# 6.819 / 6.869: Advances in Computer Vision

Deep Learning for Vision III:
Inversion, Sequences, Transfer, Applications

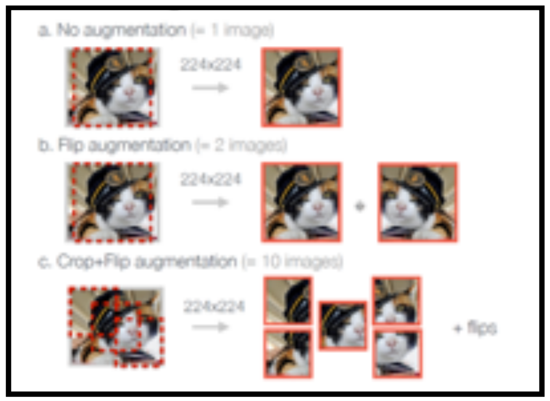
Website:
http://6.869.csail.mit.edu/fa15/

**Instructor: Yusuf Aytar**

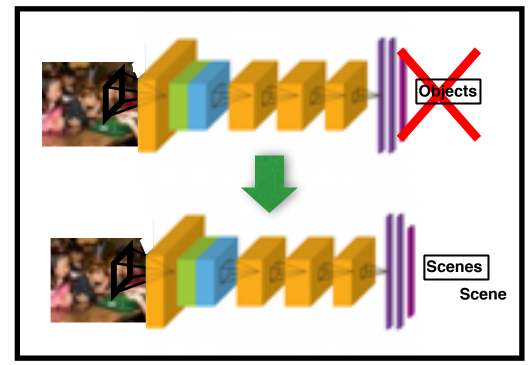
**Lecture** TR 9:30AM – 11:00AM
(Room 34-101)

# Recap on Deep Learning

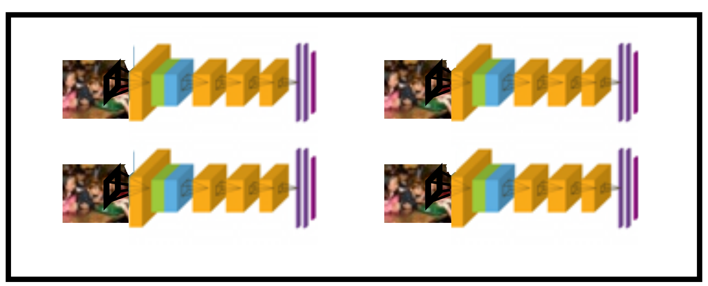
### Data Augmentation Helps



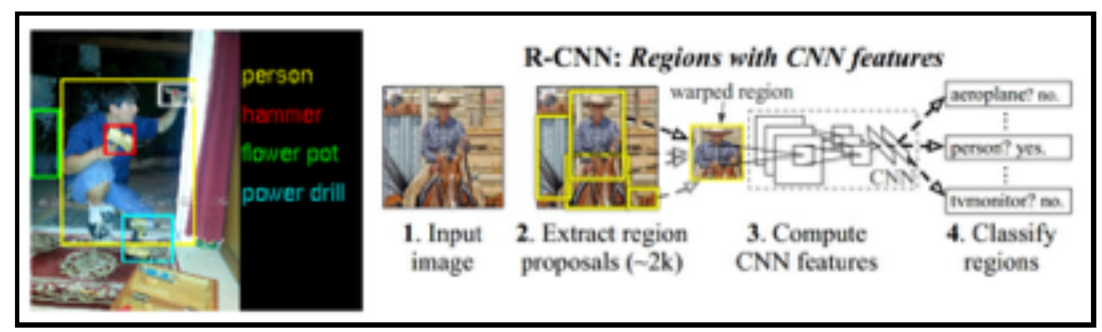

### Fine-tuning is good with limited data



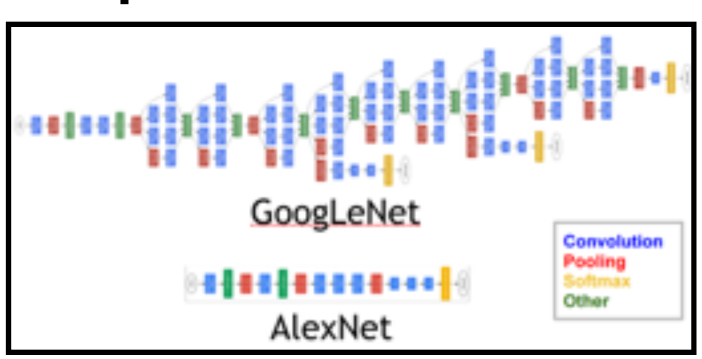

### Multiple model averaging helps



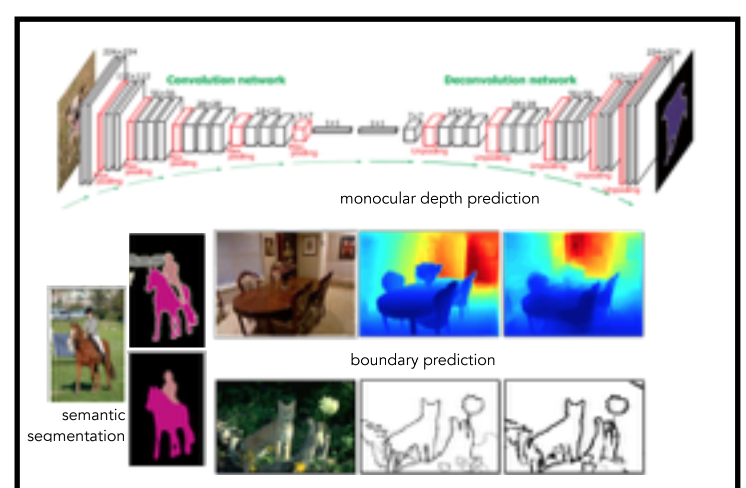
### Object Detection




### Deeper is Better




### Pixel Labeling through Deconvolution



# Overview

**Understanding Deep Networks / DeepArt**

Feature Inversion, inverting text, neural style transfer,

**Learning with Sequences**

Recurrent Neural Networks, LSTMs, Image Captioning

**Transfer in Deep Learning**

Domain Adaptation, Multi-task Learning, Domain Confusion

**Some Applications**

Face Recognition, Action Recognition

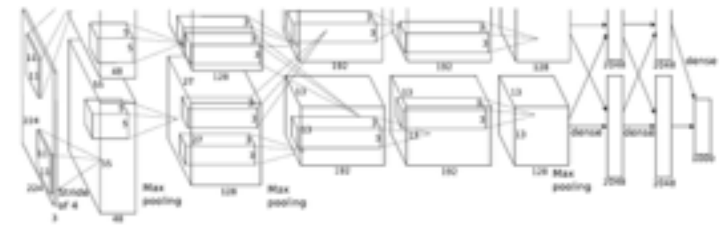
Understanding Deep Networks

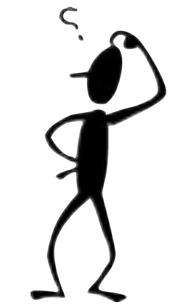
# Feature Visualization & Inversion

# Major Criticisms on Neural Networks

**Black Box**

It works but why?





**Harder to Analyze**

Non convex objective with millions of parameters

We don’t have a …

deep understanding of the method,

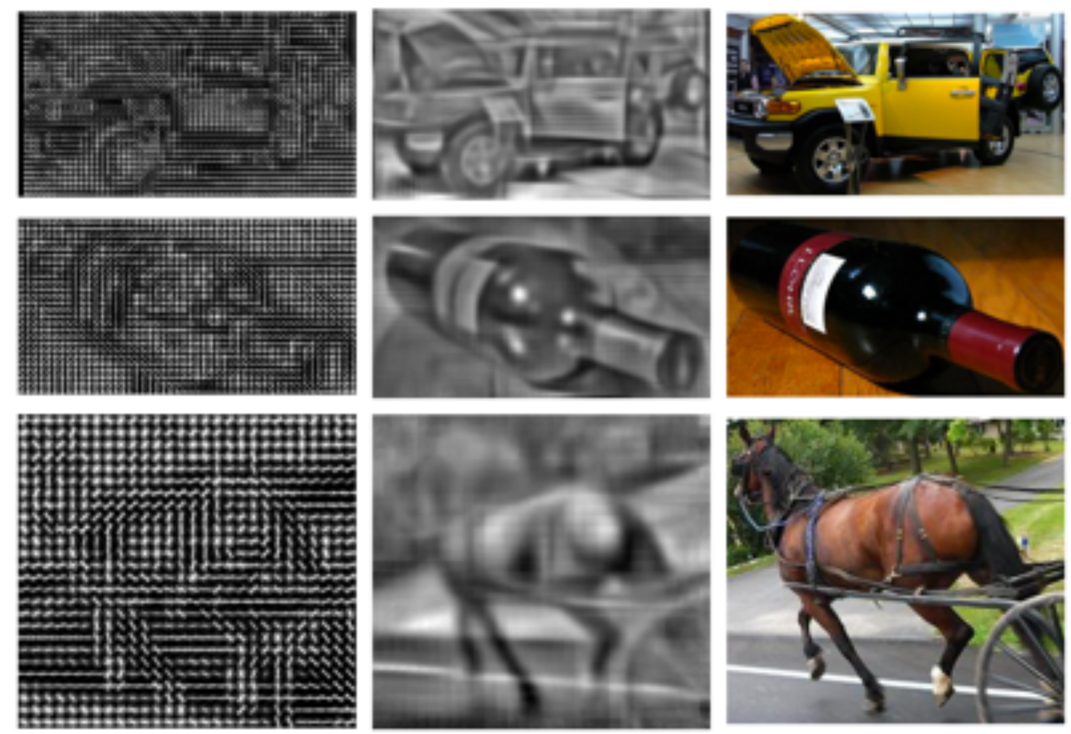
strong control over the learning mechanism.

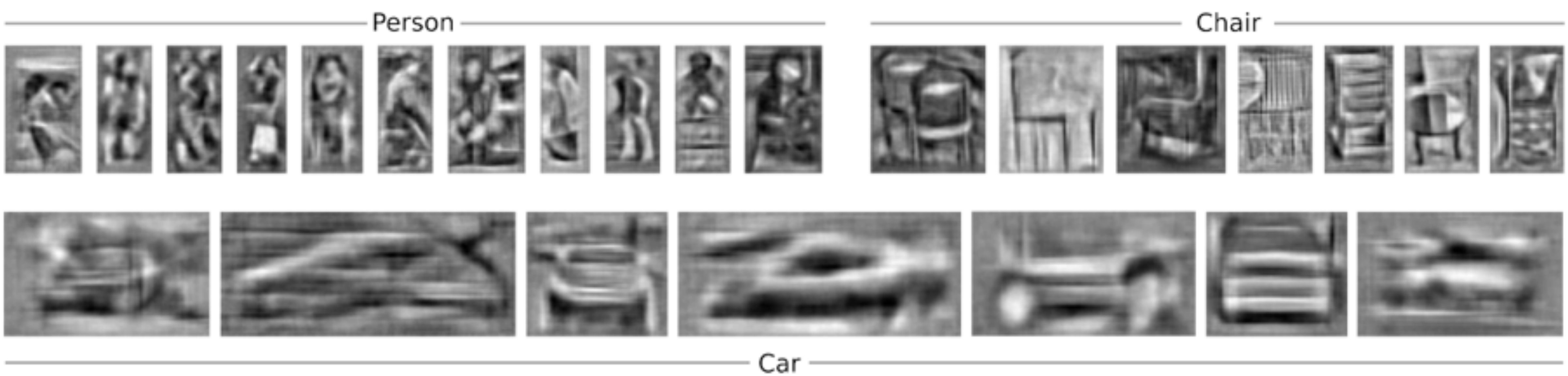
# How can we gain insight?

**Visualization / Inversion**

# **HOGgles:**
# Visualizing Object Detection Features

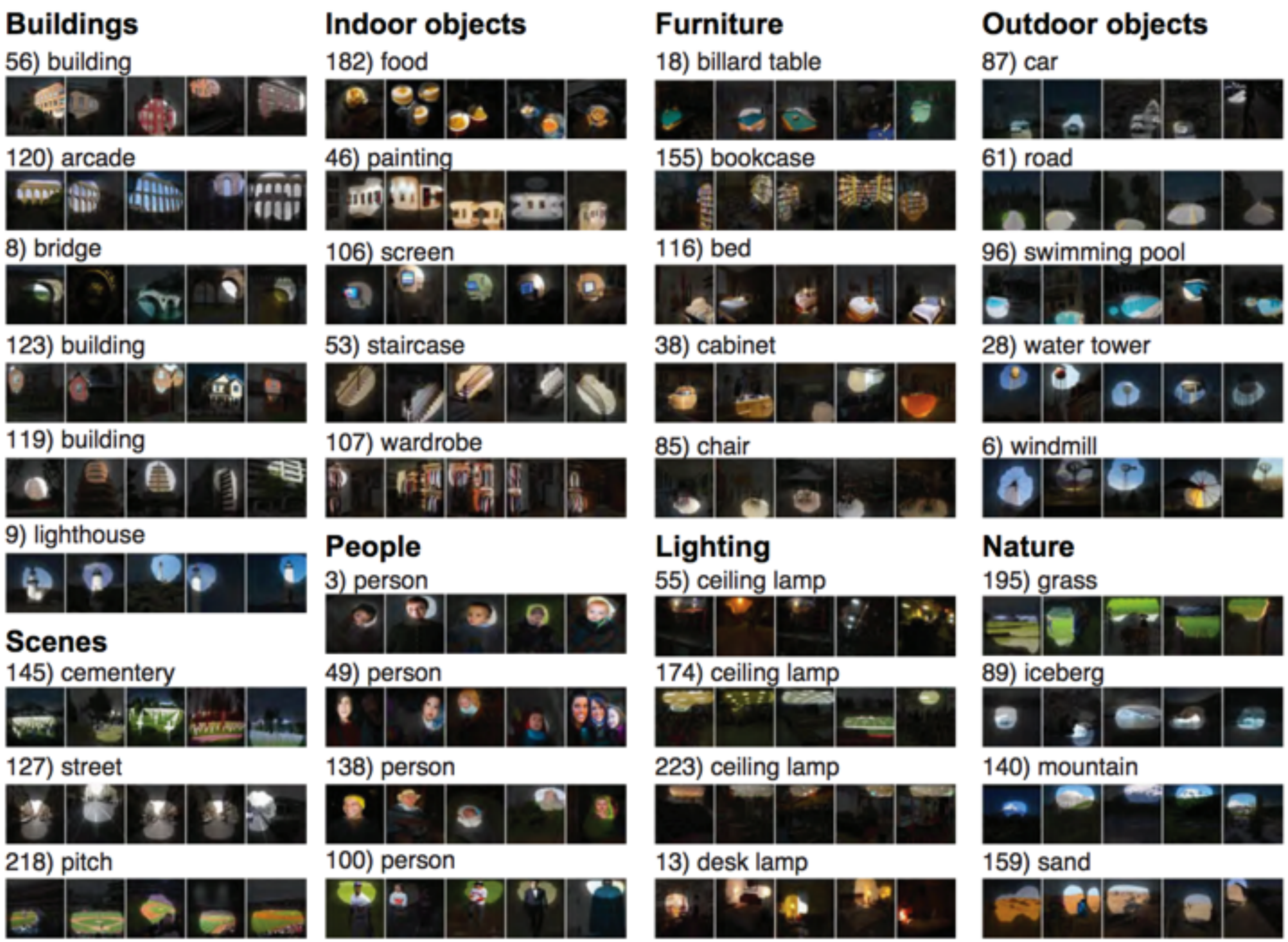
(a) Human Vision (b) HOG Vision






C. Vondrick, A. Khosla, T. Malisiewicz, A. Torralba. "HOGgles: Visualizing Object Detection Features", ICCV'13

# Emerging Object Detectors in Scene CNNS

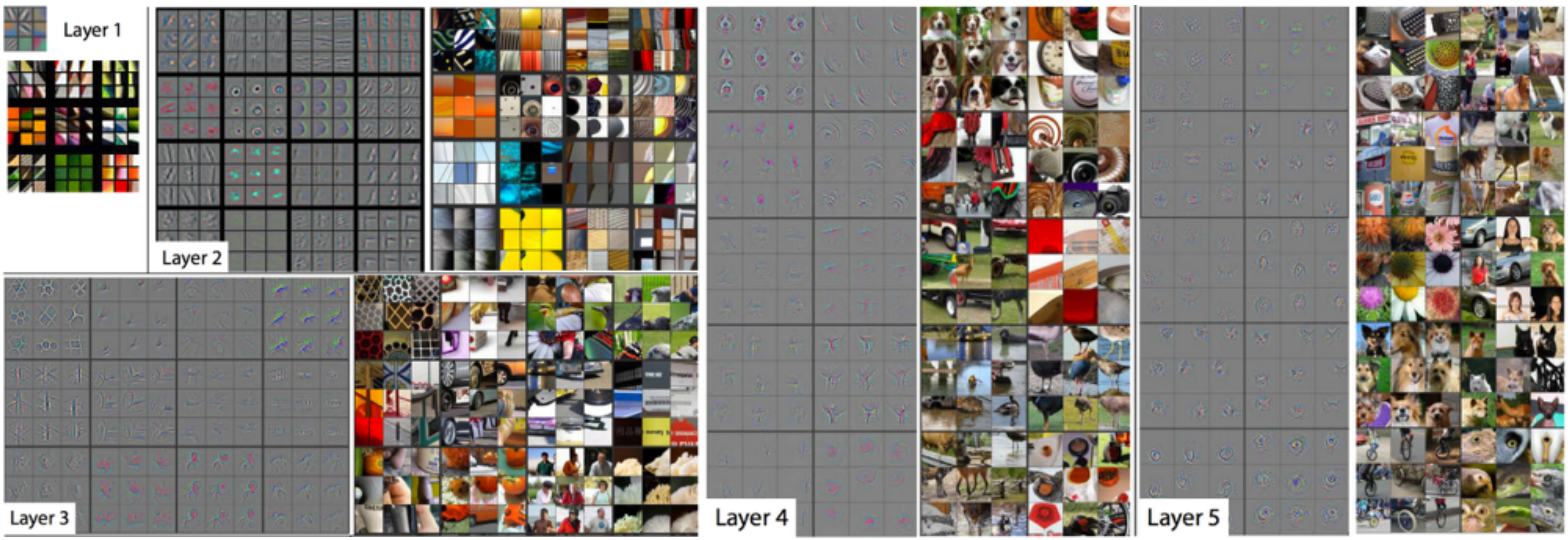



OBJECT DETECTORS EMERGE IN DEEP SCENE CNNS, Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, Antonio Torralba, ICLR 2015

# Visualizing and Understanding Convolutional Networks

corners & edge/color conjunctions

objects with pose variations



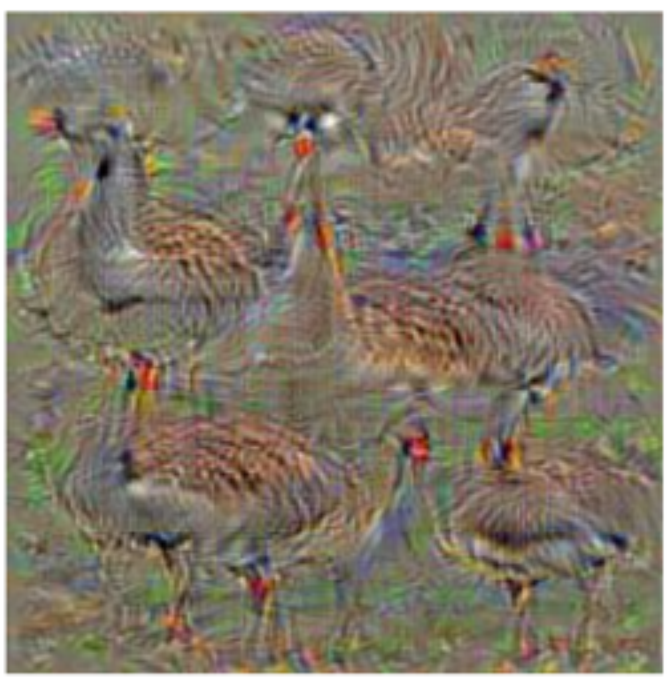

similar textures

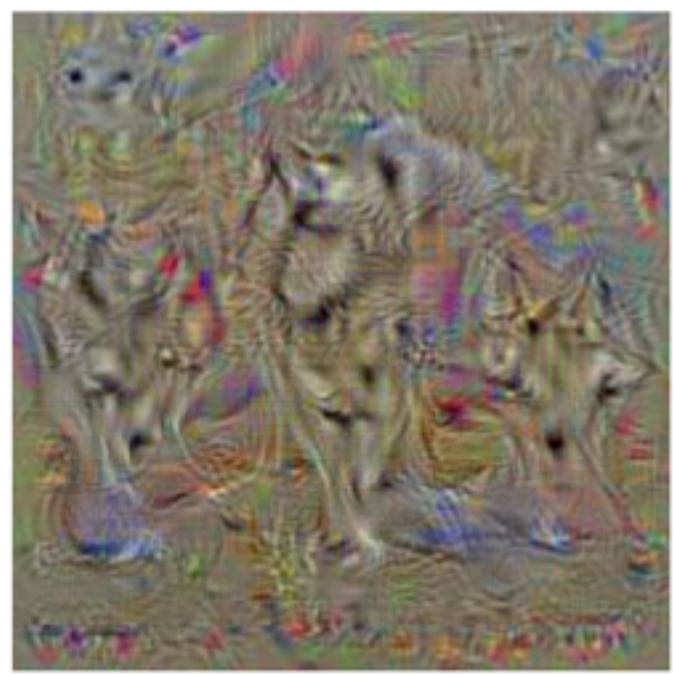
Object parts (dog-face, bird-legs)

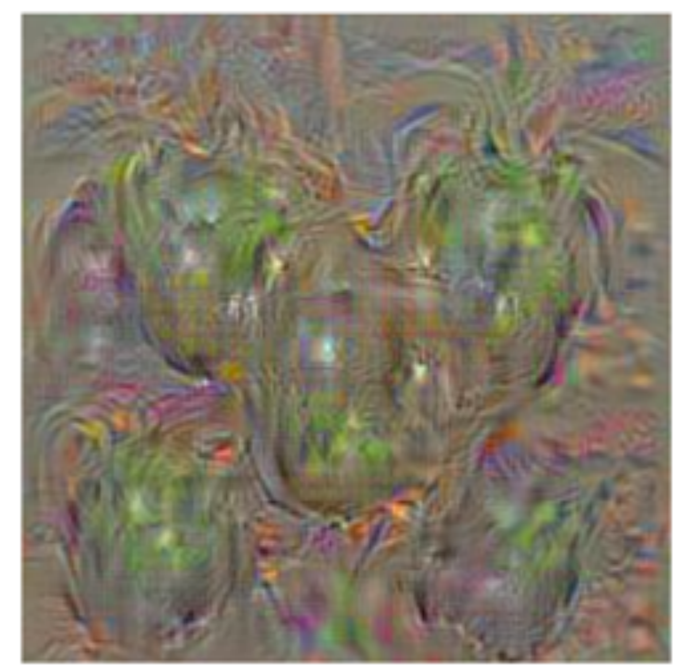
Visualizing and Understanding Convolutional Networks, Matthew, D. Zeiler and Rob Fergus, ECCV'14

# Deep Inside Convolutional Networks: Visualizing Image Classification Models

What are these objects?

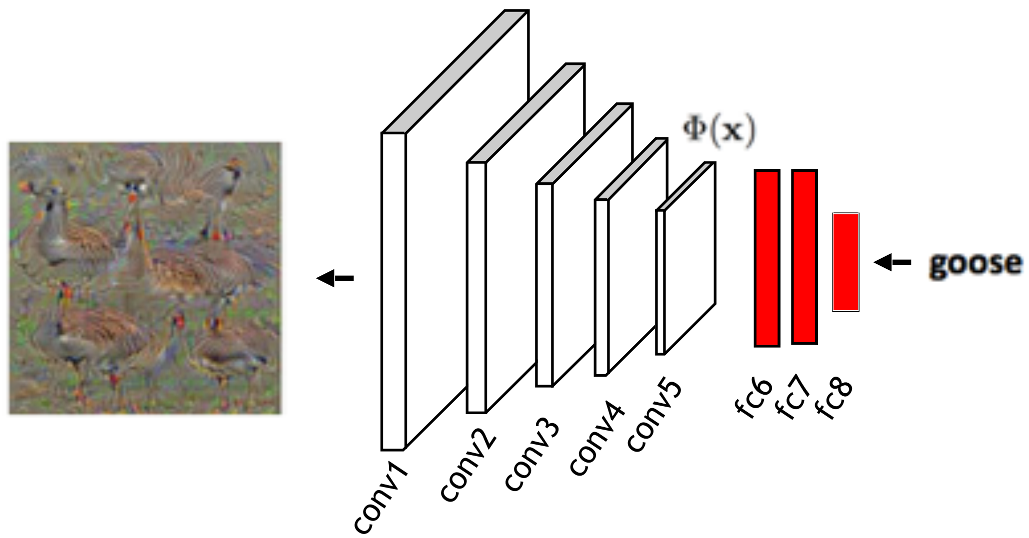




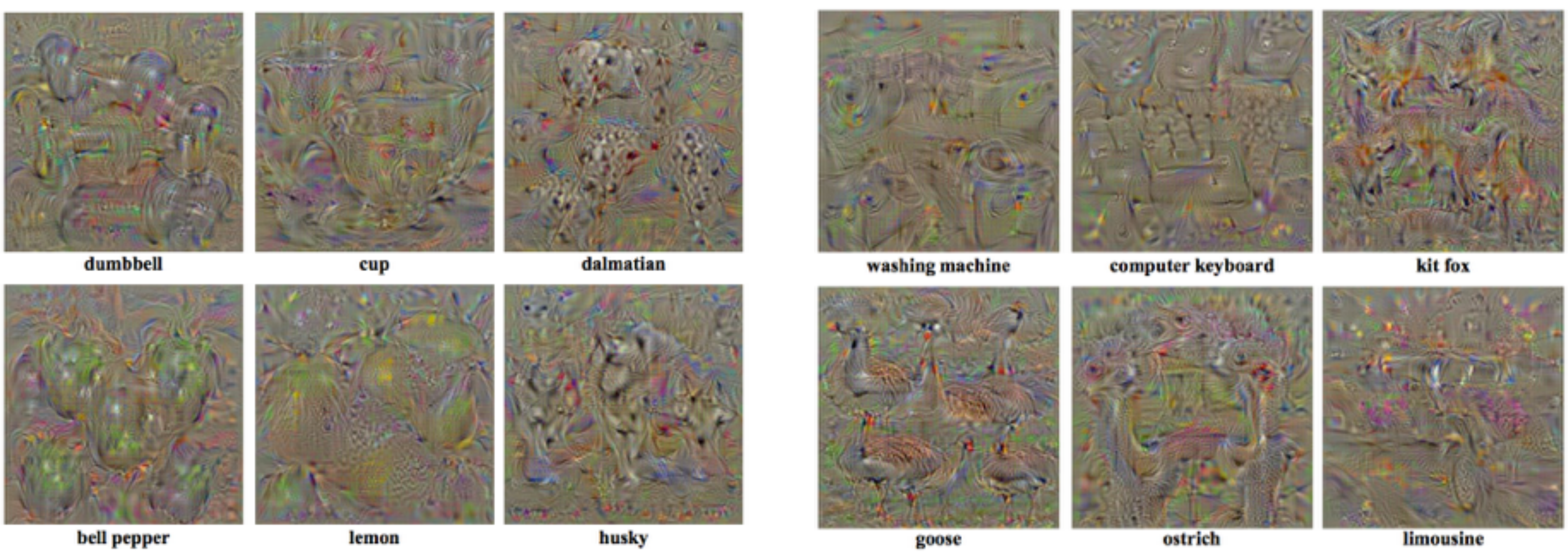


Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps, Karen Simonyan Andrea Vedaldi Andrew Zisserman, ICLR 2014

# Deep Inside Convolutional Networks: Visualizing Image Classification Models

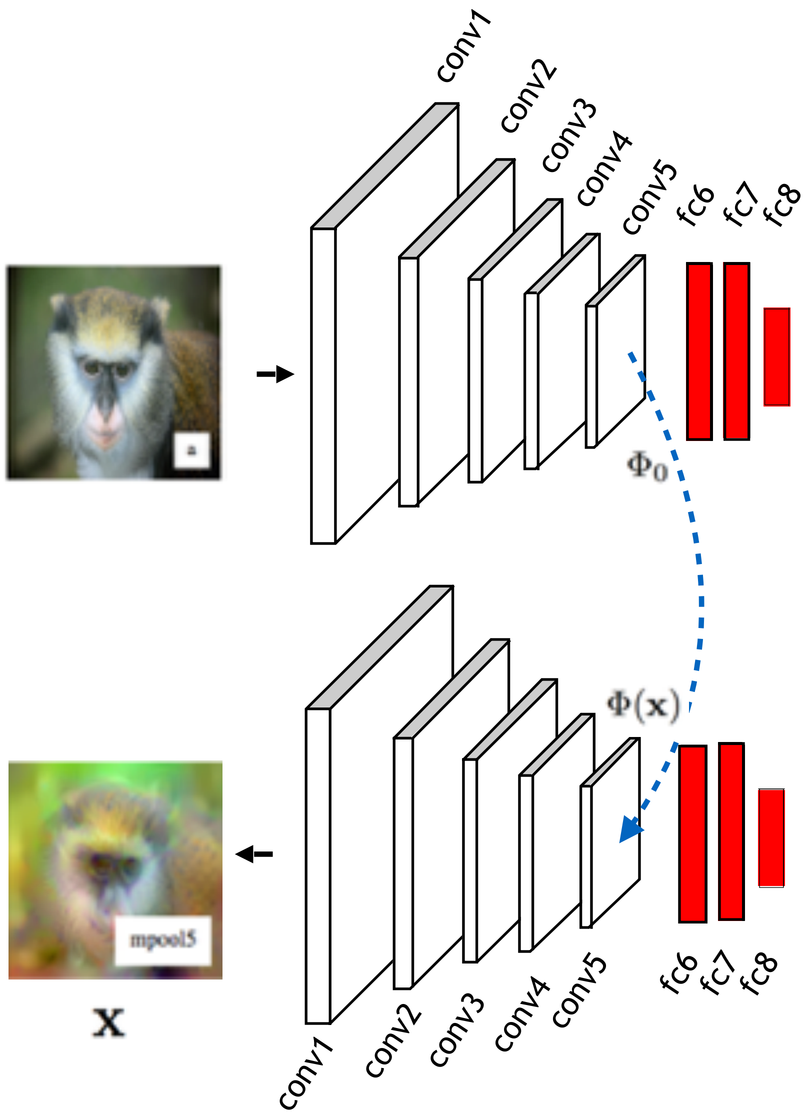



Optimized using gradient descent, initialized with the zero image. Gradients are computed using Back-propagation.




Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps, Karen Simonyan Andrea Vedaldi Andrew Zisserman, ICLR 2014

# Understanding Deep Image Representations by **Inverting** Them

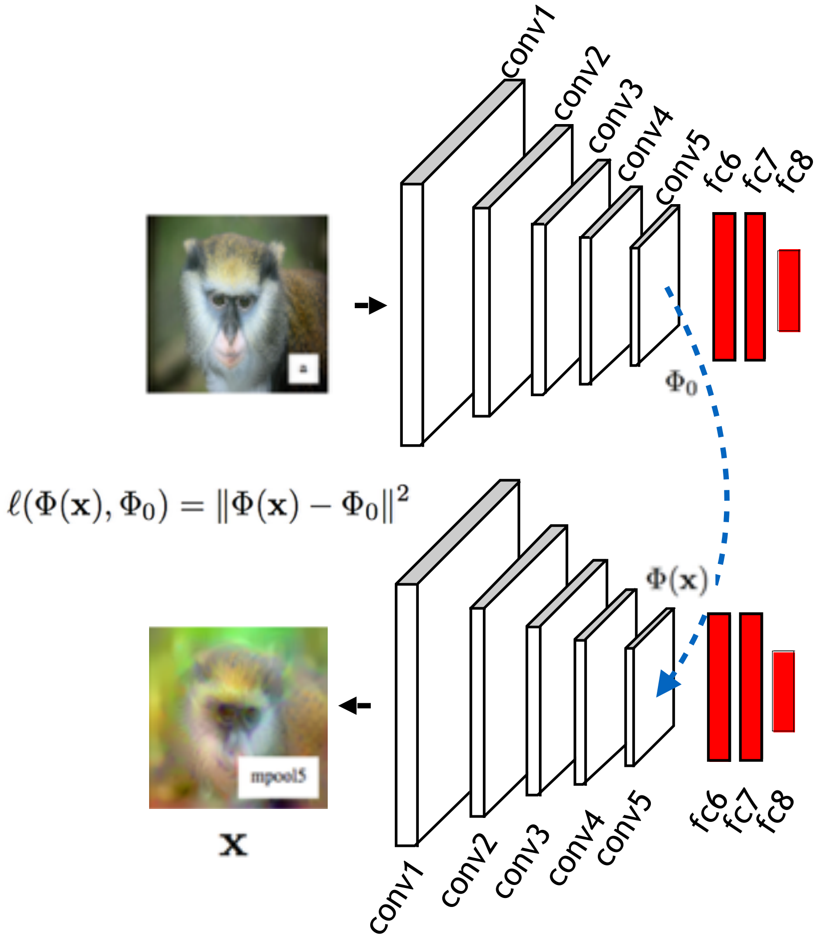



$$\ell(\Phi(\mathbf{x}), \Phi_0) = \|\Phi(\mathbf{x}) - \Phi_0\|^2$$

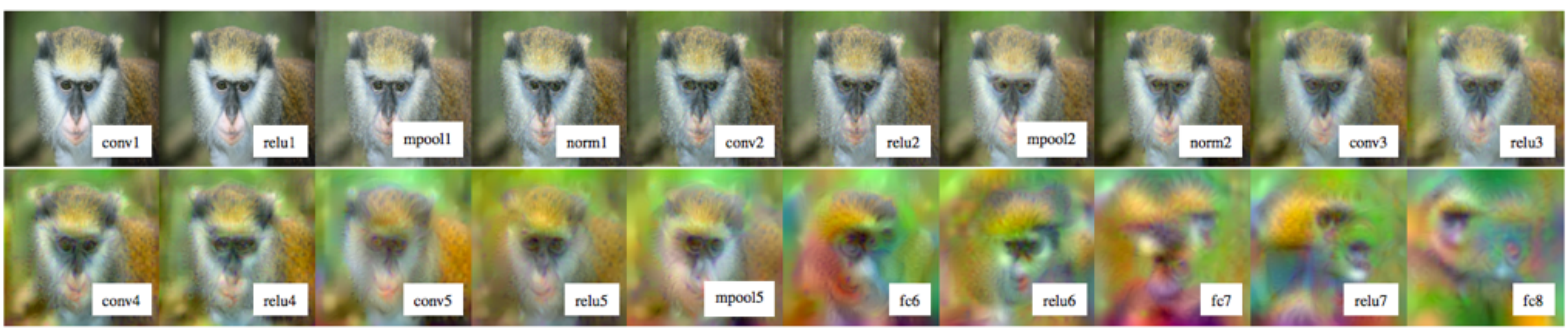
A common hypothesis is that representations collapse irrelevant differences in images (e.g. illumination or viewpoint), so that Φ should not be uniquely invertible.

Understanding Deep Image Representations by Inverting Them, Aravindh Mahendran, Andrea Vedaldi, CVPR 2015

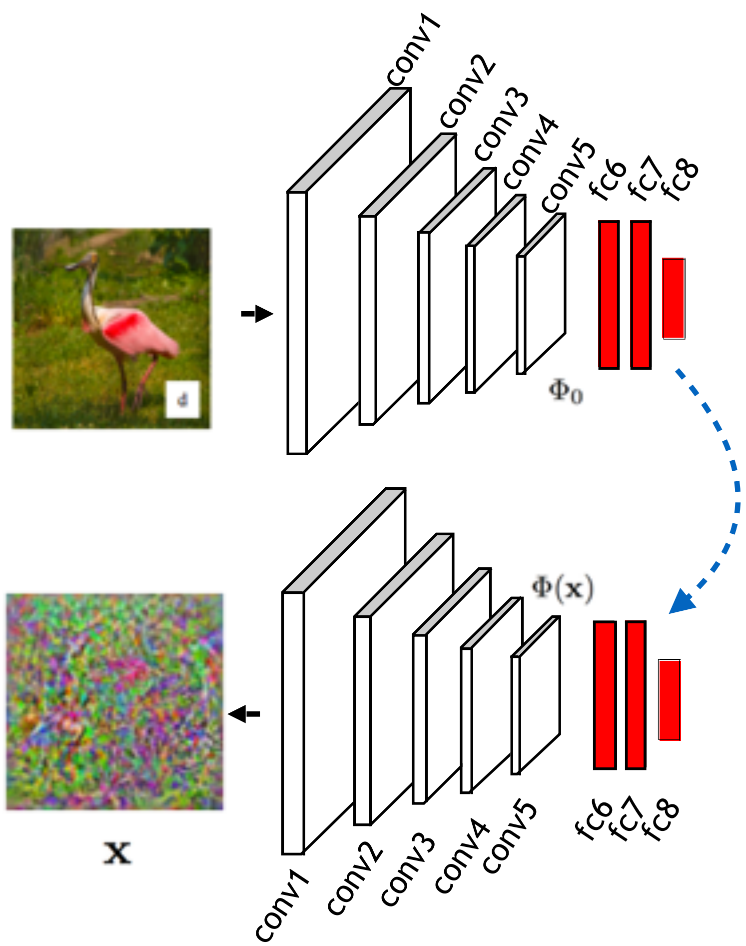
# Understanding Deep Image Representations by **Inverting** Them




Through reconstruction we obtain insights into the invariances captured by the representation.



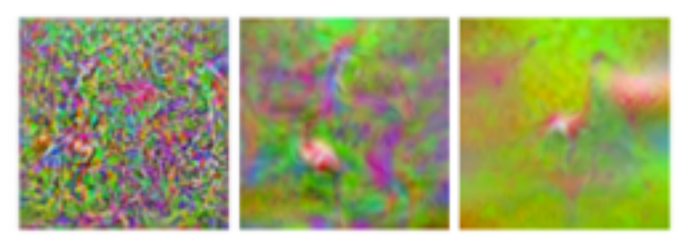
# Understanding Deep Image Representations by **Inverting** Them




$$\ell(\Phi(\mathbf{x}), \Phi_0) = \|\Phi(\mathbf{x}) - \Phi_0\|^2$$

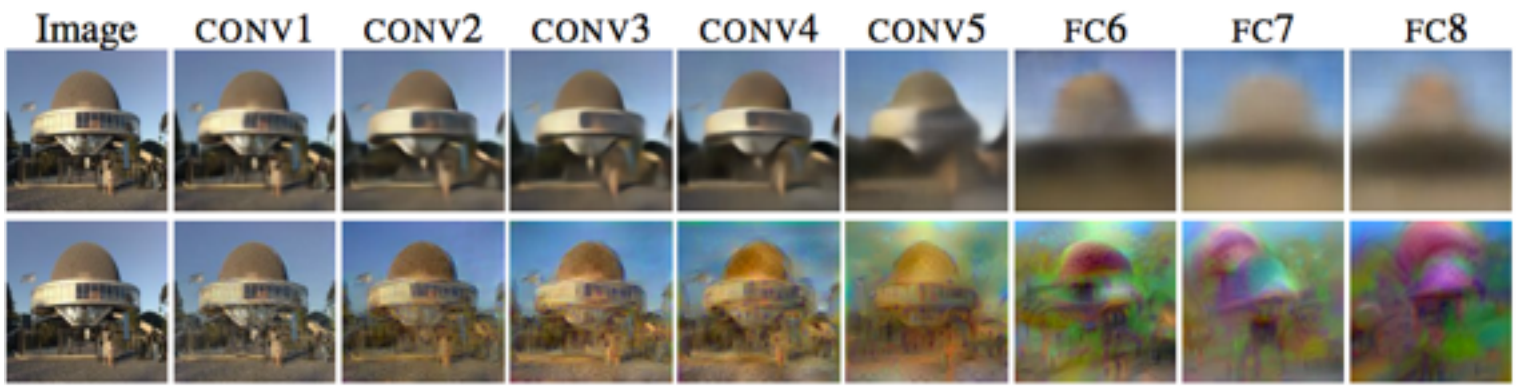
$$\mathcal{R}_\alpha(\mathbf{x}) = \|\mathbf{x}\|_\alpha^\alpha$$

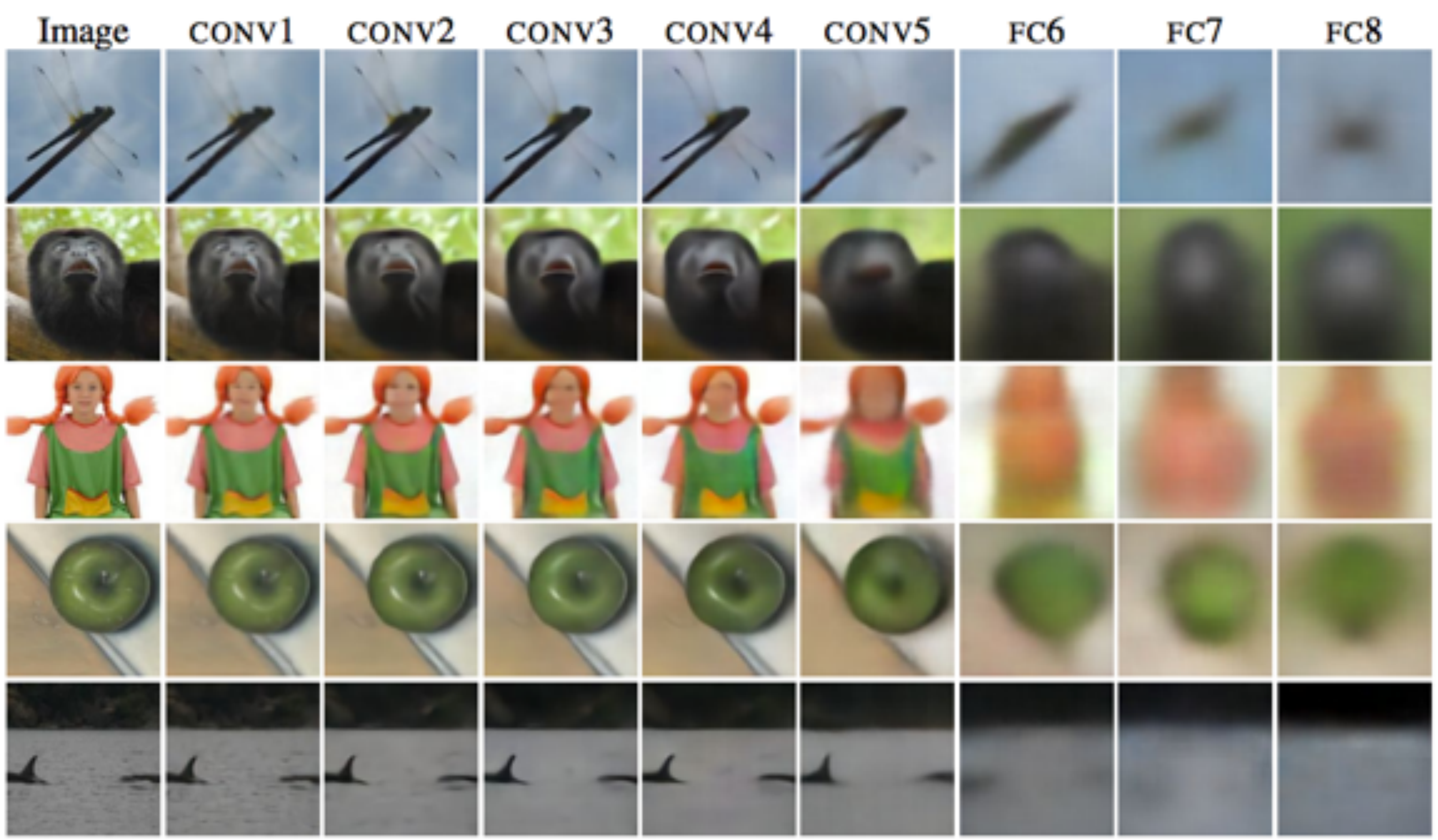
$$\mathcal{R}_{V^\beta}(\mathbf{x}) = \sum_{i,j} \left( (x_{i,j+1} - x_{ij})^2 + (x_{i+1,j} - x_{ij})^2 \right)^{\frac{\beta}{2}}$$



Regularization is important

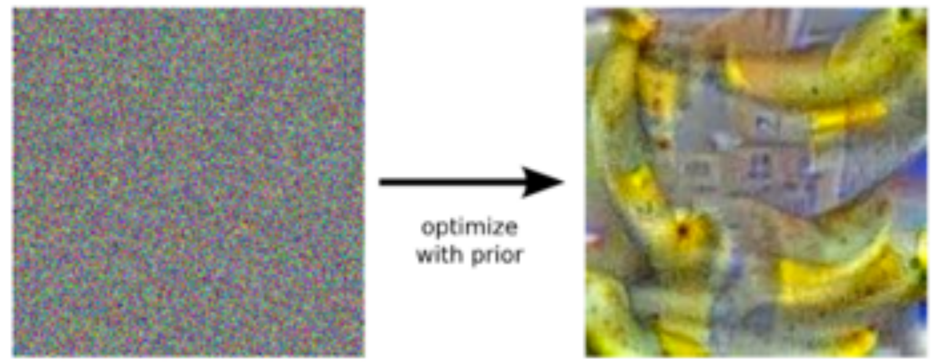
# Inverting Convolutional Networks with Convolutional Networks

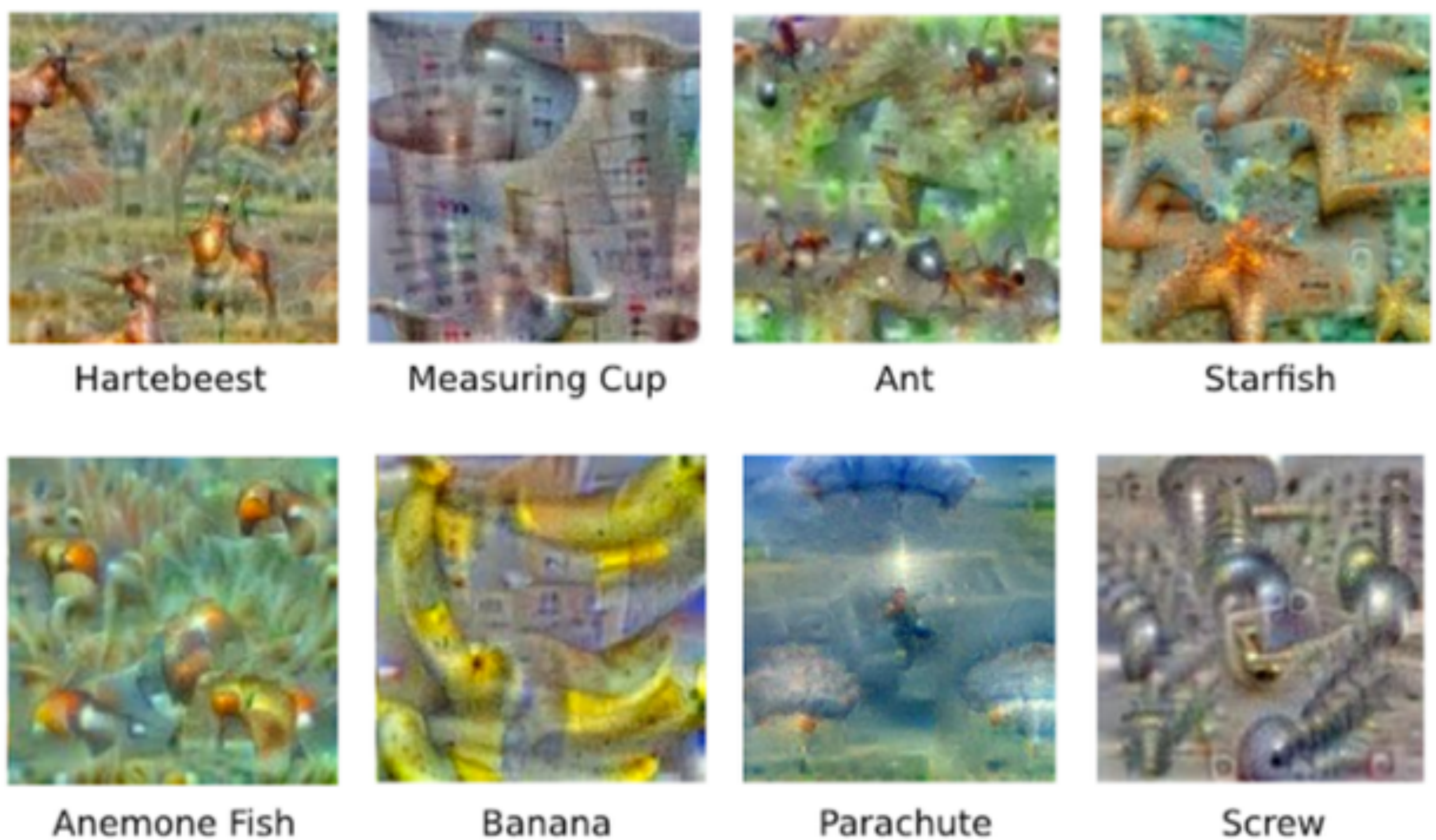



Inverting Convolutional Networks with Convolutional Networks, Alexey Dosovitskiy Thomas Brox, arXiv 2015

# Deep Dream



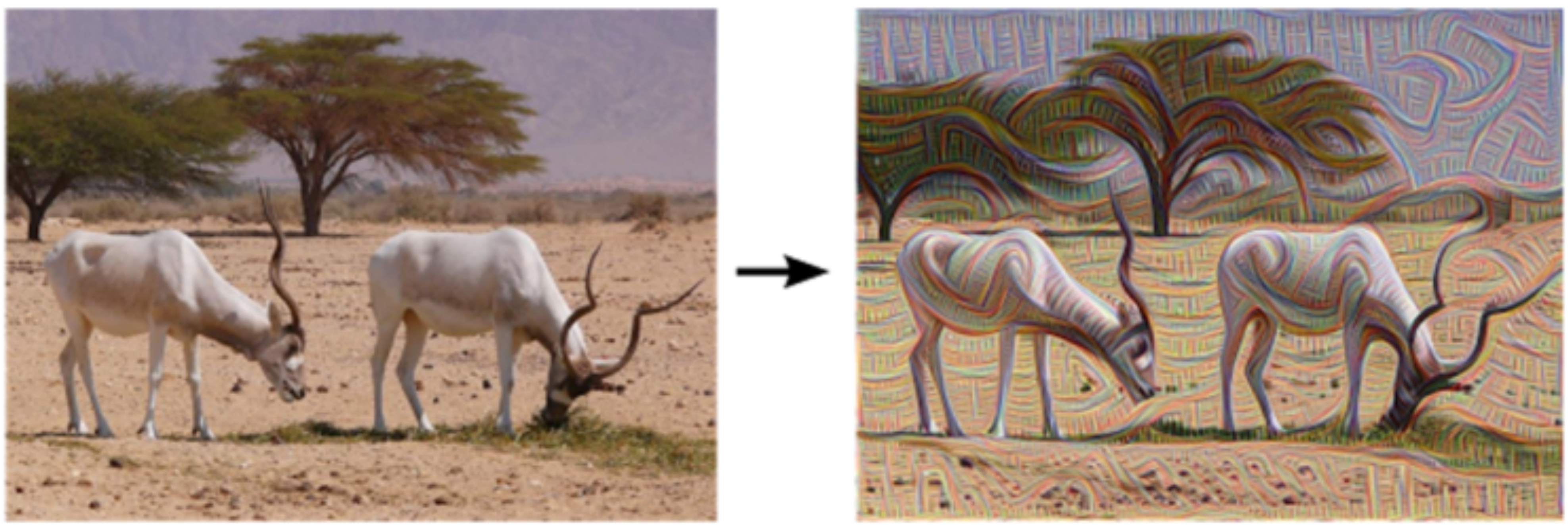




http://googleresearch.blogspot.com/2015/06/inceptionism-going-deeper-into-neural.html

# Deep Dream


http://googleresearch.blogspot.com/2015/06/inceptionism-going-deeper-into-neural.html

# Deep Dream



Simply feed the network an arbitrary image or photo and let the network analyze the picture. We then pick a layer and ask the network to enhance whatever it detected. Each layer of the network deals with features at a different level of abstraction, so the complexity of features we generate depends on which layer we choose to enhance. For example, lower layers tend to produce strokes or simple ornament-like patterns, because those layers are sensitive to basic features such as edges and their orientations.

http://googleresearch.blogspot.com/2015/06/inceptionism-going-deeper-into-neural.html

# Deep Dream

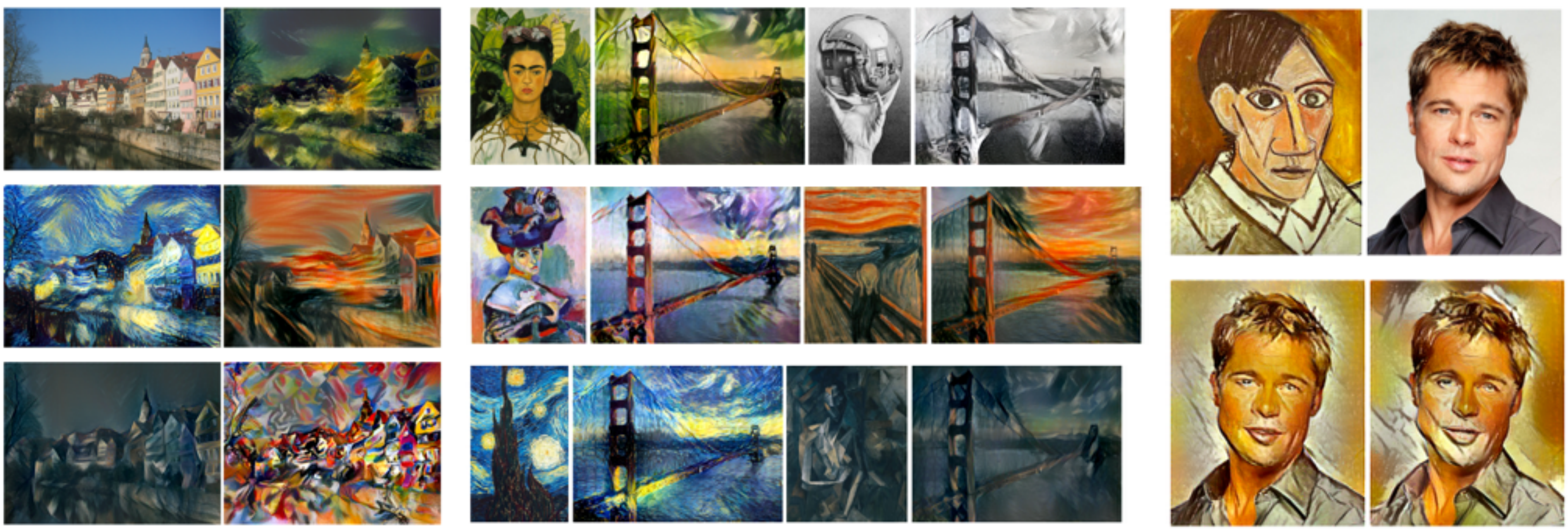
If we apply the algorithm iteratively on its own outputs and apply some zooming after each iteration, we get an endless stream of new impressions, exploring the set of things the network knows about.

# A Neural Algorithm of Artistic Style

combining the content of one image with the style of another image using convolutional neural networks.

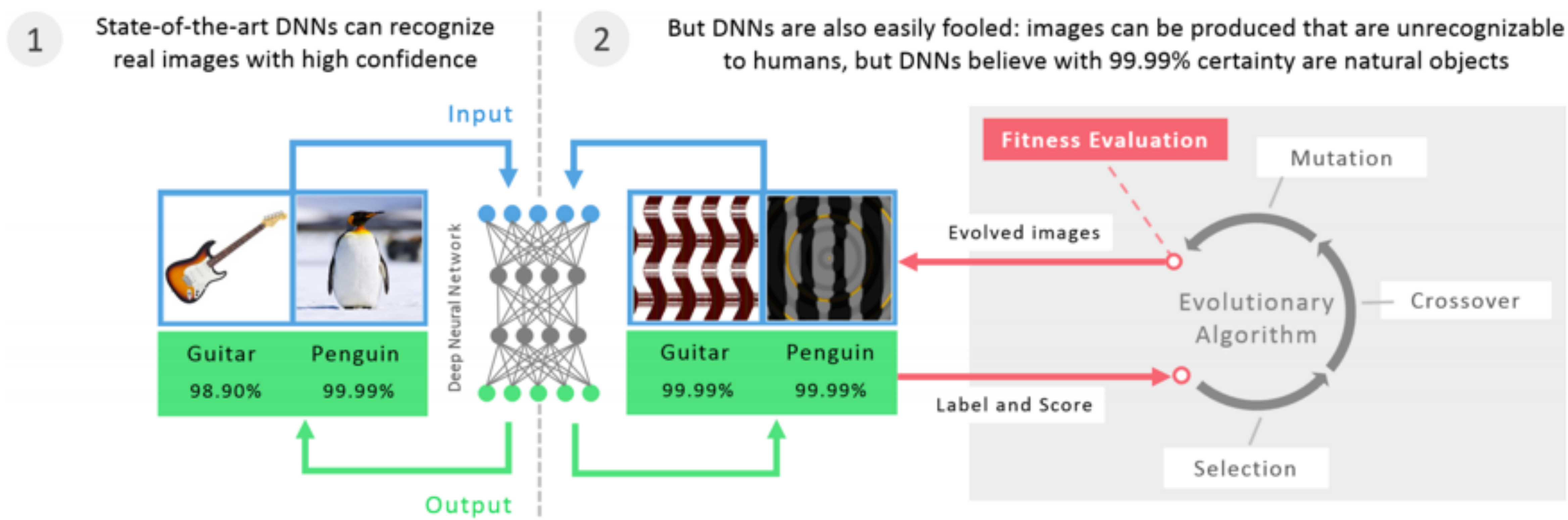
A Neural Algorithm of Artistic Style, Leon A. Gatys, Alexander S. Ecker, Matthias Bethge, arXiv 2015

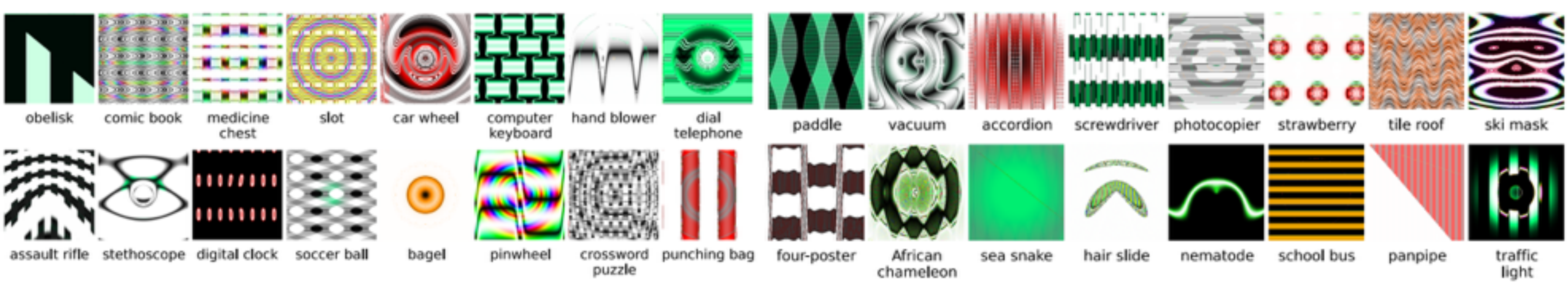
# A Neural Algorithm of Artistic Style



A Neural Algorithm of Artistic Style, Leon A. Gatys, Alexander S. Ecker, Matthias Bethge, arXiv 2015

# Deep Neural Networks are Easily Fooled
# High Confidence Predictions for Unrecognizable Images

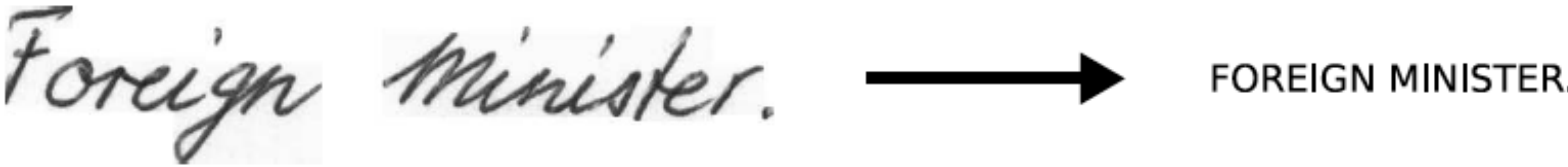


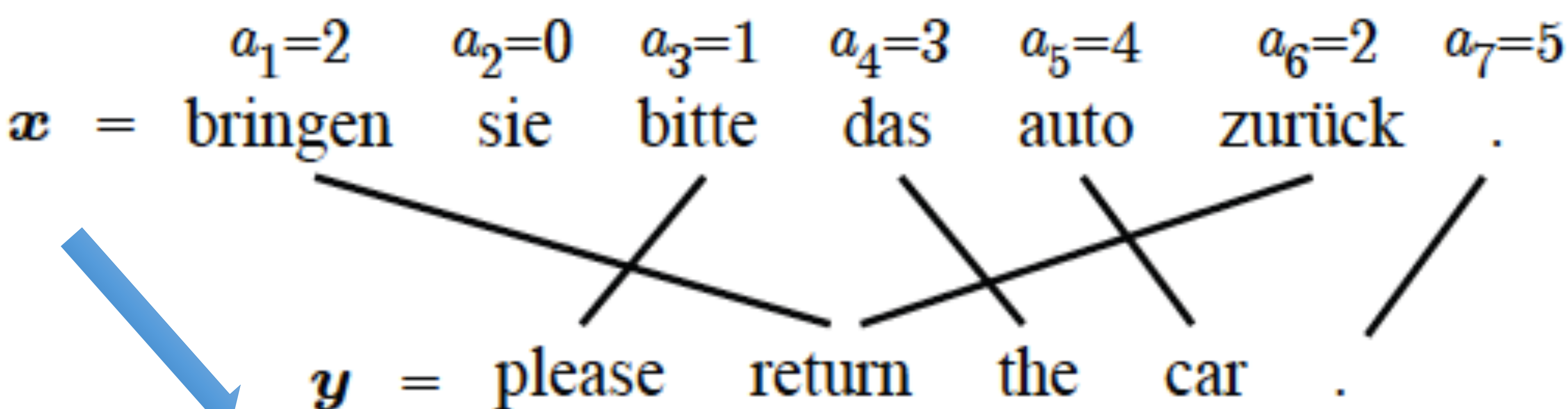

Deep Neural Networks are Easily Fooled: High Confidence Predictions for Unrecognizable Images, Anh Nguyen, Jason Yosinski, Jeff Clune, CVPR 2015

# Learning with Sequences

# Sequences are everywhere …

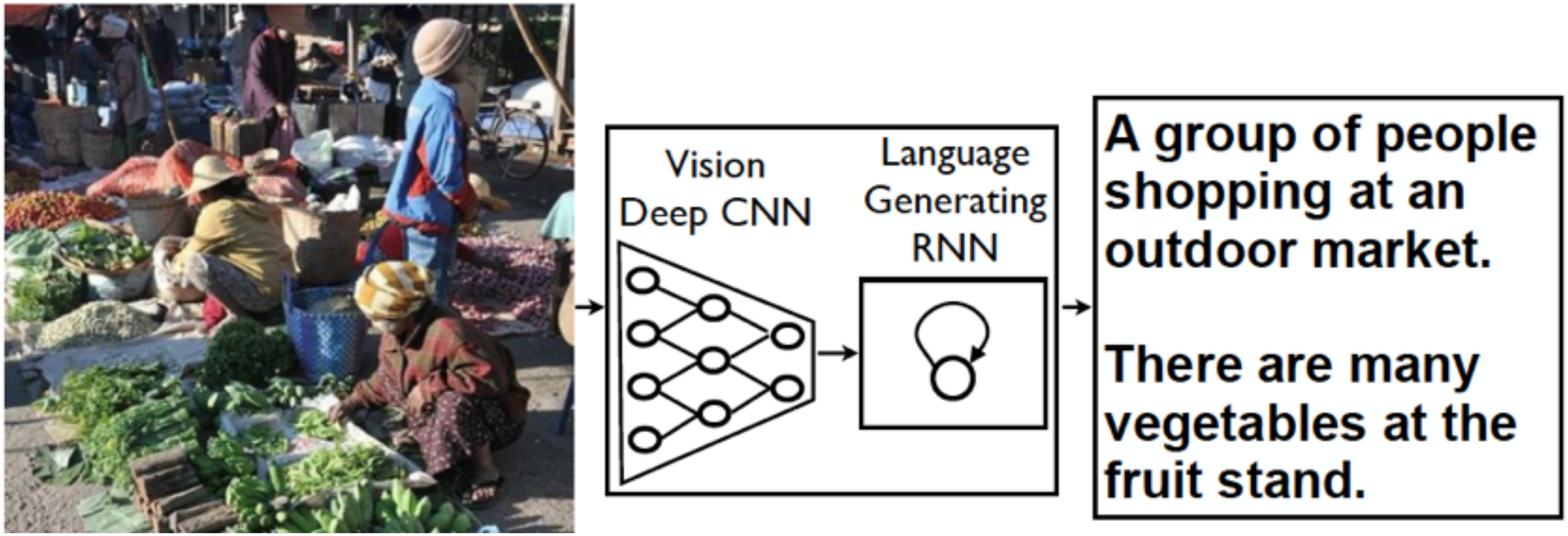




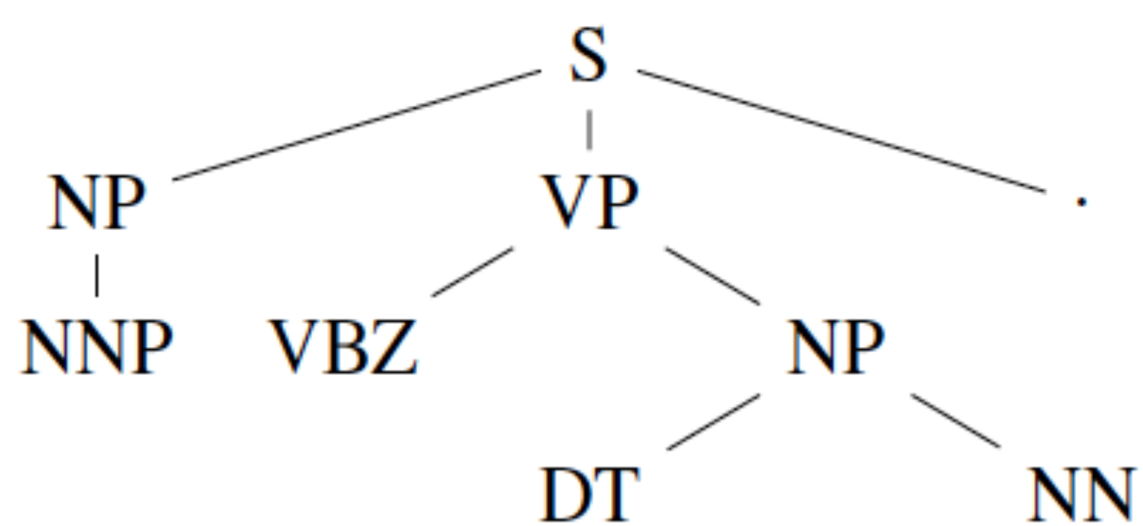



Credit: Alex Graves, Kevin Gimpel, Dhruv Batra

# Even where you might not expect a sequence…



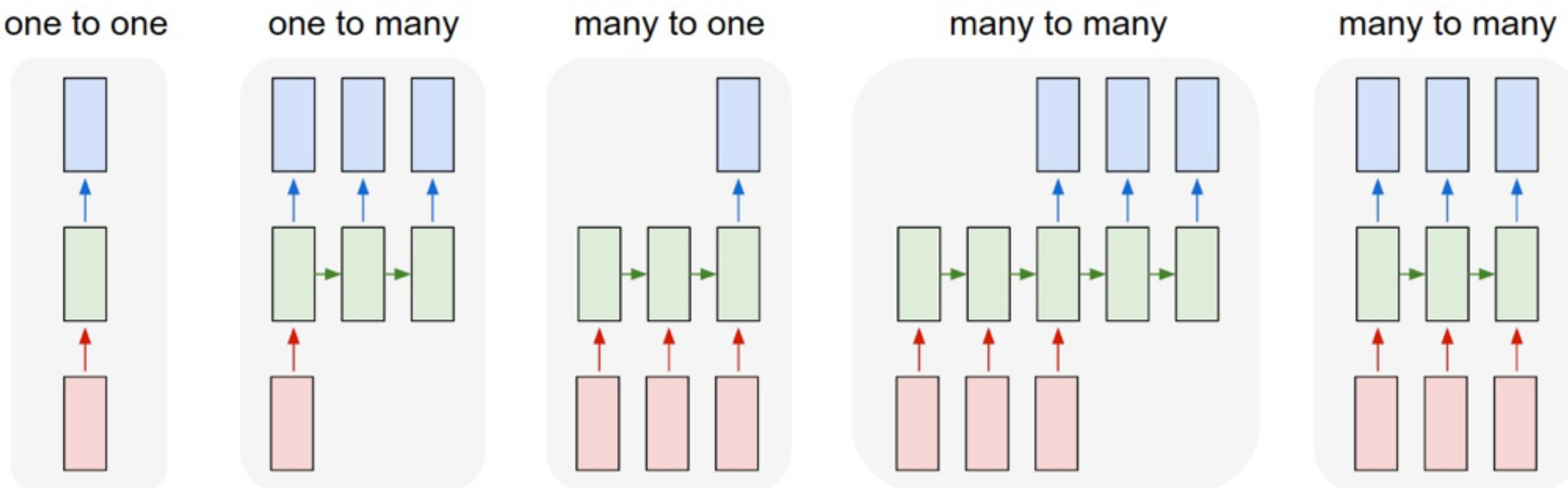

John has a dog . $\rightarrow$




John has a dog . $\rightarrow$ (S (NP NNP $)_{NP}$ (VP VBZ (NP DT NN $)_{NP}$ $)_{VP}$ . $)_{S}$

Credit: Dhruv Batra, Vinyals et al.

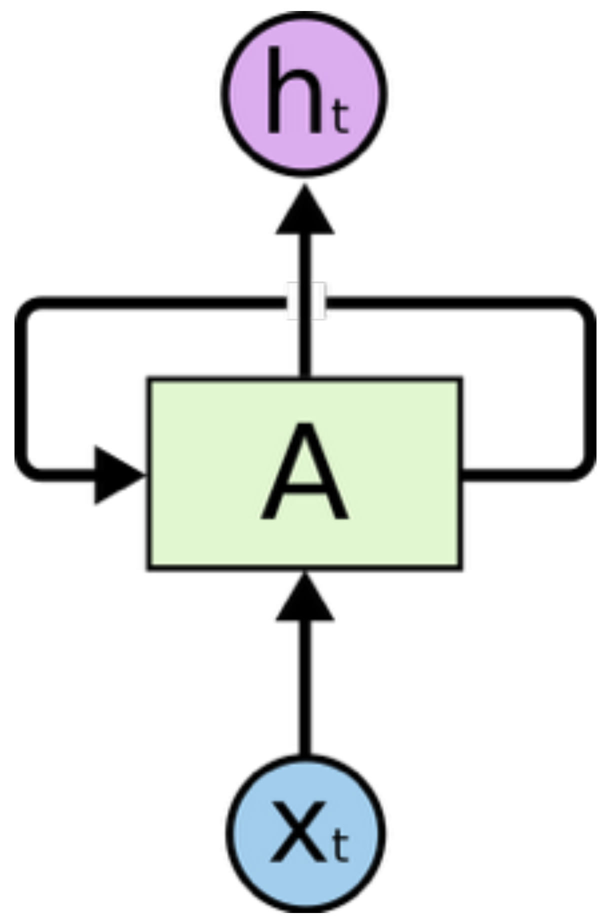
# How do we model sequences?




| | | | |
|---|---|---|---|
| Input: No sequence | Input: No sequence | Input: Sequence | Input: Sequence |
| Output: No sequence | Output: Sequence | Output: No sequence | Output: Sequence |
| Example: "standard" classification / regression problems | Example: Im2Caption | Example: sentence classification, multiple-choice question answering | Example: machine translation, video captioning, open-ended question answering, video question answering |

http://karpathy.github.io/2015/05/21/rnn-effectiveness/ Credit: Dhruv Batra, Andrej Karpathy

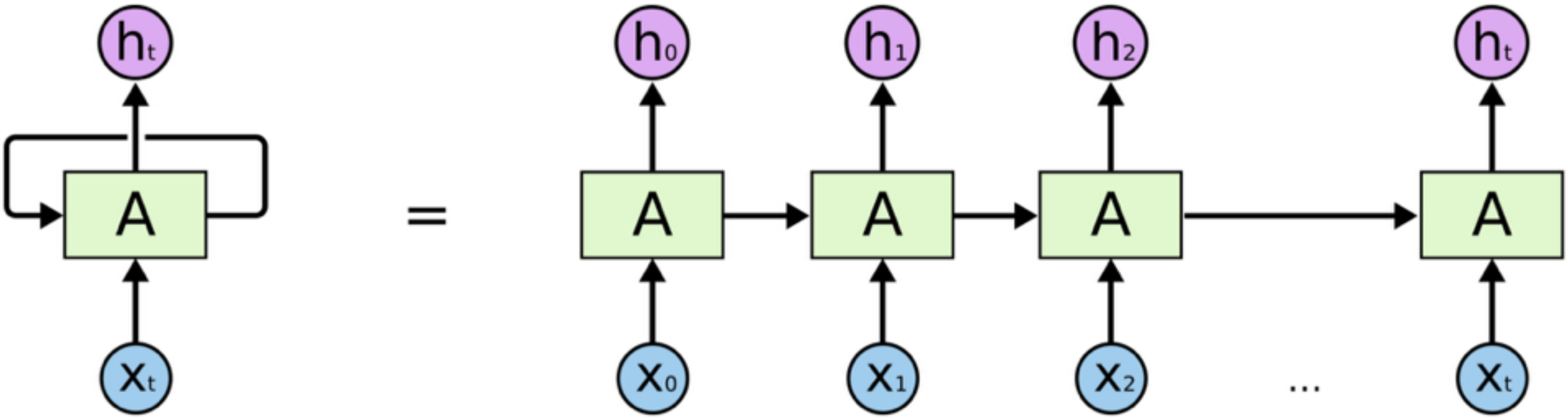
# Recurrent Neural Networks (RNNs)




In the above diagram, a chunk of neural network, $A$, looks at some input $x_i$ and outputs a value $h_i$. A loop allows information to be passed from one step of the network to the next.

Credit: Christopher Olah

# Recurrent Neural Networks (RNNs)

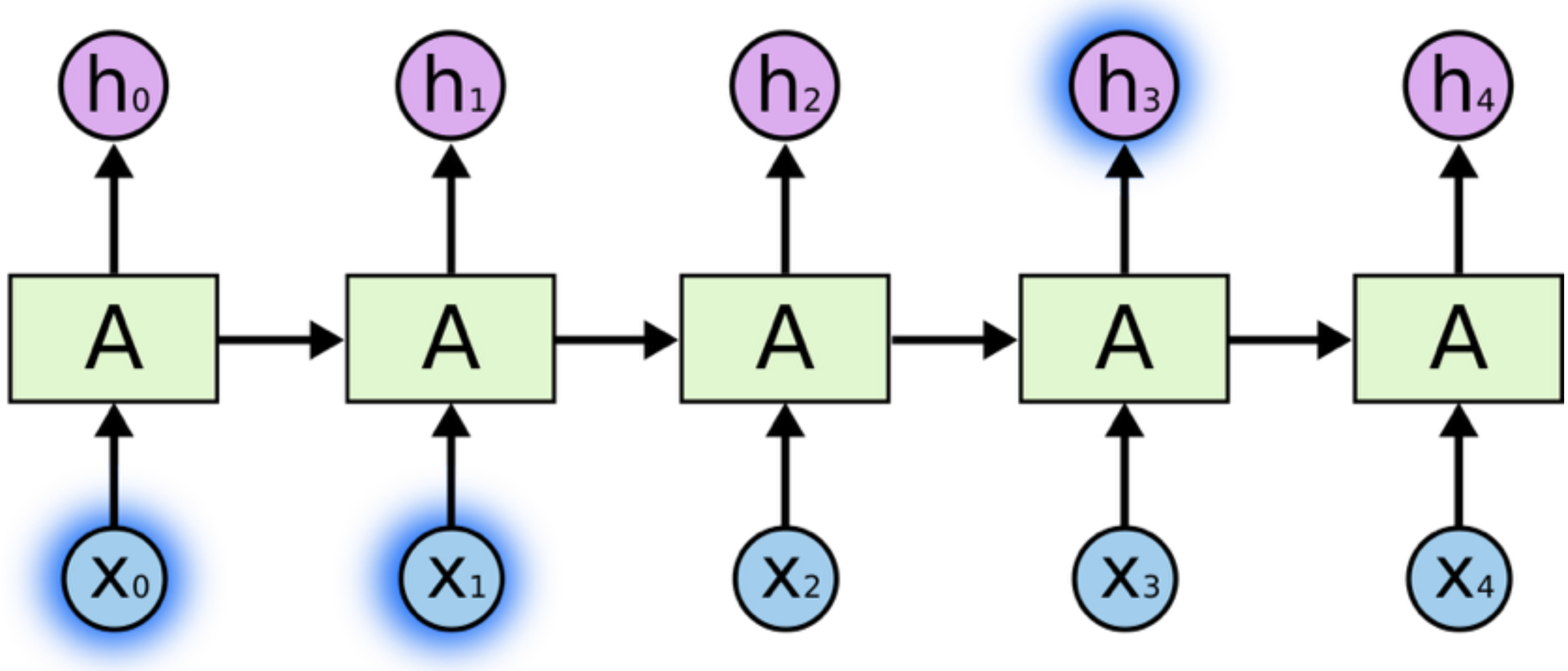



**An unrolled recurrent neural network.**

A recurrent neural network can be thought of as multiple copies of the same network, each passing a message to a successor

Credit: Christopher Olah

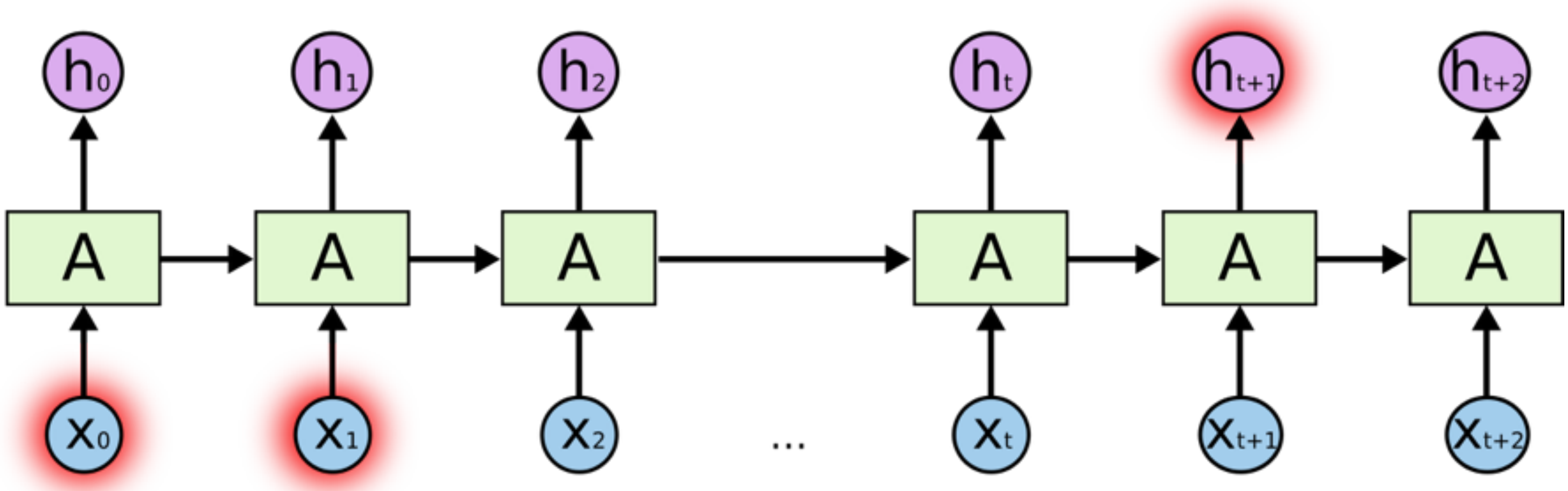
# Recurrent Neural Networks (RNNs)




When the gap between the relevant information and the place that it's needed is small, RNNs can learn to use the past information

Credit: Christopher Olah

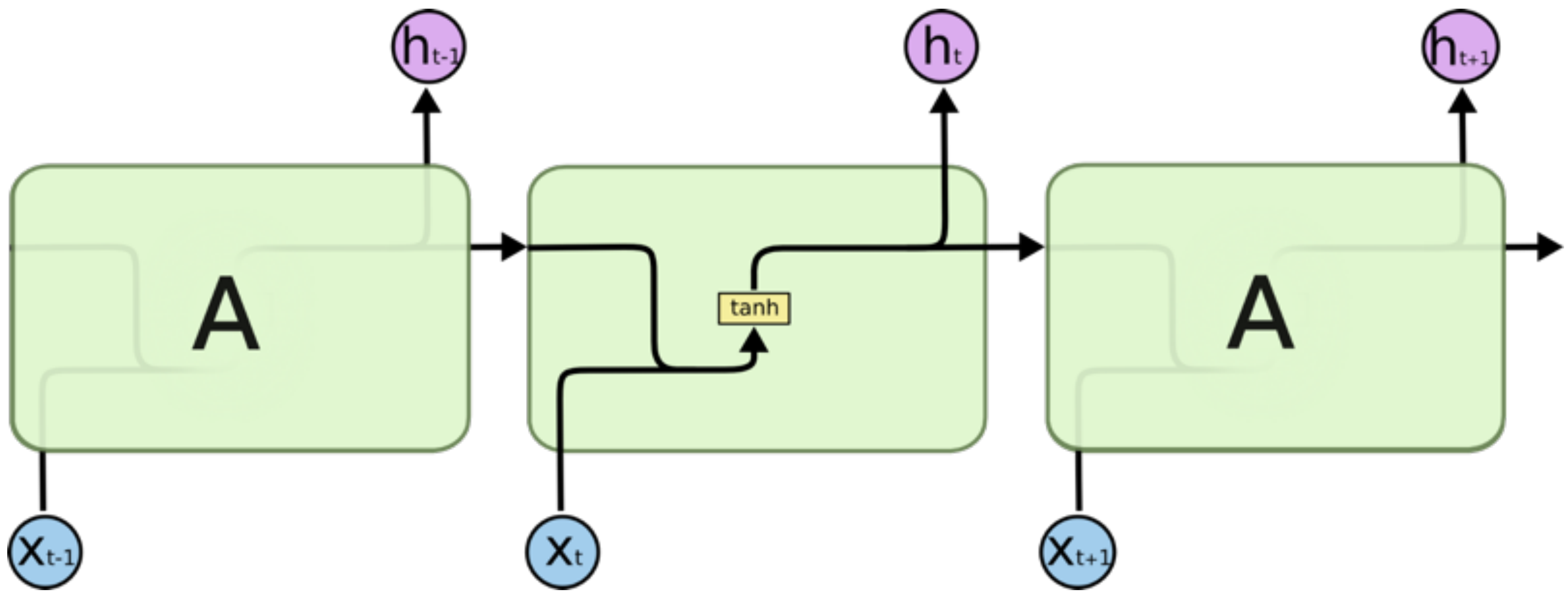
# Long-term dependencies - hard to model!




But there are also cases where we need more context.

Credit: Christopher Olah

# From plain RNNs to LSTMs

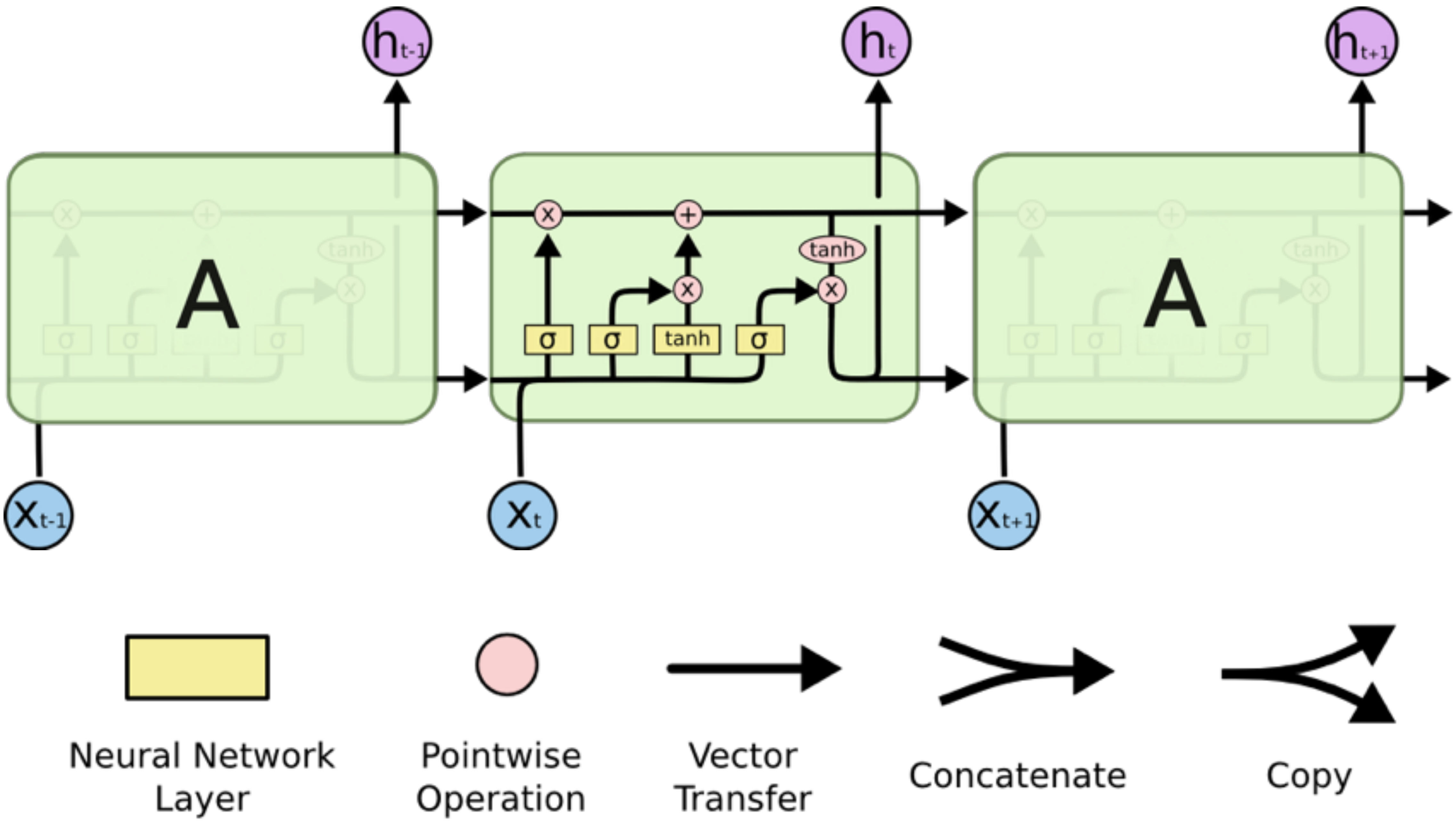



(LSTM: Long Short Term Memory Networks)

http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Credit: Christopher Olah

# From plain RNNs to LSTMs

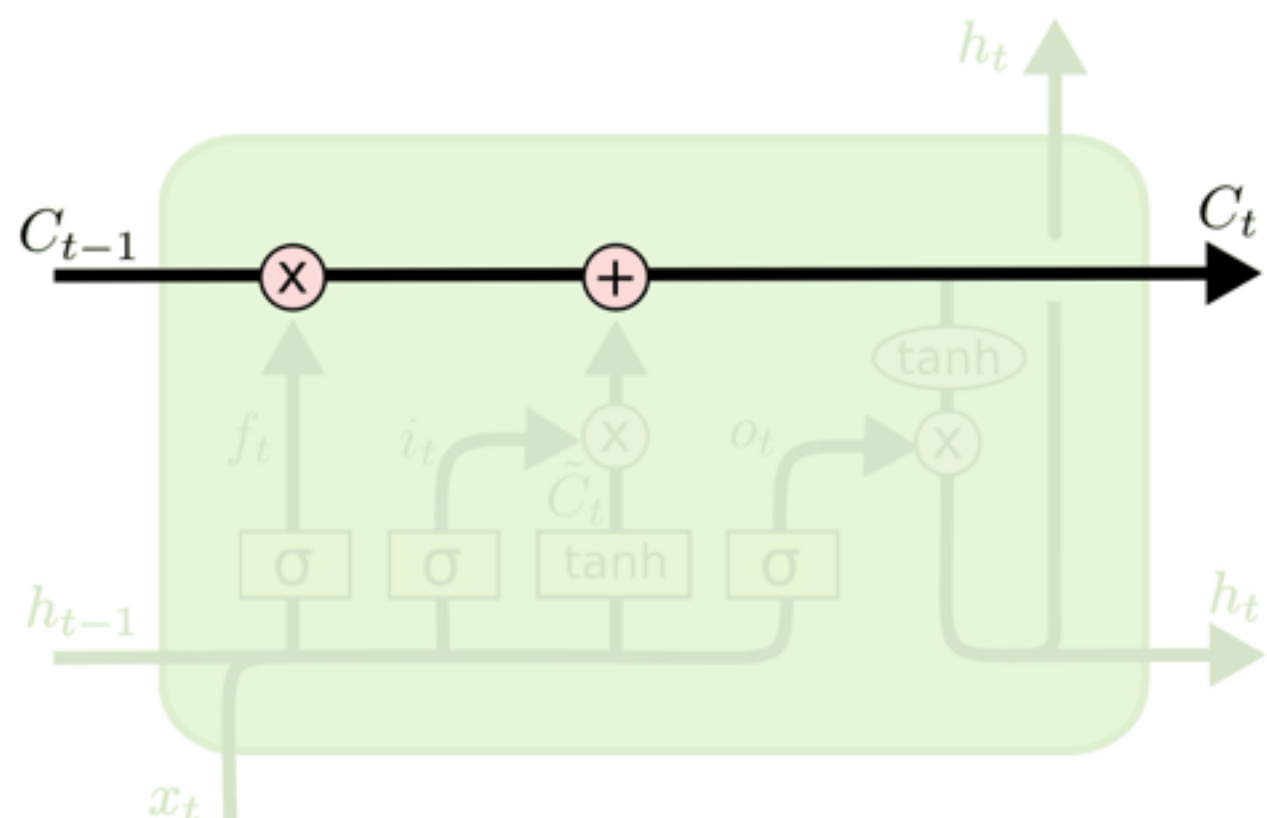



(LSTM: Long Short Term Memory Networks)

Credit: Christopher Olah

# LSTMs Step by Step: **Memory**

## Cell State / Memory

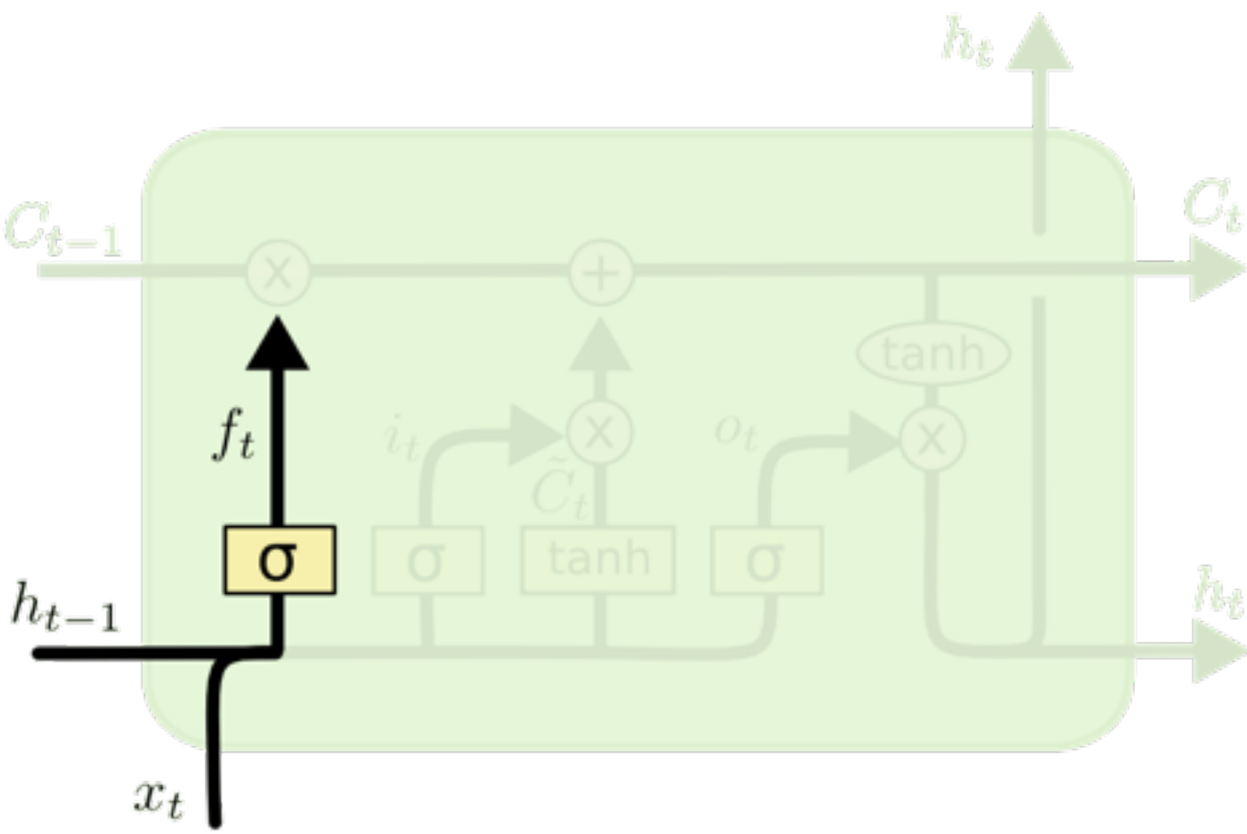



The LSTM does have the ability to remove or add information to the cell state, carefully regulated by structures called gates

Credit: Christopher Olah

# LSTMs Step by Step: **Forget Gate**

Should we continue to remember this "bit" of information or not?



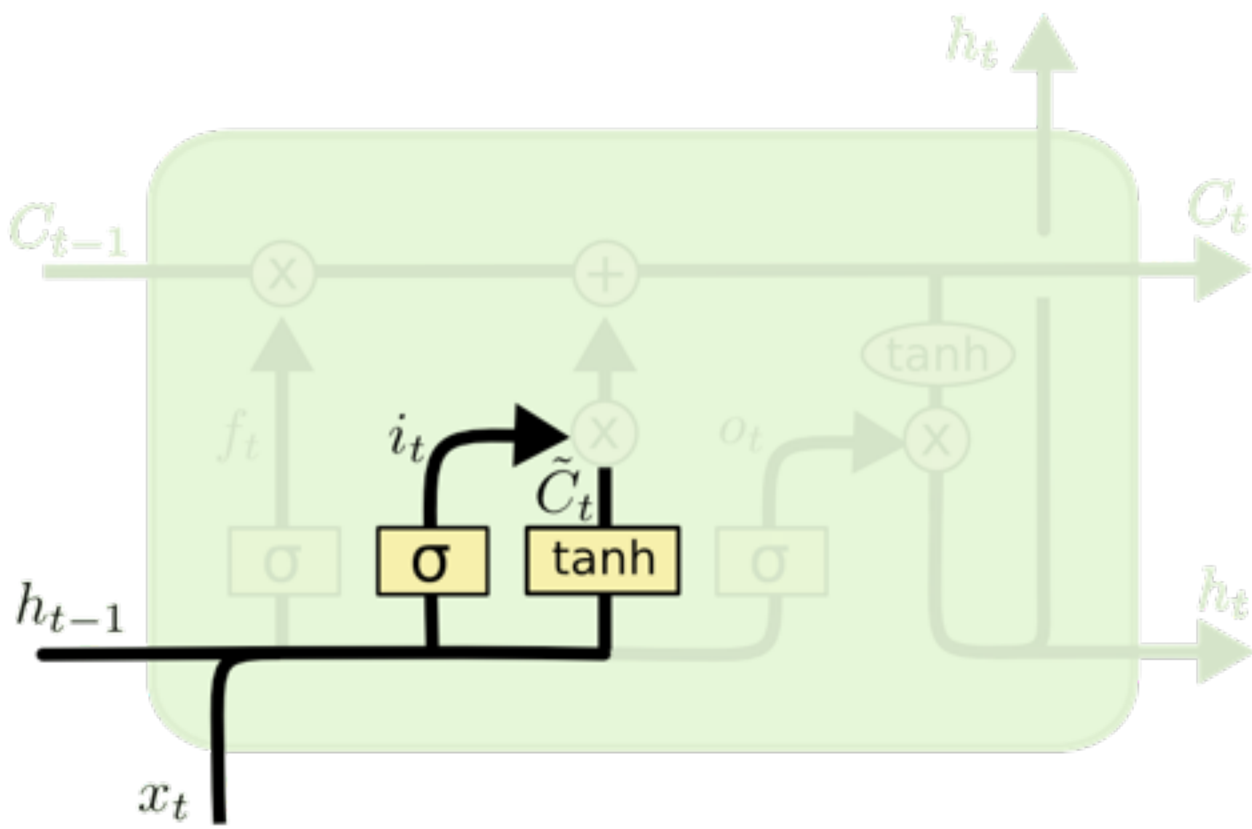

$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \ + \ b_f\right)$$

The first step in our LSTM is to decide what information we're going to throw away from the cell state. This decision is made by a sigmoid layer called the "forget gate layer."

Credit: Christopher Olah

# LSTMs Step by Step: **Input Gate**

Should we update this “bit” of information or not? If so, with what?



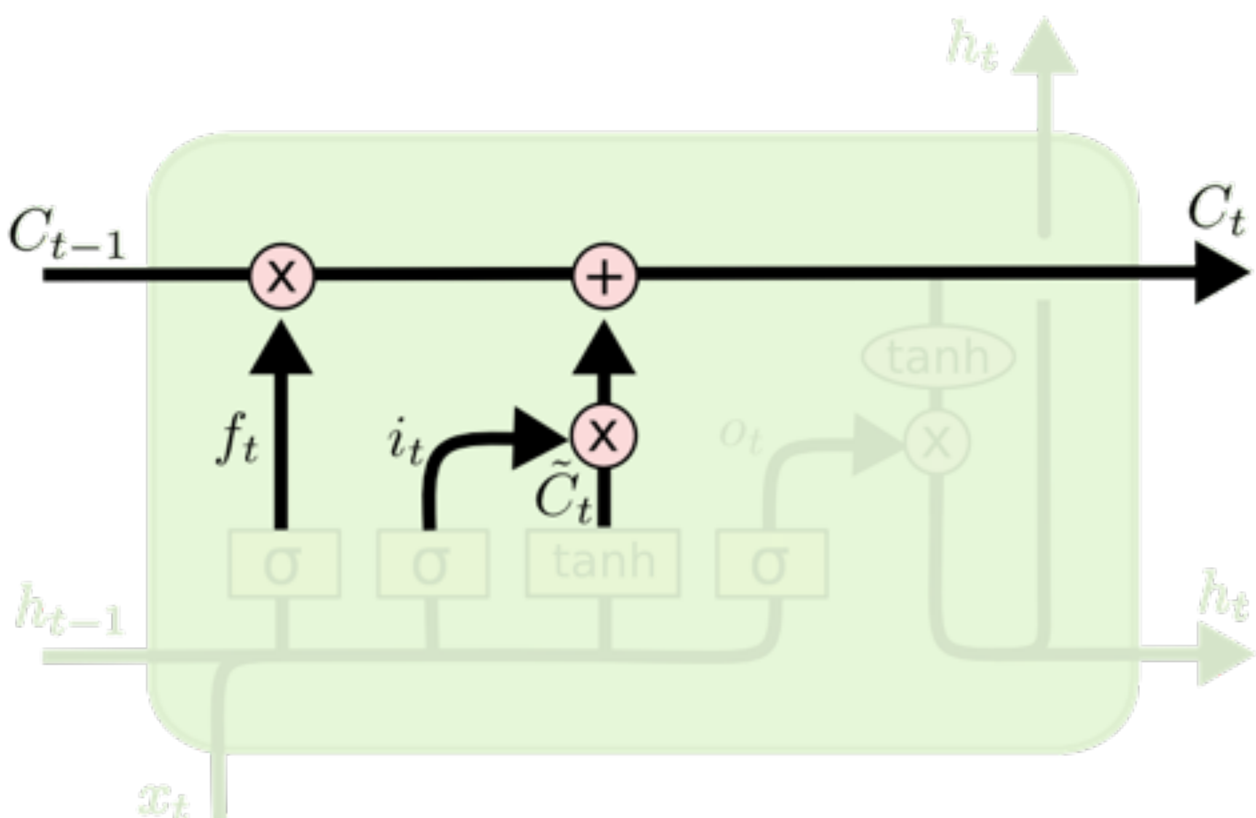

$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \ + \ b_i\right)$$
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \ + \ b_C)$$

The next step is to decide what new information we’re going to store in the cell state. This has two parts. First, a sigmoid layer called the “input gate layer” decides which values we’ll update. Next, a tanh layer creates a vector of new candidate values, $\tilde{C}_t$, that could be added to the state.

Credit: Christopher Olah

# LSTMs Step by Step: **Memory Update**

Decide what will be kept in the cell state/memory



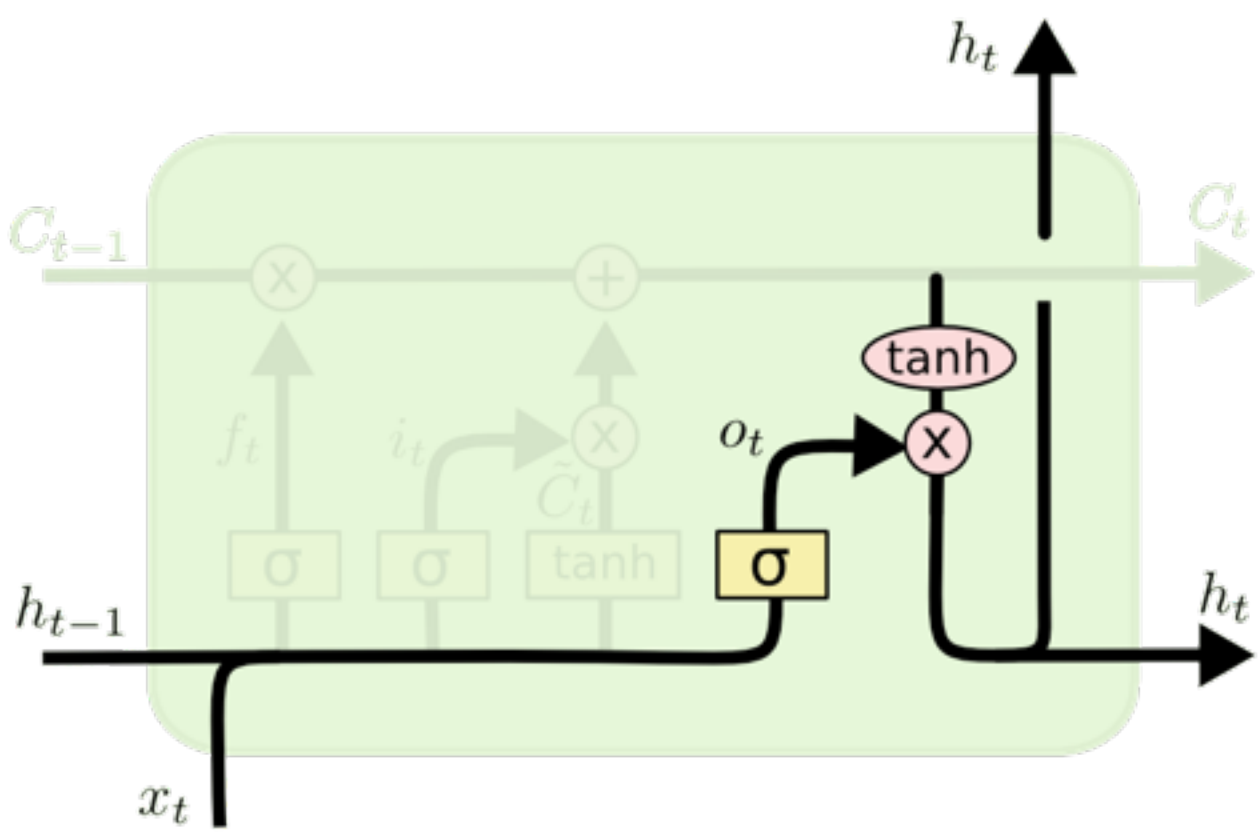

Forget that Memorize this

$$C_t = \underbrace{f_t * C_{t-1}}_{\text{Forget that}} + \underbrace{i_t * \tilde{C}_t}_{\text{Memorize this}}$$

Credit: Christopher Olah

# LSTMs Step by Step: **Output Gate**

Should we output this "bit" of information?

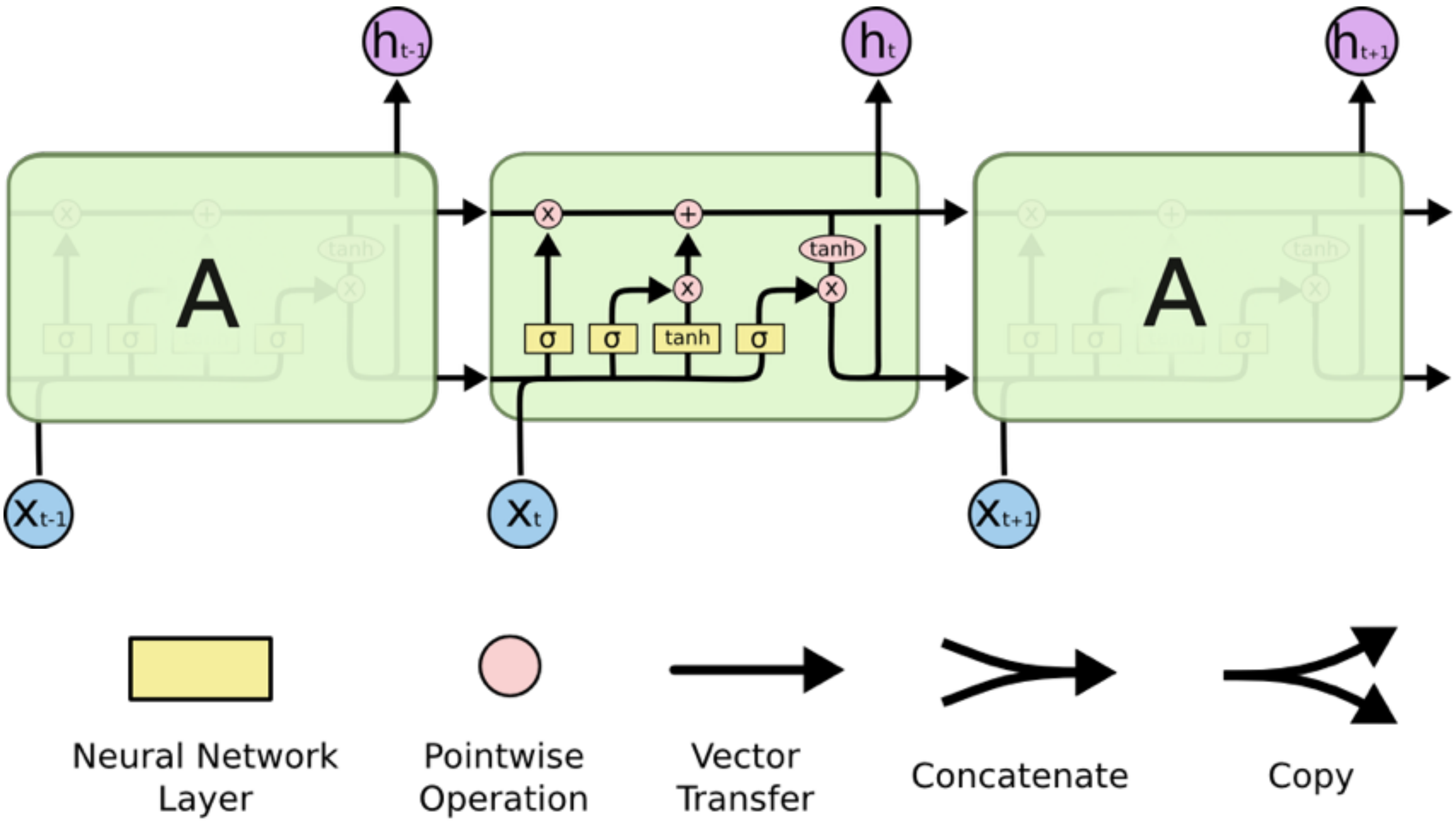



$$o_t = \sigma\left(W_o\ [h_{t-1}, x_t]\ +\ b_o\right)$$
$$h_t = o_t * \tanh\left(C_t\right)$$

This output will be based on our cell state, but will be a filtered version. First, we run a sigmoid layer which decides what parts of the cell state we're going to output. Then, we put the cell state through tanh (to push the values to be between −1 and 1) and multiply it by the output of the sigmoid gate, so that we only output the parts we decided to.

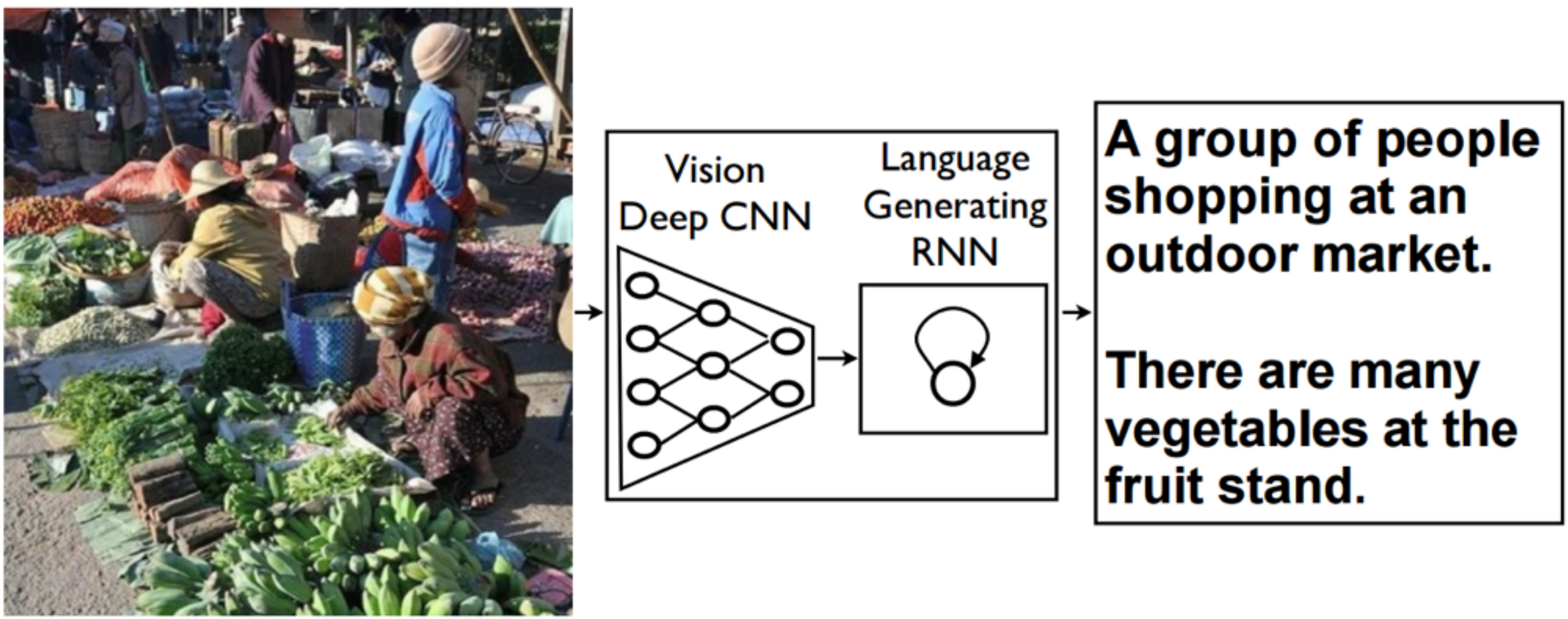
Credit: Christopher Olah

# Complete LSTM - A pretty sophisticated cell



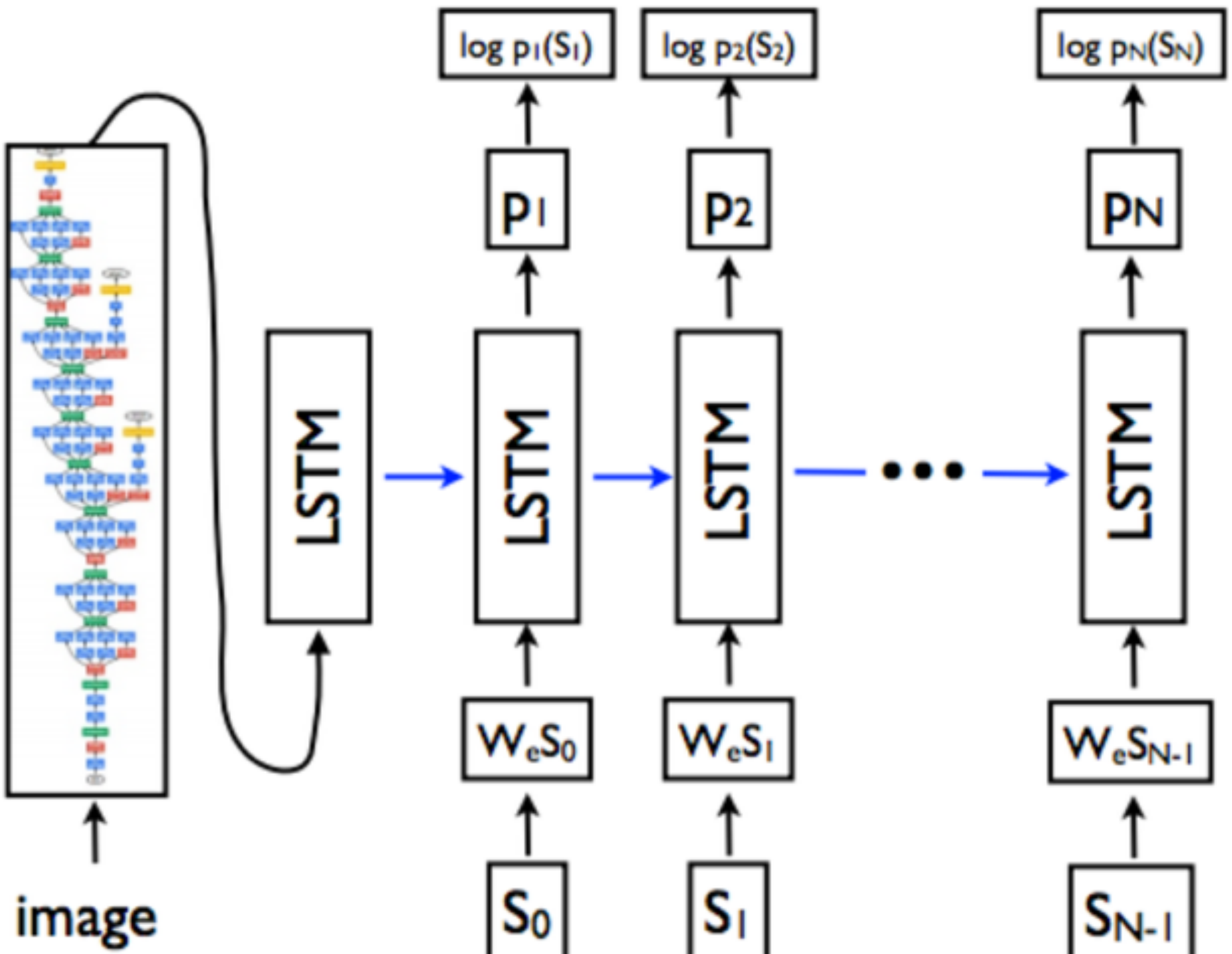

Credit: Christopher Olah

# Show and Tell: A Neural Image Caption Generator




Show and Tell: A Neural Image Caption Generator, Vinyals et. al., CVPR 2015

# Show and Tell: A Neural Image Caption Generator





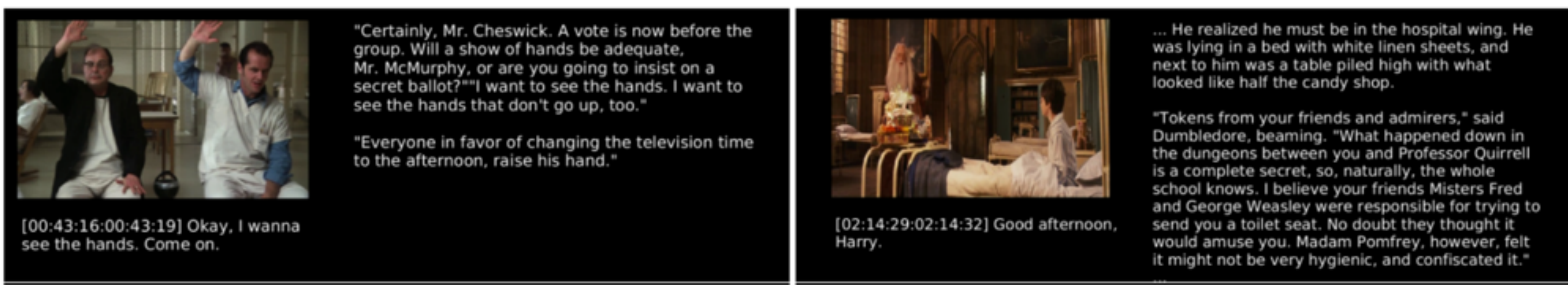
Show and Tell: A Neural Image Caption Generator, Vinyals et. al., CVPR 2015

# Image Caption Generator Results


https://pdollar.wordpress.com/2015/01/21/image-captioning/

Show and Tell: A Neural Image Caption Generator, Vinyals et. al., CVPR 2015

# Aligning Books and Movies

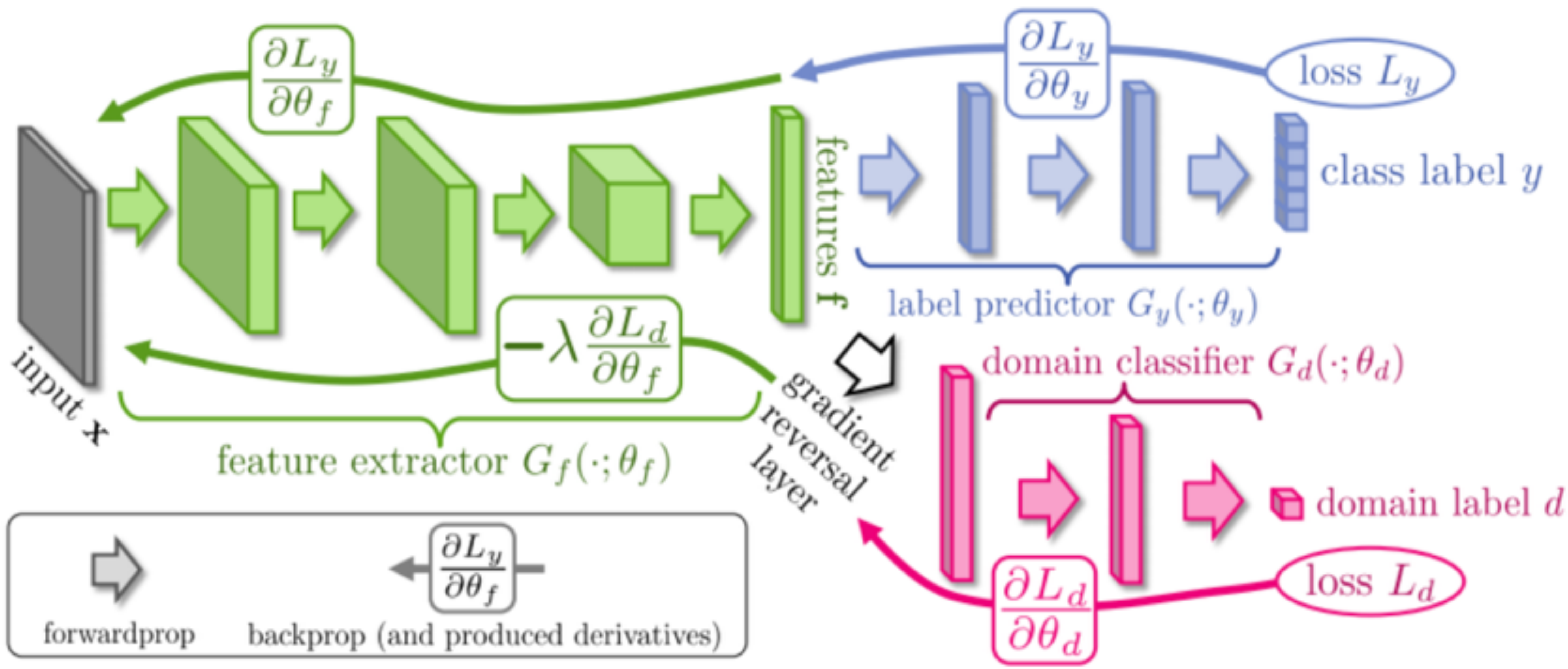




Describing movie clips via the book: a shot from the movie and its corresponding paragraph (plus one before and after) from the book.

Aligning Books and Movies: Towards Story-like Visual Explanations by Watching Movies and Reading Books, Zhu et.al., ICCV 2015

# Domain Adaptation & Multi-task Learning in Deep Convolutional Networks

# Unsupervised Domain Adaptation by Backpropagation

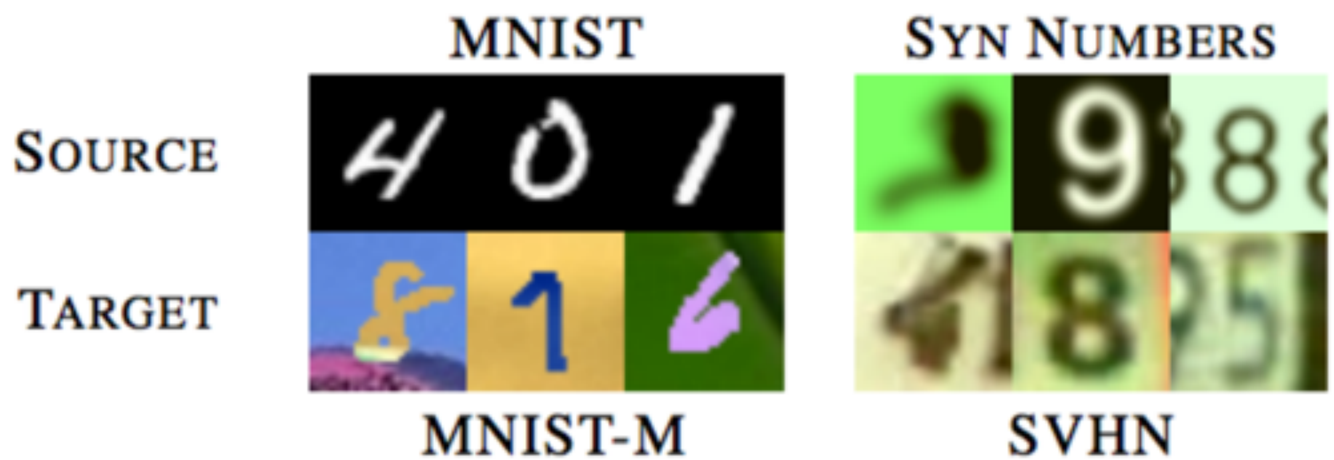



The approach promotes the emergence of "deep" features that are …

(i) discriminative for the main learning task on the source domain,

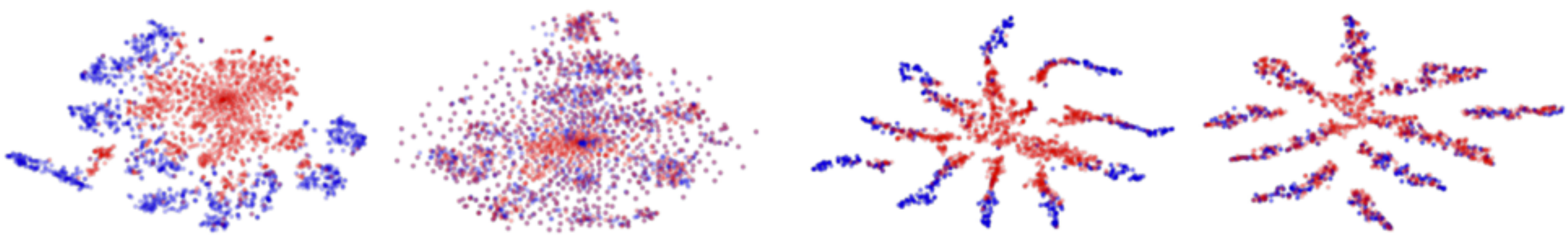
(ii) invariant with respect to the shift between the domains

# Unsupervised Domain Adaptation by Backpropagation




MNIST → MNIST-M: top feature extractor layer

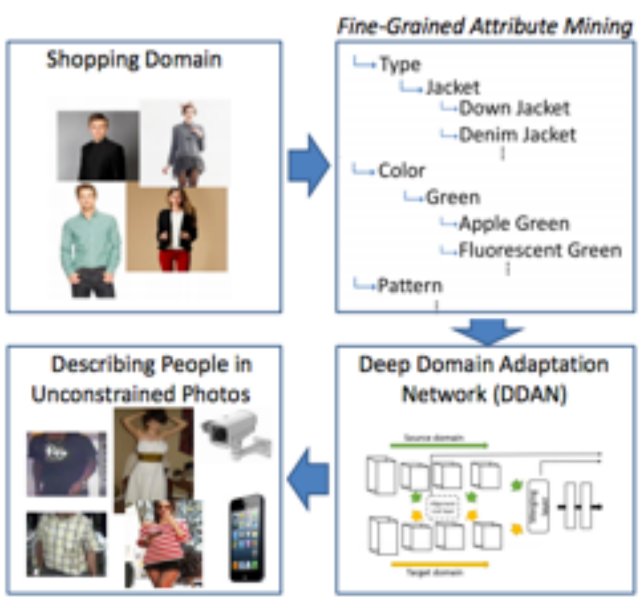
SYN NUMBERS → SVHN: last hidden layer of the label predictor



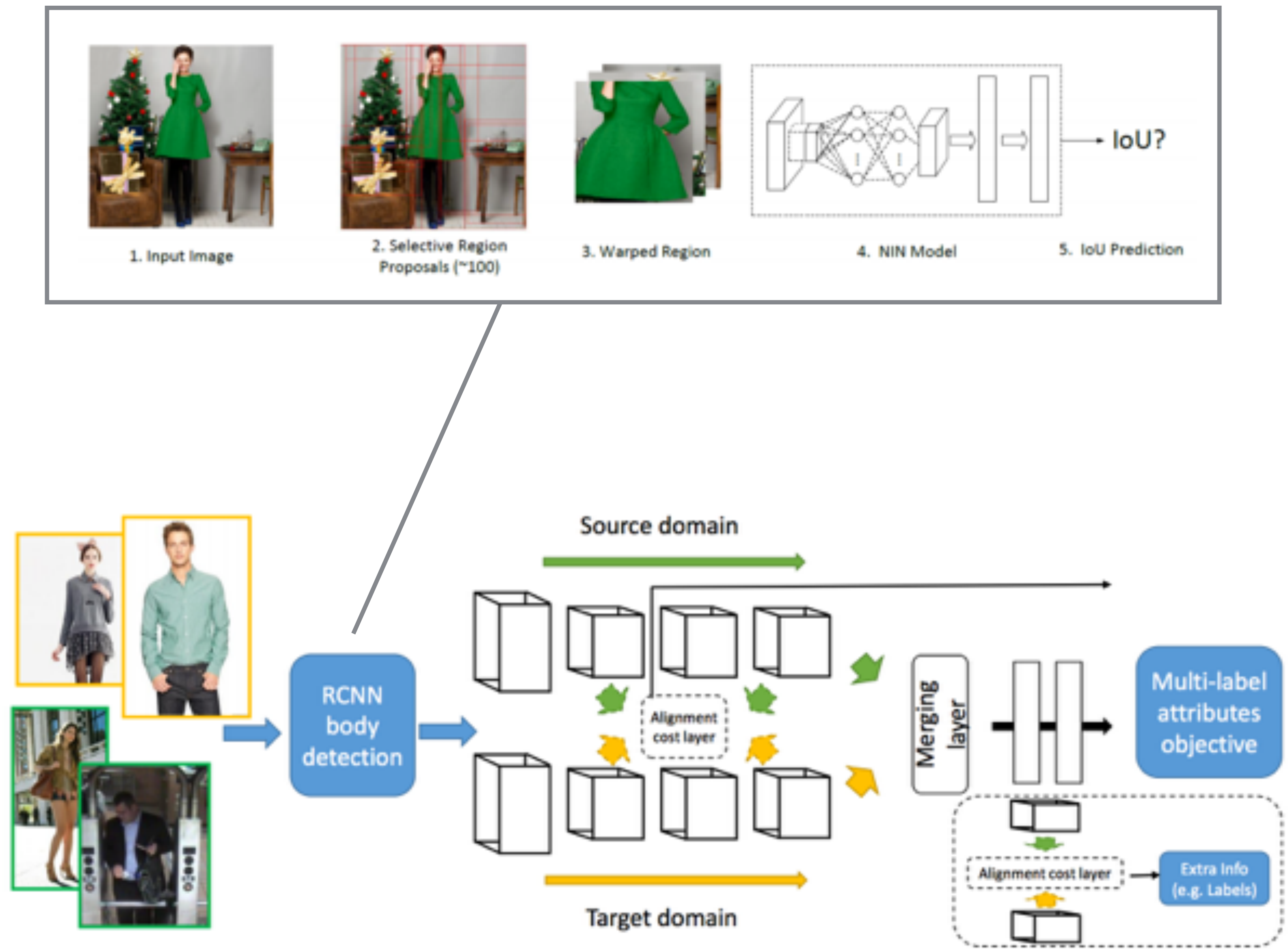
(a) Non-adapted (b) Adapted (a) Non-adapted (b) Adapted

Unsupervised Domain Adaptation by Backpropagation, Yaroslav Ganin, Victor Lempitsky, ICML 2015

# Deep Domain Adaptation for Describing People Based on Fine-Grained Clothing Attributes




A deep domain adaptation method to bridge the gap between images crawled from online shopping stores and unconstrained photos.



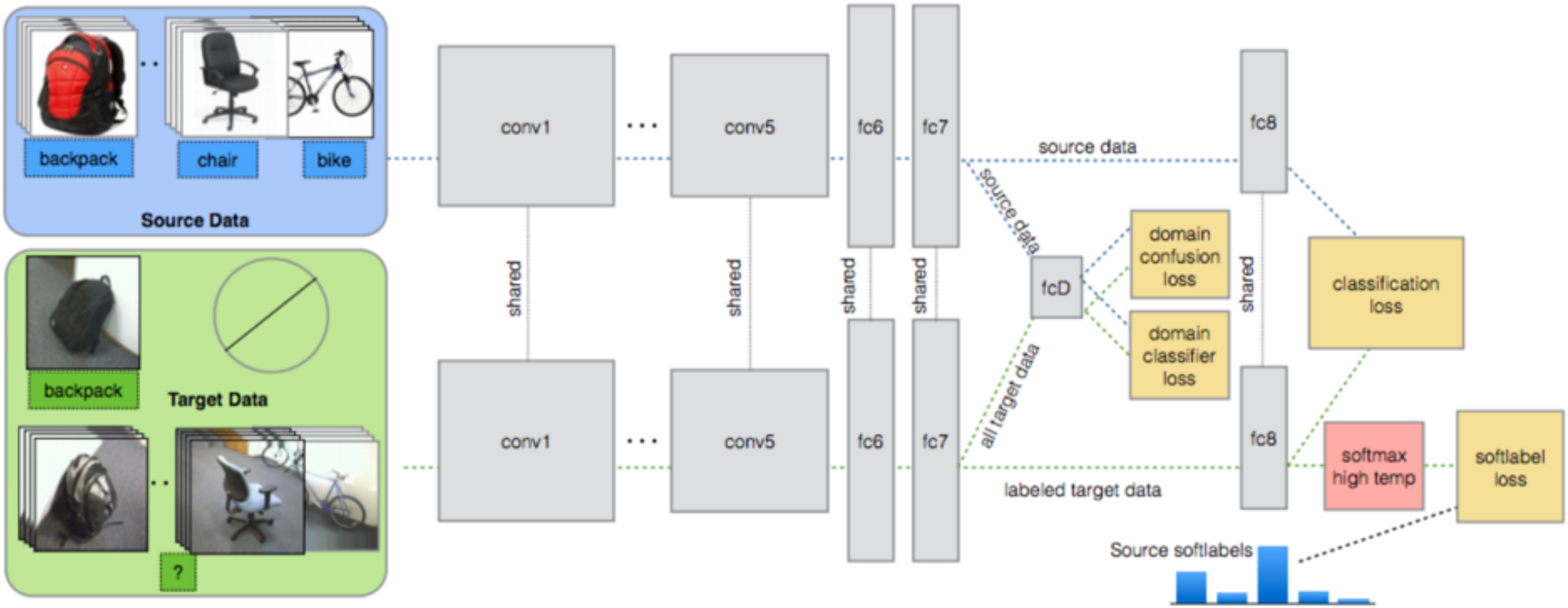

$$f(s,t) = ||X_s - X_t|| \times \lambda\phi(s,t)$$

Deep Domain Adaptation for Describing People Based on Fine-Grained Clothing Attributes, Qiang Chen et.al., CVPR15

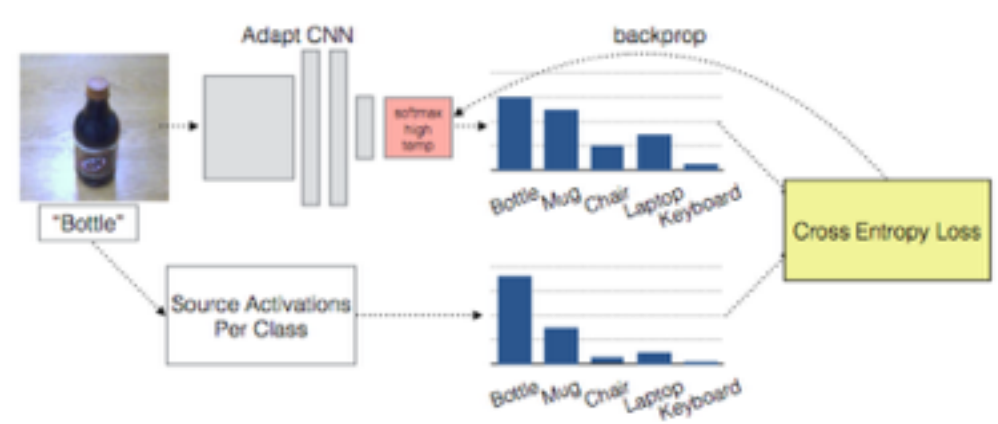
# Deep Domain Adaptation for Describing People Based on Fine-Grained Clothing Attributes

Deep Domain Adaptation for Describing People Based on Fine-Grained Clothing Attributes, Qiang Chen et.al., CVPR15

# Simultaneous Deep Transfer Across Domains and Tasks




This approach simultaneously optimizes for **domain invariance** to facilitate domain transfer and uses a **soft label distribution matching loss** to transfer information between tasks

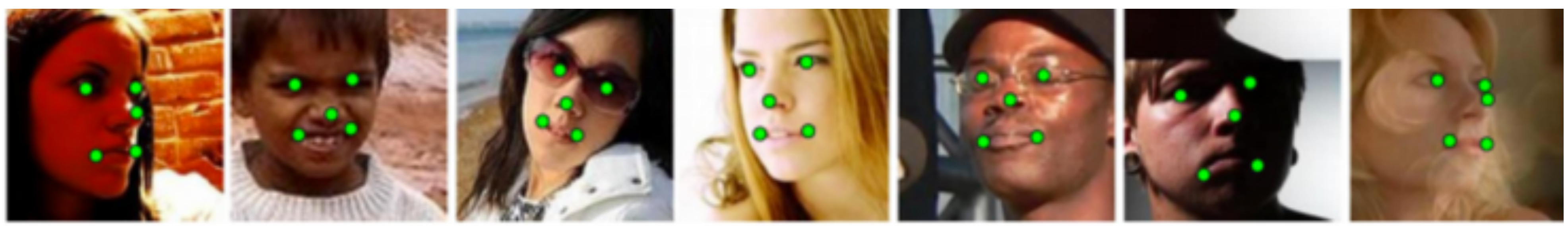



Simultaneous Deep Transfer Across Domains and Tasks, Eric Tzeng et.al., ICCV 2015

# Face Recognition with Deep Convolutional Networks

# Facial Landmark Detection by Deep Multi-task Learning

TCDCN



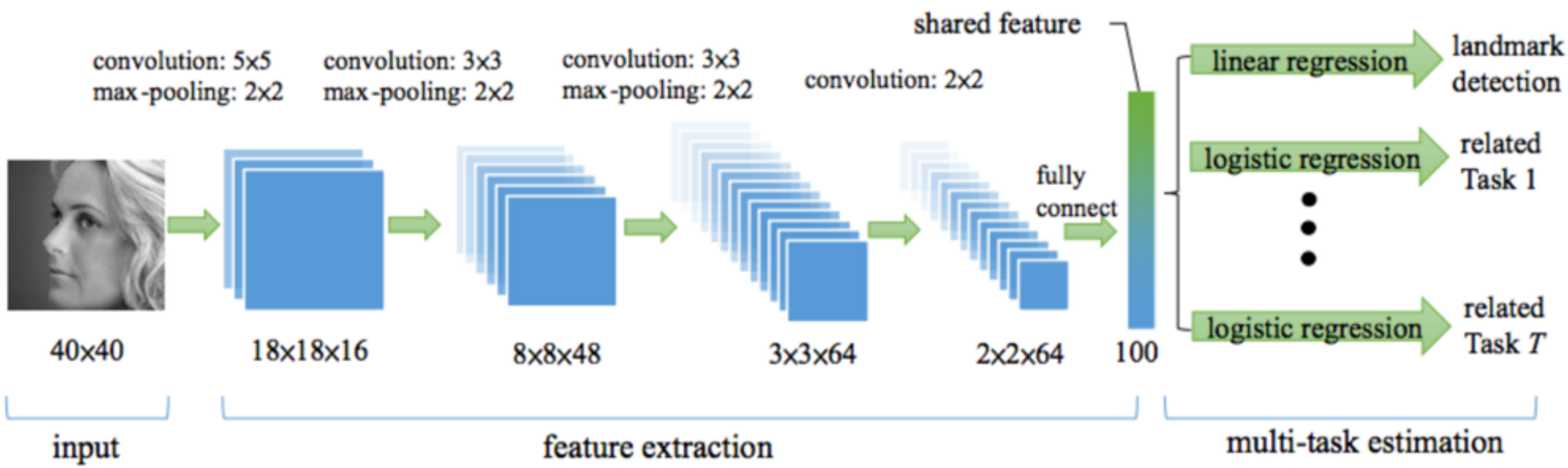
Auxiliary Tasks

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| wearing glasses | × | × | √ | × | √ | × | × |
| smiling | × | √ | × | × | × | × | × |
| gender | female | male | female | female | male | male | female |
| pose | right profile | frontal | frontal | left | frontal | frontal | right profile |

Improving facial landmark detection robustness through multi-task learning.

Facial Landmark Detection by Deep Multi-task Learning, Zhang et. al., ECCV 2014

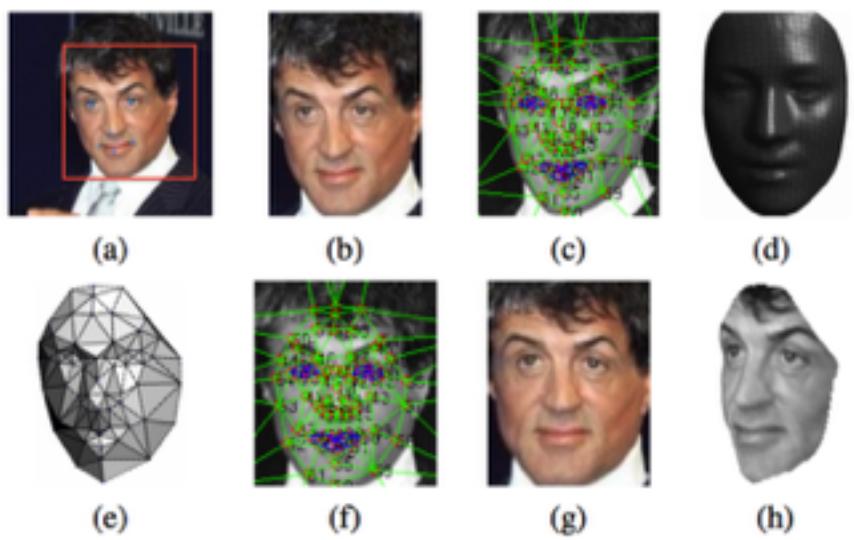
# Facial Landmark Detection by Deep Multi-task Learning




Improving facial landmark detection robustness through multi-task learning.

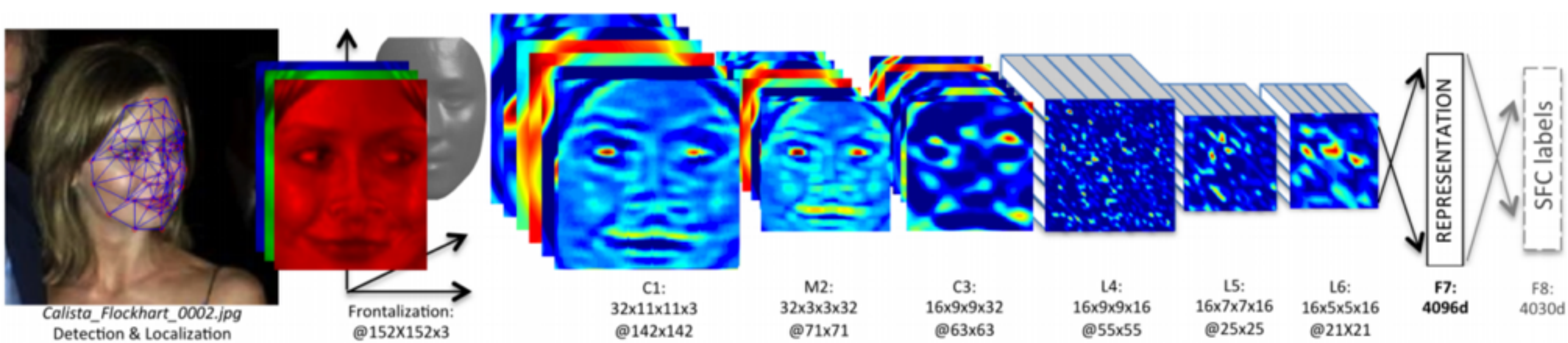
Facial Landmark Detection by Deep Multi-task Learning, Zhang et. al., ECCV 2014

# DeepFace: Closing the Gap to Human-Level Performance in Face Verification




Alignment / Frontalization

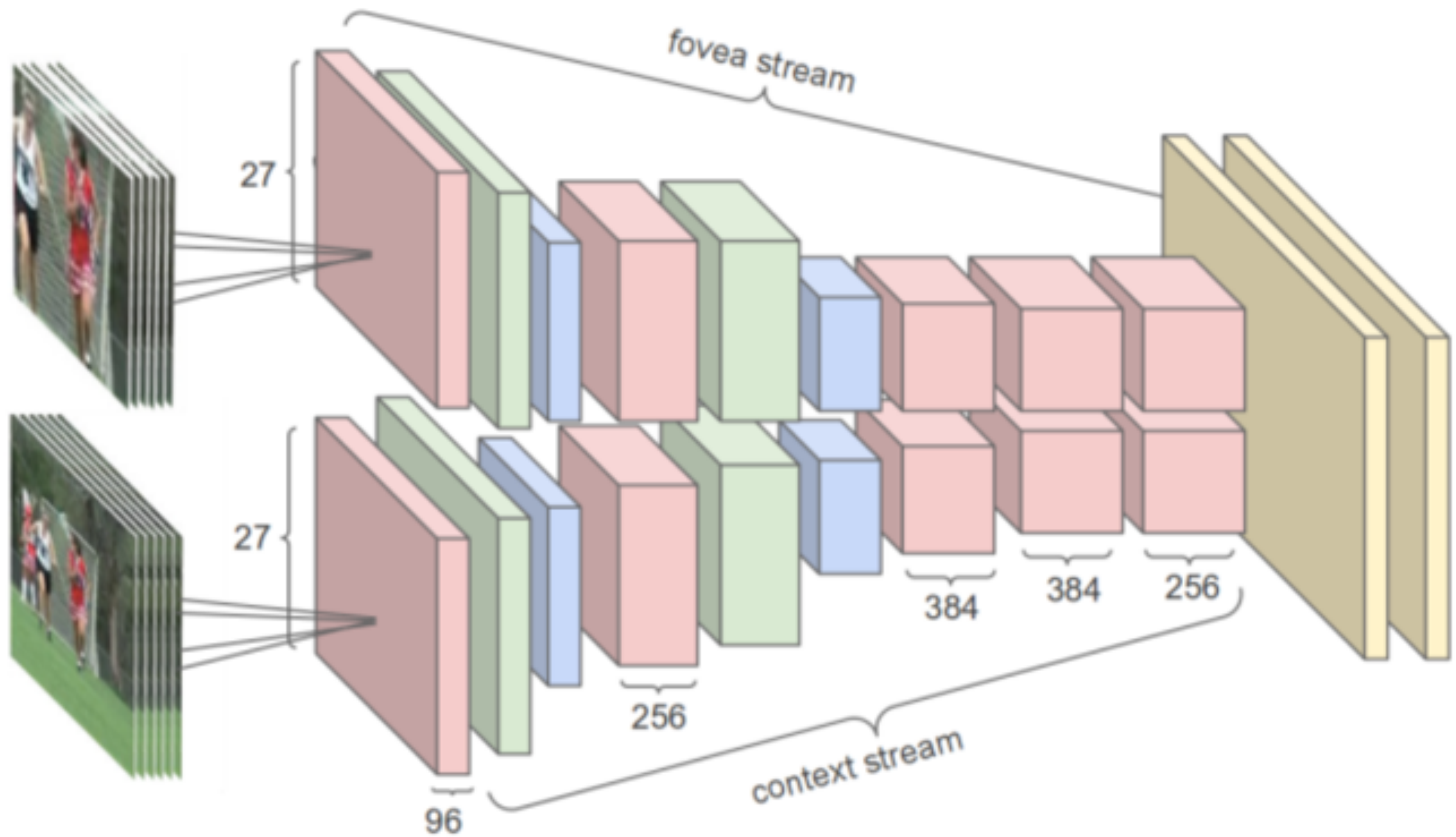
The recognition accuracy is approaching to human performance for face verification




DeepFace: Closing the Gap to Human-Level Performance in Face Verification, Taigman et.al., CVPR 2014

# Action Recognition with Deep Convolutional Networks

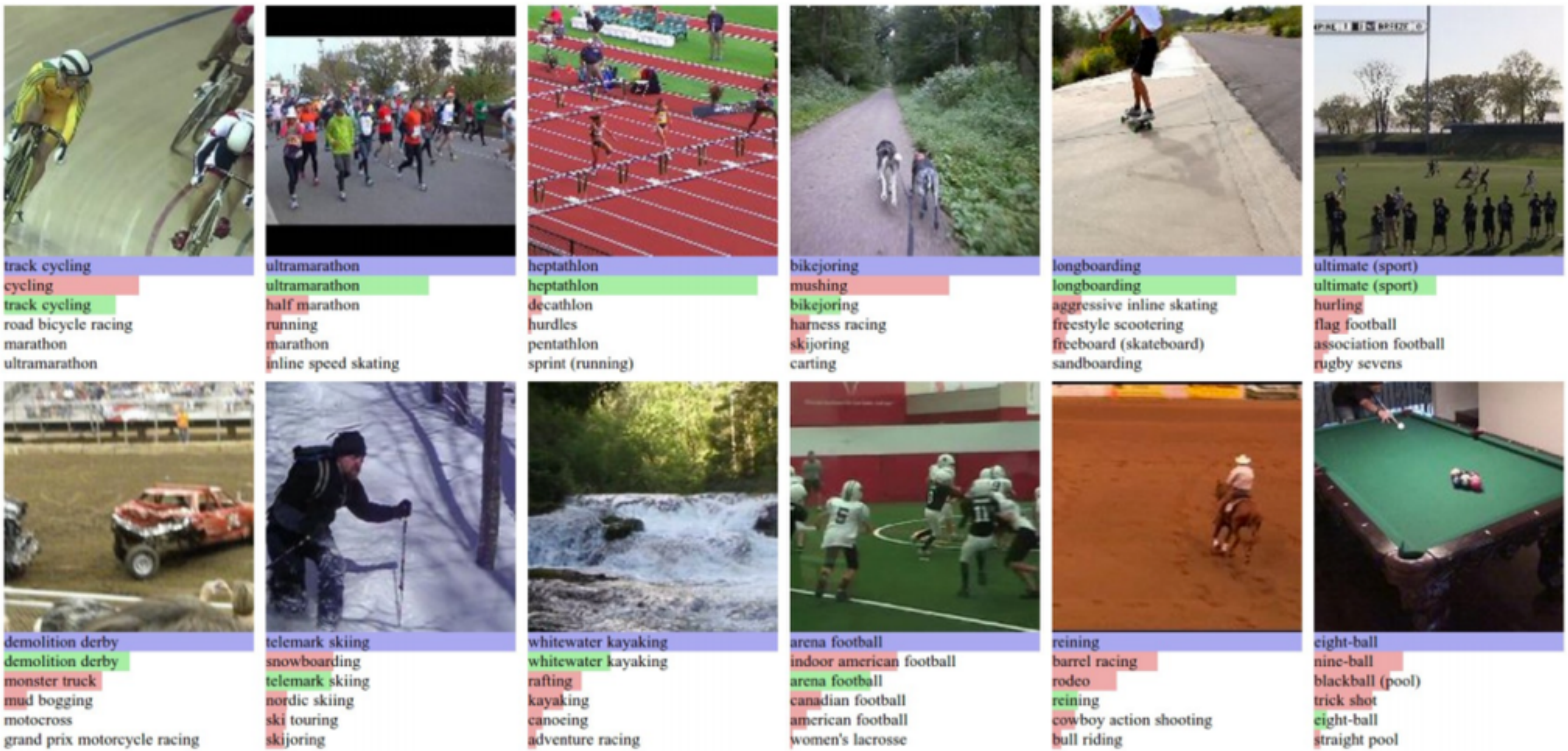
# Large-scale Video Classification with Convolutional Neural Networks



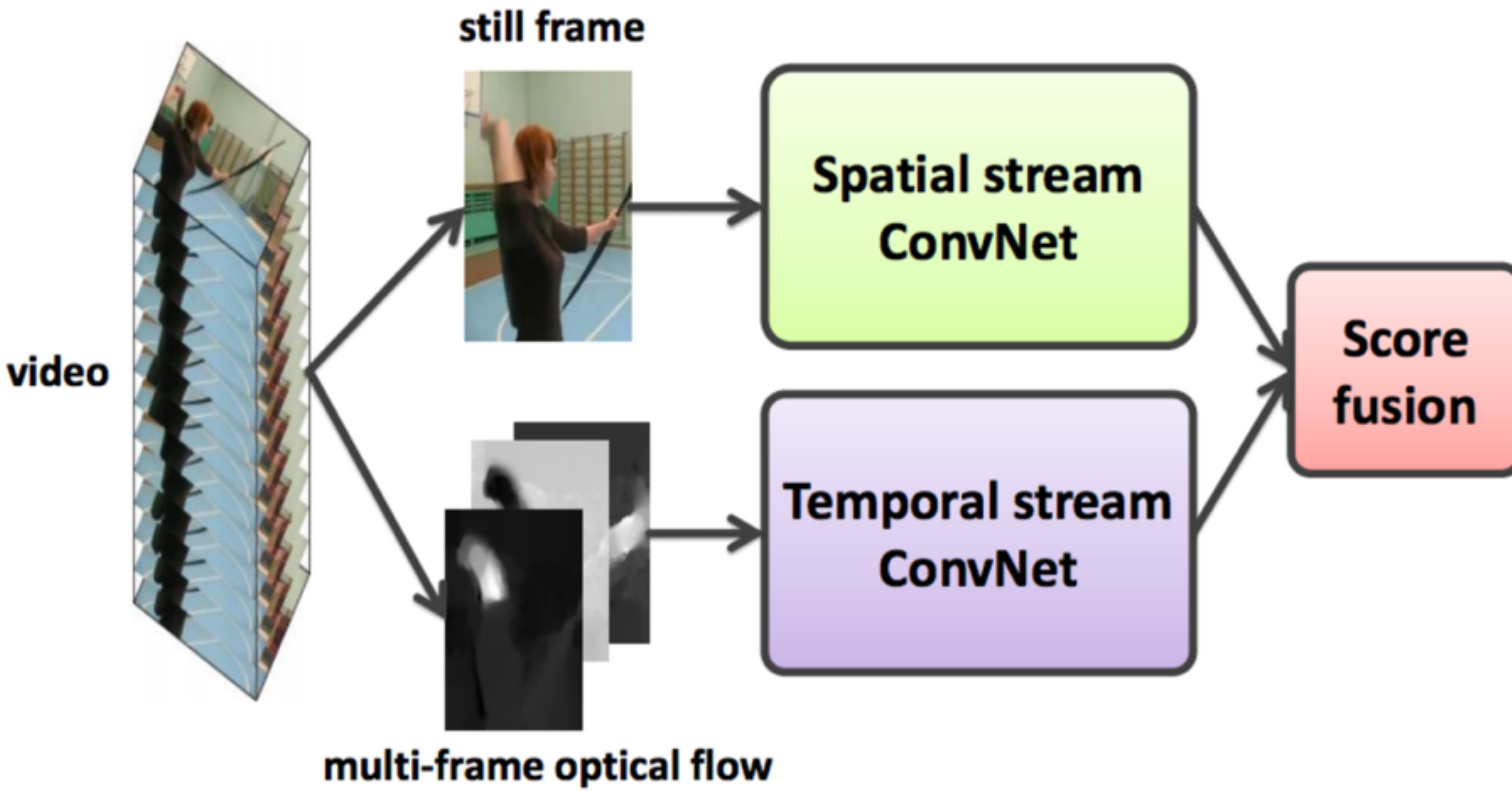

Multi-resolution CNN architecture. Input frames are fed into two separate streams of processing: a context stream that models low-resolution image and a fovea stream that processes high-resolution center crop. Both streams consist of alternating convolution (red), normalization (green) and pooling (blue) layers. Both streams converge to two fully connected layers (yellow)

Large-scale Video Classification with Convolutional Neural Networks, Karpathy et.al., CVPR 2014

# Large-scale Video Classification with Convolutional Neural Networks



# Two-Stream Convolutional Networks for Action Recognition




Two-Stream Convolutional Networks for Action Recognition in Videos, Karen Simonyan, Andrew Zisserman, NIPS 2014