

11-695: AI Engineering

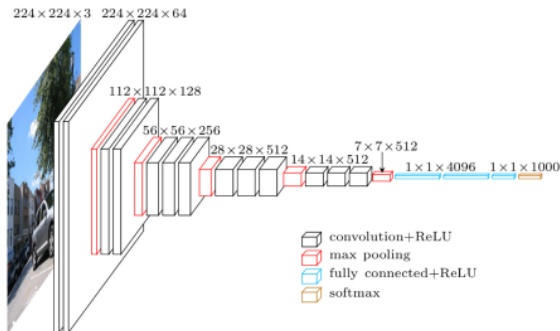
Convolution II

Spring 2019

① Recap

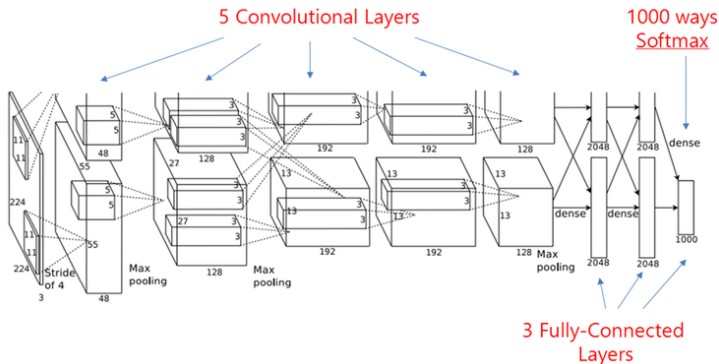
② Convolution Architectures (cont'd)

③ What CNNs Learn?

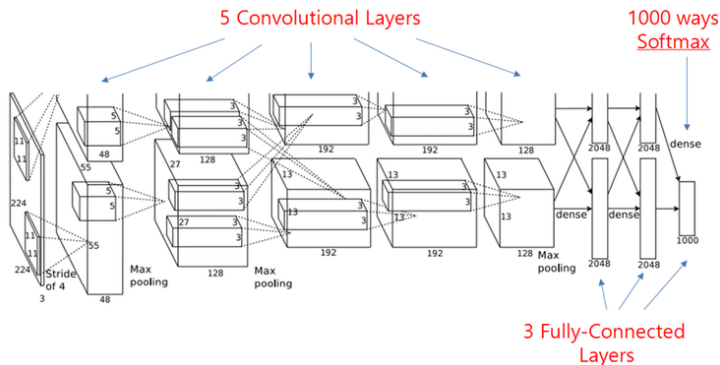


- Motivation of Convolution
- Operation of Convolution
- Convolutional Layers
- Pooling Layers
- LeNet-5 and AlexNet and VGG

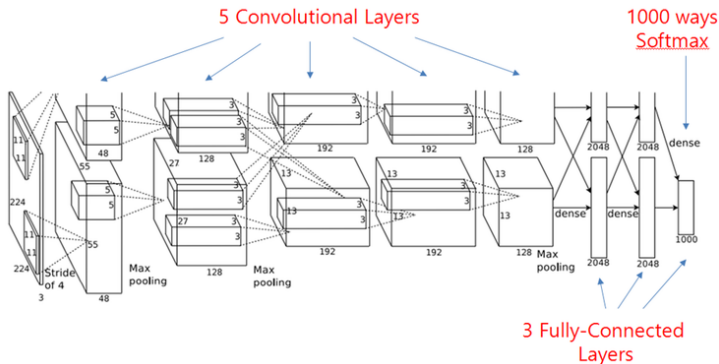
- 1 Recap
- 2 Convolution Architectures (cont'd)
- 3 What CNNs Learn?



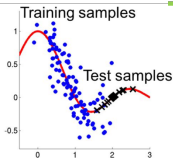
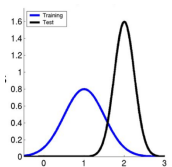
- Total params: 62,367,776 (check the module I slides)



- Total params: 62,367,776 (check the module I slides)
- 58,621,952 params in FC layers ($\approx 94\%$) $\rightarrow$ a pain!



- Total params: 62,367,776 (check the module I slides)
- 58,621,952 params in FC layers ($\approx 94\%$) $\rightarrow$ a pain!
- First use: ReLU, norm layer, heavy augmentation, dropout, momentum



Input: Values of x over a mini-batch: $\mathcal{B} = \{x_1 \dots x_m\}$;

Parameters to be learned: γ, β

Output: $\{y_i = \text{BN}_{\gamma, \beta}(x_i)\}$

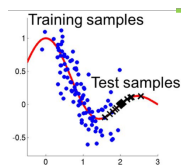
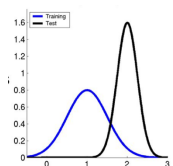
$$\mu_{\mathcal{B}} \leftarrow \frac{1}{m} \sum_{i=1}^m x_i \quad // \text{ mini-batch mean}$$

$$\sigma_{\mathcal{B}}^2 \leftarrow \frac{1}{m} \sum_{i=1}^m (x_i - \mu_{\mathcal{B}})^2 \quad // \text{ mini-batch variance}$$

$$\hat{x}_i \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}} \quad // \text{ normalize}$$

$$y_i \leftarrow \gamma \hat{x}_i + \beta \equiv \text{BN}_{\gamma, \beta}(x_i) \quad // \text{ scale and shift}$$

- Covariate Shift: different distributions between train and test sets



Input: Values of x over a mini-batch: $\mathcal{B} = \{x_1 \dots x_m\}$;

Parameters to be learned: γ, β

Output: $\{y_i = \text{BN}_{\gamma, \beta}(x_i)\}$

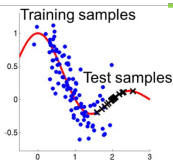
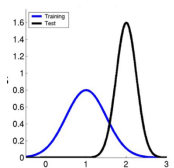
$$\mu_{\mathcal{B}} \leftarrow \frac{1}{m} \sum_{i=1}^m x_i \quad // \text{ mini-batch mean}$$

$$\sigma_{\mathcal{B}}^2 \leftarrow \frac{1}{m} \sum_{i=1}^m (x_i - \mu_{\mathcal{B}})^2 \quad // \text{ mini-batch variance}$$

$$\hat{x}_i \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}} \quad // \text{ normalize}$$

$$y_i \leftarrow \gamma \hat{x}_i + \beta \equiv \text{BN}_{\gamma, \beta}(x_i) \quad // \text{ scale and shift}$$

- Covariate Shift: different distributions between train and test sets
- Internal Covariate Shift: a covariate shift between 2 layers in NN



Input: Values of x over a mini-batch: $\mathcal{B} = \{x_1 \dots x_m\}$;

Parameters to be learned: γ, β

Output: $\{y_i = \text{BN}_{\gamma, \beta}(x_i)\}$

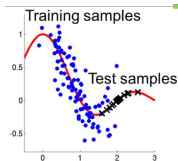
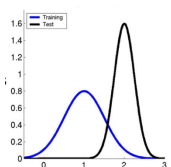
$$\mu_{\mathcal{B}} \leftarrow \frac{1}{m} \sum_{i=1}^m x_i \quad // \text{ mini-batch mean}$$

$$\sigma_{\mathcal{B}}^2 \leftarrow \frac{1}{m} \sum_{i=1}^m (x_i - \mu_{\mathcal{B}})^2 \quad // \text{ mini-batch variance}$$

$$\hat{x}_i \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}} \quad // \text{ normalize}$$

$$y_i \leftarrow \gamma \hat{x}_i + \beta \equiv \text{BN}_{\gamma, \beta}(x_i) \quad // \text{ scale and shift}$$

- Covariate Shift: different distributions between train and test sets
- Internal Covariate Shift: a covariate shift between 2 layers in NN
- Fix: scale unto unit Gaussian distribution $\mathcal{N}(0, \mathcal{I}) \rightarrow$ differentiable



Input: Values of x over a mini-batch: $B = \{x_1 \dots x_m\}$;

Parameters to be learned: γ, β

Output: $\{y_i = \text{BN}_{\gamma, \beta}(x_i)\}$

$$\mu_B \leftarrow \frac{1}{m} \sum_{i=1}^m x_i \quad // \text{ mini-batch mean}$$

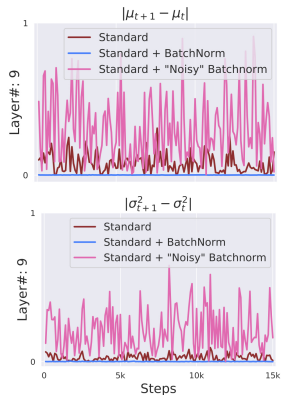
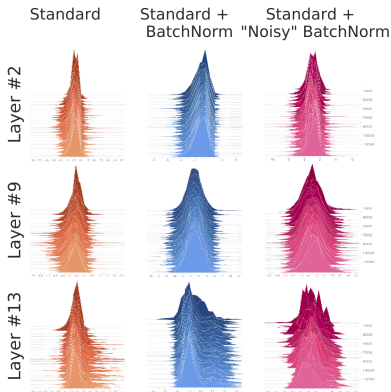
$$\sigma_B^2 \leftarrow \frac{1}{m} \sum_{i=1}^m (x_i - \mu_B)^2 \quad // \text{ mini-batch variance}$$

$$\hat{x}_i \leftarrow \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}} \quad // \text{ normalize}$$

$$y_i \leftarrow \gamma \hat{x}_i + \beta \equiv \text{BN}_{\gamma, \beta}(x_i) \quad // \text{ scale and shift}$$

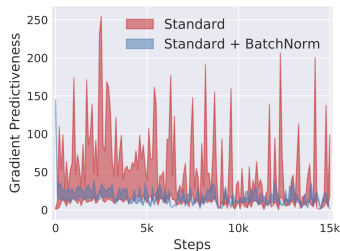
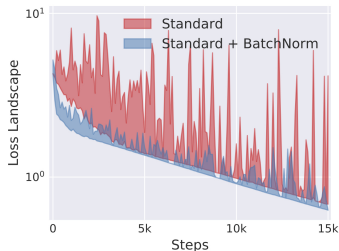
- Training time: Take batch, compute $(\mu_{tr}, \sigma_{tr}) \rightarrow$ new training values $\rightarrow$ feed to the next layer
- Test time: use the whole test set instead of each batch

Argument: Not about Co-Variate Shift

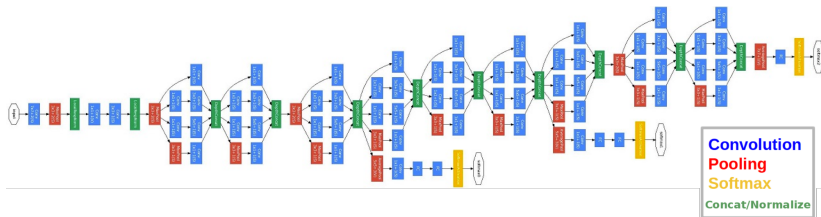


- Adding co-variate shift does not reduce the effect of batchnorm

Argument: Not about Co-Variate Shift

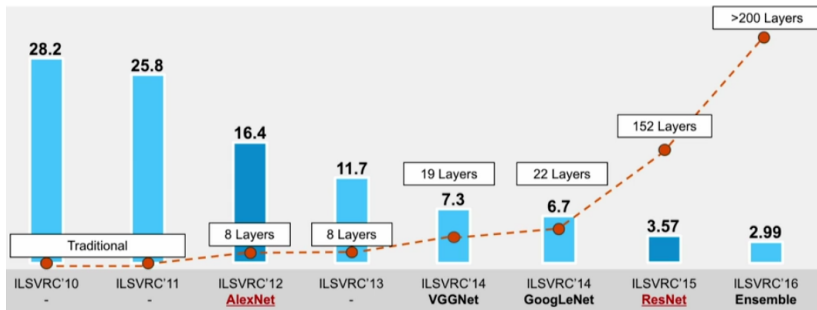


- Adding co-variate shift does not reduce the effect of batchnorm
- Batch norm seems to make the landscap smoother for optimization

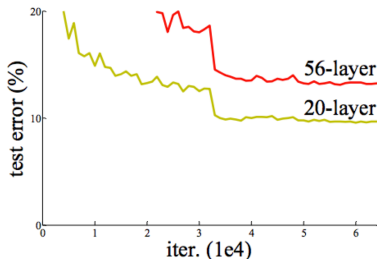
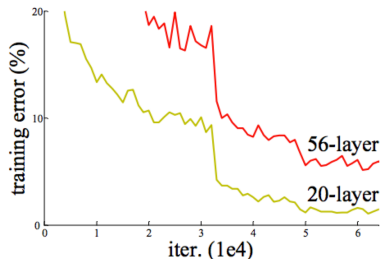


- Inception module: *concatenate* 1x1, 3x3, 5x5 conv and a pooling
- Auxiliary losses with discounted factor of 0.3
- Use of average pooling to reduce parameters → reduce params and hence overfitting at FC layers

Deep Trend?

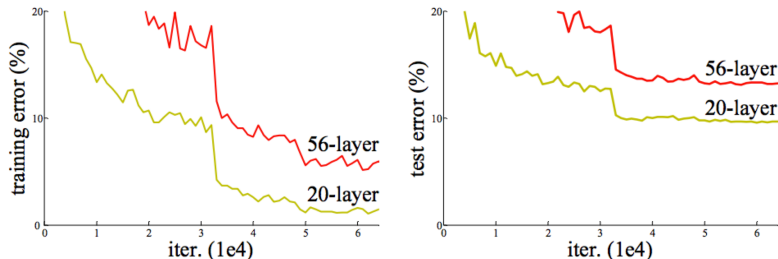


Is more layers always better?



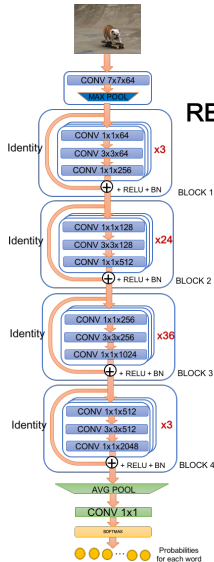
- LeNet 5 layers → AlexNet 8 layers → VGG 16 layers → GoogLeNet 22 layers
- Until it seems to saturate... (why?).

Is more layers always better?

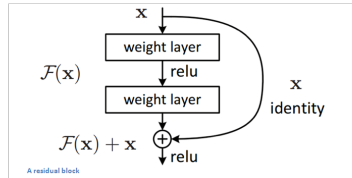


- LeNet 5 layers → AlexNet 8 layers → VGG 16 layers → GoogLeNet 22 layers
- Until it seems to saturate... (why?). We need something more than just increasing no. of layers (and resources)

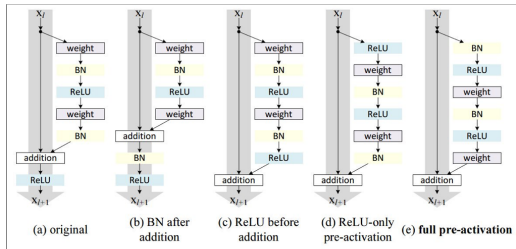
ResNet (2015)

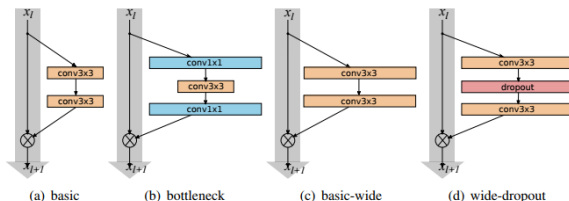


RESNET 200

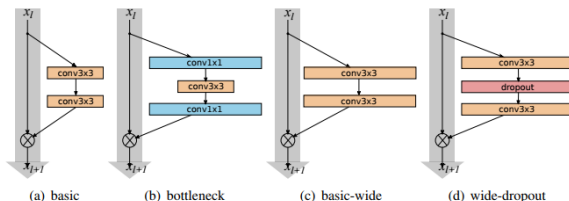


TYPES of RESNET BLOCKS



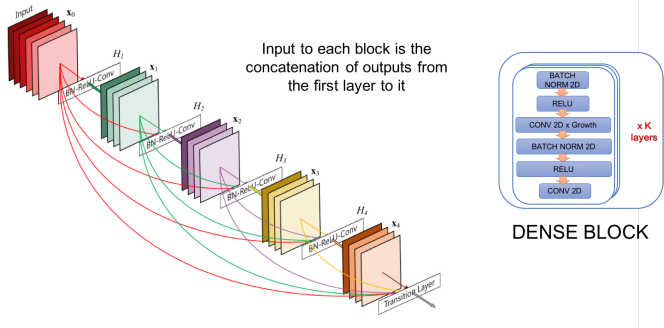


- Going wide instead of deeper
- To increase feature reuse, and decrease training time
- Same as ResNet, two important uses of:

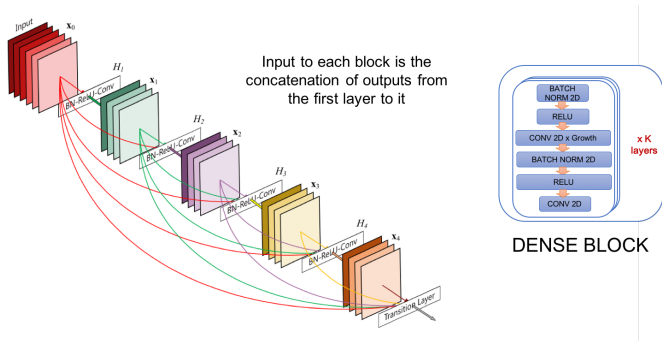


- Going wide instead of deeper
- To increase feature reuse, and decrease training time
- Same as ResNet, two important uses of:
 - Residual connection to alleviate gradient vanishing and create ensembling effect
 - Use of 1×1 conv layer to reduce dimensionality (why?)

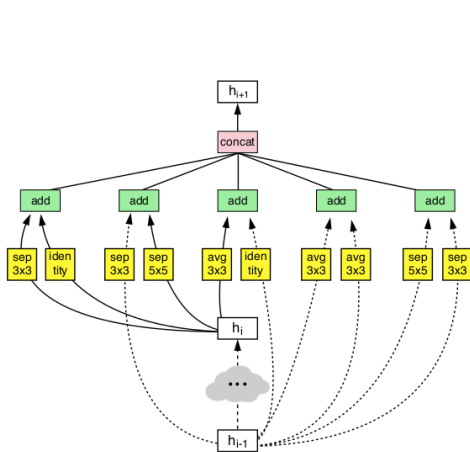
[► In Practice](#)



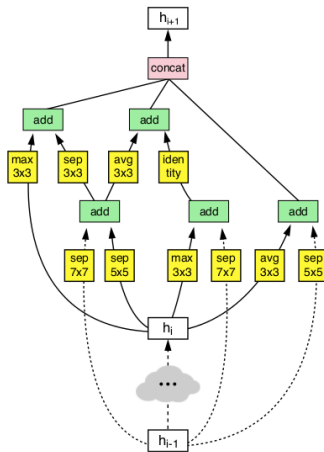
- Same motivation of feature reuse, with a different implementation
- It grows with layers



- Same motivation of feature reuse, with a different implementation
- It grows with layers
- But it also has a measure to reduce it by a compressing transition layer which outputs $\theta * m$ layers ($0 < \theta < 1$)



Normal Cell



Reduction Cell

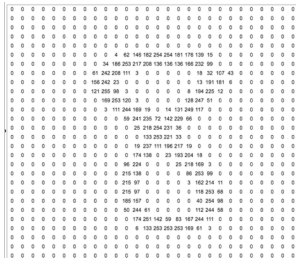
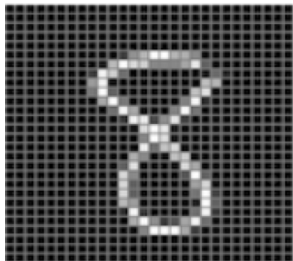
- As a result from Neural Architecture Search

Image credit: Barret Zoph - Google AI

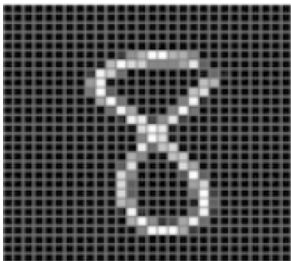
Break

- 1 Recap
- 2 Convolution Architectures (cont'd)
- 3 What CNNs Learn?

Is CNN a blackbox to human?



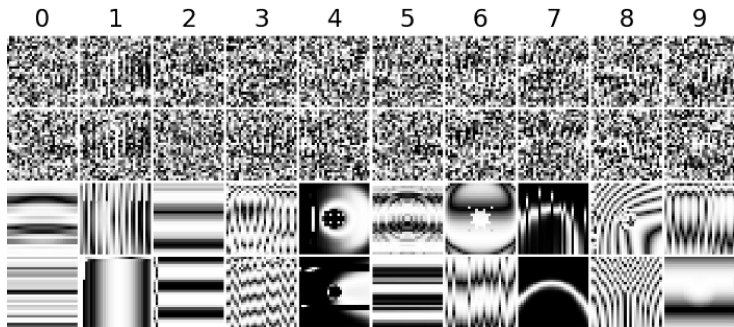
- Ugly truth: Human's perception and machine's are different
- Human struggled to make machine understand vision (the way it thinks useful)

[illegible]

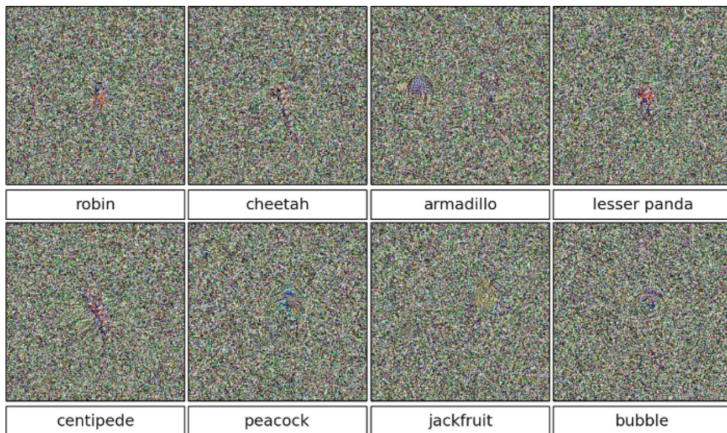
- Ugly truth: Human's perception and machine's are different
- Human struggled to make machine understand vision (the way it thinks useful)
- Now human makes effort to understand more about machine's language/perception of vision
- Although how NN works in general is still a myth...

11-695: AI Engineering

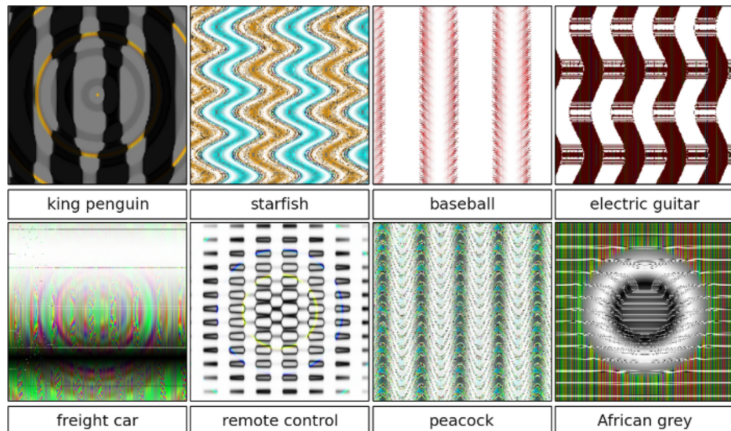
Spring 2019 19 / 29



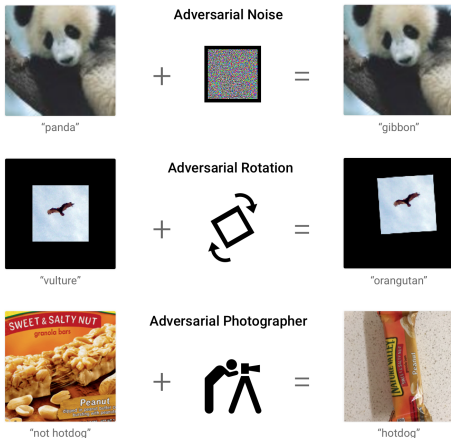
- Machine: 99.99% vs. Human: $\approx 0\%$



- Machine: 99.99% vs. Human: $\approx 0\%$

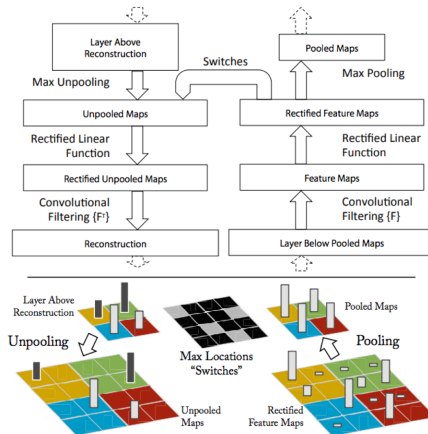


- Machine: 99.99% vs. Human: $\approx 0\%$



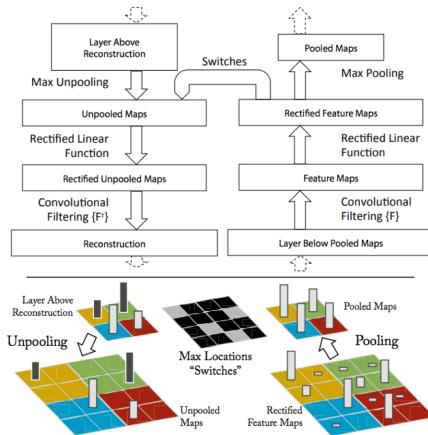
- Panda + 0.07 Noise = Gibbon

Translate back from machine to human?

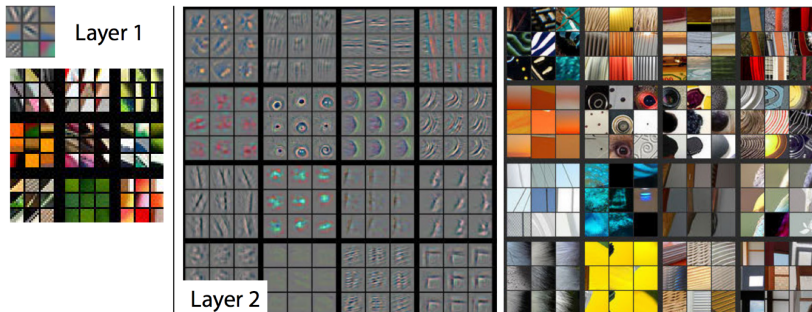


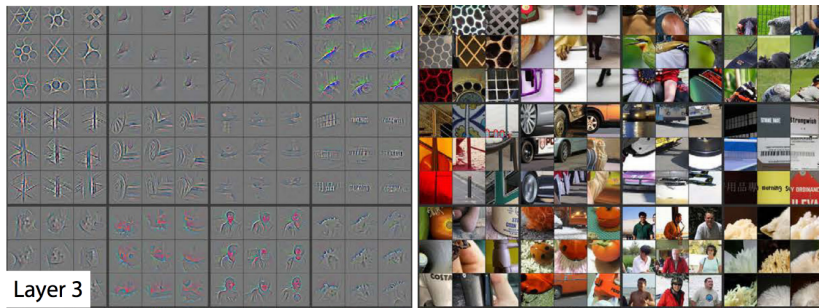
- Convolution $\iff$ Deconvolution, but

Translate back from machine to human?



- Convolution $\iff$ Deconvolution, but
- Is CNN an invertable transformation?





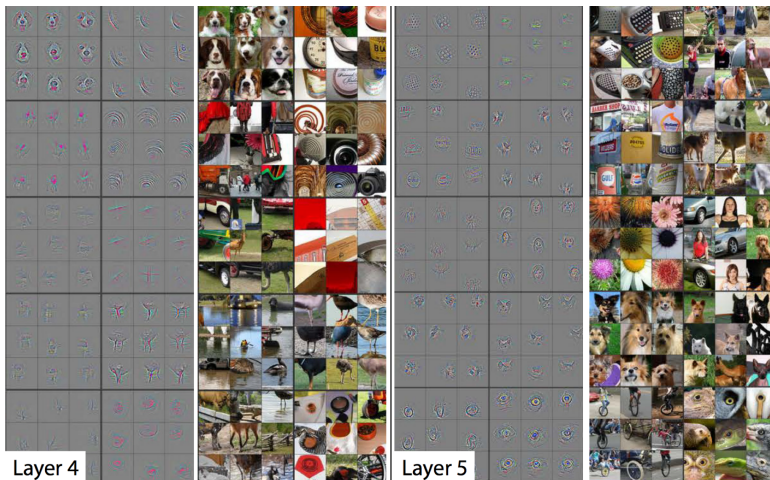
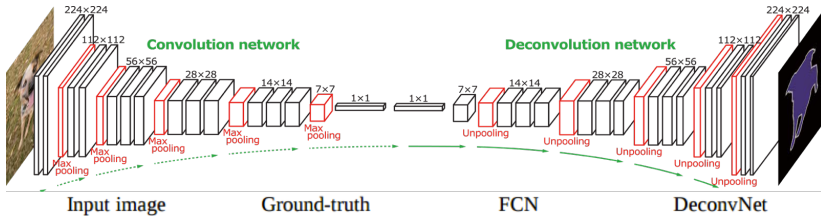


Image credit: Zeiler and Fergus



End of Module II.