Visualizing and Understanding CNN

Topic

- **By weights**
- **By activations**
- **By gradients**
- **Extensions**

Copyright 2019 ITRI 工業技術研究院

By weights

Look at one of neurons

Copyright 2019 ITRI 工業技術研究院

By weights
First Layer: Visualize Filters

AlexNet:
64 x 3 x 11 x 11

ResNet-18:
64 x 3 x 7 x 7

ResNet-101:
64 x 3 x 7 x 7

DenseNet-121:
64 x 3 x 7 x 7

Krizhevsky, “One weird trick for parallelizing convolutional neural networks”, arXiv 2014
He et al, “Deep Residual Learning for Image Recognition”, CVPR 2016
Huang et al, “Densely Connected Convolutional Networks”, CVPR 2017

Copyright 2019 ITRI 工業技術研究院

Visualize the filters/kernels (raw weights)

We can visualize filters at higher layers, but not that interesting

(these are taken from ConvNetJS CIFAR-10 demo)

layer 1 weights

16 x 3 x 7 x 7

layer 2 weights

20 x 16 x 7 x 7

layer 3 weights

20 x 20 x 7 x 7

Copyright 2019 ITRI 工業技術研究院

By activations

Look at feature maps in convolutional layers or feature vectors in fully-connected layers

Copyright 2019 ITRI 工業技術研究院

Last Layer

FC7 layer

4096-dimensional feature vector for an image (layer immediately before the classifier)

Run the network on many images, collect the feature vectors

Copyright 2019 ITRI 工業技術研究院

Last Layer: Nearest Neighbors

4096-dim vector

Test image

L2 Nearest neighbors in <u>feature</u> space

Recall: Nearest neighbors in <u>pixel</u> space

Krizhevsky et al, “ImageNet Classification with Deep Convolutional Neural Networks”, NIPS 2012.
Figures reproduced with permission.

Copyright 2019 ITRI 工業技術研究院

Last Layer: Dimensionality Reduction

Visualize the “space” of FC7 feature vectors by reducing dimensionality of vectors from 4096 to 2 dimensions

Simple algorithm: Principle Component Analysis (PCA)

More complex: **t-SNE**

Van der Maaten and Hinton, “Visualizing Data using t-SNE”, JMLR 2008
Figure copyright Laurens van der Maaten and Geoff Hinton, 2008. Reproduced with permission.

Copyright 2019 ITRI 工業技術研究院

By activations

Van der Maaten and Hinton, “Visualizing Data using t-SNE”, JMLR 2008
Figure copyright Laurens van der Maaten and Geoff Hinton, 2008. Reproduced with permission.

Copyright 2019 ITRI 工業技術研究院

By activations

conv5 feature map is 128x13x13; visualize as 128 13x13 grayscale images

Copyright 2019 ITRI 工業技術研究院

By activations: Maximally Activating Patches

Pick a layer and a channel; e.g. conv5 is 128 x 13 x 13, pick channel 17/128

Run many images through the network, record values of chosen channel

Visualize image patches that correspond to maximal activations

Springenberg et al, “Striving for Simplicity: The All Convolutional Net”, ICLR Workshop 2015
Figure copyright Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, Martin Riedmiller, 2015;

Copyright 2019 ITRI 工業技術研究院

By activations

Occlusion Experiments

elephant

Copyright 2019 ITRI 工業技術研究院

By gradients

With input image

gradient

Copyright 2019 ITRI 工業技術研究院

By gradients

Saliency Maps

How to tell which pixels matter for classification?

Dog

Compute gradient of (unnormalized) class score with respect to image pixels, take absolute value and max over RGB channels

Simonyan, Vedaldi, and Zisserman, "Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps", ICLR Workshop 2014.
Figures copyright Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman, 2014; reproduced with permission.

Copyright 2019 ITRI 工業技術研究院

Saliency Maps

Simonyan, Vedaldi, and Zisserman, “Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps”, ICLR Workshop 2014.
Figures copyright Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman, 2014; reproduced with permission.

By gradients

Intermediate features via (guided) backprop

Pick a single intermediate neuron, e.g. one value in 128 x 13 x 13 conv5 feature map

Compute gradient of neuron value with respect to image pixels

Images come out nicer if you only backprop positive gradients through each ReLU (guided backprop)

Figure copyright Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, Martin Riedmiller, 2015; reproduced with permission.

Zeiler and Fergus, "Visualizing and Understanding Convolutional Networks", ECCV 2014
Springenberg et al, "Striving for Simplicity: The All Convolutional Net", ICLR Workshop 2015

Copyright 2019 ITRI 工業技術研究院

By gradients

Intermediate features via (guided) backprop

Copyright 2019 ITRI 工業技術研究院

Fooling Images / Adversarial Examples

(1) Start from an arbitrary image
(2) Pick an arbitrary class
(3) Modify the image to maximize the class
(4) Repeat until network is fooled

Copyright 2019 ITRI 工業技術研究院

By gradients

Fooling Images / Adversarial Examples

Boat image is CC0 public domain
Elephant image is CC0 public domain

Copyright 2019 ITRI 工業技術研究院

By gradients

Without input image

Copyright 2019 ITRI 工業技術研究院

By gradients

Visualizing CNN features: Gradient Ascent

$$\arg\max_{I} \boxed{S_c(I)} - \lambda \|I\|_2^2$$

score for class c (before Softmax)

1. Initialize image to zeros

zero image

Repeat:
2. Forward image to compute current scores
3. Backprop to get gradient of neuron value with respect to image pixels
4. Make a small update to the image

Copyright 2019 ITRI 工業技術研究院

By gradients

Visualizing CNN features: Gradient Ascent

$$\arg\max_I S_c(I) - \boxed{\lambda \|I\|_2^2}$$

Simple regularizer: Penalize L2 norm of generated image

Simonyan, Vedaldi, and Zisserman, "Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps", ICLR Workshop 2014.
Figures copyright Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman, 2014; reproduced with permission.

Copyright 2019 ITRI 工業技術研究院

Extensions

DeepDream: Amplify existing features

Copyright 2019 ITRI 工業技術研究院

Extensions

DeepDream: Amplify existing features

Rather than synthesizing an image to maximize a specific neuron, instead try to **amplify** the neuron activations at some layer in the network

Choose an image and a layer in a CNN; repeat:

1. Forward: compute activations at chosen layer
2. Set gradient of chosen layer *equal to its activation*
3. Backward: Compute gradient on image
4. Update image

Equivalent to:

$$I^* = \arg\max_I \sum_i f_i(I)^2$$

Mordvintsev, Olah, and Tyka, "Inceptionism: Going Deeper into Neural Networks", Google Research Blog. Images are licensed under CC-BY

Copyright 2019 ITRI 工業技術研究院

Extensions

Copyright 2019 ITRI 工業技術研究院

Extensions

Copyright 2019 ITRI 工業技術研究院

Image is licensed under CC-BY 4.0

Extensions

Copyright 2019 ITRI 工業技術研究院

Extensions

Multi-scale processing

Copyright 2019 ITRI 工業技術研究院

Extensions

MIT places dataset

Copyright 2019 ITRI 工業技術研究院

Extensions

Feature Inversion

Copyright 2019 ITRI 工業技術研究院

Extensions

Feature Inversion

Given a CNN feature vector for an image, find a new image that:

- Matches the given feature vector
- "looks natural" (image prior regularization)

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

Given feature vector

Features of new image

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

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

Total Variation regularizer
(encourages spatial smoothness)

Mahendran and Vedaldi, "Understanding Deep Image Representations by Inverting Them", CVPR 2015

Copyright 2019 ITRI 工業技術研究院

Extensions

Feature Inversion

Reconstructing from different layers of VGG-16

Mahendran and Vedaldi, "Understanding Deep Image Representations by Inverting Them", CVPR 2015
Figure from Johnson, Alahi, and Fei-Fei, "Perceptual Losses for Real-Time Style Transfer and Super-Resolution", ECCV 2016. Copyright Springer, 2016.
Reproduced for educational purposes.

Copyright 2019 ITRI 工業技術研究院

Extensions

Neural Texture Synthesis: Gram Matrix

Copyright 2019 ITRI 工業技術研究院

Extensions

Neural Texture Synthesis: Gram Matrix

This image is in the public domain.

Each layer of CNN gives C x H x W tensor of features; H x W grid of C-dimensional vectors

Outer product of two C-dimensional vectors gives C x C matrix measuring co-occurrence

Copyright 2019 ITRI 工業技術研究院

Extensions

Neural Texture Synthesis: Gram Matrix

This image is in the public domain.

Each layer of CNN gives C x H x W tensor of features; H x W grid of C-dimensional vectors

Outer product of two C-dimensional vectors gives C x C matrix measuring co-occurrence

Average over all HW pairs of vectors, giving **Gram matrix** of shape C x C

Efficient to compute; reshape features from

C x H x W to =C x HW

then compute $G = FF^T$

Copyright 2019 ITRI 工業技術研究院

Extensions

Neural Texture Synthesis

$$E_l = \frac{1}{4N_l^2 M_l^2} \sum_{i,j} \left(G_{ij}^l - \hat{G}_{ij}^l\right)^2 \qquad \mathcal{L}(\vec{x}, \hat{\vec{x}}) = \sum_{l=0}^{L} w_l E_l$$

1. Pretrain a CNN on ImageNet (VGG-19)
2. Run input texture forward through CNN, record activations on every layer; layer i gives feature map of shape $C_i \times H_i \times W_i$
3. At each layer compute the *Gram matrix* giving outer product of features:

$$G_{ij}^l = \sum_k F_{ik}^l F_{jk}^l \text{ (shape } C_i \times C_i\text{)}$$

4. Initialize generated image from random noise
5. Pass generated image through CNN, compute Gram matrix on each layer
6. Compute loss: weighted sum of L2 distance between Gram matrices
7. Backprop to get gradient on image
8. Make gradient step on image
9. GOTO 5

Gatys, Ecker, and Bethge, "Texture Synthesis Using Convolutional Neural Networks", NIPS 2015
Figure copyright Leon Gatys, Alexander S. Ecker, and Matthias Bethge, 2015. Reproduced with permission.

Copyright 2019 ITRI 工業技術研究院

Extensions

Neural Texture Synthesis

Reconstructing texture from higher layers recovers larger features from the input texture

Gatys, Ecker, and Bethge, "Texture Synthesis Using Convolutional Neural Networks", NIPS 2015
Figure copyright Leon Gatys, Alexander S. Ecker, and Matthias Bethge, 2015. Reproduced with permission.

Copyright 2019 ITRI 工業技術研究院

Neural Style Transfer: Using Feature or Gram Reconstruction

Texture synthesis
(Gram reconstruction)

Feature reconstruction

Figure from Johnson, Alahi, and Fei-Fei, "Perceptual Losses for Real-Time Style Transfer and Super-Resolution", ECCV 2016. Copyright Springer, 2016. Reproduced for educational purposes.

Copyright 2019 ITRI 工業技術研究院

Extensions

Gatys, Ecker, and Bethge, "Image style transfer using convolutional neural networks", CVPR 2016
Figure adapted from Johnson, Alahi, and Fei-Fei, "Perceptual Losses for Real-Time Style Transfer and Super-Resolution", ECCV 2016. Copyright Springer, 2016. Reproduced for educational purposes.

Copyright 2019 ITRI 工業技術研究院

Extensions

Gatys, Ecker, and Bethge, "Image style transfer using convolutional neural networks", CVPR 2016
Figure adapted from Johnson, Alahi, and Fei-Fei, "Perceptual Losses for Real-Time Style Transfer and Super-Resolution", ECCV 2016. Copyright Springer, 2016. Reproduced for educational purposes.

Copyright 2019 ITRI 工業技術研究院

Extensions

Neural Style Transfer: Multiple Style Images

Mix style from multiple images by taking a weighted average of Gram matrices

Gatys, Ecker, and Bethge, "Image style transfer using convolutional neural networks", CVPR 2016
Figure copyright Justin Johnson, 2015.

Copyright 2019 ITRI 工業技術研究院

Extensions

Problem: Style transfer requires many forward / backward passes through VGG; very slow!

Copyright 2019 ITRI 工業技術研究院

Extensions

Fast Style Transfer

(1) Train a feedforward network for each style
(2) Use pretrained CNN to compute same losses as before
(3) After training, stylize images using a single forward pass

Johnson, Alahi, and Fei-Fei, "Perceptual Losses for Real-Time Style Transfer and Super-Resolution", ECCV 2016
Figure copyright Springer, 2016. Reproduced for educational purposes.

Copyright 2019 ITRI 工業技術研究院

Summary

Many methods for understanding CNN representations

Weights: no visually-meaningful results
Activations: Maximal patches, occlusion to indicate where to trigger the classification results.
Gradients: Saliency maps, class visualization, feature inversion, Gram matrix inversion, DeepDream, Style Transfer.

Copyright 2019 ITRI 工業技術研究院