(5 reasons) Why Deep Learning Works

Mattia Cerrato

9^{th} October, 2017

So, Deep Learning works...

Deep Learning/Neural Networks is gaining traction in many Machine Learning and Information Retrieval areas

So, Deep Learning works...

Deep Learning/Neural Networks is gaining traction in many Machine Learning and Information Retrieval areas

- Computer Vision: object recognition, segmentation...
- Machine Listening/Speech Recognition: speech to text, genre identification...
- Natural Language Processing: machine translation, word embeddings...
- Miscellaneous tasks: neural style transfer

Computer Vision

Object/Image classification + location: ImageNet

1.2M train images, 100k test images, 1k different classes

Computer Vision

NIPS 2012 - ImageNet Classification with Deep Convolutional Neural Networks (Krizhevsky et al.) [1]

Achieves $\sim$15% top-5 error rate

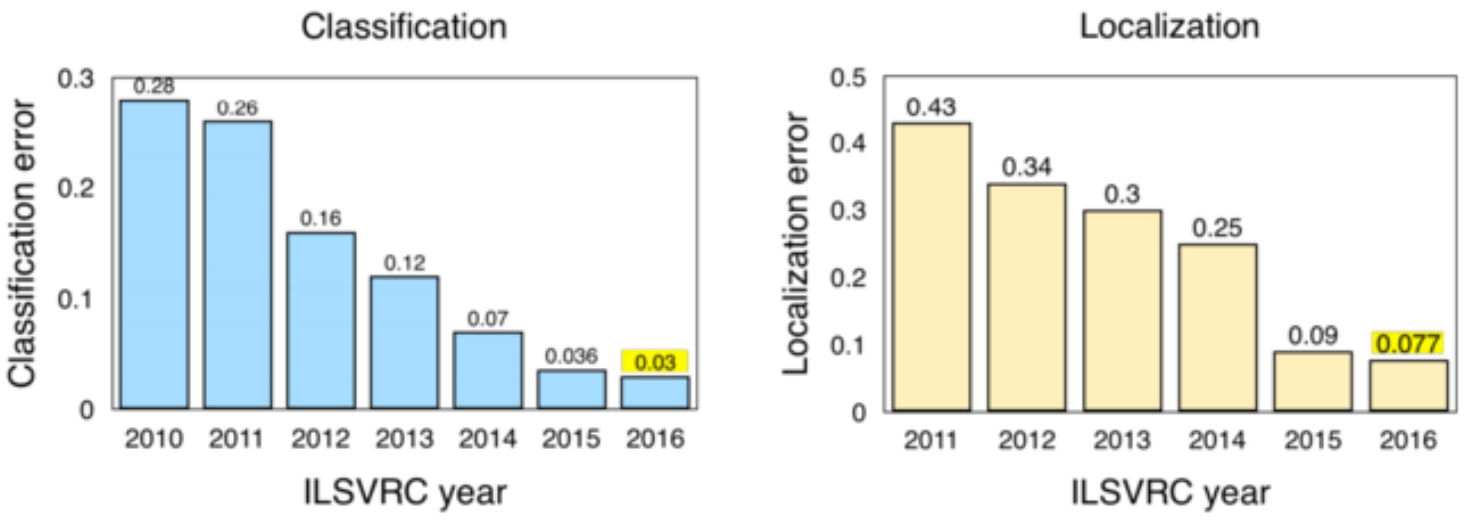
Computer Vision

NIPS 2012 - ImageNet Classification with Deep Convolutional Neural Networks (Krizhevsky et al.) [1]

Achieves $\sim$15% top-5 error rate

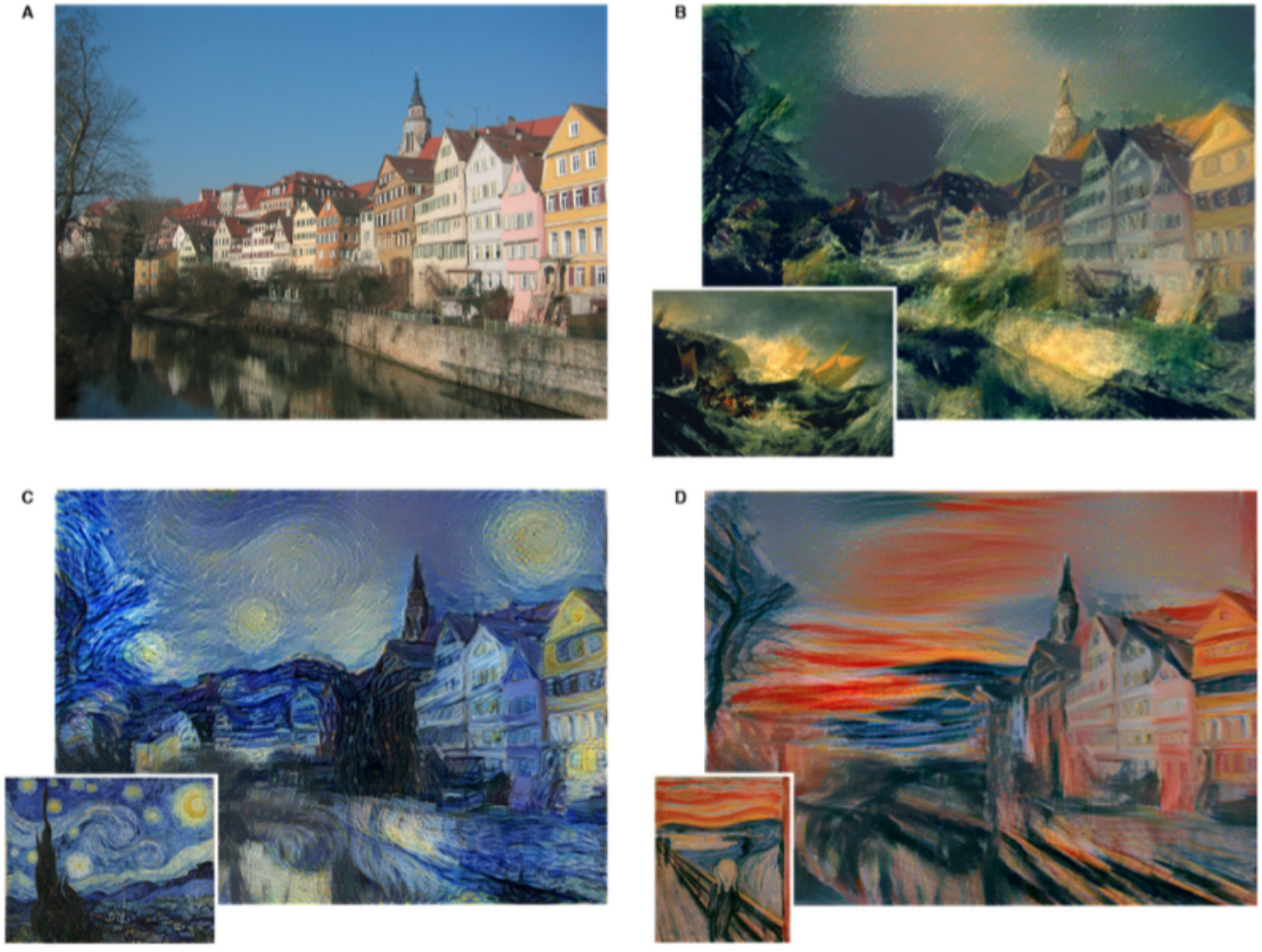
ImageNet Competition 2016: top models have $<$3% error rate

Computer Vision



From "A Year in Computer Vision" [2]

Neural Transfer



From "A Neural Algorithm of Artistic Style" [3]

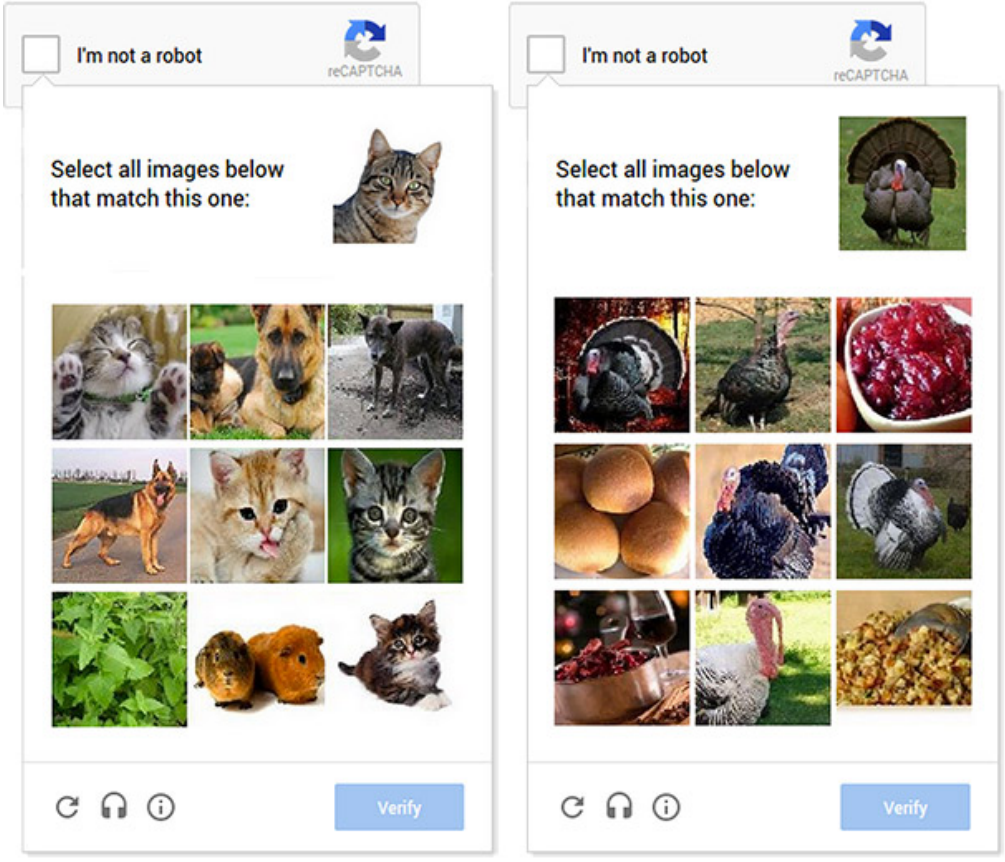
Why is this happening

There are a lot of factors involved in the success Deep Learning is enjoying

Why is this happening

There are a lot of factors involved in the success Deep Learning is enjoying

Labeled data



Why is this happening

There are a lot of factors involved in the success Deep Learning is enjoying

- Labeled data
- Computational resources
- Ease of use and library quality
- Brain analogies maybe? Learning as humans do

Why is this happening

There are a lot of factors involved in the success Deep Learning is enjoying

- Labeled data
- Computational resources
- Ease of use and library quality
- Brain analogies maybe? Learning as humans do
- Good models probably

Let's focus on the models.

Why is this happening

Deep Learning lacks an unified theory

Why is this happening

Deep Learning lacks an unified theory

xkcd, "Machine Learning" [4]

Why is this happening

Deep Learning lacks an unified theory

How do you choose an architecture? How much time until convergence? Any error rate guarantees? Why is it working at all?

5 reasons why Deep Learning works

We looked into some arguments made by the DL community and report them

Not all of them actually answer the same question

5 reasons why Deep Learning works

We looked into some arguments made by the DL community and report them

Not all of them actually answer the same question

- Why does it work "well"?
- Why does it work "better than other models"?
- Why does it work at all?

5 reasons why Deep Learning works

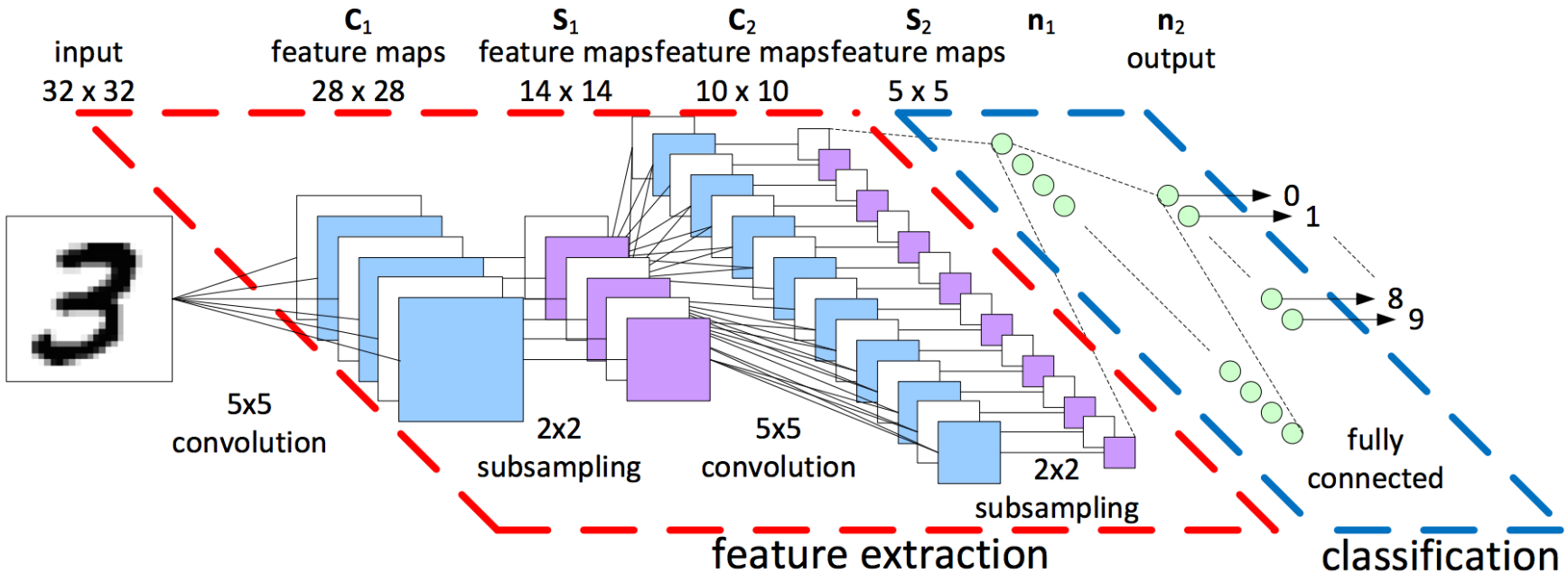
We looked into some arguments made by the DL community and report them

Not all of them actually answer the same question

- Why does it work "well"?
- Why does it work "better than other models"?
- Why does it work at all?

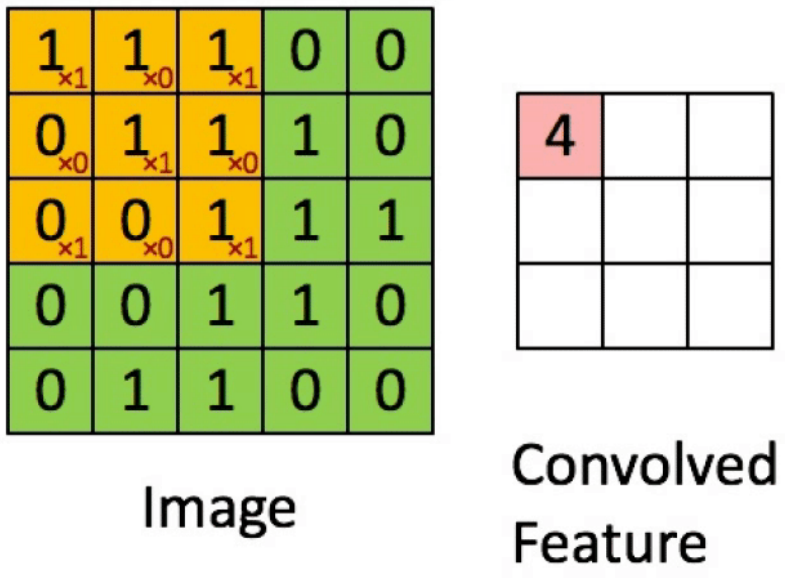
All arguments are easier to understand when applied to Convolutional Neural Networks

Convolutional Neural Networks



From "Speed sign detection and recognition by convolutional neural networks" [5]

Convolution is feature extraction



From Prof. Isola, Ali Ghodsi's slides [6]

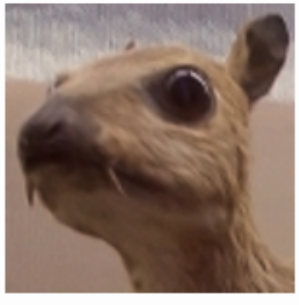
Convolution is feature extraction

$$\star \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} =$$

(identity)

$$\star \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} * 1/9 =$$

(blurring)



$$\star \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix} =$$

(edge detection)

ConvNets are learning how to extract features

ConvNets are learning how to extract features

Each convolutional level/layer in a ConvNet is actually extracting features

ConvNets are learning how to extract features

Each convolutional level/layer in a ConvNet is actually extracting features

Very often, the last layer is densely connected $\rightarrow$ perceptron, MLP

ConvNets are learning how to extract features

Each convolutional level/layer in a ConvNet is actually extracting features

Very often, the last layer is densely connected $\rightarrow$ perceptron, MLP

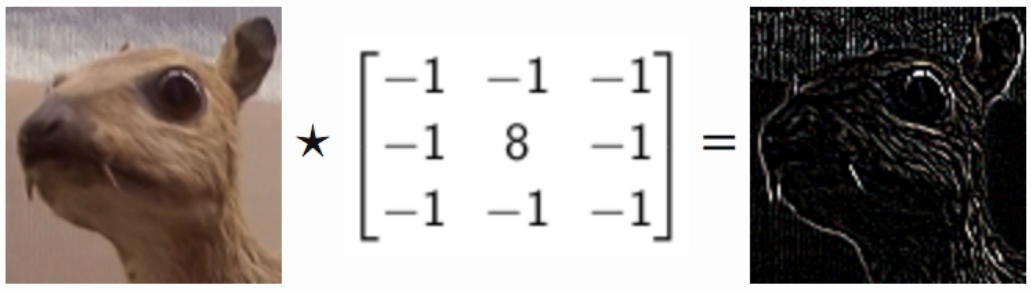
We are learning to "make things easy" for the classifier by learning **good convolution filters**

ConvNets are learning how to extract features

Each convolutional level/layer in a ConvNet is actually extracting features

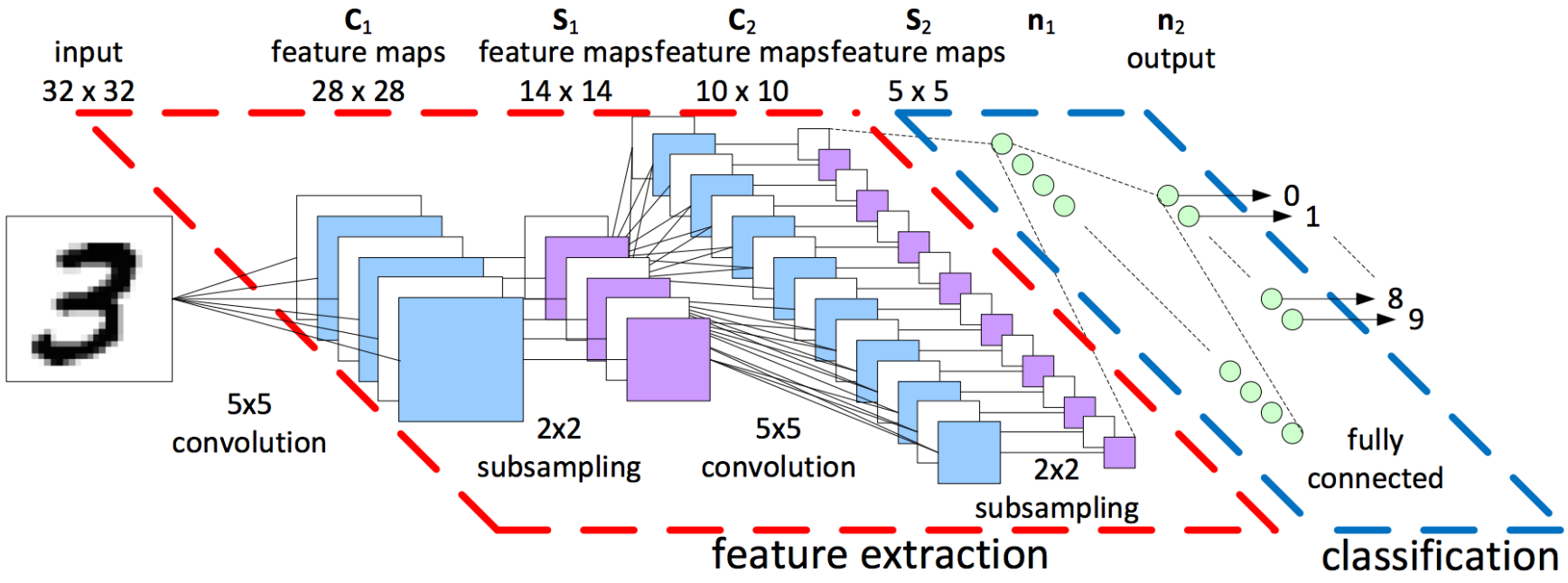
Very often, the last layer is densely connected $\rightarrow$ perceptron, MLP

We are learning to "make things easy" for the classifier by learning **good convolution filters**



ConvNets are also learning to classify

ConvNets are also learning to classify



From "Speed sign detection and recognition by convolutional neural networks" [5]

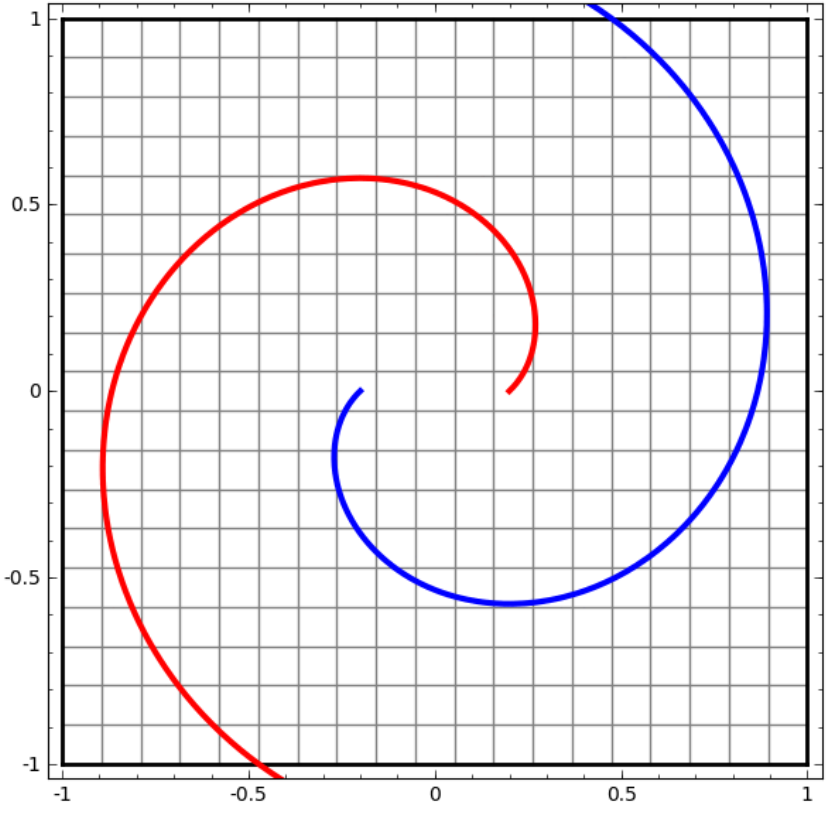
ConvNets are also learning to classify

Using the same algorithm (fancy gradient descent), ConvNets are also learning to classify

However, the perceptron offers good performance only when the dataset is linearly separable

$\rightarrow$ a ConvNet learns **how to disentangle the input space**

ConvNets disentangle the input space



From C. Olah, "Neural Networks, Manifolds, and Topology" [5]

Reason 1

Deep Neural Networks learn ~~optimal~~ good features which can disentangle the input space

More evidence: using the last layer of a ConvNet as input for a linear SVM works great! [7]

Reason 1 Rebuttal

Is depth needed at all?

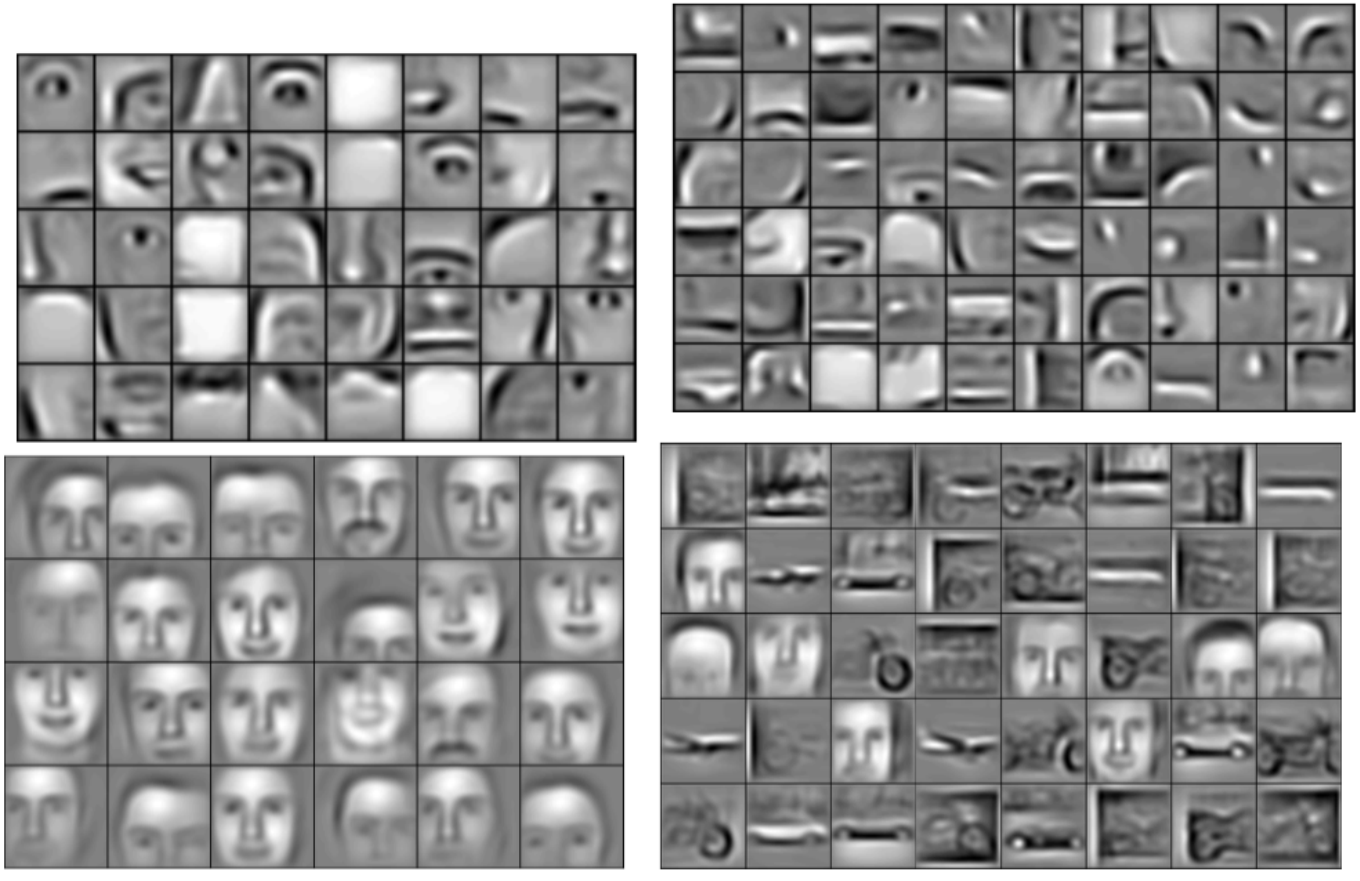
We also know we only need one hidden layer to approximate any function... [8]

Deep Networks extract a hierarchy of features

A lot of empirical evidence points to the fact that deep networks learn generic filters in the first few levels, which become more complex as we go deeper

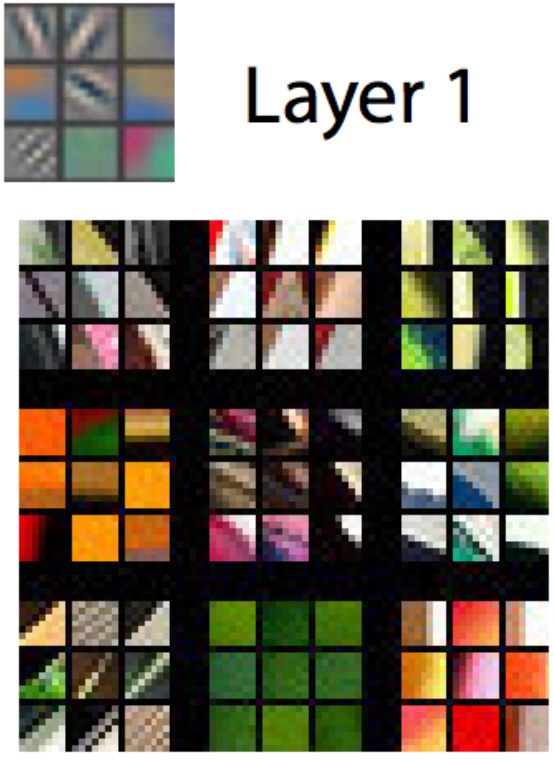
Feature Hierarchy: Empirical Evidence (1 of 2)

Lee et al. [9], Deep Belief Networks:



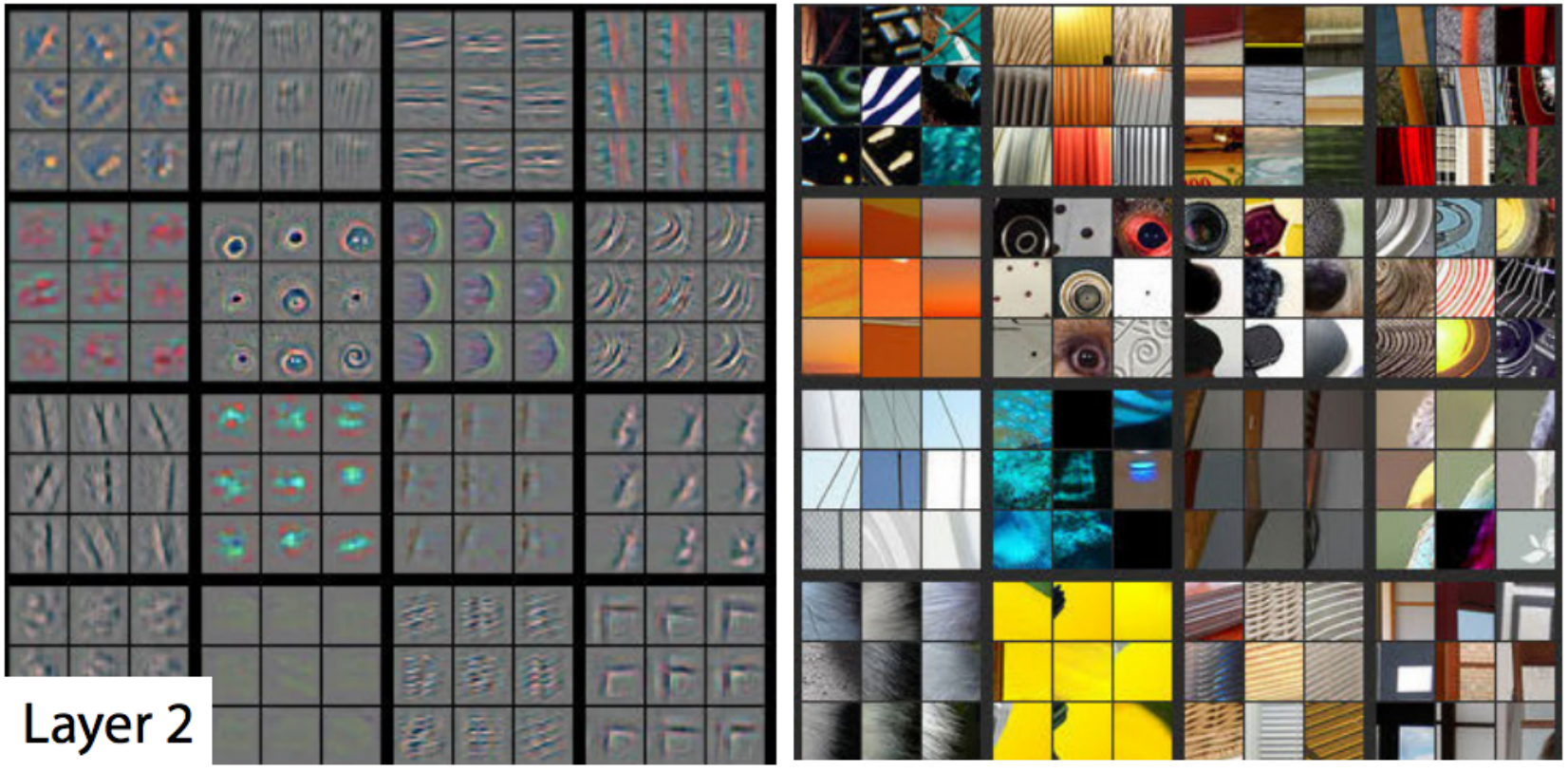
Feature Hierarchy: Empirical Evidence (2 of 2)

Zeiler et al. [10], Deconvolutional Networks: input space reconstruction



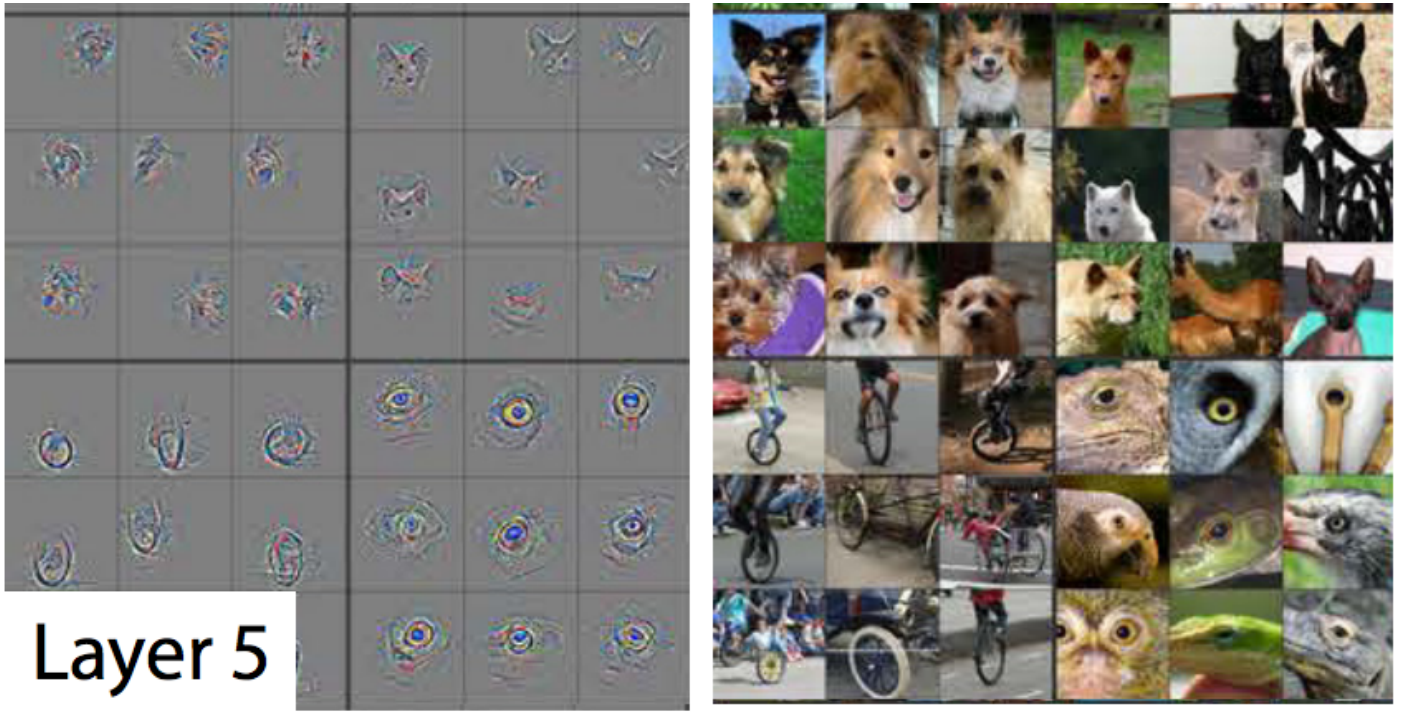
Feature Hierarchy: Empirical Evidence (2 of 2)

Zeiler et al. [10], Deconvolutional Networks: input space reconstruction



Feature Hierarchy: Empirical Evidence (2 of 2)

Zeiler et al. [10], Deconvolutional Networks: input space reconstruction



Reason 2

Deep Networks tend to learn a hierarchy of semantically meaningful features

Reason 2 Rebuttal

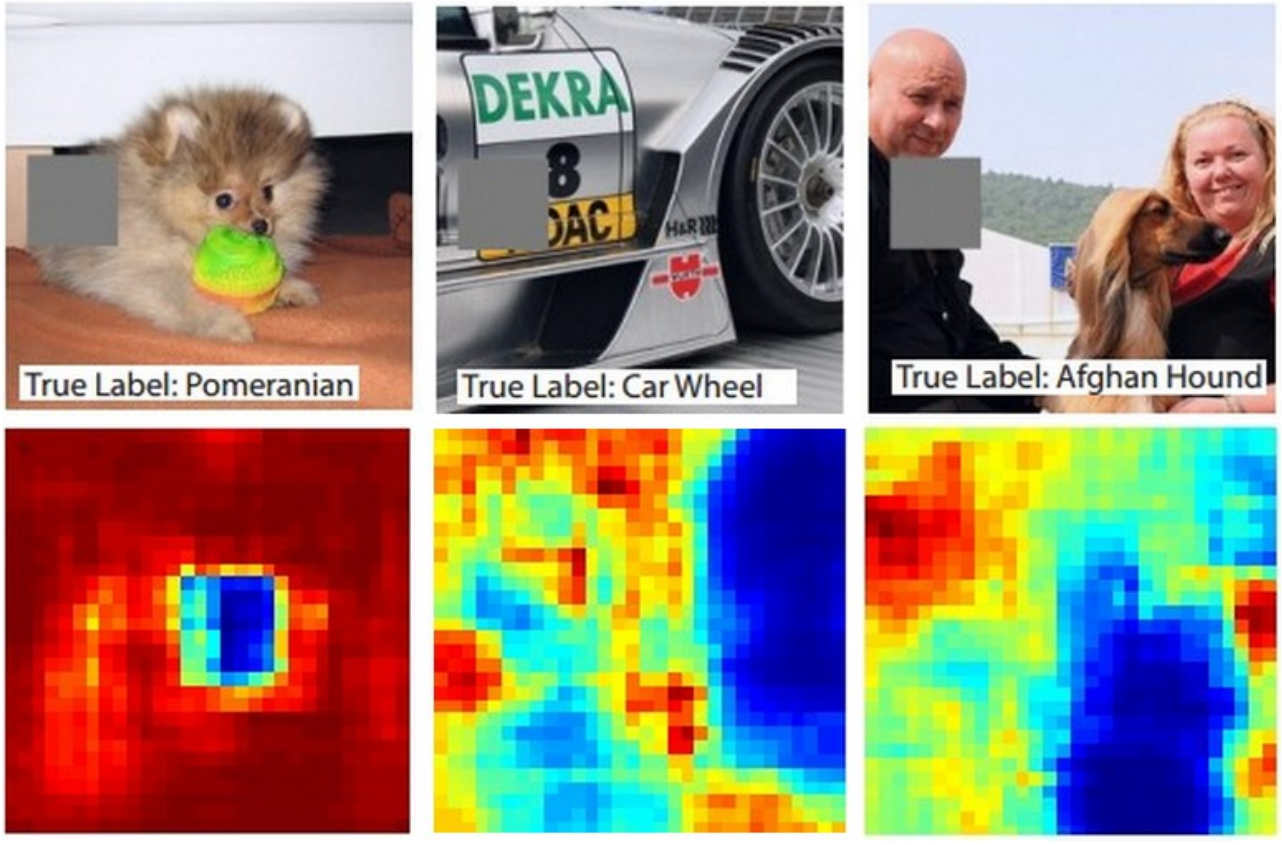
Are the models consistently doing that?

How do they even generalize? Is there a generalization principle?

Is a hierarchy useful anyway?

More Empirical Evidence (3 of ~~2~~ 3)

Zeiler et al. [10], occlusion invariance (not!)



Local representations

Hinton, 1984 [11]

"Given a network of simple computing elements and some entities to be represented, the most straightforward scheme is to use one computing element for each entity. This is called a local representation."

Local representations

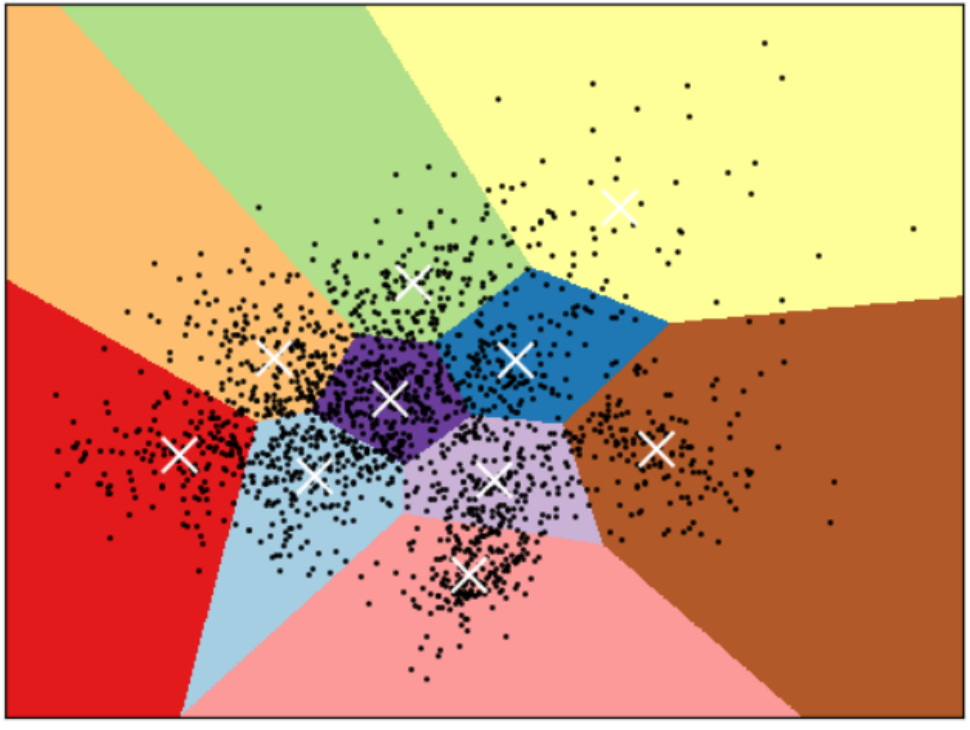
Bengio: Algorithms which learn local representations learn parameters for each region they find in input space (think centroids, support vectors...)

$\rightarrow$ the number of distinguishable regions is linear in the number of parameters

Local representations: Clustering

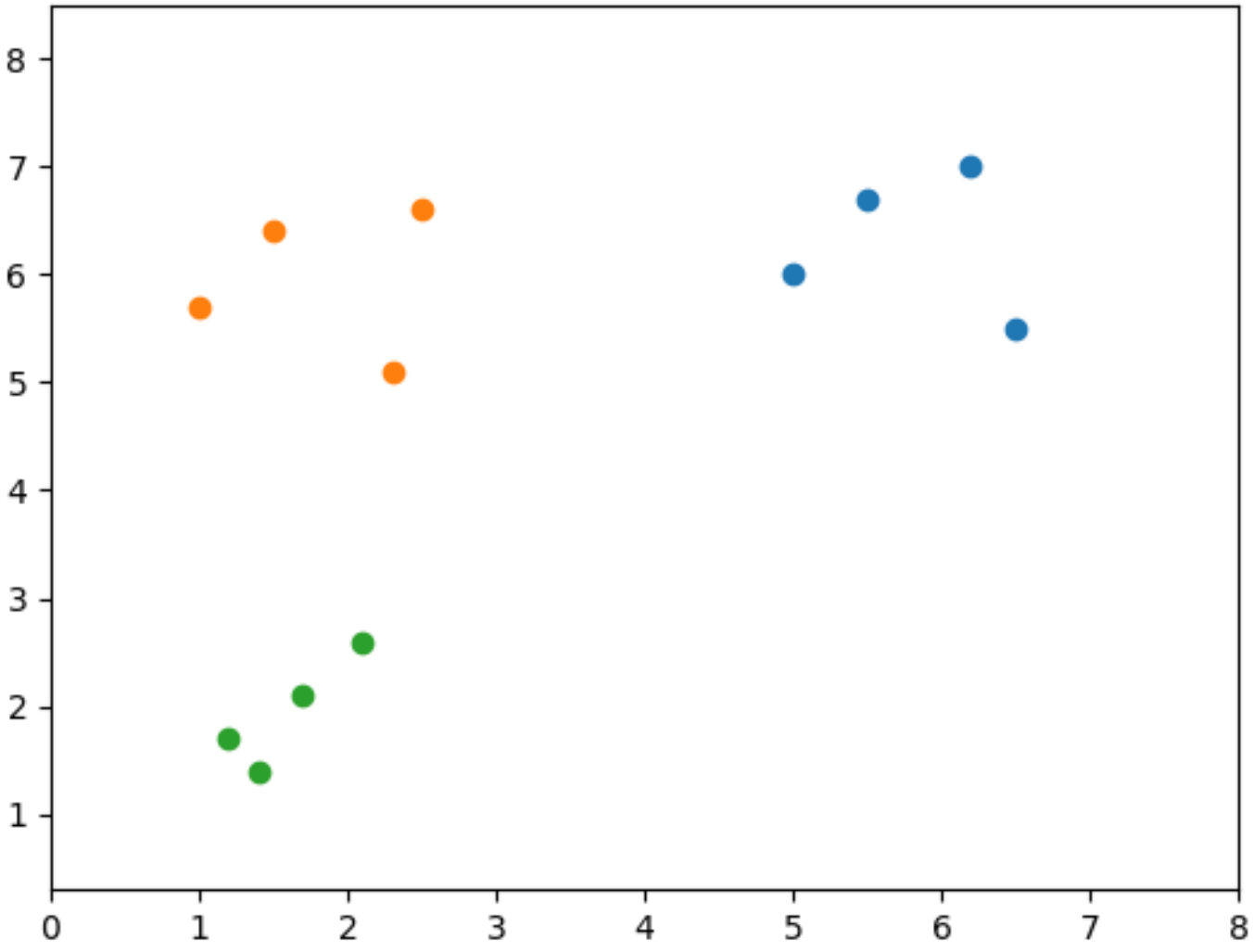
$\rightarrow$ the number of distinguishable regions is linear in the number of parameters

$\rightarrow$ n regions, n centroids, n support vectors...

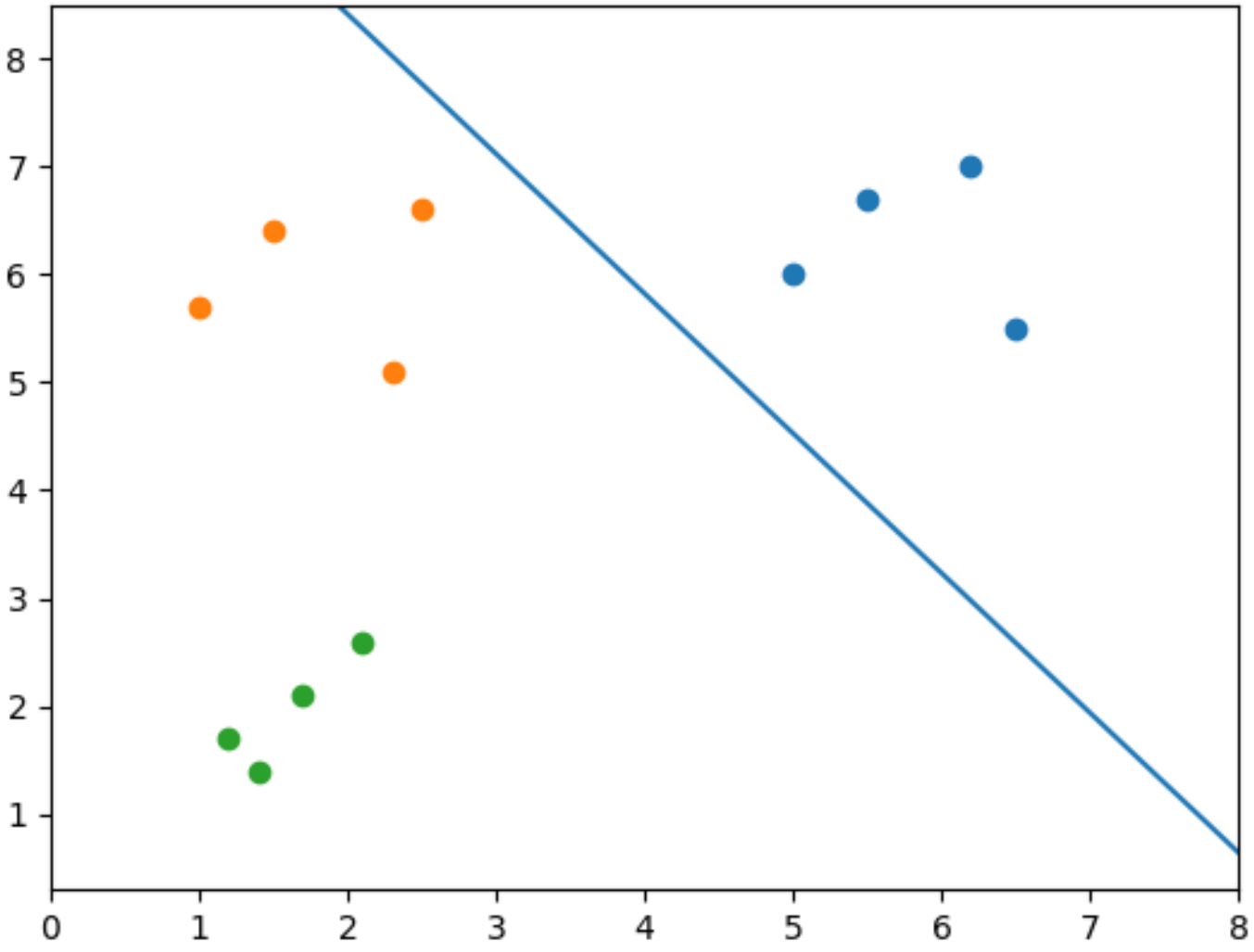


A clustering example from the scikit-learn website [12]

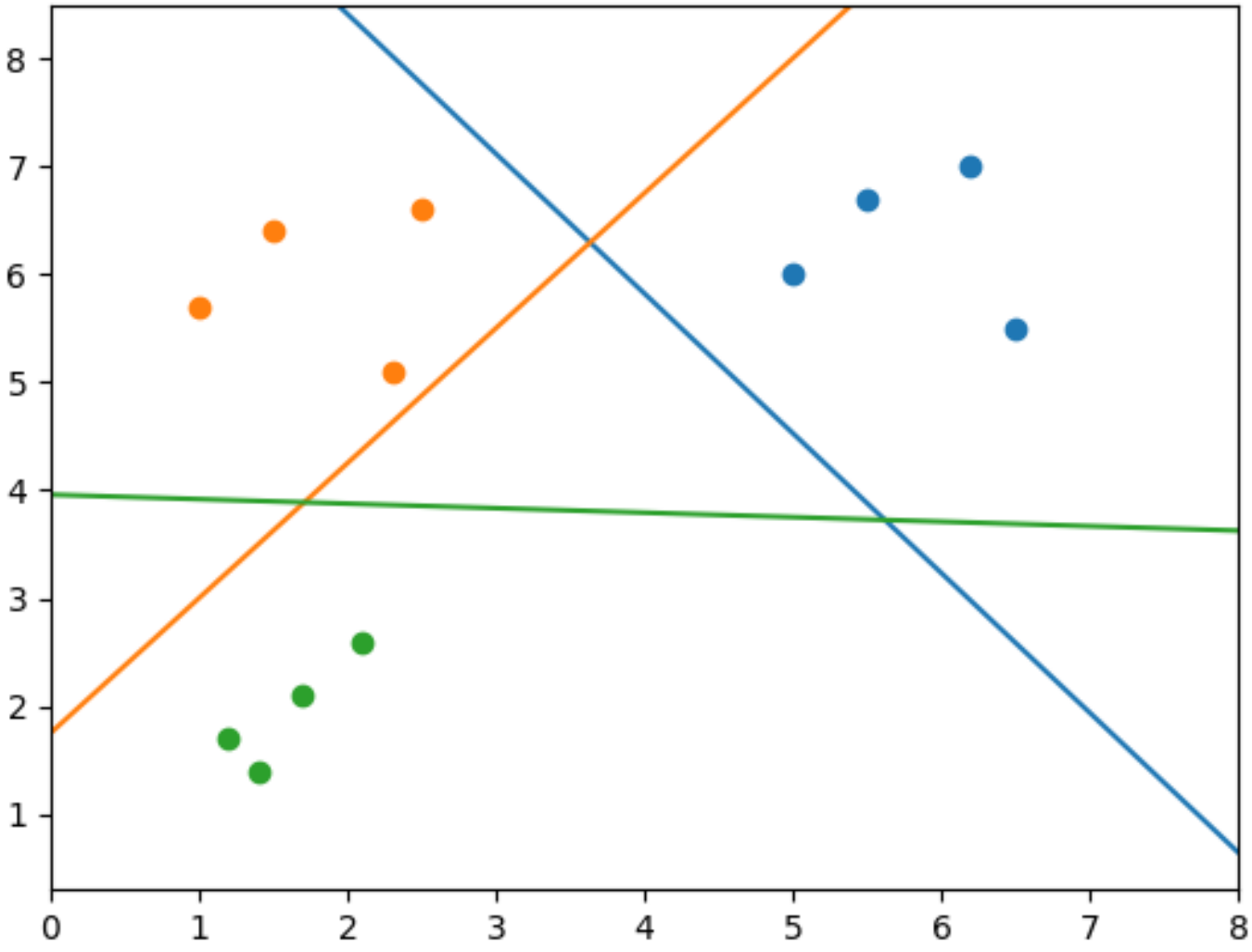
Local representations: SVM



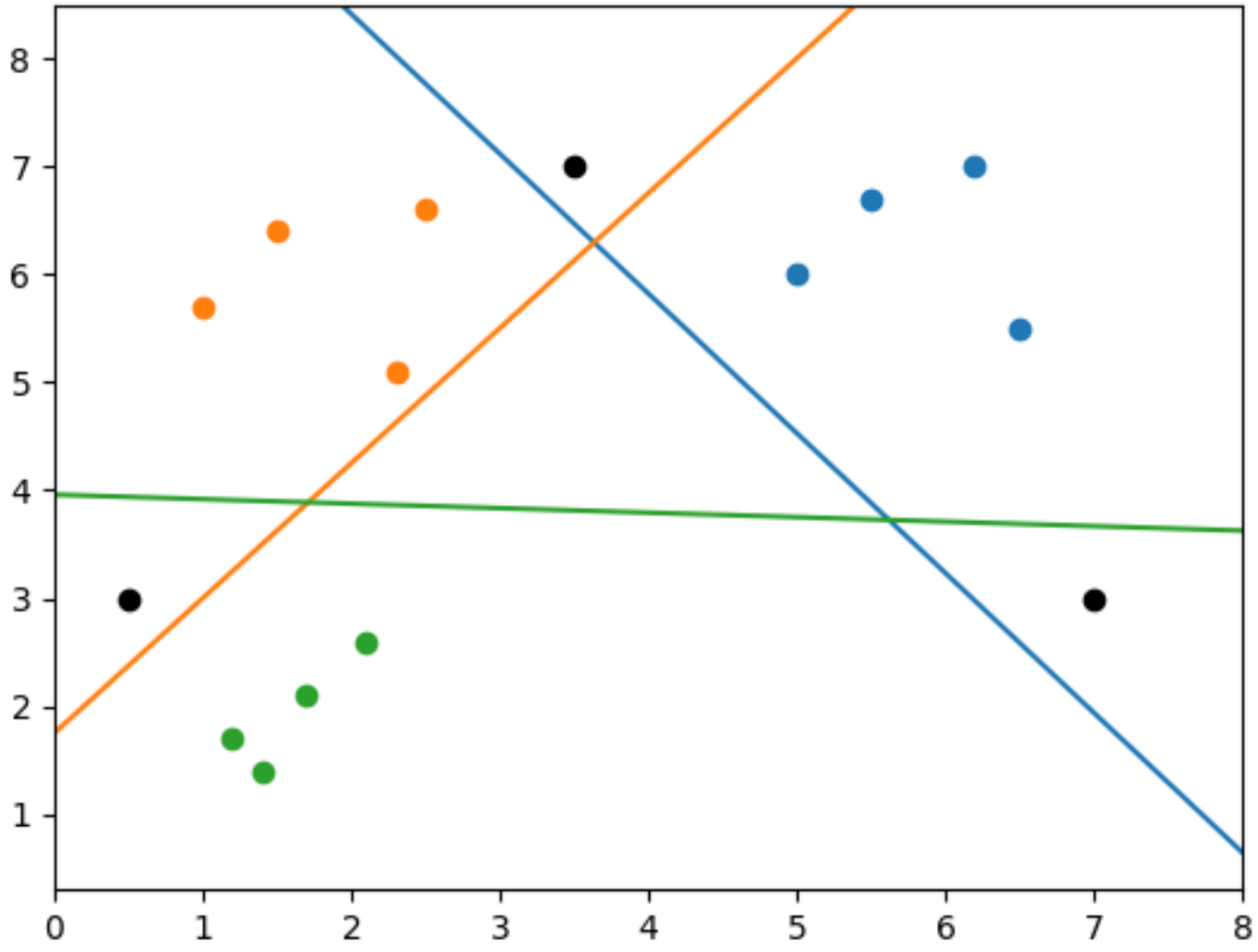
Local representations: SVM



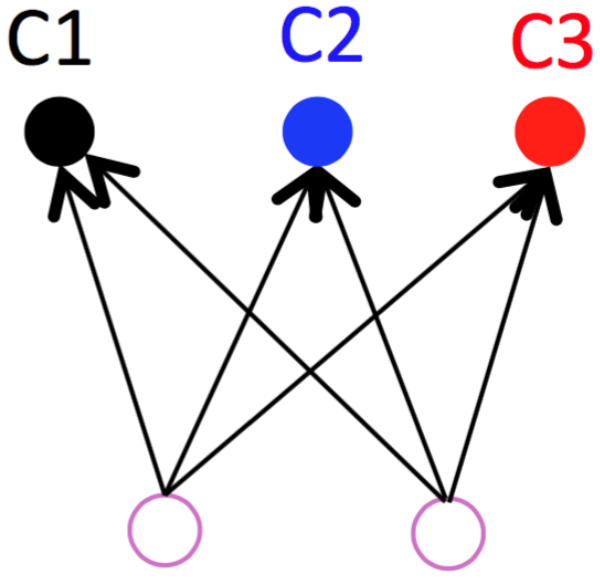
Local representations: SVM

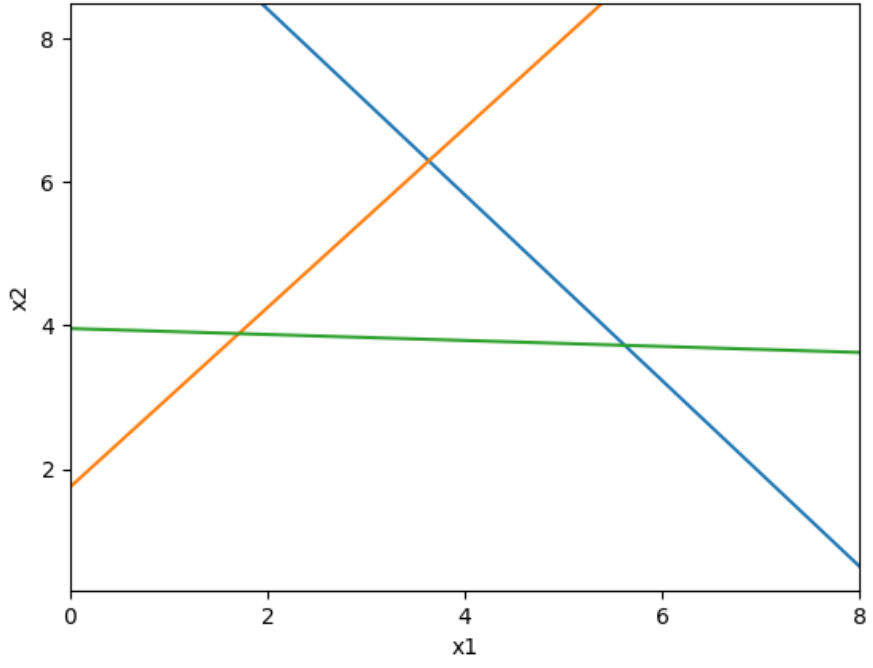


Local representations: SVM



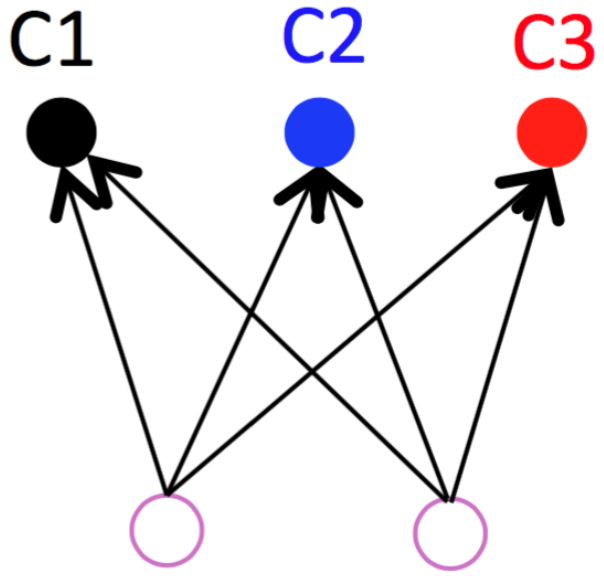
Distributed Representations: MLP example

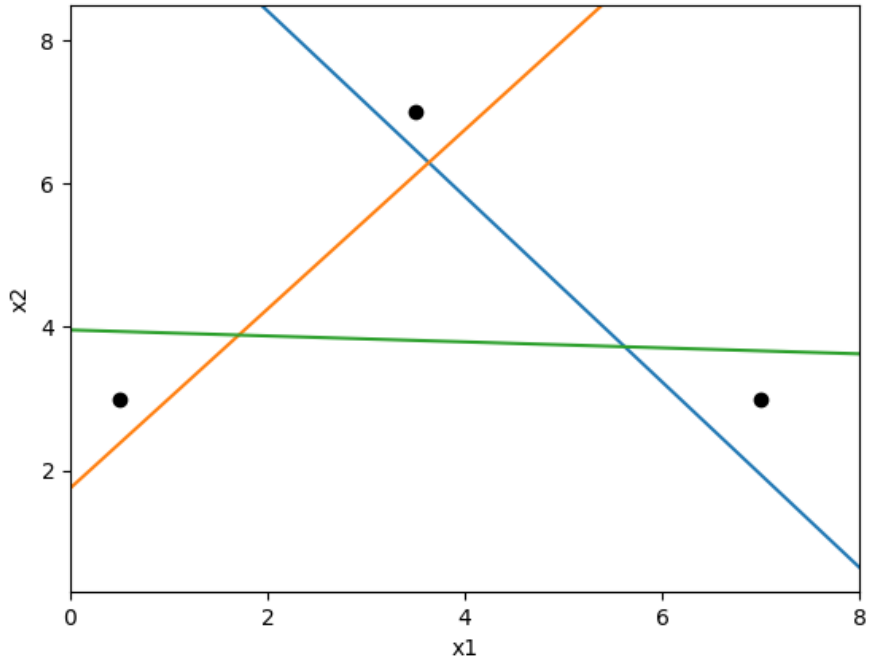




Distributed Representations: MLP example

3 hidden units $\rightarrow$ 3 hyperplanes $\rightarrow$ 7 regions (general formula is quadratic)





Distributed Representations - Hinton 1984

"Each active unit represents a 'micro-feature' of an item"

Distributed Representations - Hinton 1984

"Each active unit represents a 'micro-feature' of an item"

"Many people [...] can rapidly retrieve the item that satisfies the following partial description: It is an actor, it is intelligent, it is a politician."

Distributed Representations - Hinton 1984

"Each active unit represents a 'micro-feature' of an item"

"Many people [...] can rapidly retrieve the item that satisfies the following partial description: It is an actor, it is intelligent, it is a politician."

"All we need to do is modify the interactions between units so as to create a new stable pattern of activity. If this is done by modifying a large number of connections very slightly, the creation of a new pattern need not disrupt the existing representations."

Distributed Representations - Montufar et al. 2014

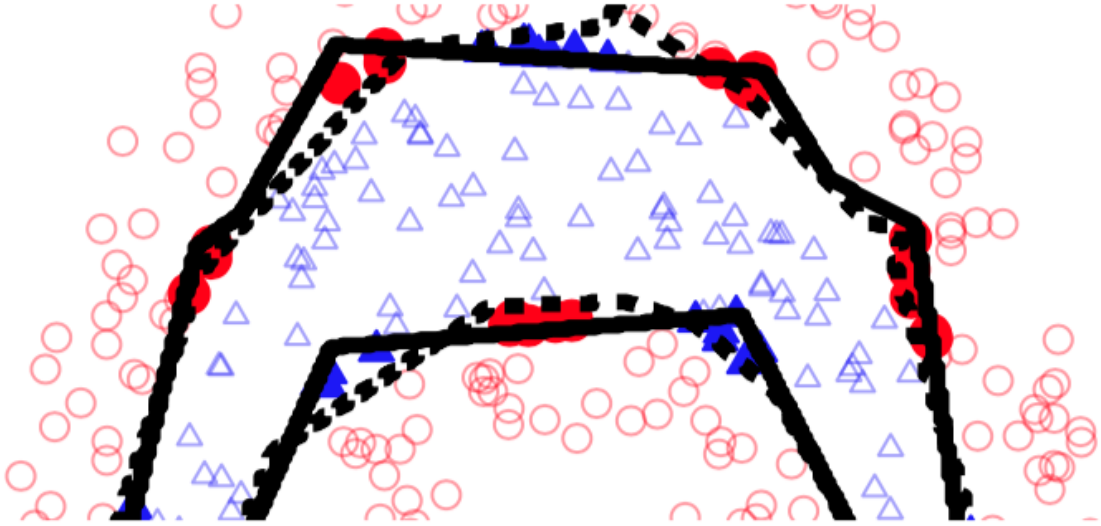
Montufar, Bengio et al.: in distributed representations, each parameter influences many regions, not just the local neighbors. $\rightarrow$ generalization to regions we don't have examples for

Distributed representations are useful when **the world is compositional**

Many visual cues are combined and aid in the final classification decision

Distributed Representations - Montufar et al. 2014

Bengio: "Deep distributed representations" fight the curse of dimensionality [13]



Deep networks with the same number of parameters (dashed line) can learn a higher number of linear regions compared to shallow nets

Distributed Representations - Montufar et al. 2014

"[...] each layer of a deep model is able to identify pieces of its input in such a way that the composition of layers identifies an exponential number of input regions. This results in exponentially replicating the complexity of the functions computed in the higher layers of the model."

Reason 3 + 4

Distributed representations exploit a prior assumption about the data/the world: compositionality as a generalization principle ($\sim$ highest margin in SVM)

Deep distributed representations fight the curse of dimensionality

Reason 3 + 4 rebuttal

How do we even get these representations?

Deep neural models have non-convex loss functions, so we should not be able to optimize them well enough

How much do we lose in non-convexity? (1)

How much do we lose in non-convexity? (1)

Choromanska, LeCun et al. 2015 [14]: local minima are focused in a narrow band, and the number of local minima outside this error region decreases exponentially with the size of the network $\rightarrow$ **many local minima have similar, "good enough" performance**

"It has to be emphasized however that this connection relies on a number of possibly unrealistic assumptions.", they do still provide experiments and empirical evidence

How much do we lose in non-convexity? (2)

Dauphin, Bengio et al. 2014 [15]: in high dimensions, there are a lot of saddle points and not that many local minima

Gradient descent can get slowed down dramatically by saddle points, and even get stuck (note that adaptive momentum algorithms like ADAGRAD, Adam can escape saddles - but no guarantees)

The authors introduce a "saddle-free Newton method" that can escape saddle points

Non-convexity is good?

LeCun 2007: Convexity is great for learning, but it can lead to unsuitable architectures [16]

Collobert et al. 2006: non-convex loss function can actually benefit SVMs! [17] Empirical evidence, of course

Still, evidence to focus on **proper representation** over **inference guarantees**

Reason 5

Reason 5: non-convexity is not that bad / good architectures are more important

For the unconvinced...

A few words from Yann LeCun

Who is afraid of non-convex loss functions?

Yann LeCun

The Courant Institute of Mathematical Sciences

New York University

Theoretical Guarantees are overrated

- When Empirical Evidence suggests a fact for which we don't have theoretical guarantees, it just means the theory is inadequate.
- When empirical evidence and theory disagree, the theory is wrong.
- Let's not be afraid of methods for which we have no theoretical guarantee, particularly if they have been shown to work well
- But, let's aggressively look to those theoretical guarantees.
- We should use our theoretical understanding to expand our creativity, not to restrict it.

Thank you!

References I

[1] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. "ImageNet Classification with Deep Convolutional Neural Networks". *Advances in Neural Information Processing Systems 25*. Ed. by F. Pereira et al. Curran Associates, Inc., 2012, pp. 1097–1105. URL: http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf.

[2] *A Year in Computer Vision*. URL: http://www.themtank.org/a-year-in-computer-vision.

[3] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. "A neural algorithm of artistic style". *arXiv preprint arXiv:1508.06576* (2015).

[4] URL: https://xkcd.com/1838/.

References II

[5] Maurice Peemen, Bart Mesman, and Henk Corporaal. "Speed sign detection and recognition by convolutional neural networks".

[6] URL: http://i-systems.github.io/HSE545/machine%20learning%20all/16%20Deep%20learning/CNN.html.

[7] Ali Sharif Razavian et al. "CNN features off-the-shelf: an astounding baseline for recognition". *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*. 2014, pp. 806–813.

[8] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. "Multilayer feedforward networks are universal approximators". *Neural Networks* 2.5 (1989), pp. 359 –366. ISSN: 0893-6080. DOI: https://doi.org/10.1016/0893-6080(89)90020-8. URL: http://www.sciencedirect.com/science/article/pii/0893608089900208.

References III

[9] Honglak Lee et al. "Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations". *Proceedings of the 26th annual international conference on machine learning*. ACM. 2009, pp. 609–616.

[10] Matthew D Zeiler and Rob Fergus. "Visualizing and understanding convolutional networks". *European conference on computer vision*. Springer. 2014, pp. 818–833.

[11] Geoffrey E Hinton. "Distributed representations". (1984).

[12] URL: http://scikit-learn.org/stable/auto_examples/cluster/plot_kmeans_digits.html#sphx-glr-auto-examples-cluster-plot-kmeans-digits-py/.

References IV

[13] Guido F Montufar et al. "On the number of linear regions of deep neural networks". *Advances in neural information processing systems*. 2014, pp. 2924–2932.

[14] Anna Choromanska et al. "The loss surfaces of multilayer networks". *Artificial Intelligence and Statistics*. 2015, pp. 192–204.

[15] Yann N Dauphin et al. "Identifying and attacking the saddle point problem in high-dimensional non-convex optimization". *Advances in neural information processing systems*. 2014, pp. 2933–2941.

[16] Yann Lecun. "Who is afraid of nonconvex loss functions?" (2007).

[17] Ronan Collobert et al. "Trading convexity for scalability". *Proceedings of the 23rd international conference on Machine learning*. ACM. 2006, pp. 201–208.