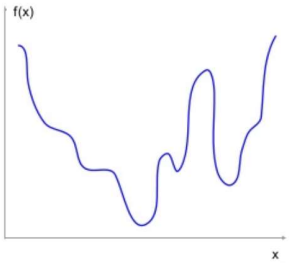
Visualizing and Understanding Deep Neural Networks

Kai-Lung Hua (花凱龍)

Some insights in the Loss Landscape

Training neural networks requires minimizing a high-dimensional non-convex loss function



Example of a non-convex function

This means that, in theory, you can get stuck in a local minima

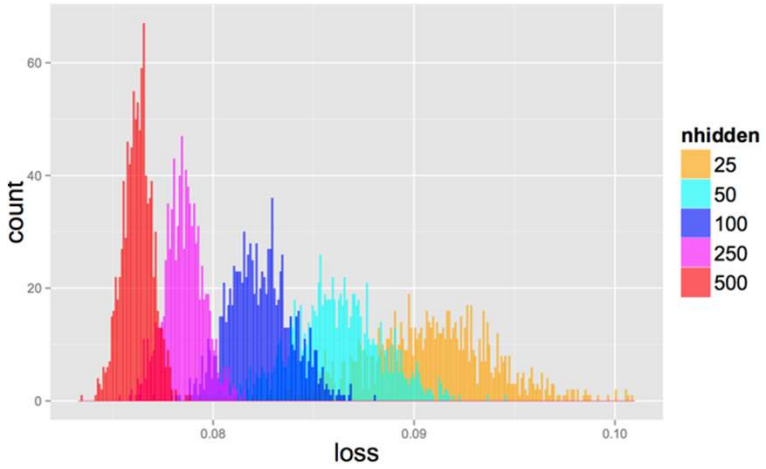
Some insights in the Loss Landscape

This implies that we cannot train these networks using local search procedures like stochastic gradient descent.

But how come in it works extremely well in practice!?
Why is it that local minima seems to be not such an issue!?

3 of 97

The Loss Surfaces of Multilayer Networks
Choromanska, A., Henaff, M., Mathieu, M., Arous, G. B., & LeCun, Y. (2015)

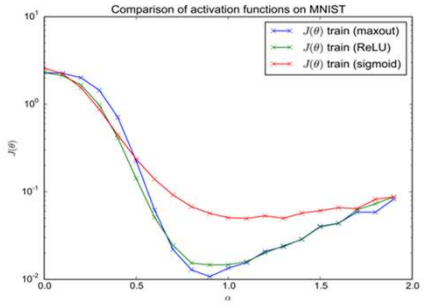


As network size grows you have **many** more local minima, but they all get better and their band shrinks.

4 of 97

Qualitatively Characterizing Neural Network Optimization Problems
Goodfellow, I. J., Vinyals, O., & Saxe, A. M. (2014)

Loss Function Experiment:
Start at some $\boldsymbol{w_0}$, train the network to get $\boldsymbol{w_T}$ with a much better loss. Now linearly interpolate $\boldsymbol{w}$ from $\boldsymbol{w_0}$ to $\boldsymbol{w_T}$, $\big((1-\alpha)w_0 + \alpha w_T\big)$. What does the loss look like along the way?



If we only knew the direction we could solve the entire neural network with a single line search.

5 of 97

Qualitatively Characterizing Neural Network Optimization Problems
Goodfellow, I. J., Vinyals, O., & Saxe, A. M. (2014)

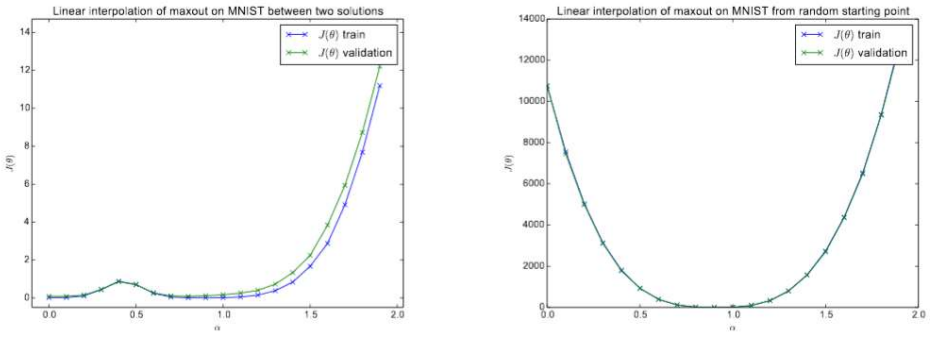
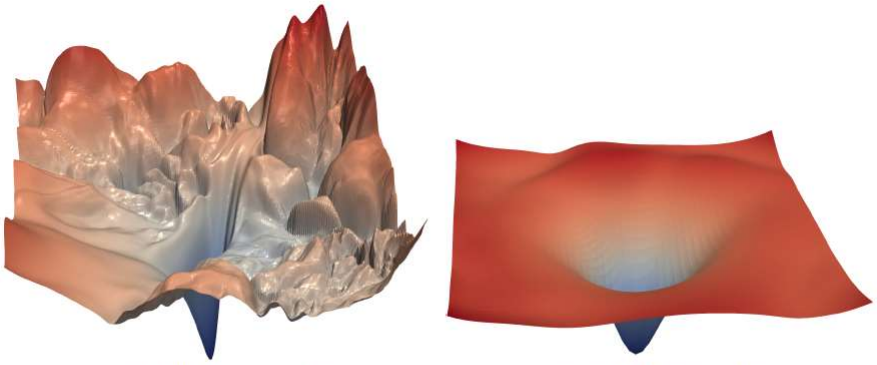


Figure 5: Here we use linear interpolation to search for local minima. Left) By interpolating between two different SGD solutions, we show that each solution is a different local minimum within this 1-D subspace. Right) If we interpolate between a random point in space and an SGD solution, we find no local minima besides the SGD solution, suggesting that local minima are rare.

6 of 97

Visualizing the Loss Landscape of Neural Nets
Li, H., Xu, Z., Taylor, G., & Goldstein, T. (2017)



(a) without skip connections (b) with skip connections

Figure 1: The loss surfaces of ResNet-56 with/without skip connections. The vertical axis is logarithmic to show dynamic range. The proposed filter normalization scheme is used to enable comparisons of sharpness/flatness between the two figures.

7 of 97

Understanding Learned Models

8 of 97

Understanding Neural Networks Through Deep Visualization
Yosinski, J., Clune, J., Nguyen, A., Fuchs, T., & Lipson, H. (2015)
http://yosinski.com/deepvis

9 of 97

Slide Credit: Fei-Fei Li, Andrej Karpathy, Justin Johnson

What do Deep Neural Networks Learn?

Some approaches / experiments people have done:

- Visualize patches that maximally activate neurons
- Visualize the weights
- Visualize the representation space (e.g. with t-SNE)
- Occlusion Experiments
- Deconvolution approaches (single backward pass)
- Optimization approaches

10 of 97

Slide Credit: Fei-Fei Li, Andrej Karpathy, Justin Johnson

Visualize patches that maximally activate neurons

Rich feature hierarchies for accurate object detection and semantic segmentation
Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2014)

one-stream AlexNet

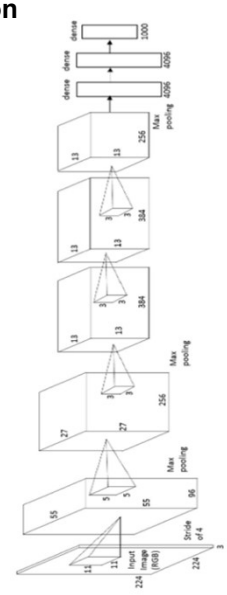


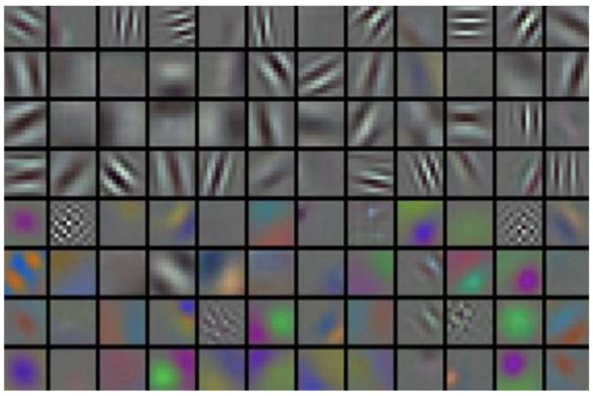
Figure 4: Top regions for six pool$_5$ units. Receptive fields and activation values are drawn in white. Some units are aligned to concepts, such as people (row 1) or text (4). Other units capture texture and material properties, such as dot arrays (2) and specular reflections (6).

Slide Credit: Fei-Fei Li, Andrej Karpathy, Justin Johnson

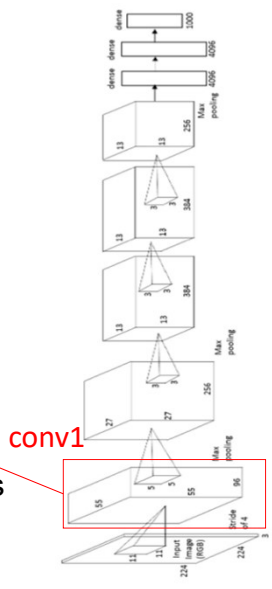
Visualizing the weights

13 of 97

Visualize the filters/kernels (raw weights)



AlexNet



We get these gabor-like filters! Which is consistent with past theories going as far back as Hubel & Wiesel's experiments on a cat's visual cortex.

Note: This approach is only interpretable on the first layer =(

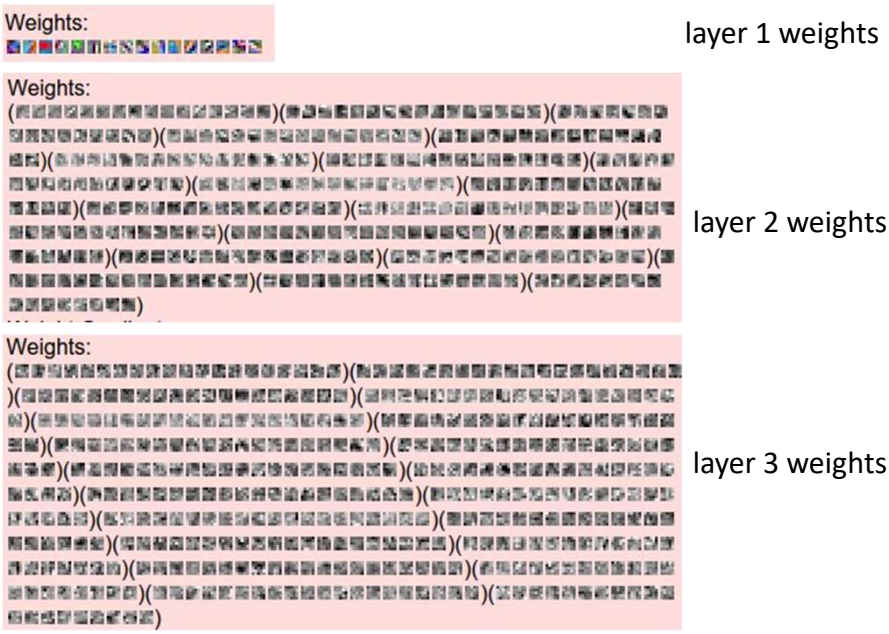
14 of 97

Slide Credit: Fei-Fei Li, Andrej Karpathy, Justin Johnson

Visualize the filters/kernels (raw weights)

you can still do it for higher layers, it's just not that interesting

(these are taken from ConvNetJS CIFAR-10 demo)



layer 1 weights

layer 2 weights

layer 3 weights

15 of 97

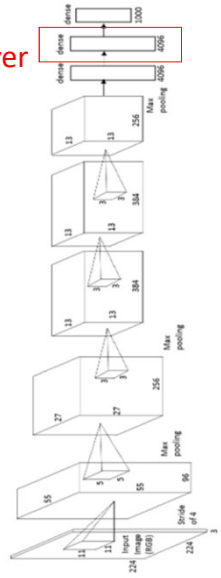
Slide Credit: Fei-Fei Li, Andrej Karpathy, Justin Johnson

Visualizing the representation space

16 of 97

Visualizing Representations

fc7 layer



4096-dimensional "code" for an image
(layer immediately before the classifier)

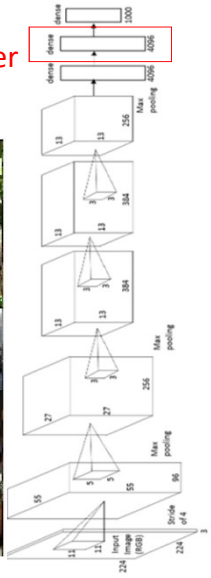
can collect the code for many images

17 of 97

Slide Credit: Fei-Fei Li, Andrej Karpathy, Justin Johnson

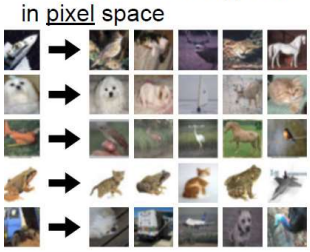
Visualizing Representations

fc7 layer



Test image L2 Nearest neighbors in feature space

Recall: Nearest neighbors in pixel space



18 of 97

Slide Credit: Fei-Fei Li, Andrej Karpathy, Justin Johnson

Visualizing Representations

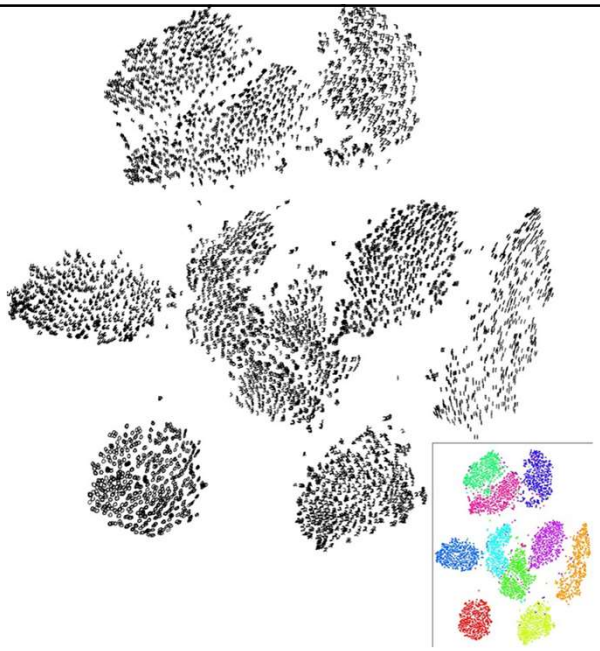
t-SNE visualization

[van der Maaten & Hinton]
t-distributed Stochastic Neighbor Embedding

Embed high-dimensional points so that locally, pairwise distances are conserved

i.e. similar things end up in similar places. dissimilar things end up wherever

Right: Example embedding of MNIST digit images (0-9) in 2D

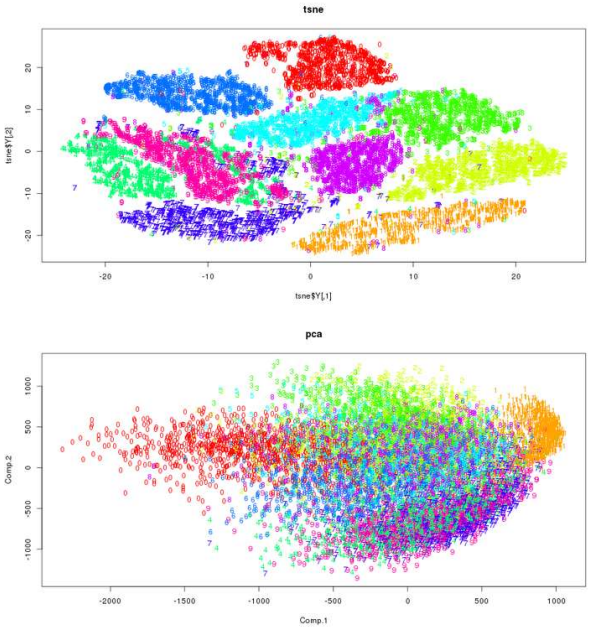


19 of 97

Slide Credit: Fei-Fei Li, Andrej Karpathy, Justin Johnson

t-SNE Embeddings

Generally does a better job of separating classes compared to PCA:



20 of 97

Slide Credit: Fei-Fei Li, Andrej Karpathy, Justin Johnson

t-SNE Embeddings

A t-SNE embedding puts similar items close to each other in 2 or 3-D.
It is often useful to cluster the data in the embedding space.
Since the clusters are not "compact" (sphere-like), its best to use a density-based clustering like DBSCAN or HDBSCAN

Density-based spatial clustering of applications with noise (DBSCAN)
Hierarchical DBSCAN (HDBSCAN)

21 of 97

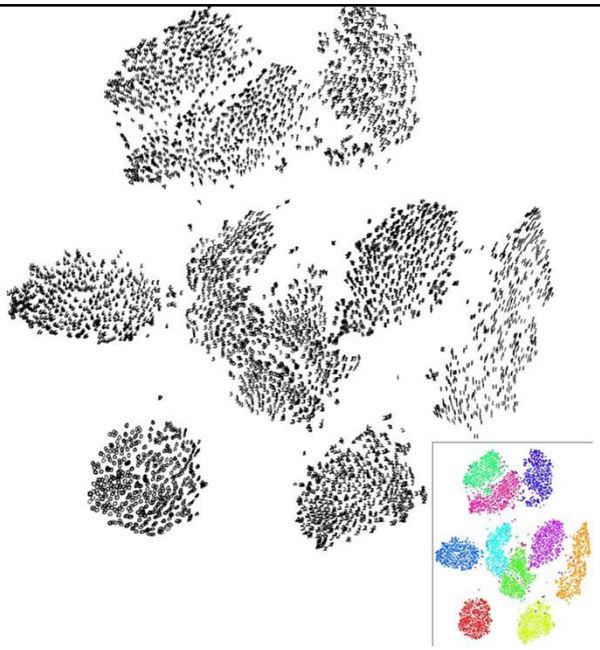
Slide Credit: Fei-Fei Li, Andrej Karpathy, Justin Johnson

t-SNE Implementation

Fast Stochastic Neighbor Embedding
[van der Maaten 2013]

Aside: t-SNE is an iterative algorithm and expensive $O(N\text{^}2)$ for datasets with N points.

Its common to use an approximation (Barnes-Hut-SNE) which is $O(N \log N)$ and which can manage millions of points.



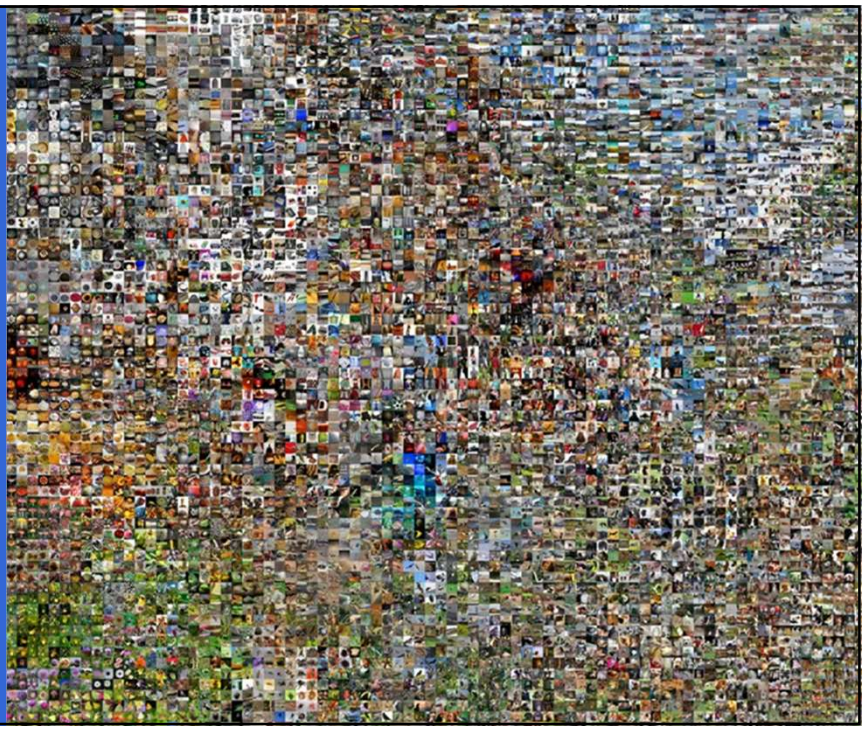
22 of 97

Slide Credit: Fei-Fei Li, Andrej Karpathy, Justin Johnson

t-SNE visualization:

two images are placed nearby if their CNN codes are close. See more:

http://cs.stanford.edu/people/karpathy/cnnembed/



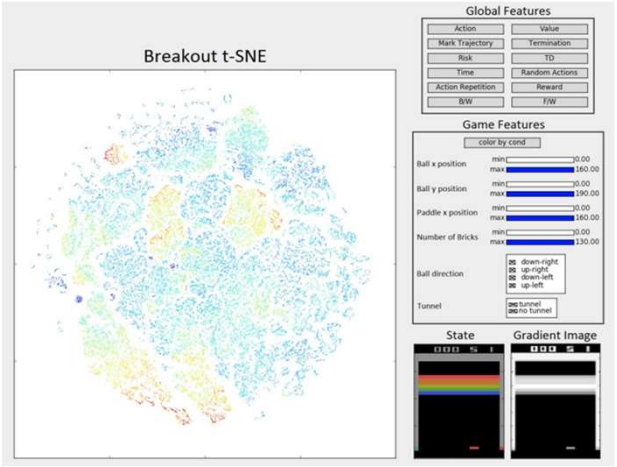
Graying the black box: Understanding DQNs
Zahavy, T., Ben-Zrihem, N., & Mannor, S. (2016)

Playing Atari with Deep Reinforcement Learning, Mnih et al. 2013

24 of 97

Slide Credit: Fei-Fei Li, Andrej Karpathy, Justin Johnson

Graying the black box: Understanding DQNs
Zahavy, T., Ben-Zrihem, N., & Mannor, S. (2016)

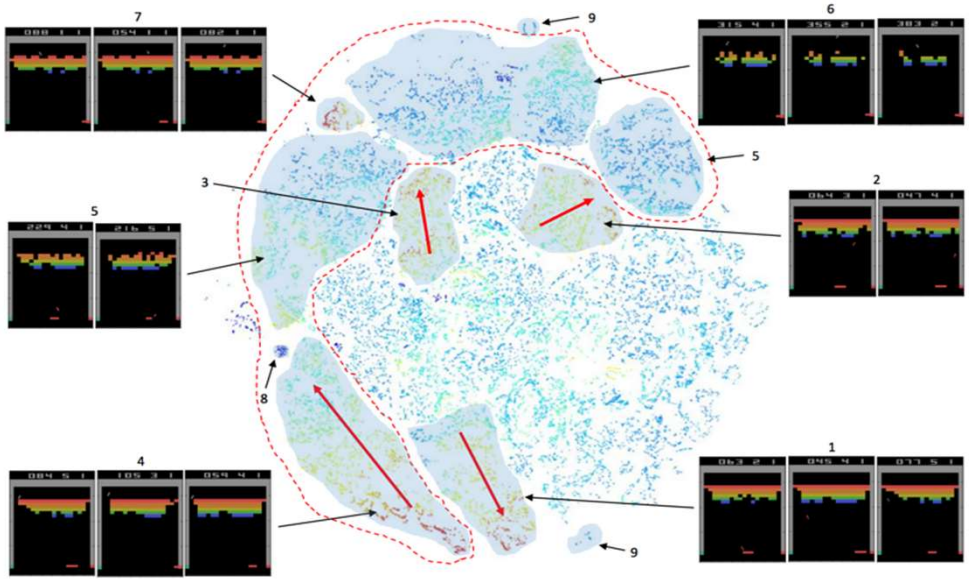


The embedding shows clustering of the *activations of the agent's policy network* for different frames of breakout.

25 of 97

Slide Credit: Fei-Fei Li, Andrej Karpathy, Justin Johnson

Visualizing the network's "strategy"

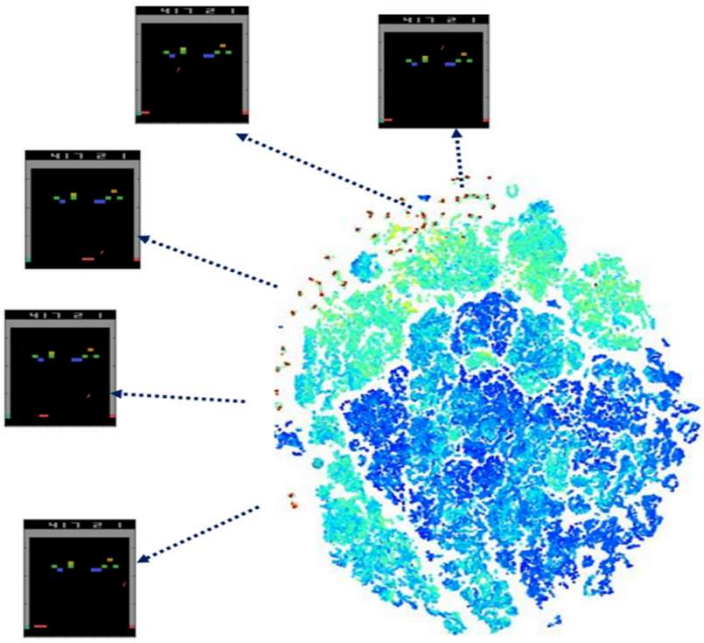


26 of 97

Slide Credit: Fei-Fei Li, Andrej Karpathy, Justin Johnson

Identify “policy bugs”: state clusters where the agent spends a very long time

(e.g. failing to hit the last few blocks over and over again for a very long time)



27 of 97

Slide Credit: Fei-Fei Li, Andrej Karpathy, Justin Johnson

Occlusion Experiments

28 of 97

Visualizing and understanding convolutional networks

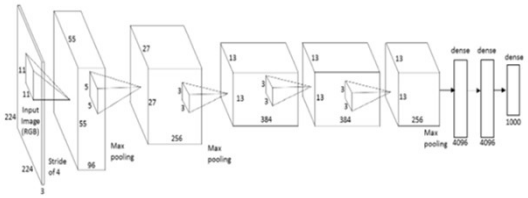
Zeiler, M. D., & Fergus, R. (2014)

(as a function of the position of the gray square in the original image)

Slide Credit: Fei-Fei Li, Andrej Karpathy, Justin Johnson

Deconvolution Approaches

1. Feed image into net

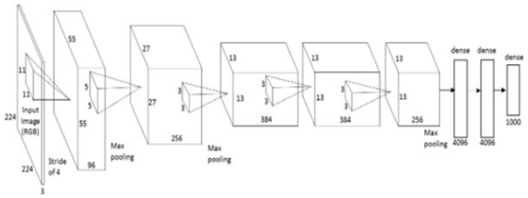


Q: how can we compute the gradient of any arbitrary neuron in the network w.r.t. the image?

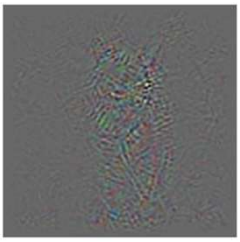
31 of 97

Slide Credit: Fei-Fei Li, Andrej Karpathy, Justin Johnson

1. Feed image into net



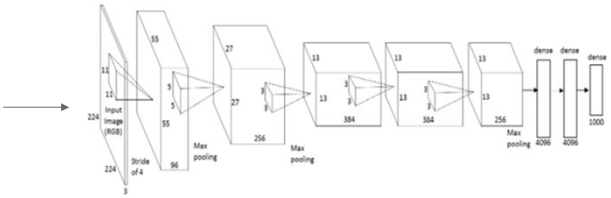
2. Pick a layer, set the gradient there to be all zero except for one 1 for some neuron of interest
3. Backprop to image:



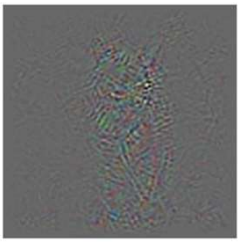
32 of 97

Slide Credit: Fei-Fei Li, Andrej Karpathy, Justin Johnson

1. Feed image into net



2. Pick a layer, set the gradient there to be all zero except for one 1 for some neuron of interest
3. Backprop to image:



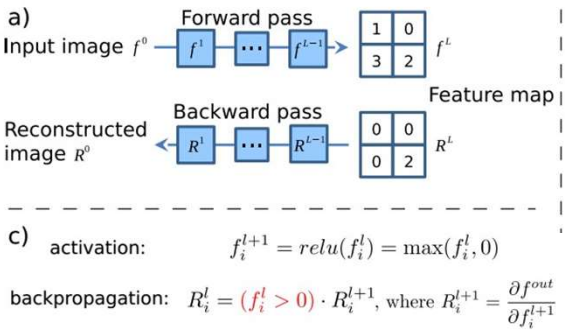
"Guided backpropagation:" instead

33 of 97

Slide Credit: Fei-Fei Li, Andrej Karpathy, Justin Johnson

Deconv approaches

[Visualizing and Understanding Convolutional Networks, Zeiler and Fergus 2014]
[Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps, Simonyan et al., 2014]
[Striving for Simplicity: The all convolutional net, Springenberg, Dosovitskiy, et al., 2015]



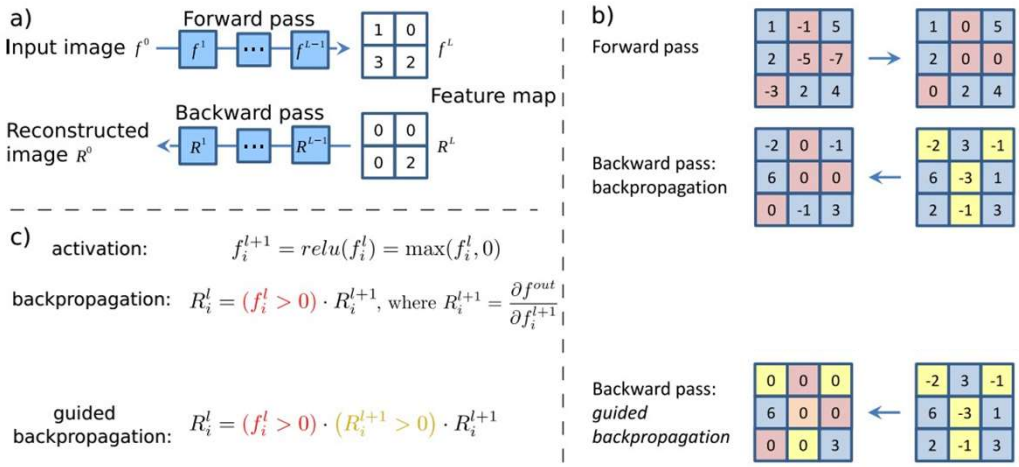
Backward pass for a ReLU (will be changed in Guided Backprop)

34 of 97

Slide Credit: Fei-Fei Li, Andrej Karpathy, Justin Johnson

Deconv approaches

[Visualizing and Understanding Convolutional Networks, Zeiler and Fergus 2014]
[Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps, Simonyan et al., 2014]
[Striving for Simplicity: The all convolutional net, Springenberg, Dosovitskiy, et al., 2015]



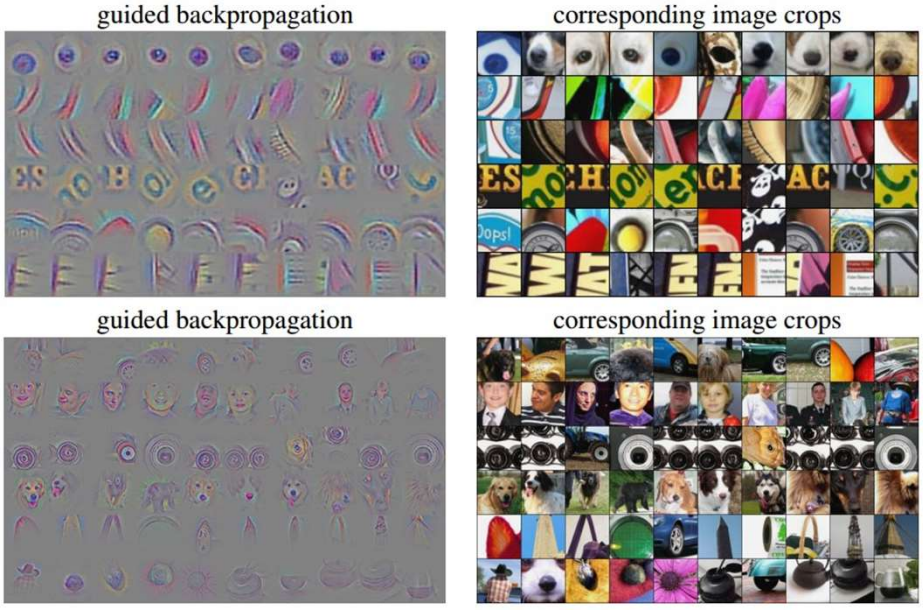
35 of 97

Slide Credit: Fei-Fei Li, Andrej Karpathy, Justin Johnson

Visualization of patterns learned by the layer **conv6** (top) and layer **conv9** (bottom) of the network trained on ImageNet.

Each row corresponds to one filter.

The visualization using "guided backpropagation" is based on the top 10 image patches activating this filter taken from the ImageNet dataset.



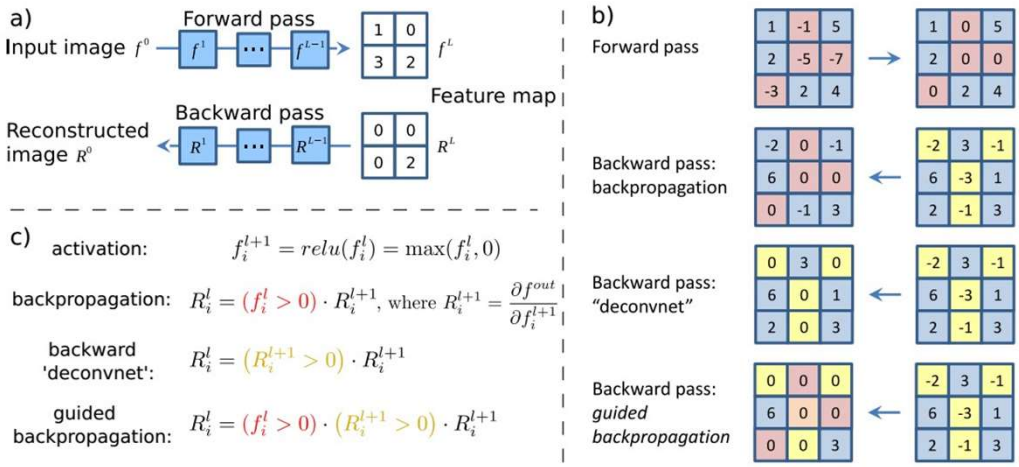
[Striving for Simplicity: The all convolutional net, Springenberg, Dosovitskiy, et al., 2015]

36 of 97

Slide Credit: Fei-Fei Li, Andrej Karpathy, Justin Johnson

Deconv approaches

[Visualizing and Understanding Convolutional Networks, Zeiler and Fergus 2014]
[Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps, Simonyan et al., 2014]
[Striving for Simplicity: The all convolutional net, Springenberg, Dosovitskiy, et al., 2015]

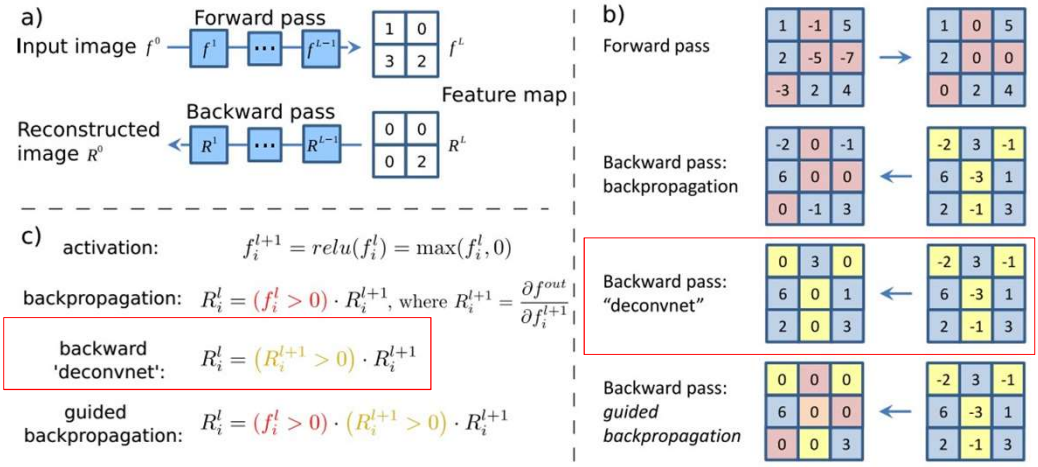


37 of 97

Slide Credit: Fei-Fei Li, Andrej Karpathy, Justin Johnson

Deconv approaches

[Visualizing and Understanding Convolutional Networks, Zeiler and Fergus 2014]
[Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps, Simonyan et al., 2014]
[Striving for Simplicity: The all convolutional net, Springenberg, Dosovitskiy, et al., 2015]

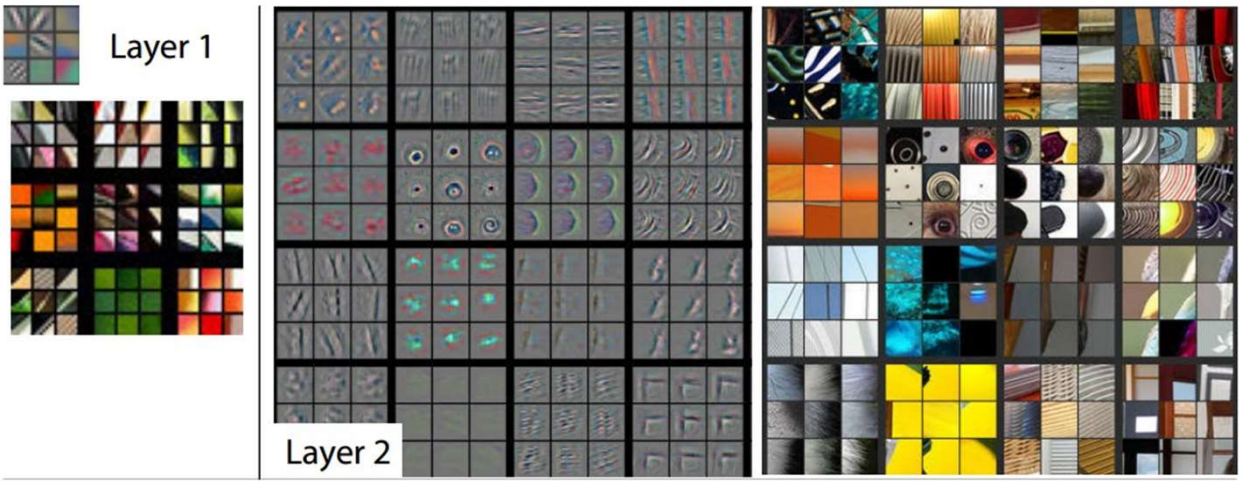


bit weird

38 of 97

Slide Credit: Fei-Fei Li, Andrej Karpathy, Justin Johnson

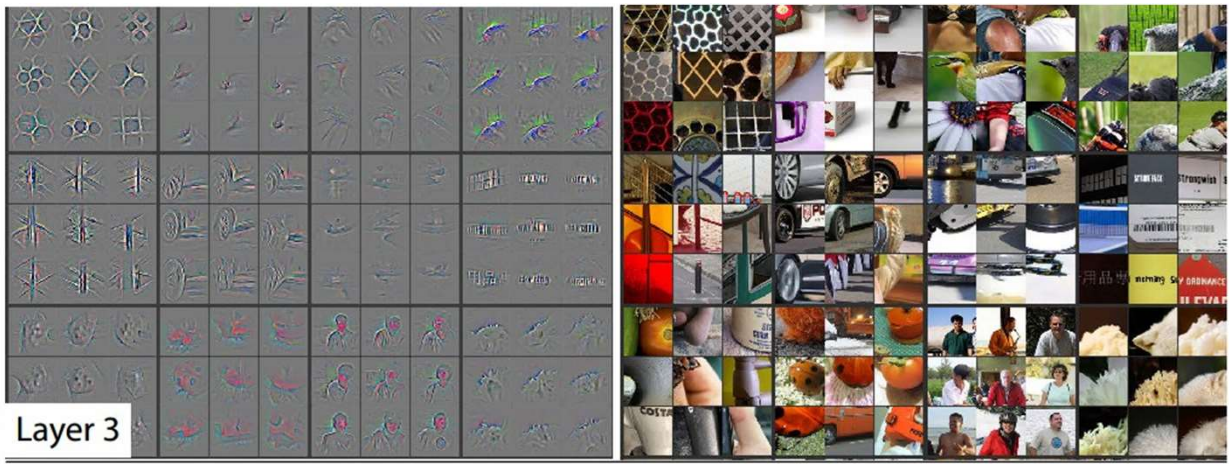
Visualizing and understanding convolutional networks
Zeiler, M. D., & Fergus, R. (2014)



39 of 97

Slide Credit: Fei-Fei Li, Andrej Karpathy, Justin Johnson

Visualizing and understanding convolutional networks
Zeiler, M. D., & Fergus, R. (2014)

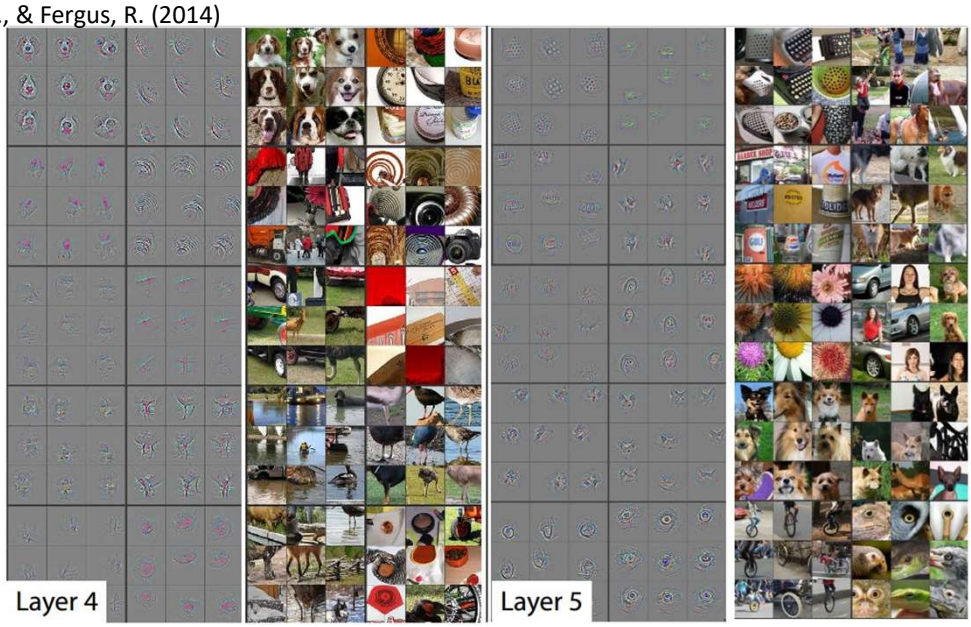


40 of 97

Slide Credit: Fei-Fei Li, Andrej Karpathy, Justin Johnson

Visualizing and understanding convolutional networks

Zeiler, M. D., & Fergus, R. (2014)



41 of 97

Slide Credit: Fei-Fei Li, Andrej Karpathy, Justin Johnson

Comparison

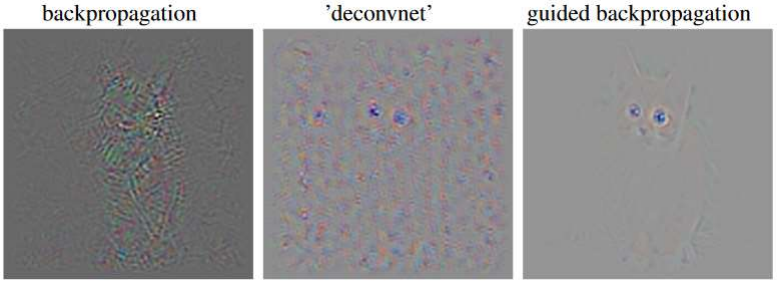


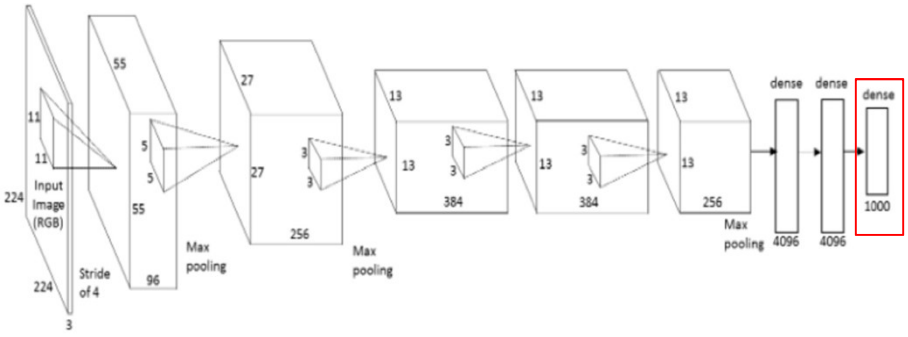
Figure 5: Visualization of descriptive image regions with different methods from the single largest activation in the pre-softmax layer global_pool of the network trained on ImageNet.

42 of 97

Optimization approaches

Deep Inside Convolutional Networks: Visualizing Image Classification Models and Saliency Maps

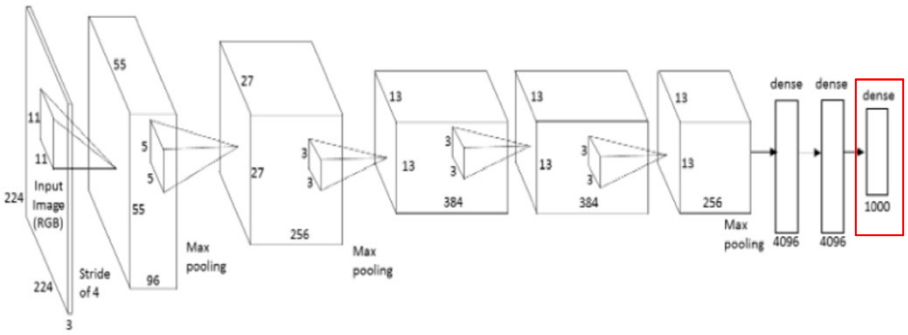
Simonyan, K., Vedaldi, A., & Zisserman, A. (2013)



Can we find an image that maximizes some class score?

Slide Credit: Fei-Fei Li, Andrej Karpathy, Justin Johnson

Deep Inside Convolutional Networks: Visualizing Image Classification Models and Saliency Maps
Simonyan, K., Vedaldi, A., & Zisserman, A. (2013)



i.e. generate an image that maximizes the class score

45 of 97

Slide Credit: Fei-Fei Li, Andrej Karpathy, Justin Johnson

Deep Inside Convolutional Networks: Visualizing Image Classification Models and Saliency Maps
Simonyan, K., Vedaldi, A., & Zisserman, A. (2013)

$$\underset{I}{\operatorname{argmax}} S_c(I) - \lambda \|I\|_2^2$$

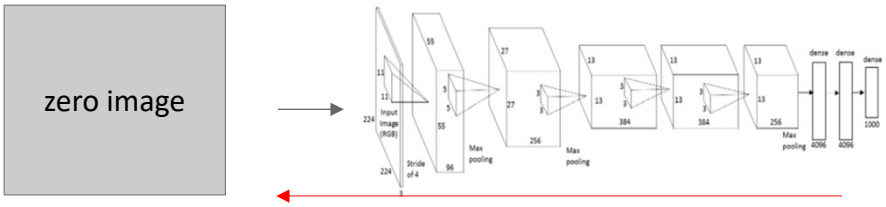
$S_c(I)$ - Score for class c (before SoftMax)
$\lambda \|I\|_2^2$ - L_2 Regularization

46 of 97

Slide Credit: Fei-Fei Li, Andrej Karpathy, Justin Johnson

Optimization to Image

1. Initialize image to zeros



2. Forward image to compute class scores
3. Set the gradient of the scores vector to be [0,0,....1,....,0]
4. Backprop to get the gradient of neuron value with respect to image pixels
5. Make a small update to the image

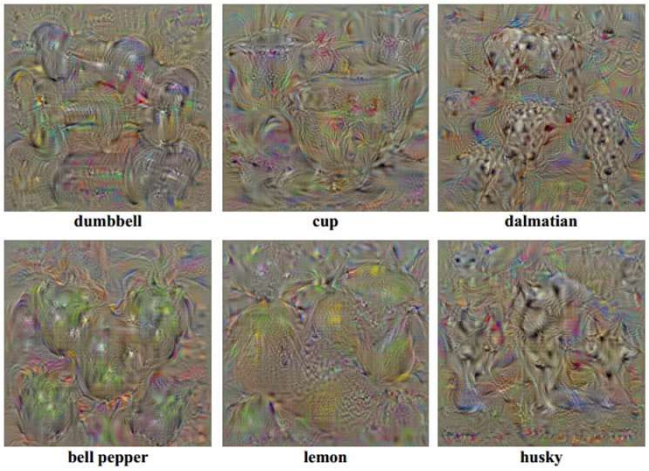
47 of 97

Slide Credit: Fei-Fei Li, Andrej Karpathy, Justin Johnson

Deep Inside Convolutional Networks: Visualizing Image Classification Models and Saliency Maps

Simonyan, K., Vedaldi, A., & Zisserman, A. (2013)

1. Generate images that maximize some class score:

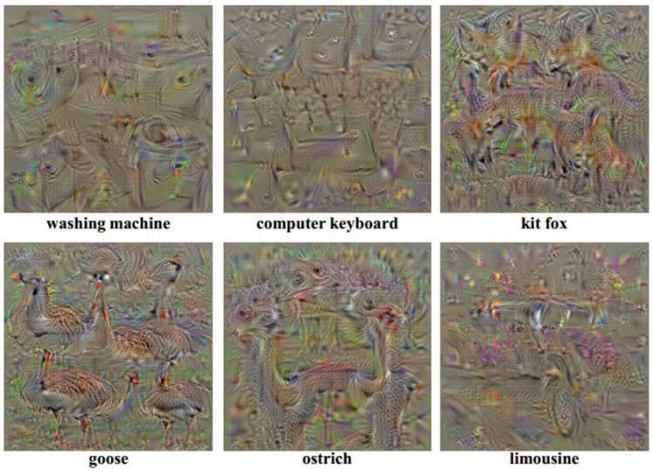


48 of 97

Slide Credit: Fei-Fei Li, Andrej Karpathy, Justin Johnson

Deep Inside Convolutional Networks: Visualizing Image Classification Models and Saliency Maps
Simonyan, K., Vedaldi, A., & Zisserman, A. (2013)

1. Generate images that maximize some class score:



49 of 97

Slide Credit: Fei-Fei Li, Andrej Karpathy, Justin Johnson

Understanding neural networks through deep visualization
Yosinski, J., Clune, J., Nguyen, A., Fuchs, T., & Lipson, H. (2015)

$$\underset{I}{\operatorname{argmax}} S_c(I) - \lambda \|I\|_2^2$$

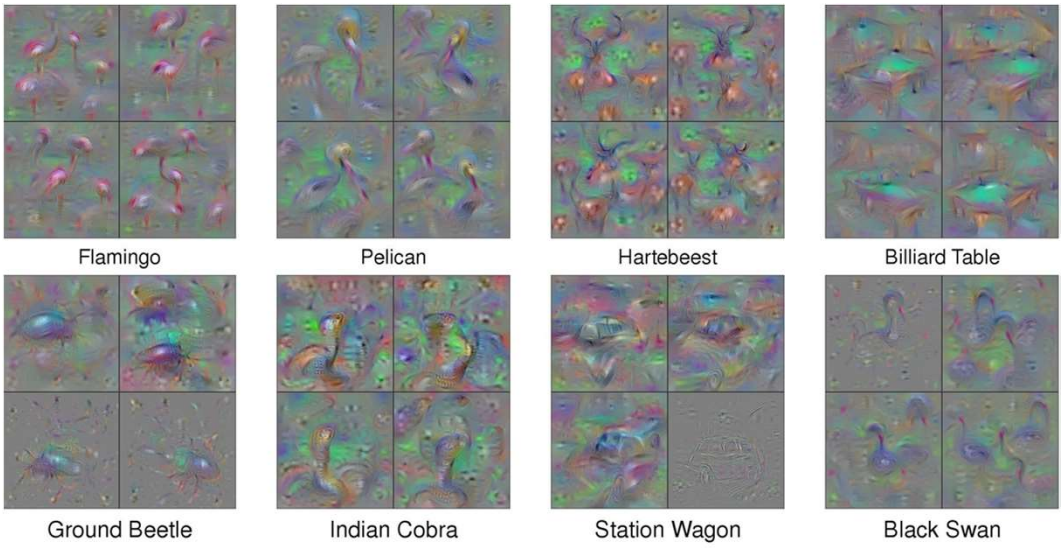
Proposed other types of regularizers beyond L_2

- Penalize high frequencies: Apply gaussian blur.
- Clip (zero) pixels with small norm using a threshold.
- Clip pixels with small contribution. Do this by ***ablation***: set pixel activation to zero, measure change in output. Zero pixels if change is small.

50 of 97

Slide Credit: Fei-Fei Li, Andrej Karpathy, Justin Johnson

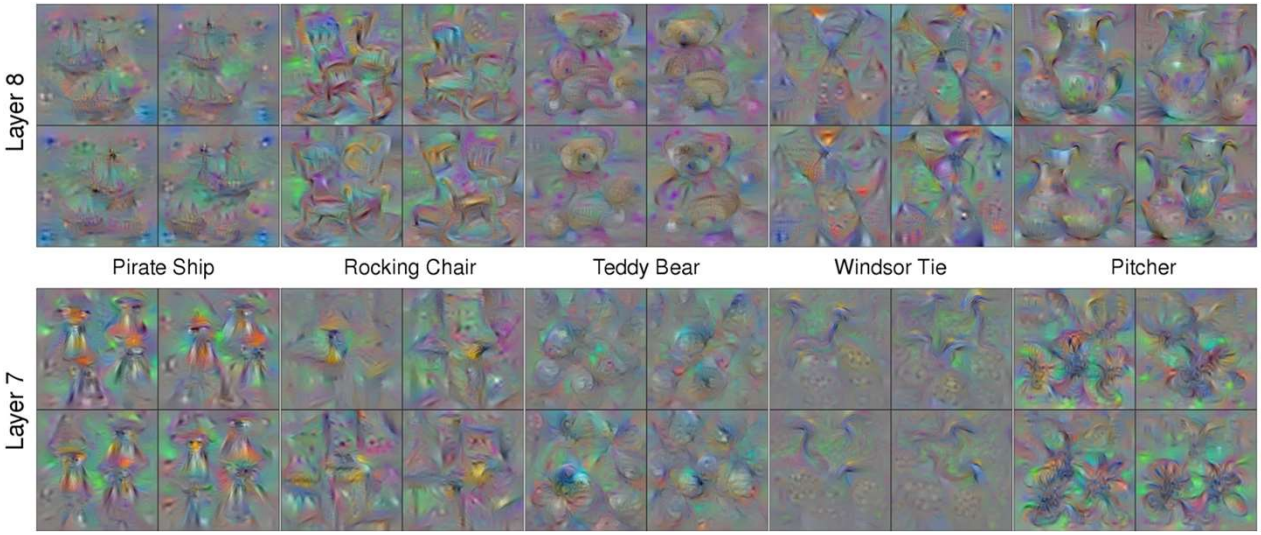
Understanding neural networks through deep visualization
Yosinski, J., Clune, J., Nguyen, A., Fuchs, T., & Lipson, H. (2015)



51 of 97

Slide Credit: Fei-Fei Li, Andrej Karpathy, Justin Johnson

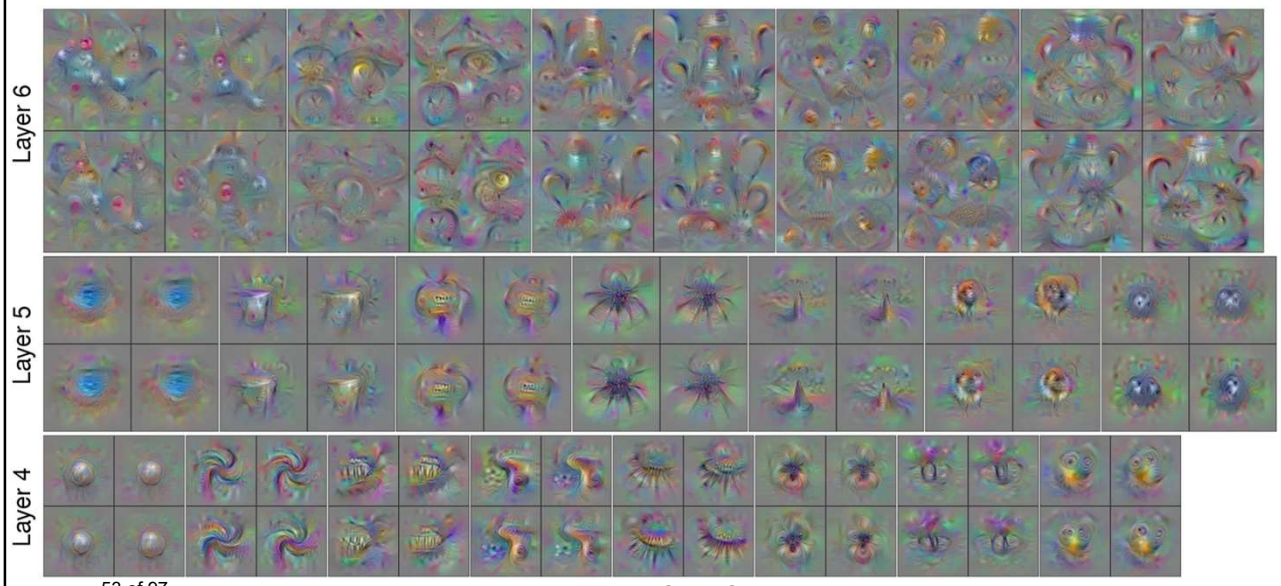
Understanding neural networks through deep visualization
Yosinski, J., Clune, J., Nguyen, A., Fuchs, T., & Lipson, H. (2015)



52 of 97

Slide Credit: Fei-Fei Li, Andrej Karpathy, Justin Johnson

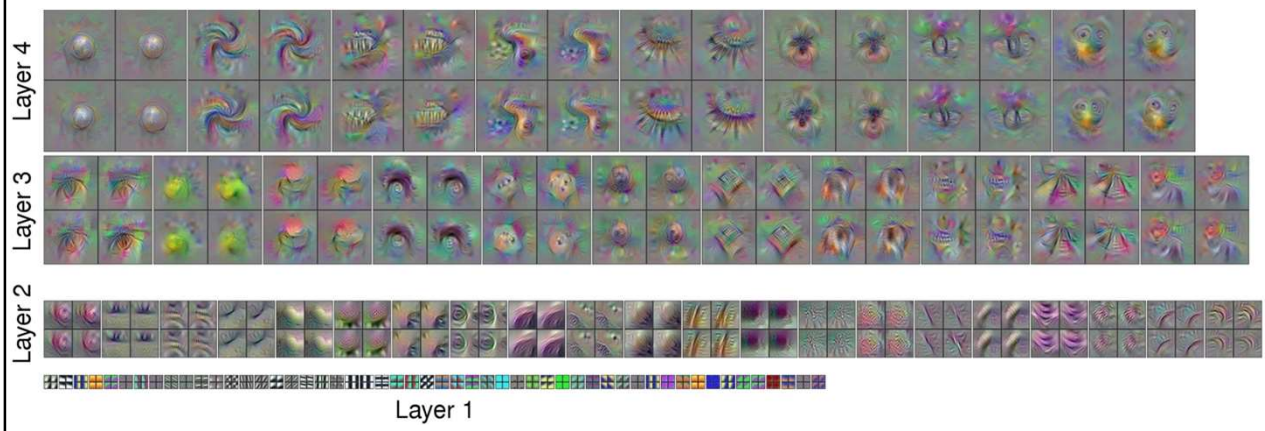
Understanding neural networks through deep visualization
Yosinski, J., Clune, J., Nguyen, A., Fuchs, T., & Lipson, H. (2015)



53 of 97

Slide Credit: Fei-Fei Li, Andrej Karpathy, Justin Johnson

Understanding neural networks through deep visualization
Yosinski, J., Clune, J., Nguyen, A., Fuchs, T., & Lipson, H. (2015)



54 of 97

Slide Credit: Fei-Fei Li, Andrej Karpathy, Justin Johnson

Multifaceted Feature Visualization: Uncovering the Different Types of Features Learned By Each Neuron in Deep Neural Networks
Nguyen, A., Yosinski, J., & Clune, J. (2016)

Reconstructions of multiple feature types (facets) recognized by the same "grocery store" neuron

Change initialization: project the training set images that maximally activate a neuron into a low-dimensional space (here, a 2D space via t-SNE), cluster the images via k-means, and average the n (here, 15) closest images to each cluster centroid to produce the initial image.

55 of 97

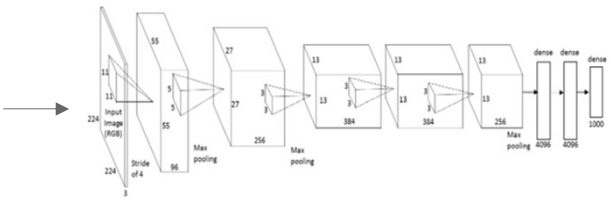
Slide Credit: Fei-Fei Li, Andrej Karpathy, Justin Johnson

Multifaceted Feature Visualization: Uncovering the Different Types of Features Learned By Each Neuron in Deep Neural Networks
Nguyen, A., Yosinski, J., & Clune, J. (2016)

pretty!

Deep Inside Convolutional Networks: Visualizing Image Classification Models and Saliency Maps
Simonyan, K., Vedaldi, A., & Zisserman, A. (2013)

1. Feed image into net



Q: Which pixels are most important for the class output on a particular image?

57 of 97

Slide Credit: Fei-Fei Li, Andrej Karpathy, Justin Johnson

Deep Inside Convolutional Networks: Visualizing Image Classification Models and Saliency Maps
Simonyan, K., Vedaldi, A., & Zisserman, A. (2013)

2. Visualize the Data gradient:

(note that the gradient on data has three channels. Here they visualize M, s.t.:

$$M_{i,j} = \max_c |w_h(i,j,c)|$$

(at each pixel take abs val, and max over channels)

$M = ?$

58 of 97

Slide Credit: Fei-Fei Li, Andrej Karpathy, Justin Johnson

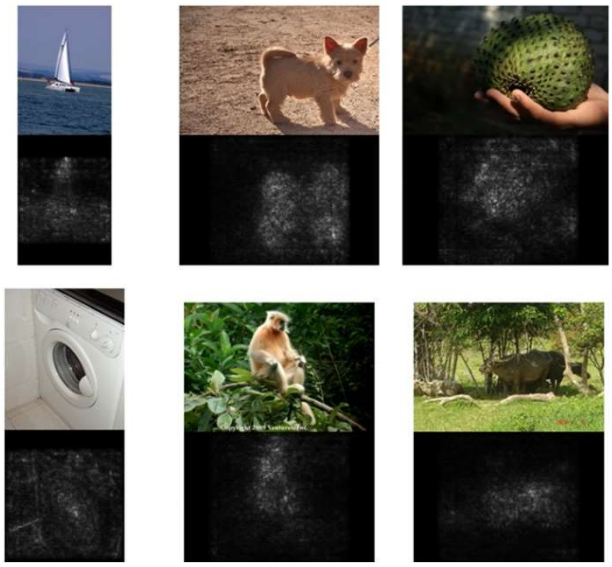
Deep Inside Convolutional Networks: Visualizing Image Classification Models and Saliency Maps
Simonyan, K., Vedaldi, A., & Zisserman, A. (2013)

2. Visualize the Data gradient:

(note that the gradient on data has three channels. Here they visualize M, s.t.:

$$M_{i,j} = \max_c |w_h(i,j,c)|$$

(at each pixel take abs val, and max over channels)



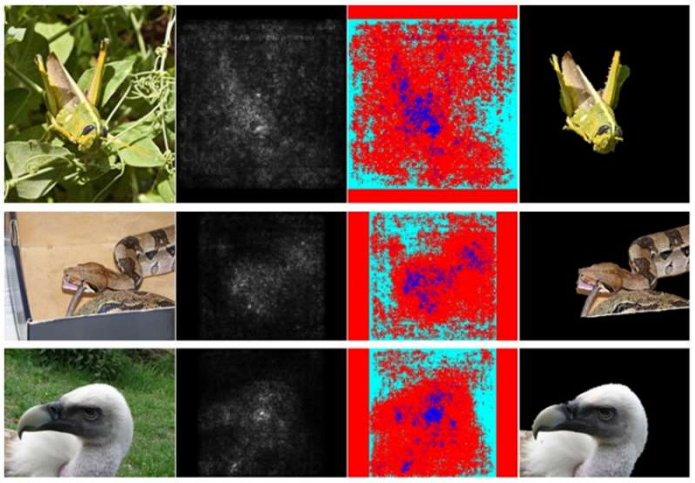
59 of 97

Slide Credit: Fei-Fei Li, Andrej Karpathy, Justin Johnson

Deep Inside Convolutional Networks: Visualizing Image Classification Models and Saliency Maps
Simonyan, K., Vedaldi, A., & Zisserman, A. (2013)

Use **grabcut** for segmentation:
- Use a box to select the object
- Compute the max class score
- Construct an saliency map for the class
- Segment the saliency map

Finds the activating object in the image

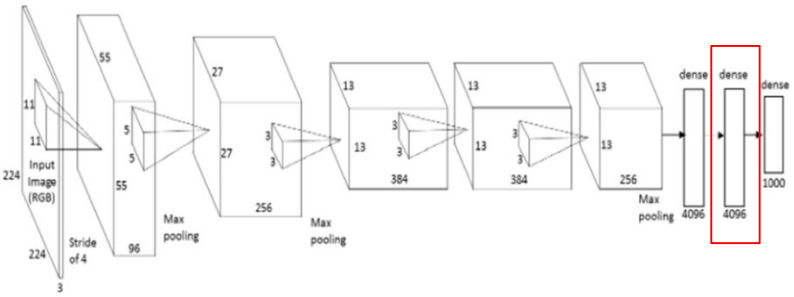


60 of 97

Slide Credit: Fei-Fei Li, Andrej Karpathy, Justin Johnson

Understanding deep image representations by inverting them
Mahendran, A., & Vedaldi, A. (2015)

Given an image feature code, is it possible to reconstruct the original image?



61 of 97

Slide Credit: Fei-Fei Li, Andrej Karpathy, Justin Johnson

Understanding deep image representations by inverting them
Mahendran, A., & Vedaldi, A. (2015)

Find an image such that:
- Its code is similar to a given code
- It "looks natural" (image prior regularization)

$$\mathbf{x}^* = \underset{\mathbf{x} \in \mathbb{R}^{H \times W \times C}}{\operatorname{argmin}} \ \ell(\Phi(\mathbf{x}), \Phi_0) + \lambda \mathcal{R}(\mathbf{x})$$

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

62 of 97

Slide Credit: Fei-Fei Li, Andrej Karpathy, Justin Johnson

Understanding deep image representations by inverting them
Mahendran, A., & Vedaldi, A. (2015)

Reconstructions from the 1000 log probabilities for ImageNet (ILSVRC) classes

original image

63 of 97

Slide Credit: Fei-Fei Li, Andrej Karpathy, Justin Johnson

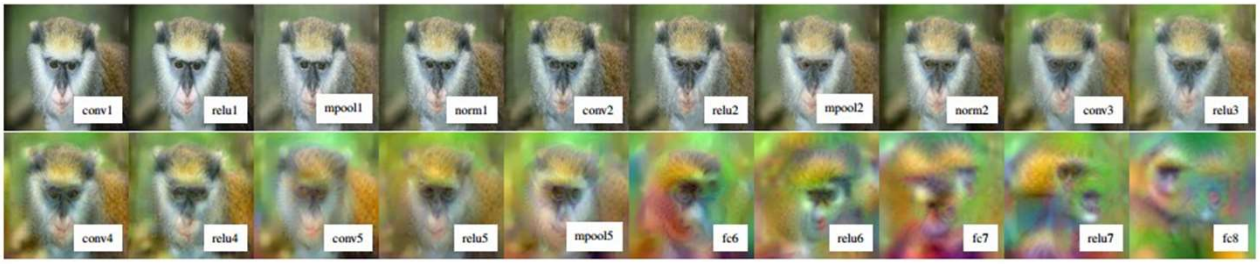
Understanding deep image representations by inverting them
Mahendran, A., & Vedaldi, A. (2015)

Reconstructions from the representation after last pooling layer (immediately before the first Fully Connected layer)

64 of 97

Slide Credit: Fei-Fei Li, Andrej Karpathy, Justin Johnson

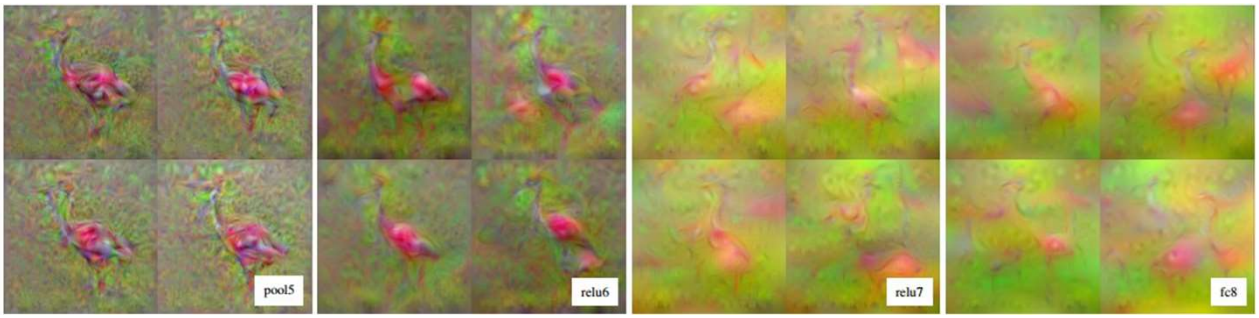
Reconstructions from intermediate layers



65 of 97

Slide Credit: Fei-Fei Li, Andrej Karpathy, Justin Johnson

Multiple reconstructions. Images in quadrants all “look” the same to the CNN (same code)



66 of 97

Slide Credit: Fei-Fei Li, Andrej Karpathy, Justin Johnson

Inverting Visual Representations with Convolutional Networks
Dosovitskiy, A., & Brox, T. (2016)

(A different **code inversion** approach from Dosovitskiy and Brox 2016)

- Requires no optimization "at test time", directly trains the image reconstructor with Euclidean loss to the original true image.

$$W^* = \arg\min_W \sum_i ||x_i - f(\Phi(x_i), W)||_2^2$$

i.e. directly train a network for the mapping: features → image.

Slide Credit: Fei-Fei Li, Andrej Karpathy, Justin Johnson

Inverting Visual Representations with Convolutional Networks
Dosovitskiy, A., & Brox, T. (2016)

Inverting SIFT features:

$$W^* = \arg\min_W \sum_i ||x_i - f(\Phi(x_i), W)||_2^2$$

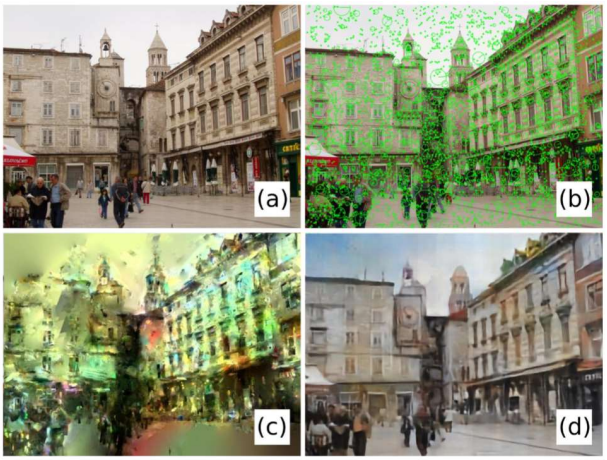
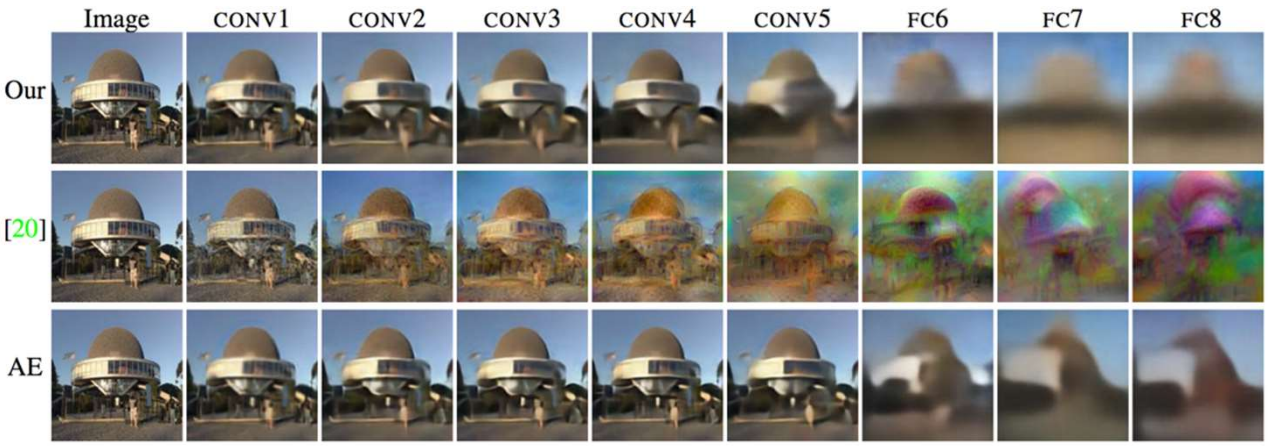


Figure 4: Reconstructing an image from SIFT descriptors with different methods. **(a)** an image, **(b)** SIFT keypoints, **(c)** reconstruction of [26], **(d)** our reconstruction.

Slide Credit: Fei-Fei Li, Andrej Karpathy, Justin Johnson

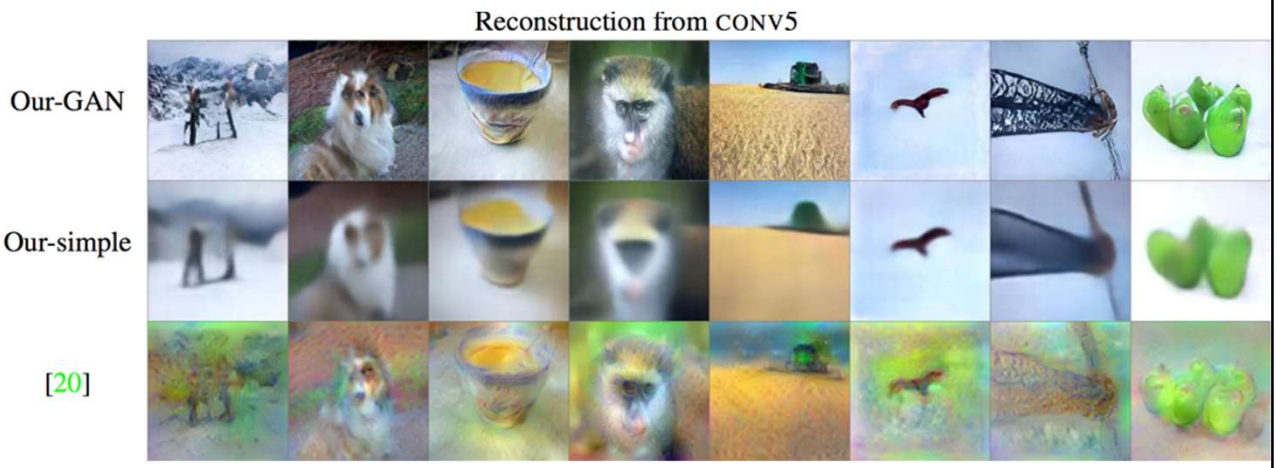
Inverting Visual Representations with Convolutional Networks
Dosovitskiy, A., & Brox, T. (2016)



69 of 97

Slide Credit: Fei-Fei Li, Andrej Karpathy, Justin Johnson

Inverting Visual Representations with Convolutional Networks
Dosovitskiy, A., & Brox, T. (2016)



70 of 97

Slide Credit: Fei-Fei Li, Andrej Karpathy, Justin Johnson

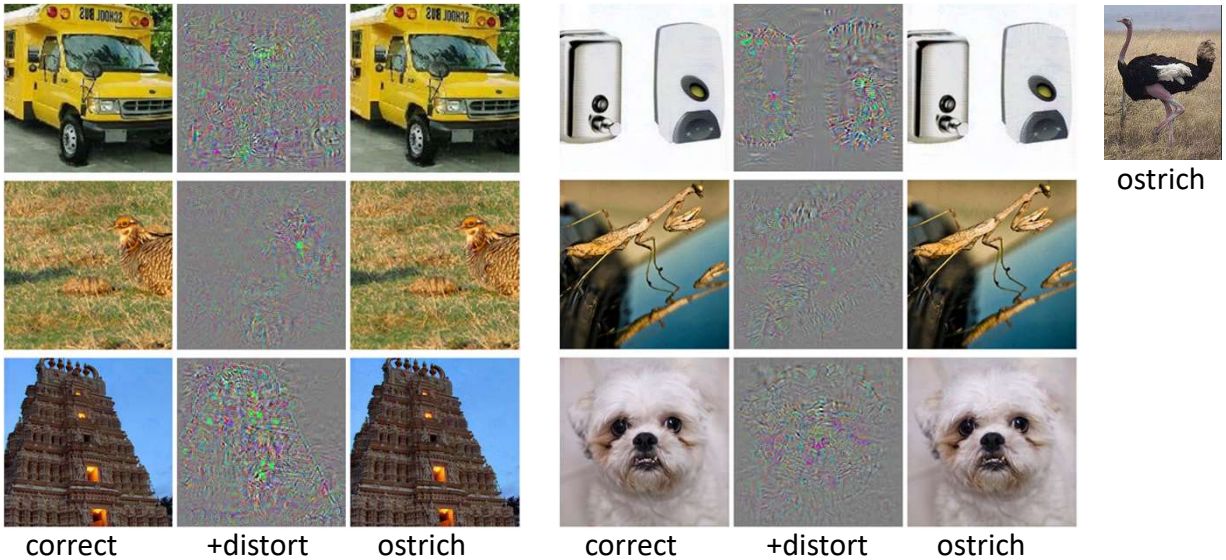
Using these ideas, we can pose an optimization over the input image to maximize any class score.

This leads to the question:
Can we use this to “fool” ConvNets?

Unfortunately, Yes!

71 of 97

Intriguing properties of neural networks
Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., & Fergus, R. (2013)

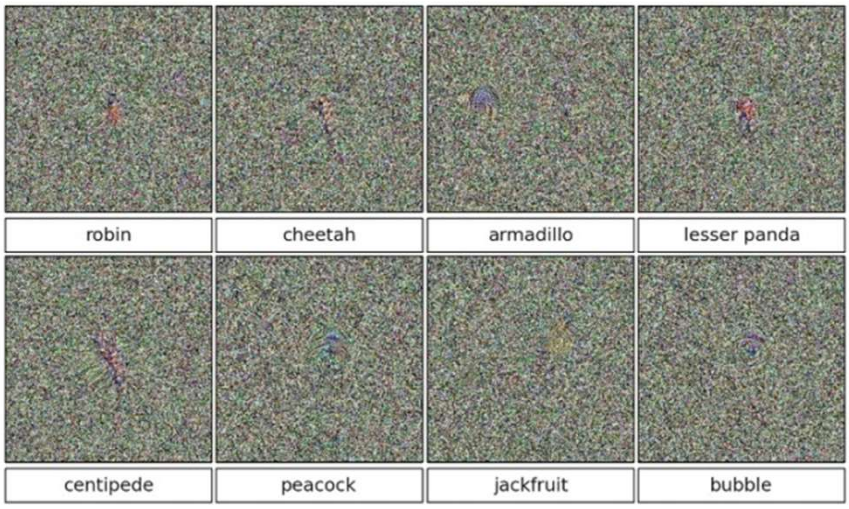


72 of 97

Slide Credit: Fei-Fei Li, Andrej Karpathy, Justin Johnson

Deep Neural Networks are Easily Fooled: High Confidence Predictions for Unrecognizable Images
Nguyen, A., Yosinski, J., & Clune, J. (2015)

> 99.6% confidences

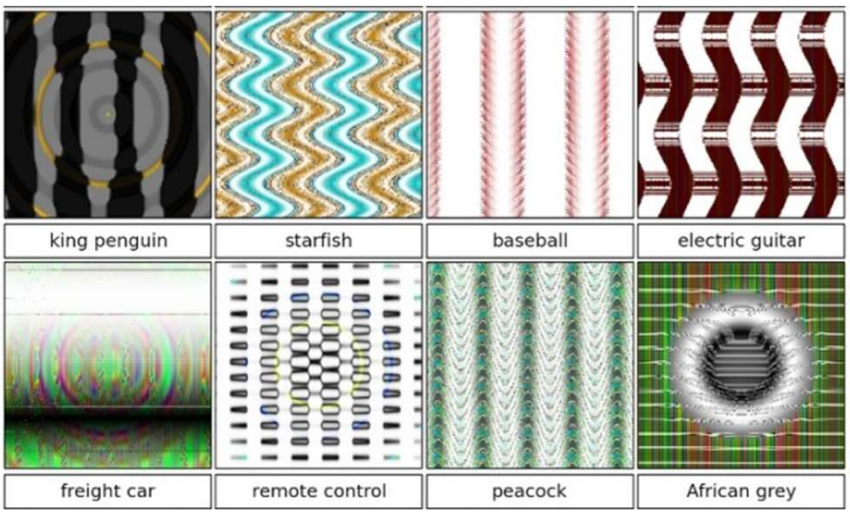


73 of 97

Slide Credit: Fei-Fei Li, Andrej Karpathy, Justin Johnson

Deep Neural Networks are Easily Fooled: High Confidence Predictions for Unrecognizable Images
Nguyen, A., Yosinski, J., & Clune, J. (2015)

> 99.6% confidences



74 of 97

Slide Credit: Fei-Fei Li, Andrej Karpathy, Justin Johnson

One pixel attack for fooling deep neural networks

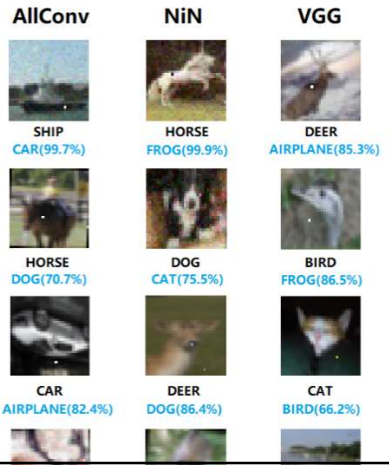
Jiawei Su*
Kyushu University
Japan
jiawei.su@inf.kyushu-u.ac.jp

Danilo Vasconcellos Vargas*
Kyushu University
Japan
vargas@inf.kyushu-u.ac.jp

Kouichi Sakurai
Kyushu University
Japan
sakurai@csce.kyushu-u.ac.jp

Abstract

Recent research has revealed that the output of Deep Neural Networks (DNN) can be easily altered by adding relatively small perturbations to the input vector. In this paper, we analyze an attack in an extremely limited scenario where only one pixel can be modified. For that we propose a novel method for generating one-pixel adversarial perturbations based on differential evolution. It requires less adversarial information and can fool more types of networks. The results show that 68.36% of the natural images in CIFAR-10 test dataset and 41.22% of the ImageNet (ILSVRC 2012) validation images can be perturbed to at least one target class by modifying just one pixel with 73.22% and 5.52% confidence on average. Thus, the proposed attack explores a different take on adversarial machine learning in an extreme limited scenario, showing that current DNNs are also vulnerable to such low dimension attacks.



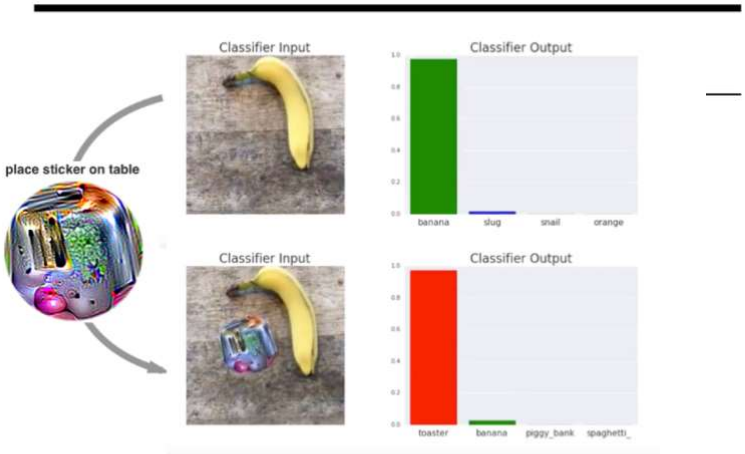


Figure 1: A real-world attack on VGG16, using a physical patch generated by the white-box ensemble method described in Section 3. When a photo of a tabletop with a banana and a notebook (top photograph) is passed through VGG16, the network reports class 'banana' with 97% confidence (top plot). If we physically place a sticker targeted to the class "toaster" on the table (bottom photograph), the photograph is classified as a toaster with 99% confidence (bottom plot). See the following video for a full demonstration: https://youtu.be/i1sp4X57TL4

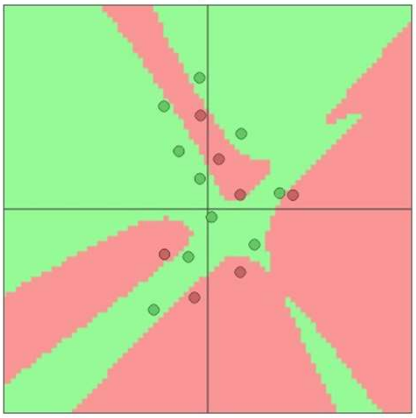
Projected Gradient Ascent (LS-PGA) [2] for discretized inputs. Other attack methods seek to modify only a small number of pixels in the image (Jacobian-based saliency map [11]), or a small patch at a fixed location of the image [13].

76 of 97

Explaining and Harnessing Adversarial Examples
Goodfellow, I. J., Shlens, J., & Szegedy, C. (2014)

"primary cause of neural networks' vulnerability to adversarial perturbation is their **linear nature**"

In particular, this is not a problem specific to Deep Learning and ConvNets.

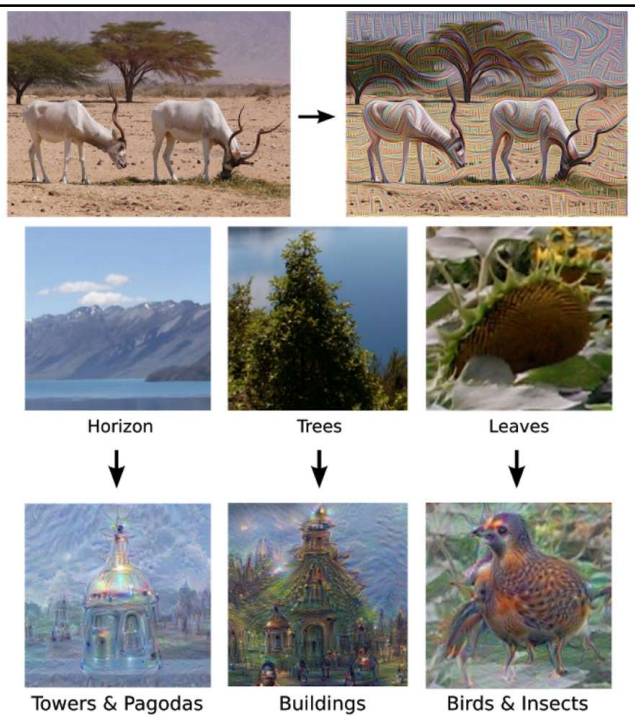


77 of 97

Slide Credit: Fei-Fei Li, Andrej Karpathy, Justin Johnson

DeepDream https://github.com/google/deepdream

78 of 97



79 of 97

DeepDream

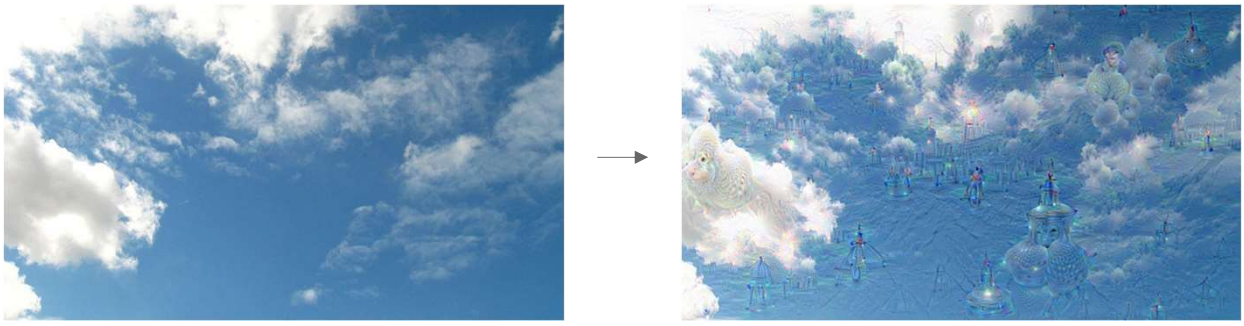
The idea is to magnify the activations by setting the gradient equal or proportional to its activation values and then backpropagating.

This makes magnifies patches that slightly looks like a particular class.

E.g. it makes anything that looks like a dog more dog like.

80 of 97

inception_4c/output



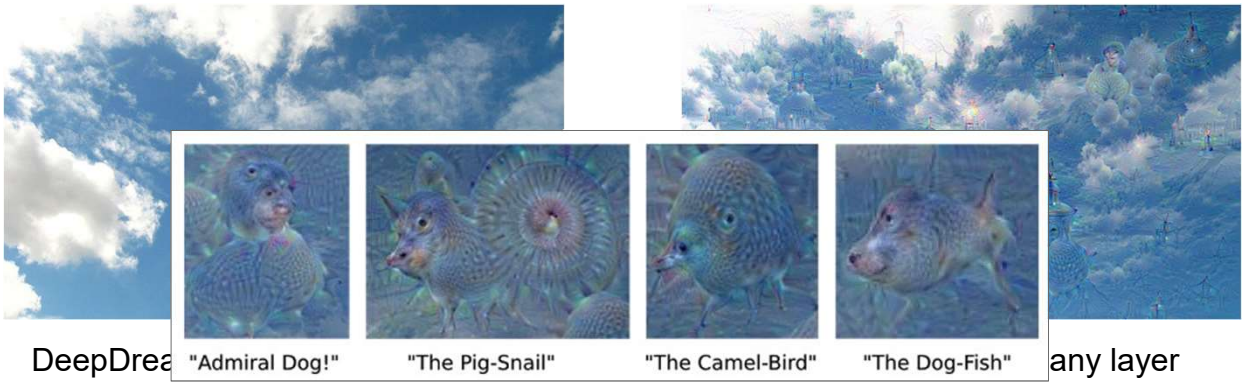
DeepDream modifies the image in a way that “boosts” all activations, at any layer

this creates a feedback loop: e.g. any slightly detected dog face will be made more and more dog like over time

81 of 97

Slide Credit: Fei-Fei Li, Andrej Karpathy, Justin Johnson

inception_4c/output



DeepDrea any layer

this creates a feedback loop: e.g. any slightly detected dog face will be made more and more dog like over time

82 of 97

Slide Credit: Fei-Fei Li, Andrej Karpathy, Justin Johnson

inception_3b/5x5_reduce

DeepDream modifies the image in a way that “boosts” all activations, at any layer

83 of 97

Slide Credit: Fei-Fei Li, Andrej Karpathy, Justin Johnson

NeuralStyle

[A Neural Algorithm of Artistic Style by Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge, 2015]

good implementation by Justin Johnson in Torch:
https://github.com/jcjohnson/neural-style

84 of 97

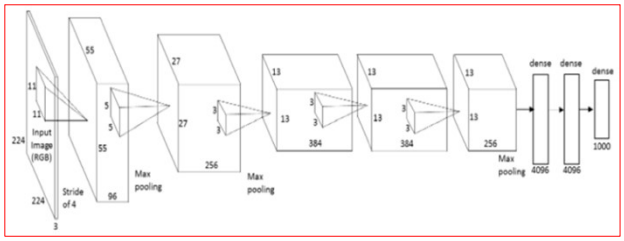
Slide Credit: Fei-Fei Li, Andrej Karpathy, Justin Johnson

make your own easily on deepart.io

85 of 97

Slide Credit: Fei-Fei Li, Andrej Karpathy, Justin Johnson

Step 1: Extract **content targets** (ConvNet activations of all layers for the given content image)



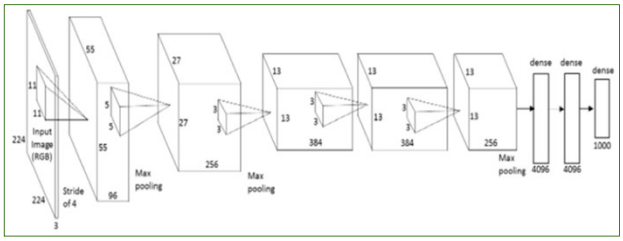
content activations

e.g.
at CONV5_1 layer we would have a [14x14x512] array of target activations

86 of 97

Slide Credit: Fei-Fei Li, Andrej Karpathy, Justin Johnson

Step 2: Extract **style targets** (Gram matrices of ConvNet activations of all layers for the given style image)



style gram matrices $G = V^{\mathrm{T}}V$

e.g.
at CONV1 layer (with [224x224x64] activations) would give a [64x64] Gram matrix of all pairwise activation covariances (summed across spatial locations)

87 of 97

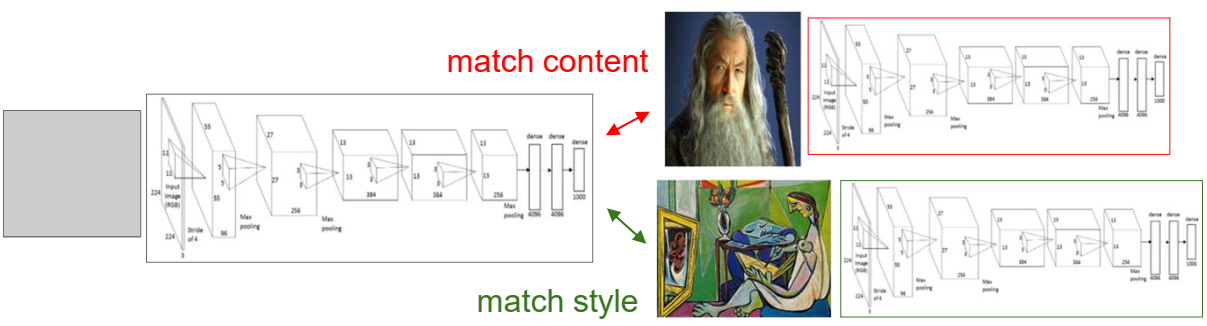
Slide Credit: Fei-Fei Li, Andrej Karpathy, Justin Johnson

Step 3: Optimize over image to have:

- The **content** of the content image (activations match content)
- The **style** of the style image (Gram matrices of activations match style)

$$\mathcal{L}_{total}(\vec{p}, \vec{a}, \vec{x}) = \alpha \mathcal{L}_{content}(\vec{p}, \vec{x}) + \beta \mathcal{L}_{style}(\vec{a}, \vec{x})$$

(+Total Variation regularization (maybe))



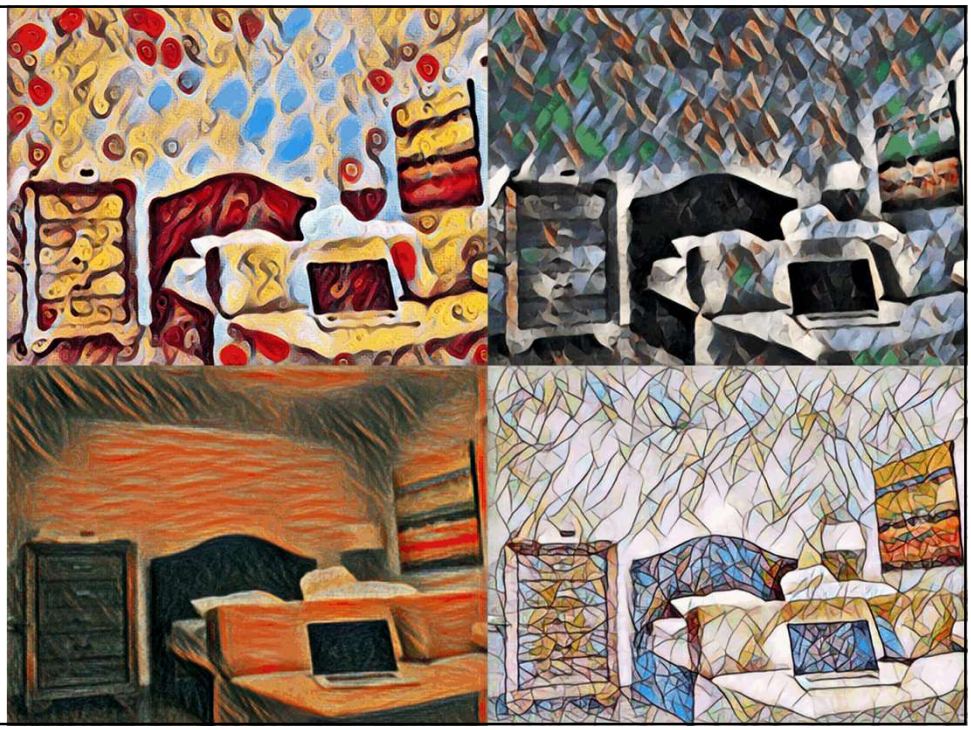
88 of 97

Slide Credit: Fei-Fei Li, Andrej Karpathy, Justin Johnson

FAST neural-style

run a webcam demo in real time:

https://github.com/jcjohnson/fast-neural-style



89 of 97

Fast Neural Style

Recall for **image code inversion**:

- Mahendran and Vedaldi 2014:
  optimize over the image such that the compute code matches a target code.
- Dosovitskiy and Brox 2015:
  train a new "inversion" network from code to image using image-image loss (e.g. L2)

Use the same idea to transform Neural Style to Fast Neural Style!

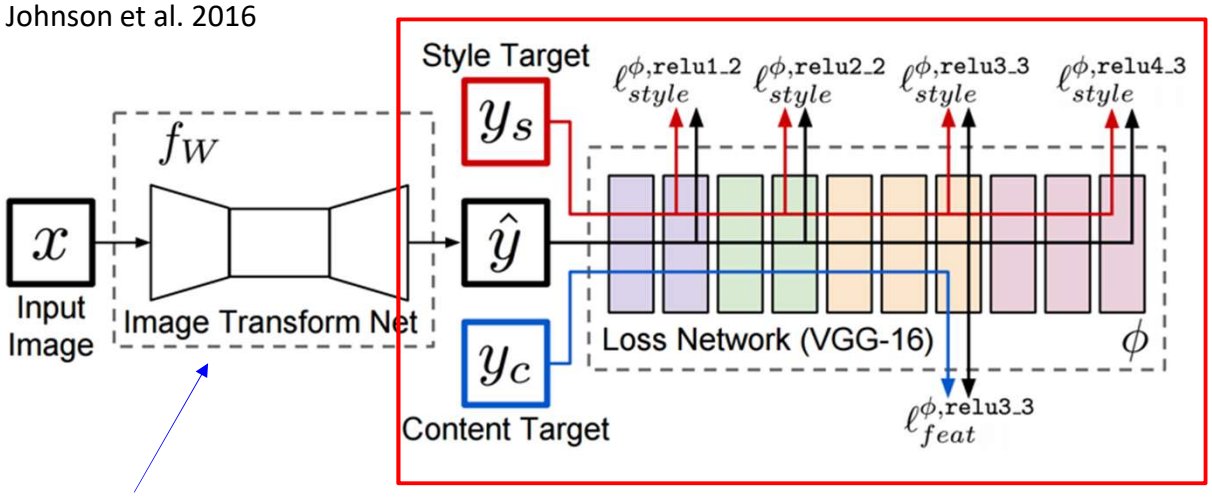
90 of 97

Slide Credit: Fei-Fei Li, Andrej Karpathy, Justin Johnson

Fast Neural Style

Johnson et al. 2016

Can also think of this as a fixed discriminator in GAN that doesn't get trained...



Train this network

All of this is just "neural style loss"

Pros: SUPER FAST (no optimization); **Cons**: network is style-specific :(

91 of 97

Slide Credit: Fei-Fei Li, Andrej Karpathy, Justin Johnson

Final Project Proposals

92 of 97

Final Project Proposals

Teams : 3-4 members

For the final project, we would like you to draft a 1-3 page proposal and a PowerPoint presentation that discusses the following:

- Problem Statement and Background
- Data Source
- Description of the Tools You Plan to Use
- Evaluation

Proposal presentation : To be announced

93 of 97

Problem Statement and Background

A high-level statement of the problem you intend to address, e.g. recognition of objects in images. Be specific - i.e. "recognition" in this case means outputting a label for the object and also a bounding box for its location.

Give background on the problem you are solving:

- Why it is important?
- What is the start of the art?
- Include some references to papers that describe the method you plan to use, or to build on.

94 of 97

Data Source

Describe the data source(s) you will use. Ideally you should make sure you can get the data before starting your project.

Describe how you plan to obtain the data, or how you got it if you already have it.

You should do any data cleaning / pre-processing work asap.

95 of 97

Evaluation

Describe how you will evaluate your model.

Describe the experiments you plan to conduct.

It is best to mimic the evaluations done on state-of-the-art papers.

96 of 97

OpenVino

Since we are working with intel we are required to use OpenVino

97 of 97

Some remarks

- Different groups can have similar topics but should have different approaches
- First come first serve basis
- Reproducing papers is okay if the code for the paper is not available online.
- If it is, you are required to modify / introduce your own ideas / contributions.

98 of 97

Last Homework: Neural Networks

- Will be posted Tonight
- Deadline Monday 11pm

99 of 97