unusual cameras / image types
- Videos
- Data with expert-level annotation (not mTurkable)
- ….

Surrogate training data often available:

- Borrow from adjacent modality
- Generate synthetic imagery (computer graphics)
- Use data augmentation to amplify data (*image-based rendering, morphing, re-synthesis,….*)

Resulting training data are shifted. *Domain adaptation* needed.

Example: Internet images -> Webcam sensor

[Saenko et al. ECCV2010]

Example: (semi-)synthetic to real

Deep Model Adaptation using Domain Adversarial Training

Assumptions and goals

- Lots of *labeled* data in the *source domain* (e.g. synthetic images)
- Lots of *unlabeled* data in the *target domain* (e.g. real images)
- **Goal**: train a **deep neural net** that does well on the **target domain**

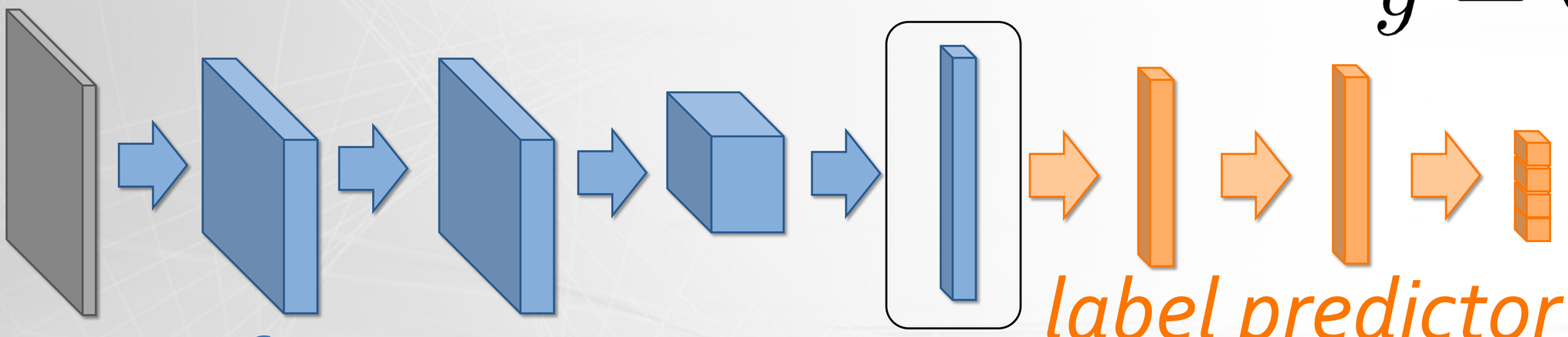
Large-scale deep unsupervised domain adaptation

Domain shift in a deep architecture

$$\mathbf{x}$$

$$\mathbf{f} = G_f(\mathbf{x}; \theta_f)$$

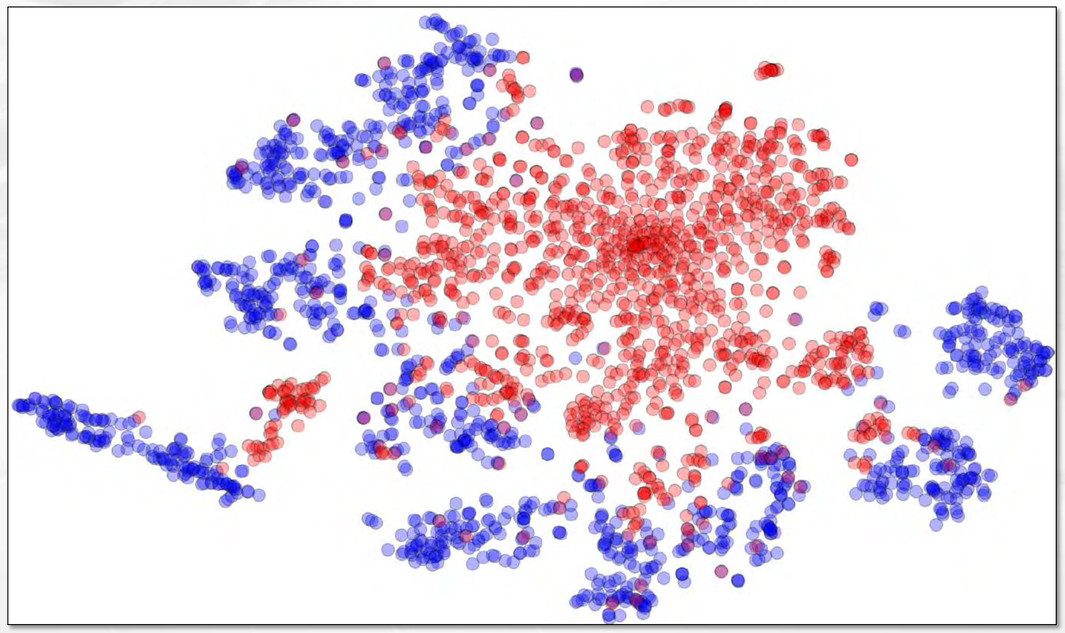
$$y = G_y(\mathbf{f}; \theta_y)$$



feature extractor

label predictor

When trained on source only, feature distributions do not match:

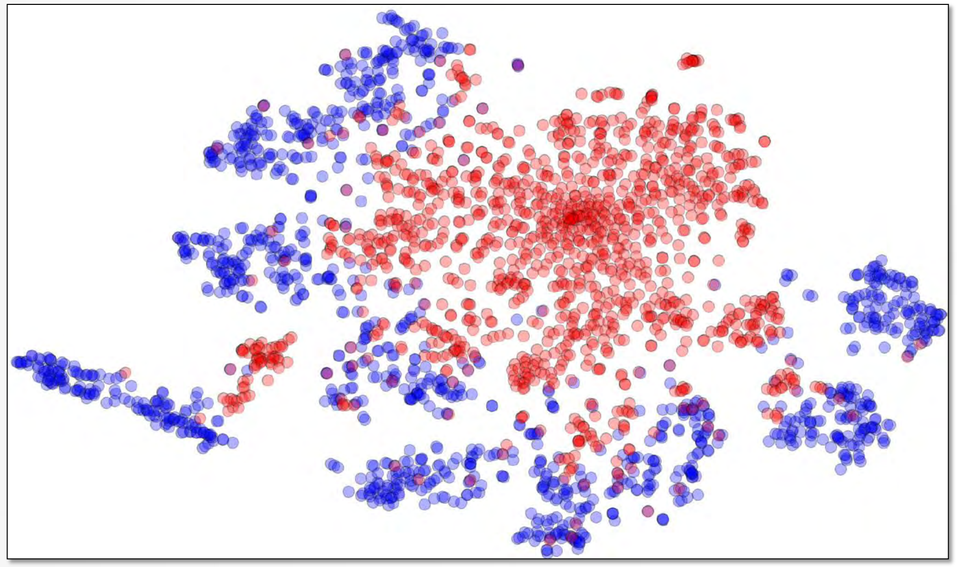


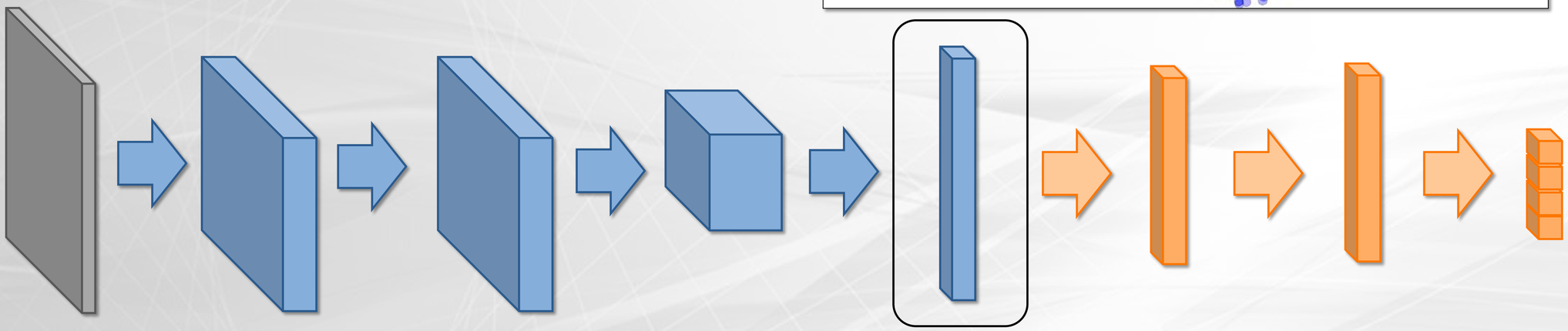
$$S(\mathbf{f}) = \{G_f(\mathbf{x}; \theta_f) \,|\, \mathbf{x} \sim S(\mathbf{x})\}$$

$$T(\mathbf{f}) = \{G_f(\mathbf{x}; \theta_f) \,|\, \mathbf{x} \sim T(\mathbf{x})\}$$

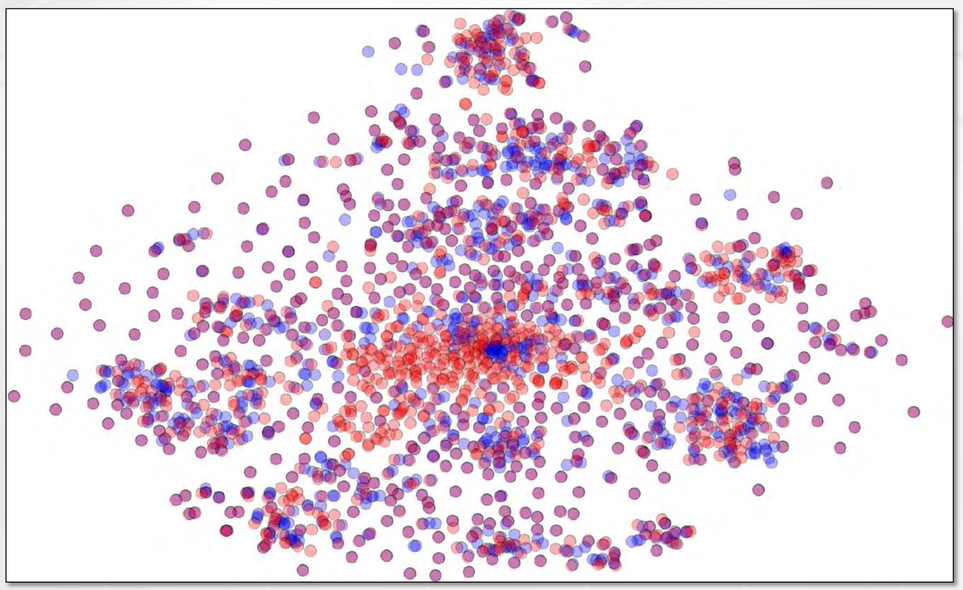
Idea 1: domain-invariant features wanted

Feature distribution without adaptation:

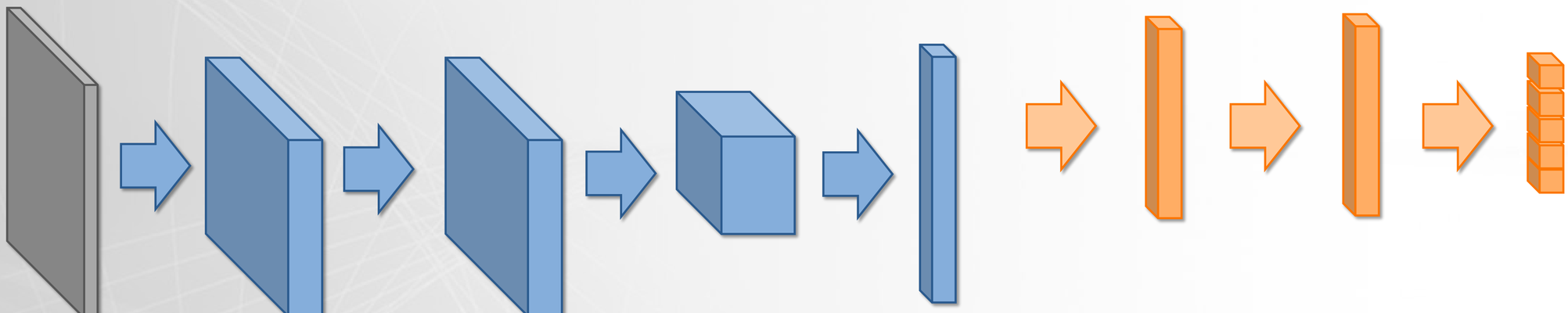




Our goal (after adaptation):

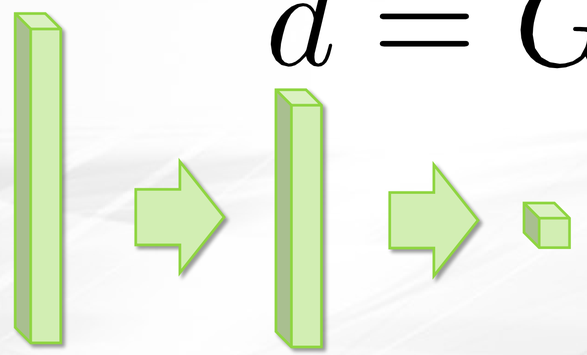


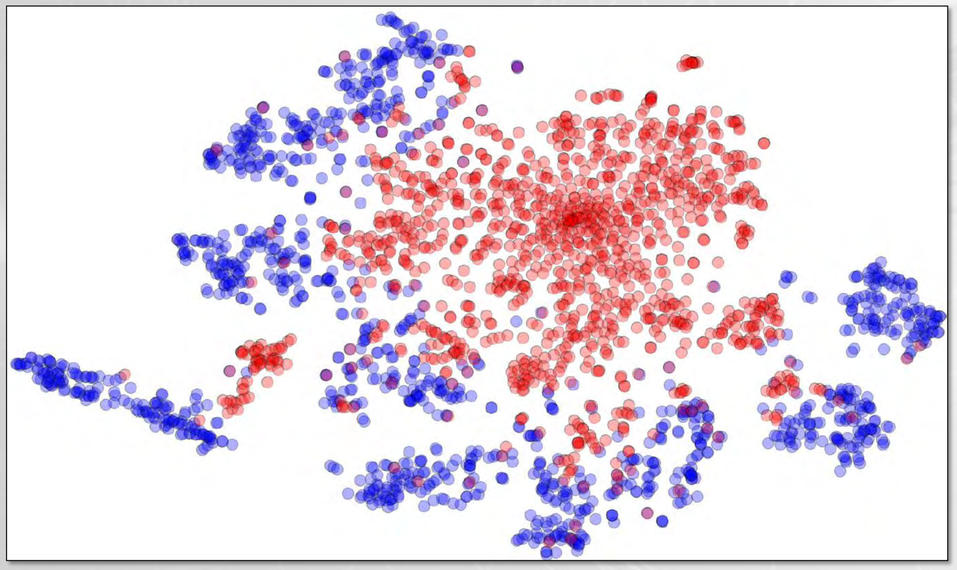
Idea 2: measuring domain shift



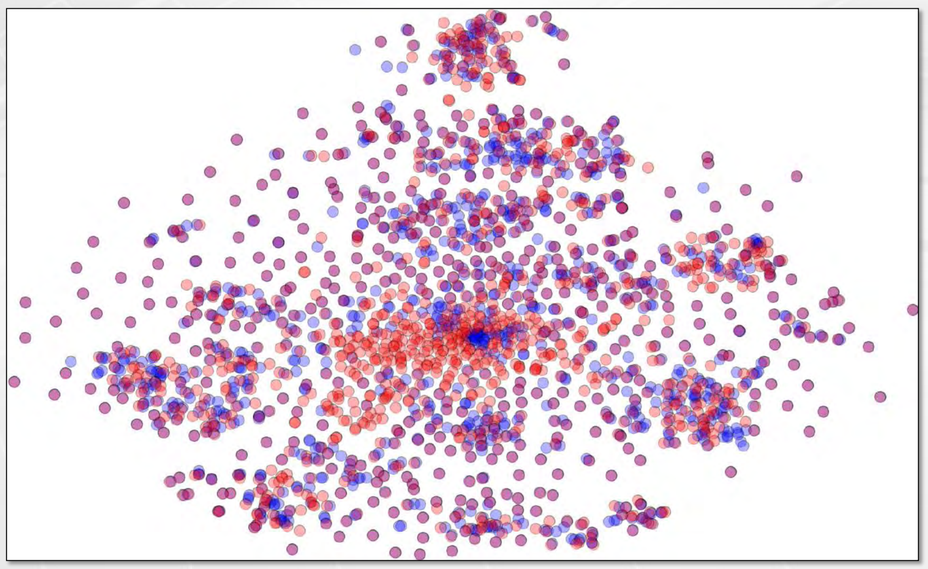
$$d = G_d(\mathbf{f}; \theta_d)$$

Domain classifier:



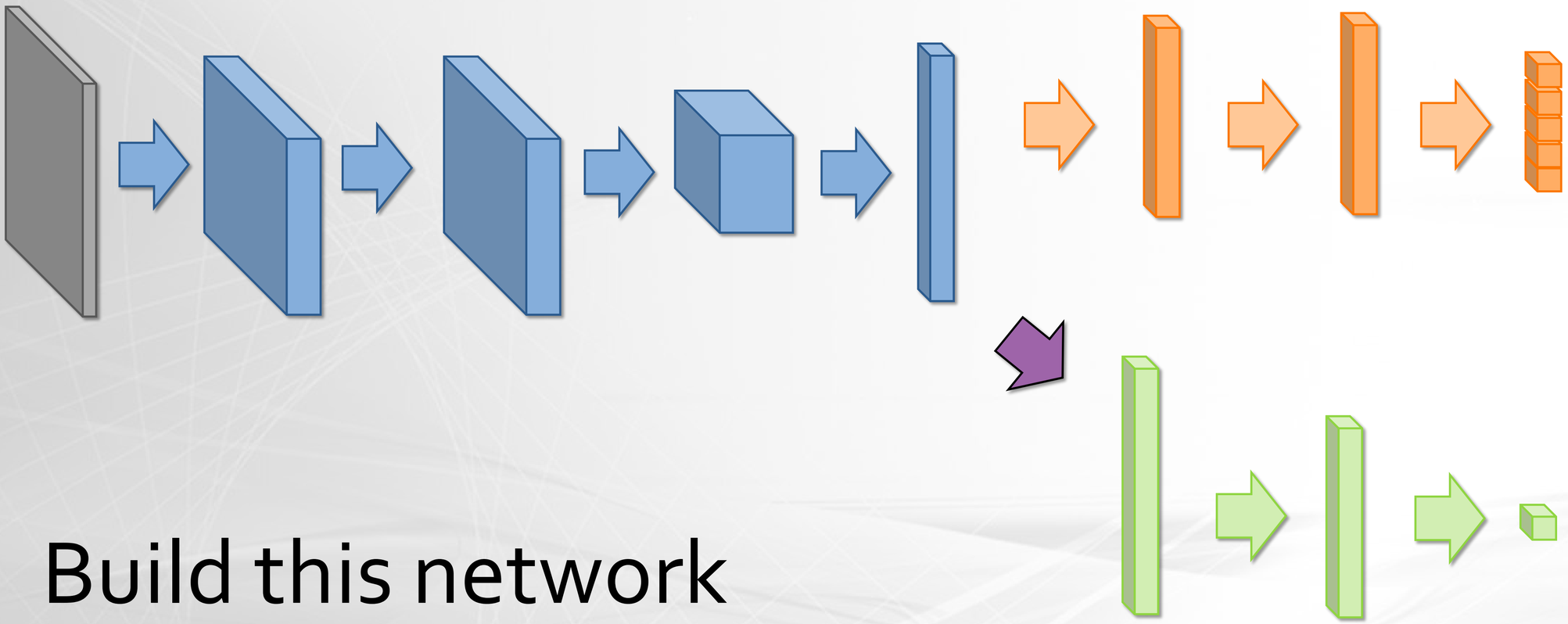


Domain loss low



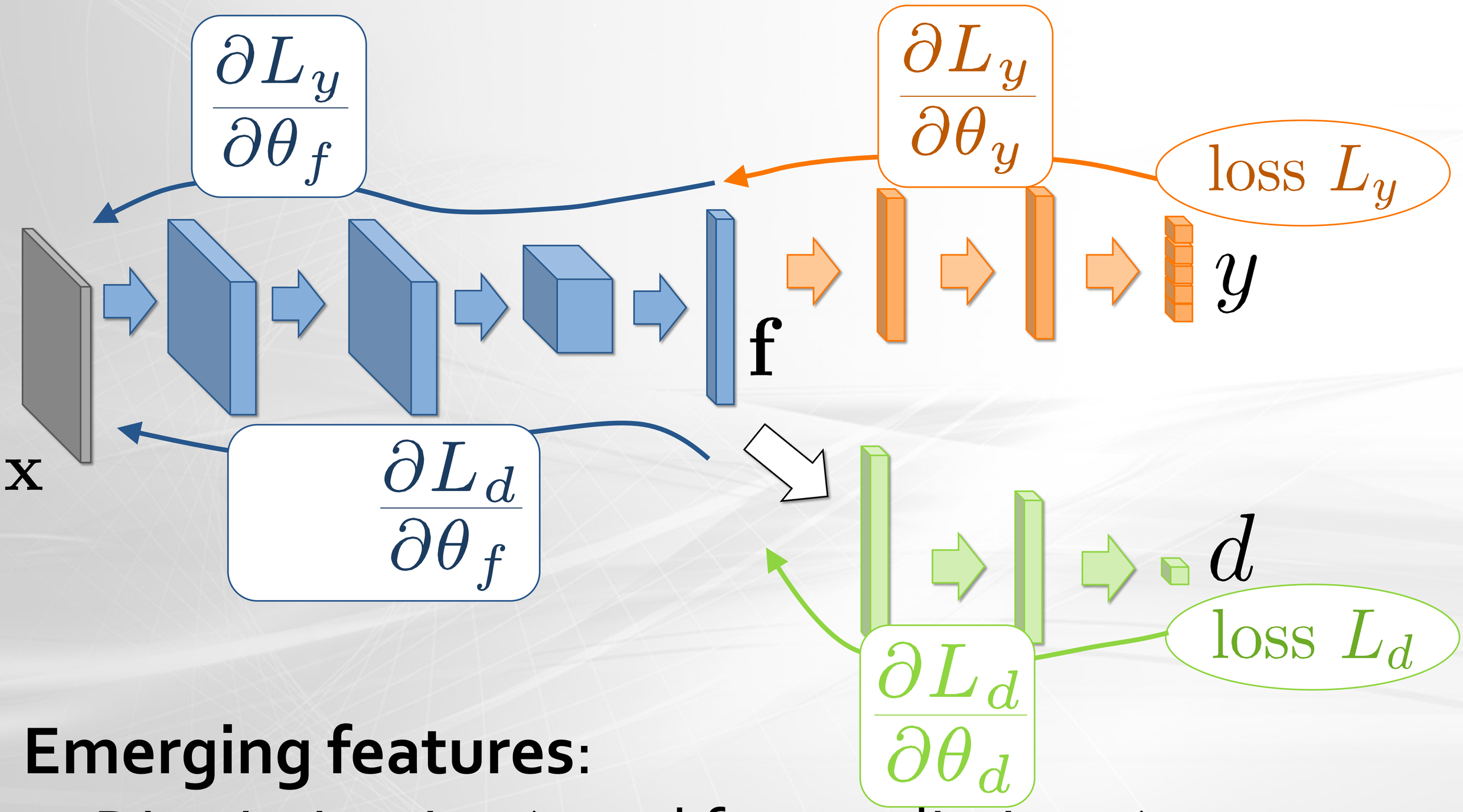
Domain loss high

Learning with adaptation



1. Build this network
2. Train feature extractor + class predictor on source data
3. Train feature extractor + domain classifier on source+target data
4. Use feature extractor + class predictor at test time

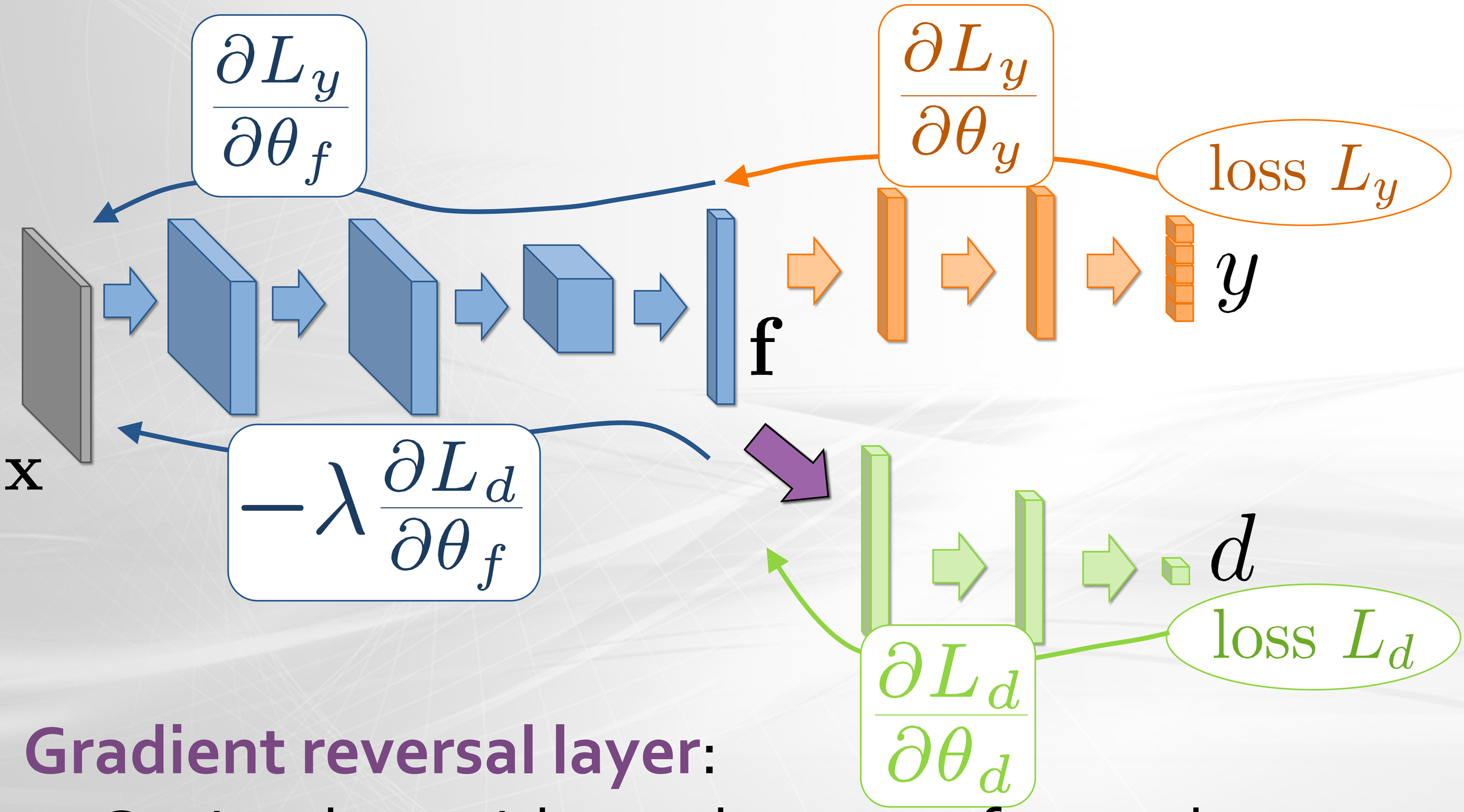
Idea 3: minimizing domain shift



Emerging features:

- Discriminative (good for predicting *y*)
- Domain-discriminative (good for predicting *d*)

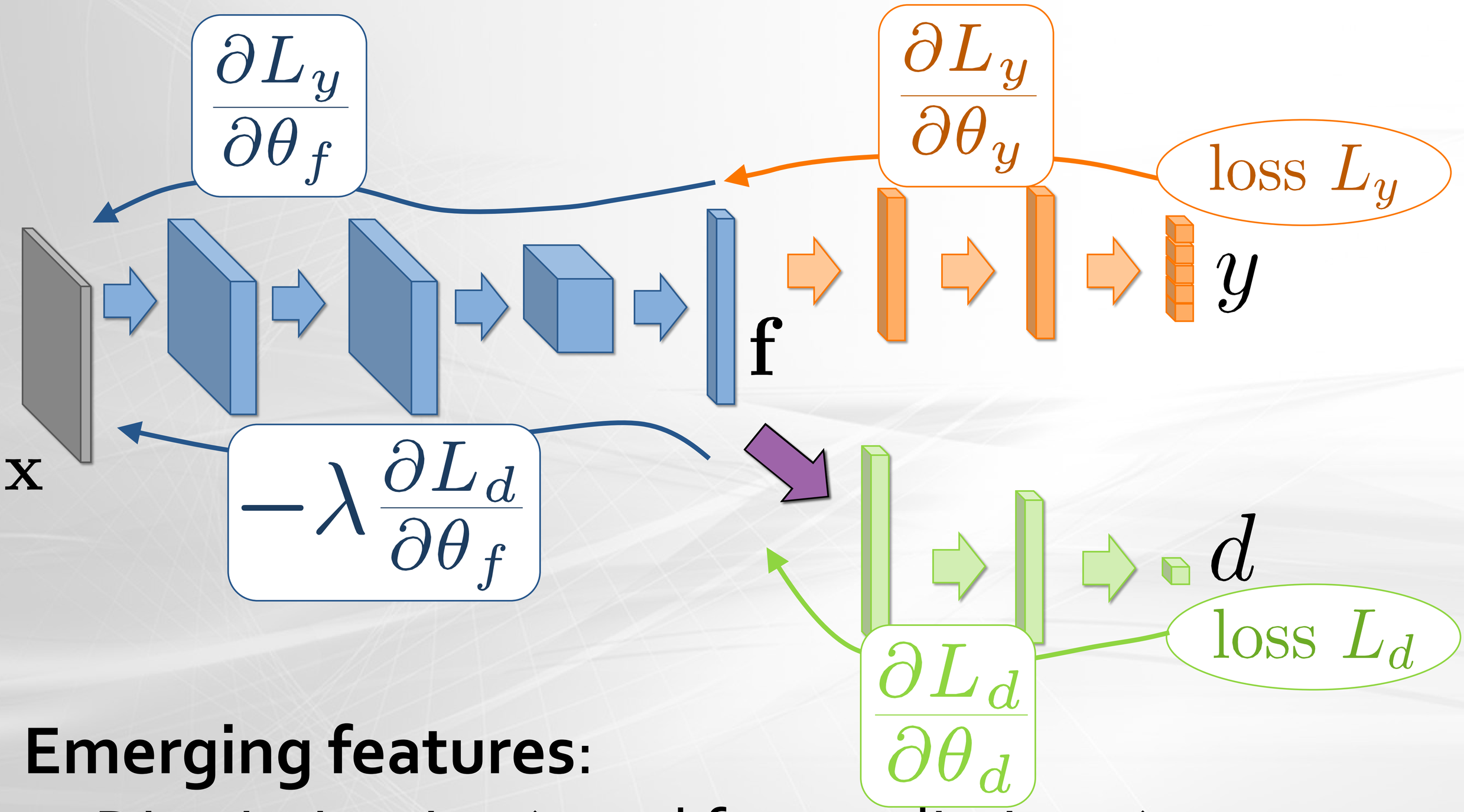
Idea 3: minimizing domain shift



Gradient reversal layer:

- Copies data without change at forwardprop
- Multiplies the gradient by -λ at backprop

Idea 3: minimizing domain shift



Emerging features:

- Discriminative (good for predicting *y*)
- Domain-invariant (not good for predicting *d*)

Gradient reversal layer

```
class GradReversalLayer : Layer {

        float lambda;

        blob forward (blob x)      {
                return x
        }

        blob backward(blob dzdy) {
                return multiply(dzdy, -lambda)
        }
}
```

Saddle point interpretation

Our objective (small label prediction loss + large domain classification loss wanted)

$$E(\theta_f, \theta_y, \theta_d) = \sum_{\substack{i=1..N \\ d_i=0}} L_y^i(\theta_f, \theta_y) - \lambda \sum_{i=1..N} L_d^i(\theta_f, \theta_d)$$

The backprop converges to a saddle point:

$$(\hat{\theta}_f, \hat{\theta}_y) = \arg\min_{\theta_f, \theta_y} E(\theta_f, \theta_y, \hat{\theta}_d)$$

$$\hat{\theta}_d = \arg\max_{\theta_d} E(\hat{\theta}_f, \hat{\theta}_y, \theta_d)\,.$$

Similar idea for generative networks:

[Goodfellow et al. Generative adversarial nets. In NIPS, 2014]

Backprop updates

Domain classification loss for the *i*th example

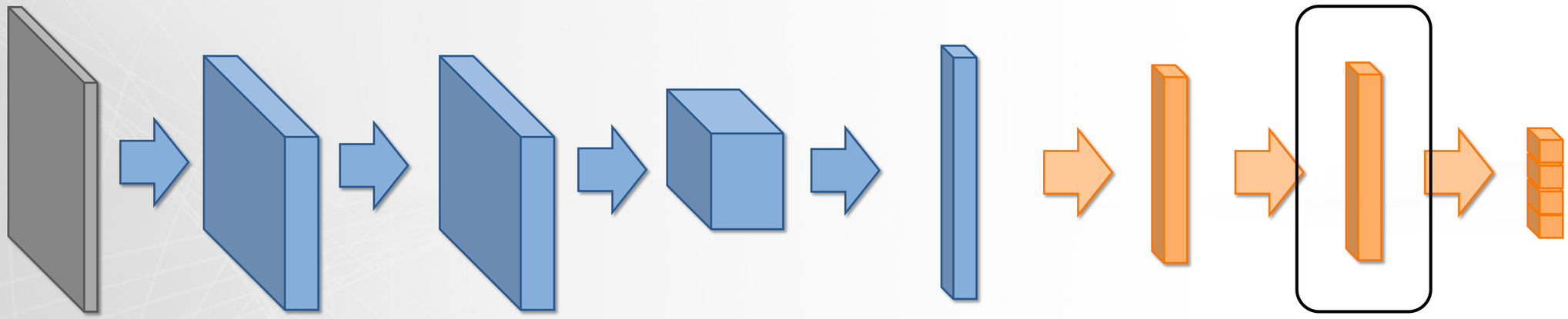
Label prediction loss for the *i*th example

$$\theta_f \longleftarrow \theta_f - \mu\left(\frac{\partial L_y^i}{\partial \theta_f} - \lambda\frac{\partial L_d^i}{\partial \theta_f}\right)$$

$$\theta_y \longleftarrow \theta_y - \mu\frac{\partial L_y^i}{\partial \theta_y}$$

$$\theta_d \longleftarrow \theta_d - \mu\frac{\partial L_d^i}{\partial \theta_d}$$

Initial experiments: baselines

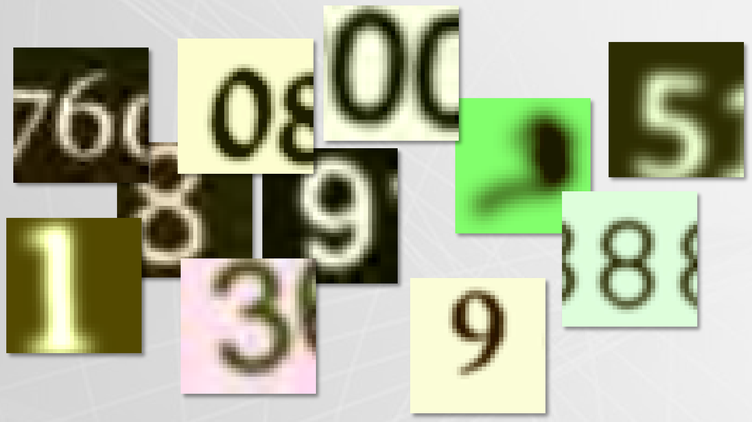


Upper bound: training on target domain **with labels**

Shallow adaptation baseline: *[Fernando et al., Unsupervised visual domain adaptation using subspace alignment. ICCV, 2013]* applied to the last-but-one layer

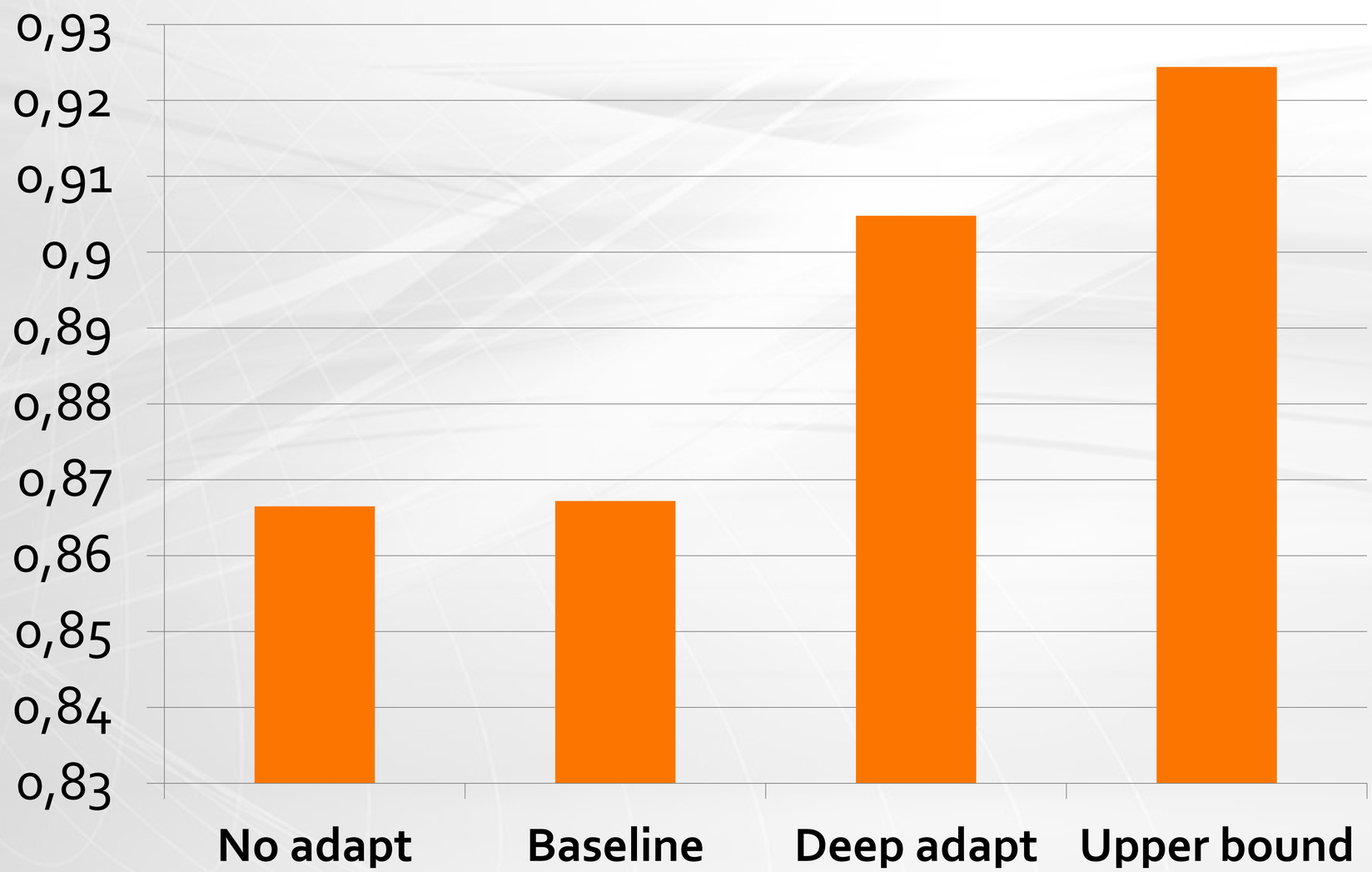
Lower bound: training on source domain only

Example: from synthetic to real

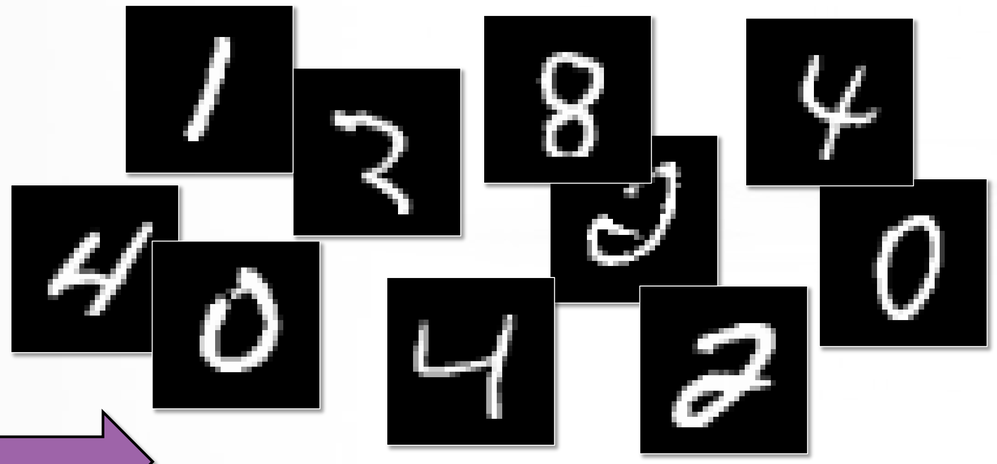


"Windows digits"

"House numbers"

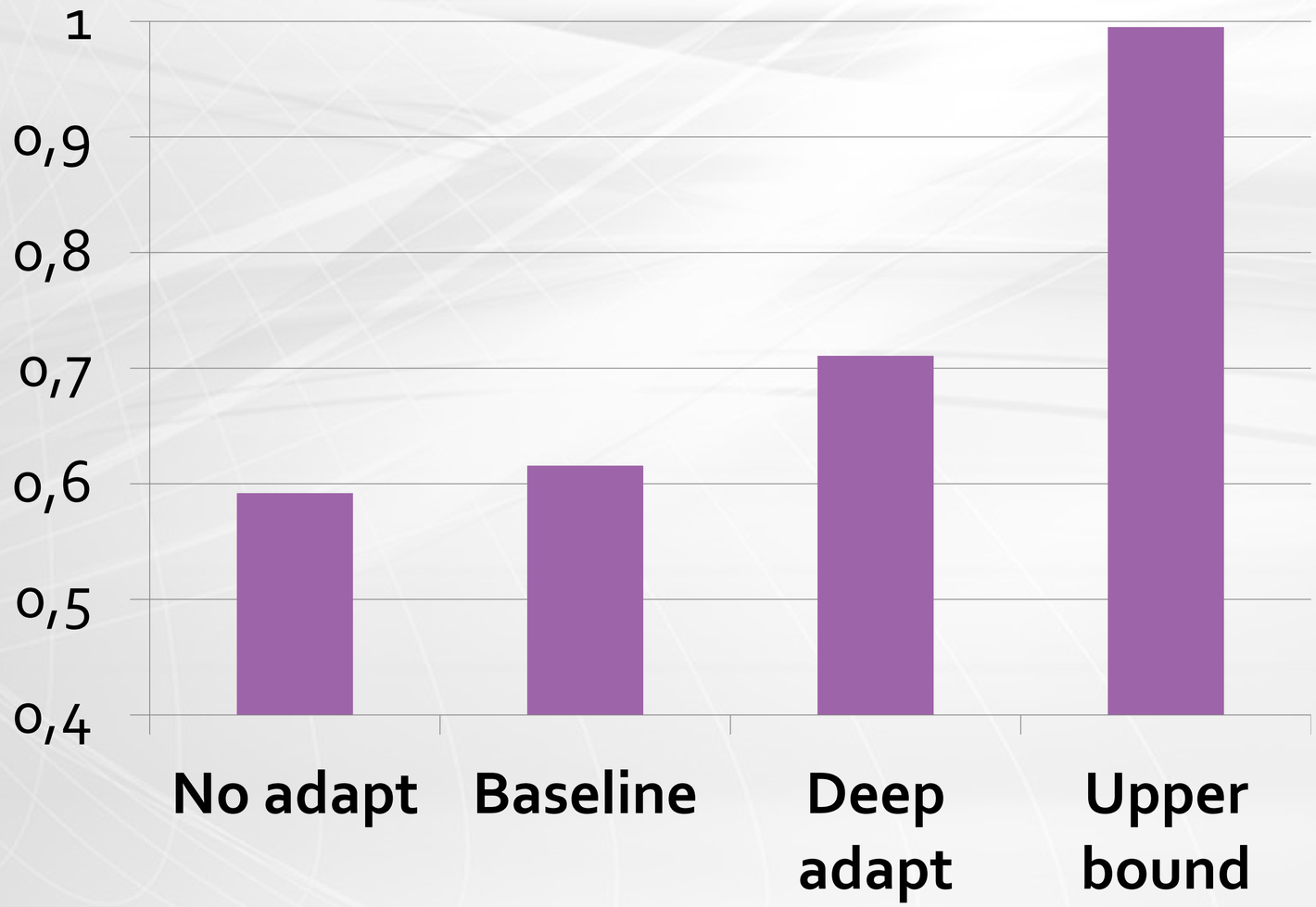


Example: large gap



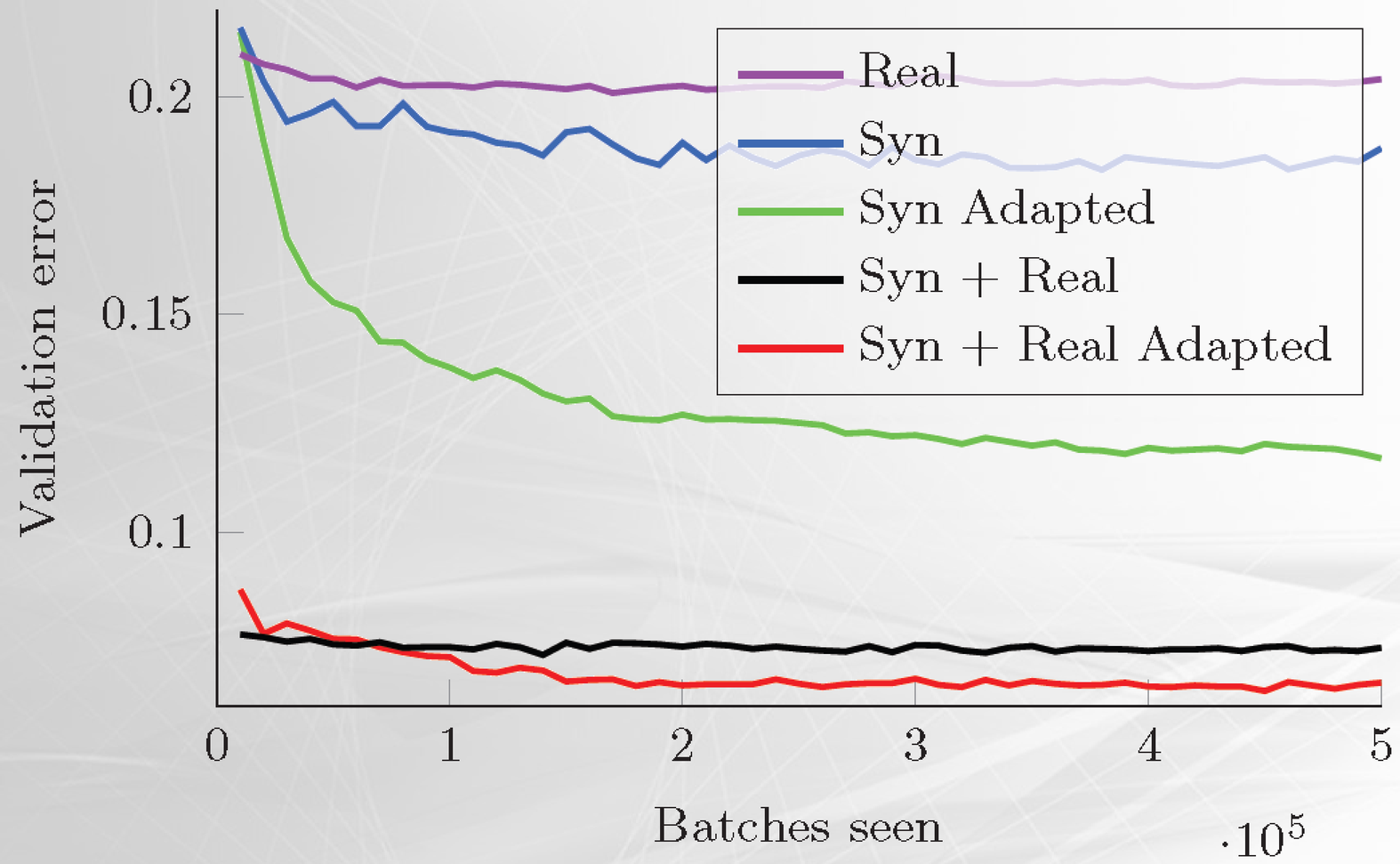
MNIST

“House numbers”



Reverse direction does not work ☹

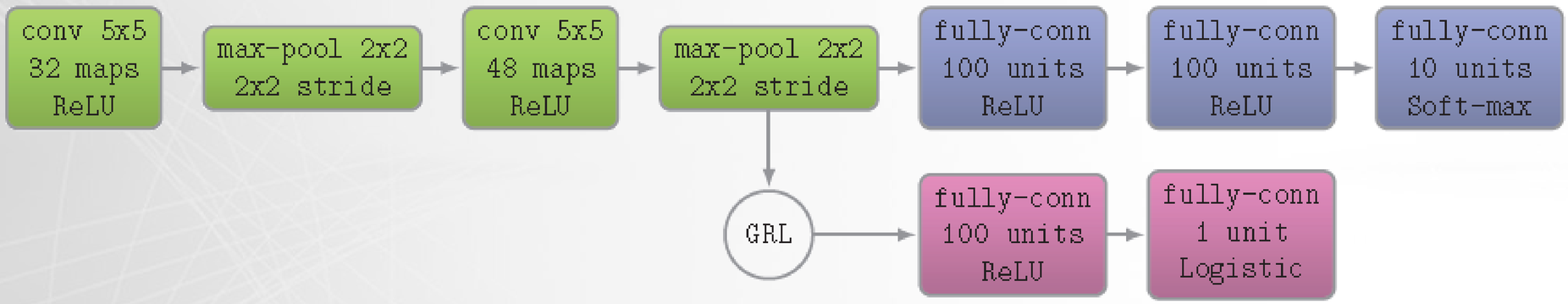
Traffic signs: semi-supervised adaptation



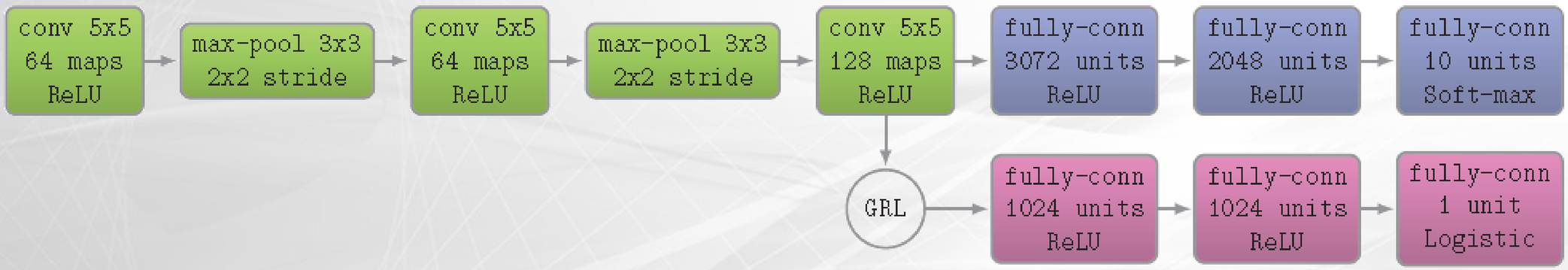
- 43 classes
- 430 Real examples
- 131000 Synthetic examples

Testing on Real only

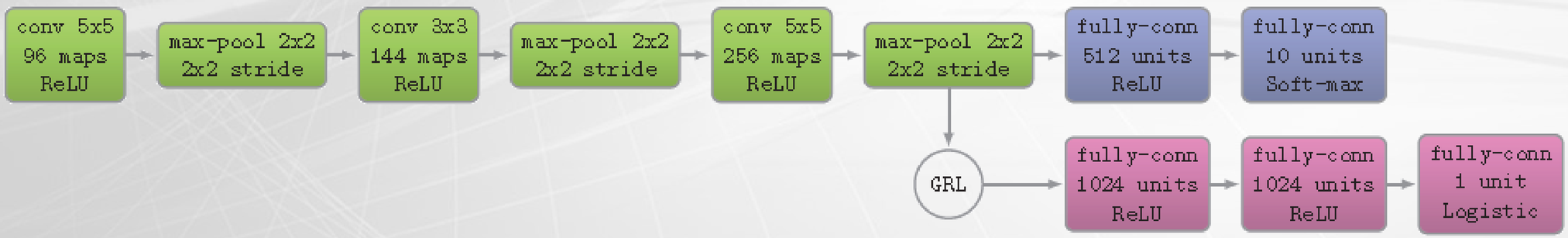
Sample architectures for image classification



(a) MNIST architecture; inspired by the classical LeNet-5 (LeCun et al., 1998).



(b) SVHN architecture; adopted from Srivastava et al. (2014).



(c) GTSRB architecture; we used the single-CNN baseline from Cireşan et al. (2012) as our starting point.

Office dataset

[Saenko et al. ECCV2010]

Deep Model Adaptation using Domain Adversarial Training

Results on Office dataset

Method / Source	Amazon	DSLR	Webcam
Target	Webcam	Webcam	DSLR
GFK(PLS, PCA) (Gong et al., 2012)	.197	.497	.6631
SA* (Fernando et al., 2013)	.450	.648	.699
DLID (Chopra et al., 2013)	.519	.782	.899
DDC (Tzeng et al., 2014)	.618	.950	.985
DAN (Long and Wang, 2015)	.685	.960	.990
Source only	.642	.961	.978
DANN	**.730**	**.964**	**.992**

Most similar approach (matches means of distributions):

[Tzeng et al. Deep domain confusion: Maximizing for domain invariance. CoRR, abs/1412.3474, 2014]

Beyond image classification

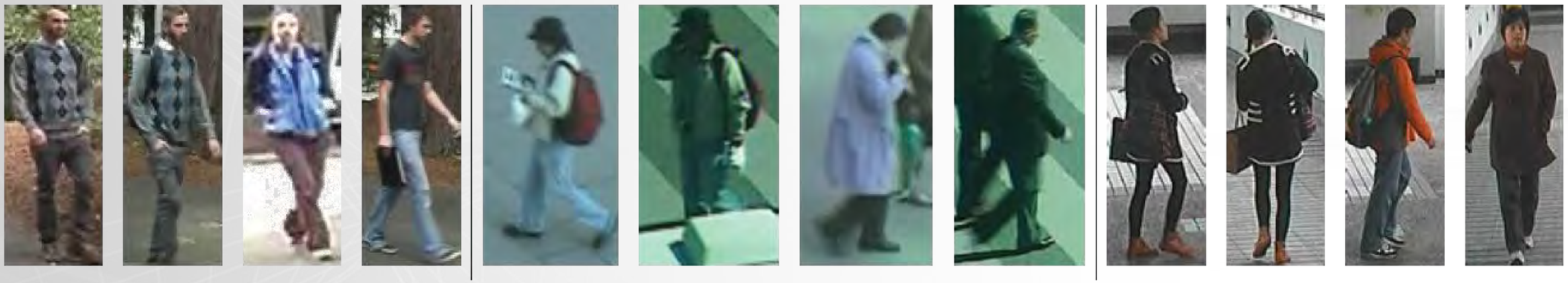
Domain-Adversarial Training of Neural Networks

Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, Victor Lempitsky, JMLR 2016

		Original data			mSDA representation		
SOURCE	TARGET	DANN	NN	SVM	DANN	NN	SVM
BOOKS	DVD	.784	.790	**.799**	.829	.824	**.830**
BOOKS	ELECTRONICS	.733	.747	**.748**	**.804**	.770	.766
BOOKS	KITCHEN	**.779**	.778	.769	**.843**	.842	.821
DVD	BOOKS	.723	.720	**.743**	.825	.823	**.826**
DVD	ELECTRONICS	**.754**	.732	.748	**.809**	.768	.739
DVD	KITCHEN	**.783**	.778	.746	.849	**.853**	.842
ELECTRONICS	BOOKS	**.713**	.709	.705	**.774**	.770	.762
ELECTRONICS	DVD	**.738**	.733	.726	**.781**	.759	.770
ELECTRONICS	KITCHEN	**.854**	**.854**	.847	.881	**.863**	.847
KITCHEN	BOOKS	**.709**	.708	.707	.718	.721	**.769**
KITCHEN	DVD	**.740**	.739	.736	**.789**	**.789**	.788
KITCHEN	ELECTRONICS	**.843**	.841	.842	.856	.850	**.861**

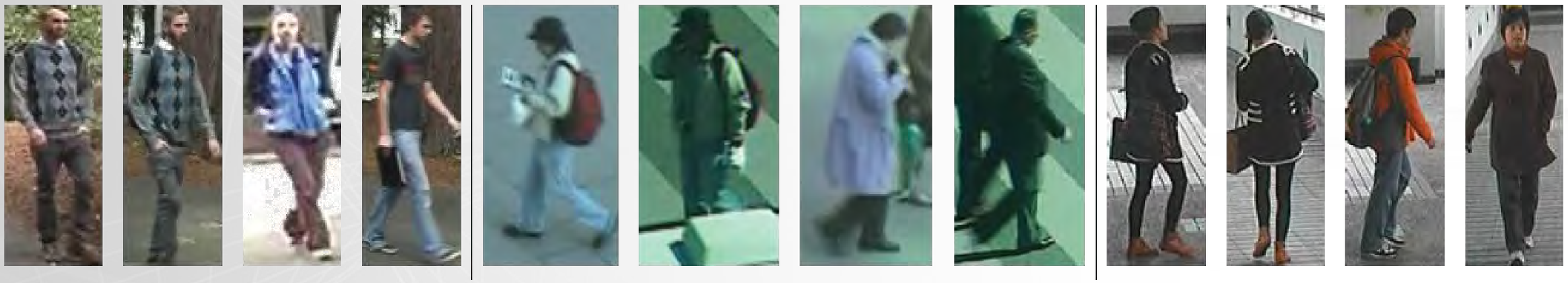
(a) Classification accuracy on the Amazon reviews data set

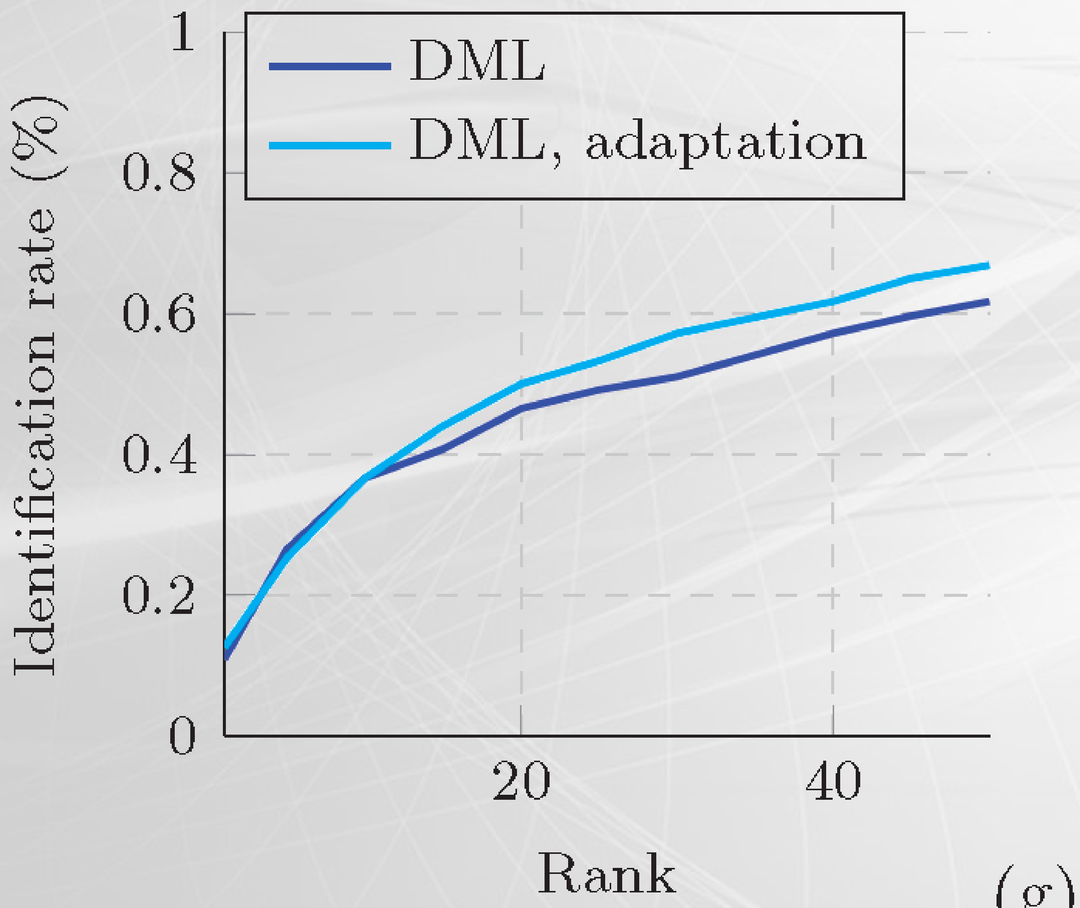
Adaptation for Person Re-identification



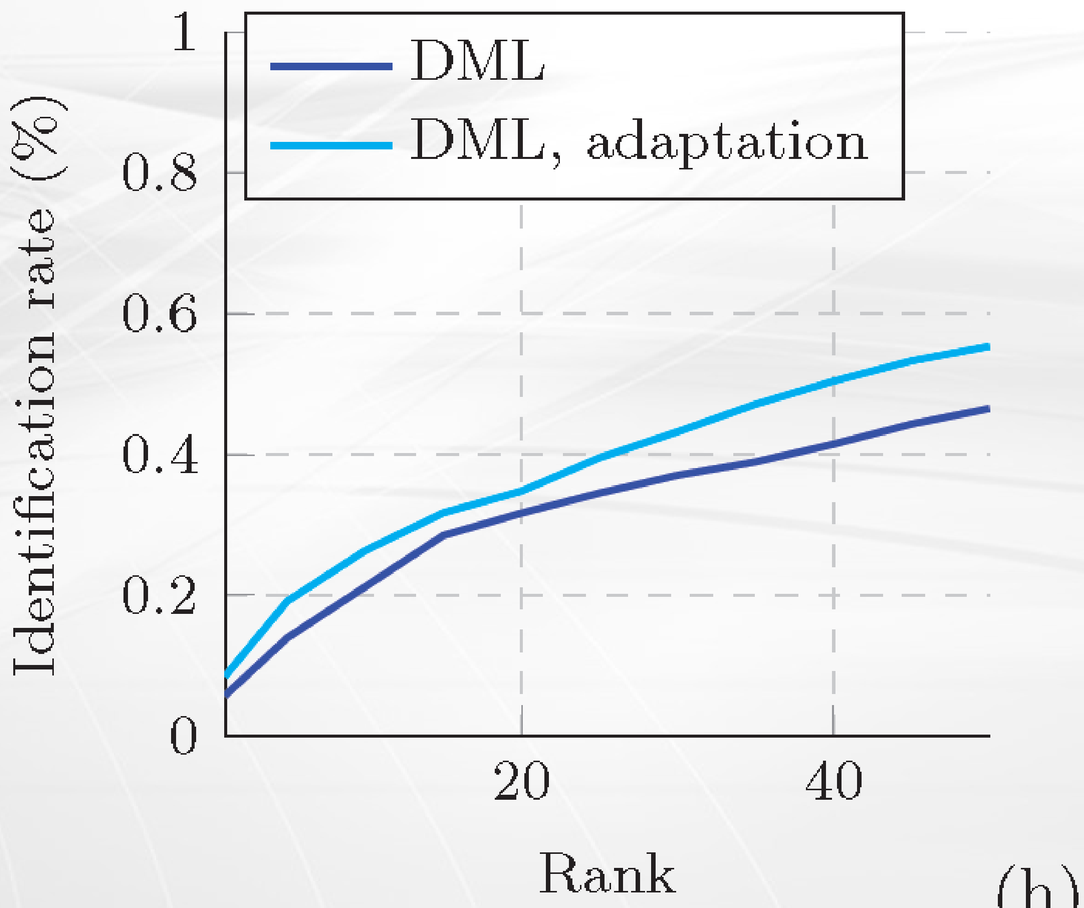
VIPER to CUHK

Adaptation for Person Re-identification





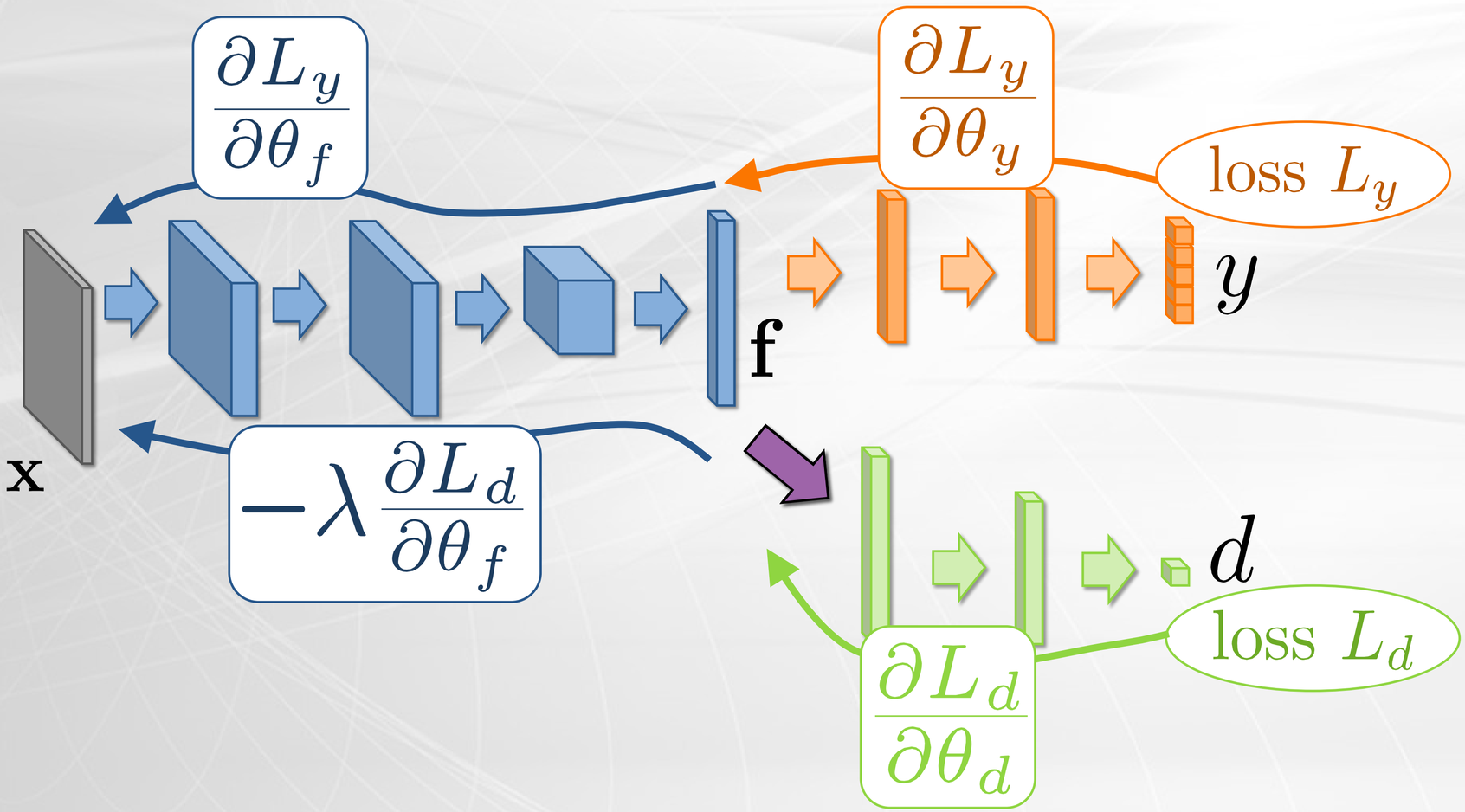
(g) VIPeR → CUHK/p1



(h) PRID → CUHK/p1

Caveats

- Domains should not be too far apart
- Early on, the gradient from the domain classification loss should not be too strong
- The trick used to obtain the results: gradually increase λ from 0 to 1



Conclusion

- Scalable method for deep unsupervised domain adaptation
- Based on simple idea. Takes few lines of code (+ defining a specific network architecture). *Caffe implementation available.*
- State-of-the-art results
- Unsupervised parameter tuning is easy (look at the domain classifier error)
- Main challenge: initialization and stepsize

http://sites.skoltech.ru/compvision/projects/grl/