

# DEEP LEARNING: REDES NEURAIS CONVOLUCIONAIS

## Aprendizado de Máquina

2

- Como programar um computador para resolver problemas muito complexos?
  - ▣ Tarefa difícil e que depende do problema a ser resolvido
- Aprendizado de Máquina
  - ▣ Estuda como dar aos computadores a capacidade de aprender a partir dos dados

# Aprendizado de Máquina

3

- Tarefa simples de se resolver
  - ▣ Reconhecer um objeto tridimensional
  
- Tarefa difícil de se resolver
  - ▣ Reconhecer um objeto tridimensional mudando o ponto de vista ou as condições de iluminação

## Motivação

4

- Apesar das melhorias recentes, os algoritmos de aprendizagem tem dificuldades para:
  - ▣ Entender cenas e descrevê-las em linguagem natural
  - ▣ Inferir conceitos semânticos suficientes a ponto de interagir com humanos

# Motivação

5

- Como extrair informações de imagens?
  - ▣ Tradicionalmente, processamos seus pixels de modo a obter representações mais abstratas
    - Presença de bordas
    - Presença de objetos com determinadas formas
    - etc
  - ▣ Alta variabilidade de tipos de estruturas presentes

# Motivação

6

- Como extrair informações de imagens?
  - ▣ Geralmente as imagens demandam uma combinação de estratégias
    - Cor dos gatos
    - Textura do camaleão
    - Etc



(a)

(b)



(c)

(d)

# Deep Learning

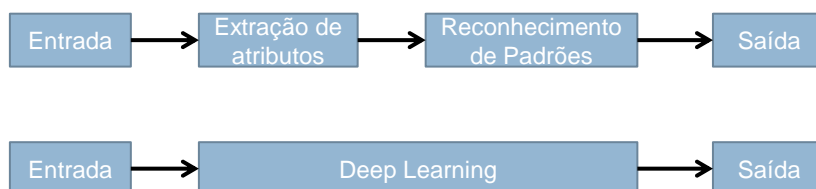
7

- Ou *Aprendizado Profundo*
- É uma sub área do aprendizado de máquina
  - ▣ Utiliza algoritmos de aprendizado que extraem significado dos dados usando uma hierarquia de múltiplas camadas que imitam as redes neurais do nosso cérebro
- Busca aprender representações de dados
  - ▣ Eficácia excepcional em padrões de aprendizagem

# Deep Learning

8

- As camadas não são projetadas por engenheiros humanos
  - ▣ As camadas são aprendidas a partir dos dados
  - ▣ Utiliza um procedimento de aprendizado para fins gerais



# Deep Learning

9

- É inspirado na forma como o cérebro funciona
  - ▣ Muitos neurônios conectados
  - ▣ A força das conexões representa um conhecimento de longo prazo
- É indicado para tarefas cujo espaço de entrada seja localmente estruturado
  - ▣ Estrutura espacial ou temporal
  - ▣ Imagens, linguagem etc.

# Deep Learning

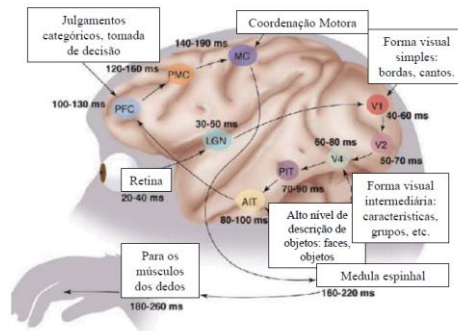
10

- Neurônios organizados em camadas hierárquicas
  - ▣ Cada camada transforma os dados de entrada em representações cada vez mais abstratas
    - Atributos de baixo nível até conceitos mais abstratos
    - Exemplo: borda -> nariz -> face
  - ▣ A camada de saída combina esses recursos para fazer previsões

# Deep Learning

11

- O cérebro humano funciona dessa forma
  - ▣ Primeira hierarquia (dados do córtex visual): sensibilidade às bordas
  - ▣ Regiões mais abaixo são sensíveis às estruturas mais complexas
    - Exemplo: rostos



# Deep Learning

12

- Processo de aprendizagem que descobre múltiplos níveis de abstração
  - ▣ Representações mais abstratas permitem extrair informações mais úteis para os classificadores
  - ▣ A profundidade está associada ao número de operações não lineares aprendidas

# Deep Learning

13

- Podemos usar uma rede MLP tendo como entrada uma imagem?
  - ▣ Tamanho da entrada em geral é muito grande!
  - ▣ Uma imagem de 200 x 200: 40000 unidades de entrada
  - ▣ Considerando as várias camadas ocultas, pode-se chegar a bilhões de parâmetros a serem ajustados

# Deep Learning

14

- Podemos usar uma rede MLP tendo como entrada uma imagem?
  - ▣ Redes são iniciadas com pesos aleatórios. Geralmente ficavam presas a mínimos locais
  - ▣ Aumento da profundidade da rede tornava difícil obter uma boa generalização

# Rede Neural Convolucional

15

- *Convolutional Neural Network - CNN*
- Proposta por Yann LeCun e Yoshua Bengio em 1995
  - ▣ É um tipo especial de rede neural com múltiplas camadas, como a MLP
  - ▣ É inspirada na sensibilidade local e orientação seletiva do cérebro humano
  - ▣ Projetada para extrair características relevantes da entrada

# Rede Neural Convolucional

16

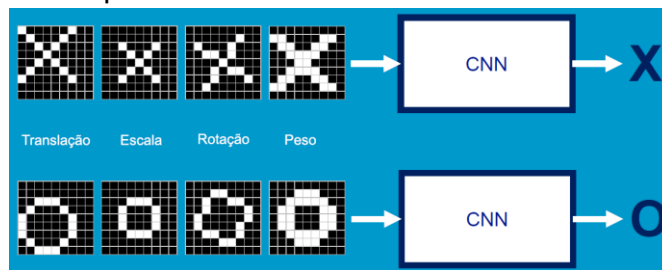
- É uma rede do tipo *feed-forward*
  - ▣ Similar ao sistema visual humano, é projetada para extrair propriedades topológicas da entrada
    - Aprendem múltiplas camadas de transformações
    - As camadas são aplicadas umas sobre as outras
- Treinamento é feito utilizando o algoritmo de *back-propagation* (ou suas variações)
  - ▣ Necessitam de grandes quantidades de dados



# Rede Neural Convolucional

17

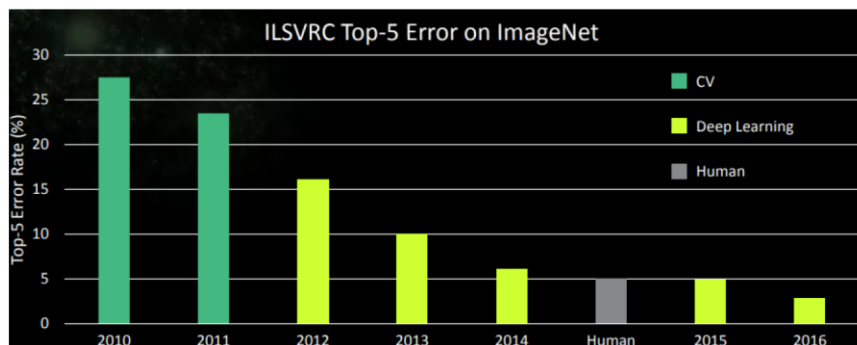
- Um conjunto de pixels de entrada se transforma em um conjunto de votos nas classes possíveis
  - ▣ São redes capazes de reconhecer dados com muita variabilidade
    - Exemplo: caracteres escritos a mão



# Rede Neural Convolucional

18

- Impacto na classificação de imagens



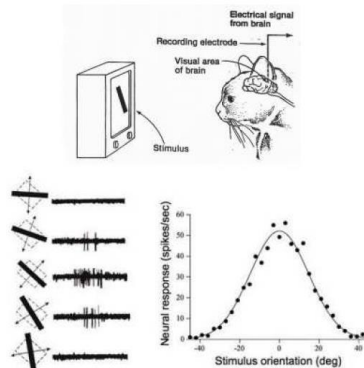
ImageNet: The "computer vision World Cup"

Fonte: <https://www.dsiac.org/sites/default/files/journals/dsiac-winter-2017-volume-4-number-1.pdf>

# Rede Neural Convolucional

19

- Hubel e Wiesel, 1962
  - ▣ Estudos com o sistema visual de gatos



# Rede Neural Convolucional

20

- Hubel e Wiesel, 1962
  - ▣ Estudos com o sistema visual de gatos
    - Descoberta do papel importante dos Campos Receptivos
    - Atuam como filtros locais
      - Filtragem Espacial
    - 2 tipos de comportamentos
      - Simple Cells: Respondem a padrões de bordas na imagem;
      - Complex Cells: Possuem campos receptivos grandes e são invariantes à posição do padrão.

# Rede Neural Convolucional

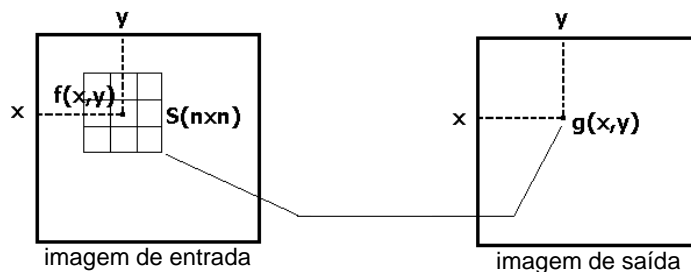
21

- Hubel e Wiesel, 1962
  - ▣ Redes convolucionais são um tipo especial de redes MLP que usam o conceito de campo receptivo local
  - ▣ Explora as correlações espaciais focando em conectividade entre unidades de processamento próximas

## Filtragem Espacial

22

- Definição
  - ▣ Também conhecidos como operadores locais ou filtros locais
  - ▣ Combinam a intensidade de um certo número de pixels, para gerar a intensidade da imagem de saída.



# Filtragem Espacial

23

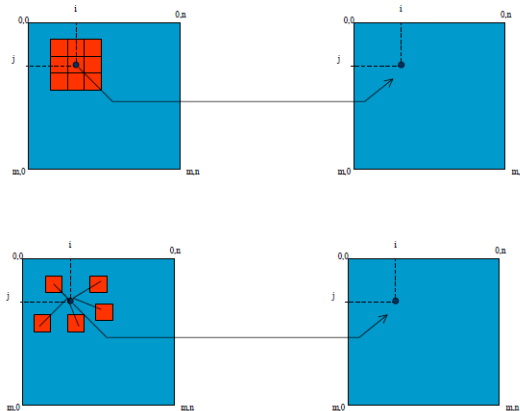
- São técnicas baseadas na convolução de

- ▣ templates

- janelas, matrizes

- ▣ tuplas

- conjunto de pixels



# Filtragem Espacial

24

- Uma grande variedade de filtros digitais podem ser implementados através da convolução no domínio do espaço
  - ▣ São os operadores locais mais utilizados em processamento de imagens, com diversas aplicações
    - Pré-processamento
    - Eliminação de ruídos
    - Suavização
    - Segmentação

# Filtragem Espacial

25

- Refere-se ao plano da imagem
  - ▣ Envolve a manipulação direta dos pixels da imagem utilizando uma máscara espacial (kernels, templates, janelas)

$\frac{1}{9} \times$	1	1	1
	1	1	1
	1	1	1

-1	0	1
-2	0	2
-1	0	1

- Valores das máscaras são chamados de coeficientes
  - ▣ O processo de filtragem é similar a um operação matemática denominada **convolução**

# Filtragem Espacial

26

- Processo de filtragem
  - ▣ Cada elemento da máscara é multiplicado pelo valor do pixel correspondente na imagem ***f***
  - ▣ A soma desses resultados é o novo valor do nível de cinza na nova imagem ***g***
  - ▣ Exemplo:  $w$  é uma janela de  $n \times n = k$  pixels. O processo de filtragem para cada pixel na imagem ***g(x,y)*** será dada por

$$g(x, y) = \sum_{i=1}^k w_i \cdot f(x, y)$$

# Filtragem Espacial

27

## □ Processo de filtragem

- $(a, b, c, d, e, f, g, h, i)$ : são os valores dos níveis de cinza na vizinhança de  $f(x, y)$
- $(w_1 a w_9)$ : são os coeficientes da máscara
- O valor do pixel  $g(x, y)$  é dado por

$$g(x, y) = w_1.a + w_2.b + w_3.c + w_4.d + w_5.e + w_6.f + w_7.g + w_8.h + w_9.i$$

Imagem ---  $f(x, y)$ 

a	b	c
d	e	f
g	h	i

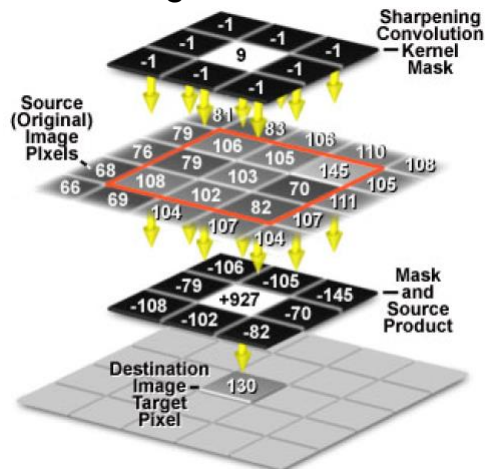
 $k = 3 \times 3 = 9$ 

$w_1$	$w_2$	$w_3$
$w_4$	$w_5$	$w_6$
$w_7$	$w_8$	$w_9$

# Filtragem Espacial

28

## □ Processo de filtragem



# Filtragem Espacial

29

- A esse processo de filtragem dá-se o nome de **convolução**
  - ▣ Desloca-se a máscara (espelhada) sobre a imagem e calcula-se a soma dos produtos em cada local
    - A equação devem ser aplicada sobre todas as posições  $x$  e  $y$  da imagem

$$w(x, y) * f(x, y) = \sum_{s=-a}^a \sum_{t=-b}^b w(s, t) f(x-s, y-t)$$

$$a = (m-1)/2 \quad b = (n-1)/2$$

Espelhamento ou rotação, feito na imagem

## Exemplo de Convolução

30

- Conjunto de operações

Desloca, Multiplica, Soma

máscara

1	0
0	1

Imagem

1	1	3	3	4
1	1	4	4	3
2	1	3	3	3
1	1	1	4	4

Resultado

2	5	7	6	*
2	4	7	7	*
3	2	7	7	*
*	*	*	*	*

1	1	3	3	4	0
0	1	4	4	3	1
2	1	3	3	3	
1	1	1	4	4	

A imagem resultado é menor do que a imagem original. Os valores marcados com \* não podem ser calculados.

# Convolução

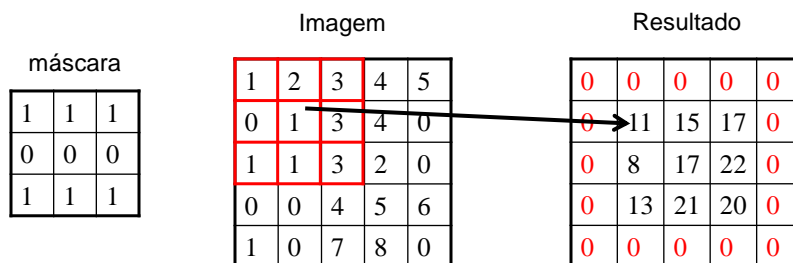
31

- Convenção
  - ▣ Nas máscaras de tamanho par ( $2 \times 2$ ,  $4 \times 4$ , ...) o resultado é colocado sobre o primeiro pixel
  - ▣ Nas máscaras de tamanho ímpar ( $3 \times 3$ ,  $5 \times 5$ , ...) o resultado é colocado sobre o pixel central
  - ▣ A imagem resultante da convolução não necessita obrigatoriamente ser menor do que a imagem original
    - Convolução aperiódica
    - Gabarito truncado

## Convolução aperiódica

32

- O valor 0 é atribuído aos resultados não calculáveis

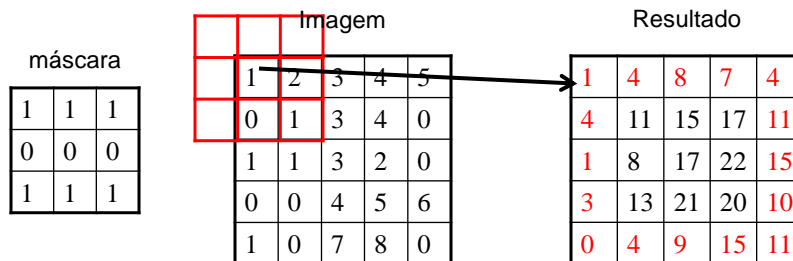




## Gabarito truncado

33

- Centra-se a máscara com o primeiro pixel da imagem atribuindo o valor 0 aos valores inexistentes na imagem



## Custo da convolução

34

- O custo computacional da convolução é alto
  - ▣ Em um imagem de tamanho  $M \times M$  e máscara  $N \times N$ , o número de multiplicações é de  $M^2 N^2$
  - ▣ Exemplo: imagem de  $512 \times 512$  e máscara de  $16 \times 16 = 67.108.864$  multiplicações.

# Máscaras de convolução

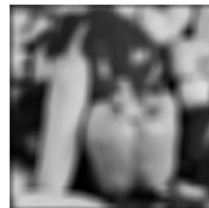
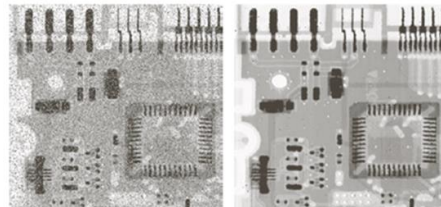
35

- O tamanho da máscara e os valores de seus coeficientes definem o tipo de filtragem produzido
- Alguns exemplos
  - ▣ Filtro Passa Baixa e média espacial (suavização)
  - ▣ Filtro Passa Alta (realce)
  - ▣ Gradientes (robert, sobel, etc): detectores de borda

# Máscaras de convolução

36

- Filtro Passa Baixa
  - ▣ Remoção de ruído
  - ▣ Suavização



## Máscaras de convolução

37

- Filtro Passa Alta
  - ▣ Realce de imagens



## Máscaras de convolução

38

- Gradientes
  - ▣ Detectores de Bordas

Original

Bordas  
horizontais (Sobel)Bordas  
verticais (Sobel)Imagem  
gradiente

## Estrutura da Rede

39

- Uma Rede Neural Convolucional é formada pela combinação de 4 tipos de camadas
  - ▣ Camada de Convolução
  - ▣ Camada ReLU
  - ▣ Camada de *Pooling*
  - ▣ Camada *Fully Connected*

## Camada de Convolução

40

- Utiliza a ideia de Campos Receptivos Locais
  - ▣ A camada calcula a saída dos neurônios conectados a regiões locais da entrada
  - ▣ Trata-se de um detector de atributos
    - Aprende automaticamente a filtrar as informações não necessárias de uma entrada usando um filtro de convolução

## Camada de Convolução

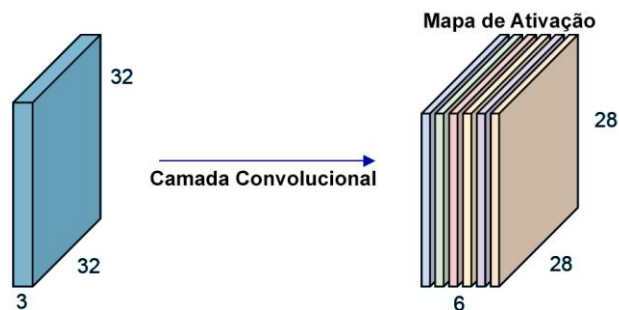
41

- Os filtros são aprendidos pelo algoritmo
  - ▣ A inicialização dos filtros é aleatória
  - ▣ O pesos dos filtros são os mesmo em qualquer uma das regiões do mapa
  - ▣ A convolução é linear
  - ▣ Facilita o paralelismo do processo

## Camada de Convolução

42

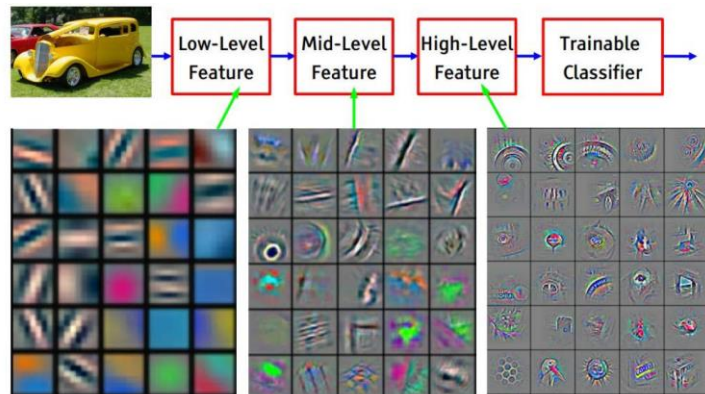
- Uma imagem permite gerar um conjunto de imagens



## Camada de Convolução

43

- Permite aprender diferentes tipos de atributos em cada camada

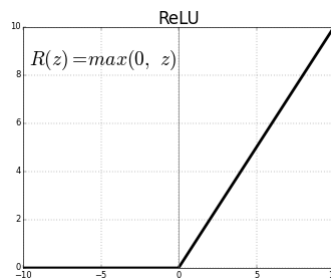


Feature visualization of convolutional net trained on ImageNet from [Zeiler & Fergus 2013]

## Camada ReLU

44

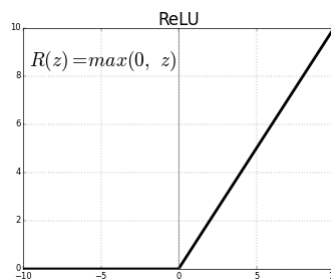
- Do inglês, **R**ectified **L**inear **U**nits
  - ▣ Função de ativação ponto-a-ponto
  - ▣ É a função de ativação usada nas redes convolucionais



## Camada ReLU

45

- Usada como função de ativação para camadas ocultas
  - ▣ Treino mais muito rápido
  - ▣ Mais expressiva que a função logística
  - ▣ Evita o problema do gradiente de fuga



## Camada ReLU

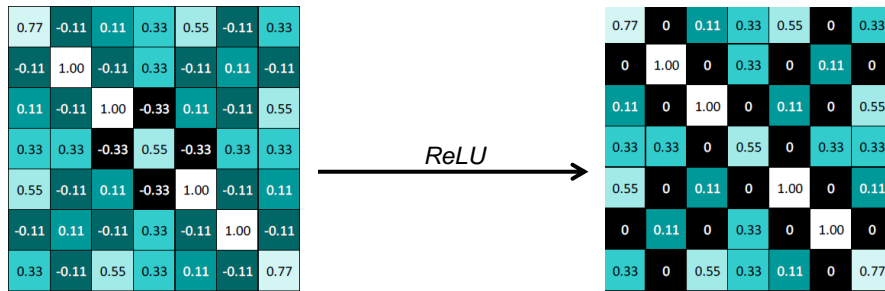
46

- Gradiente de Fuga ou Dissipação do Gradiente
  - ▣ O gradiente em redes neurais profundas é instável
    - Com isso, ele tende a explodir ou a desaparecer nas camadas anteriores
  - ▣ Atualização dos pesos da rede é proporcional à derivada parcial da função de erro com respeito ao peso em cada iteração
    - Gradiente extremamente pequeno impede o peso de ser alterado

## Camada ReLU

47

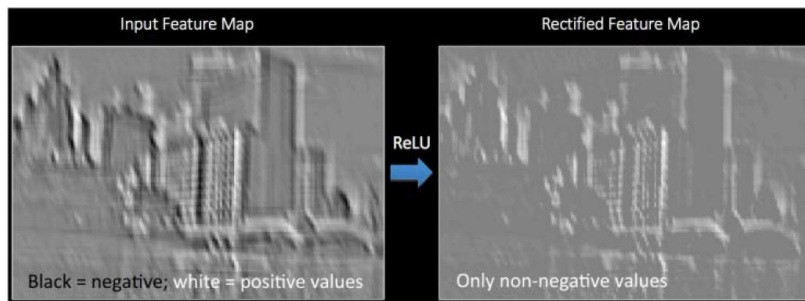
- Exemplo de aplicação nos dados



## Camada ReLU

48

- Exemplo de aplicação em uma “imagem”





## Camada de Pooling

49

- Também chamada de *downsample*
  - ▣ Reduz o tamanho das imagens de entrada
  - ▣ Reduz a sensibilidade a deslocamentos e outras formas de distorção
    - Torna a convolução invariante a translação, rotação e *shifting* (janelamento)
  - ▣ Ajuda a detectar objetos em alguns locais incomuns e reduz o tamanho da memória
- Tipos de pooling
  - ▣ Min, Max e Média

## Camada de Pooling

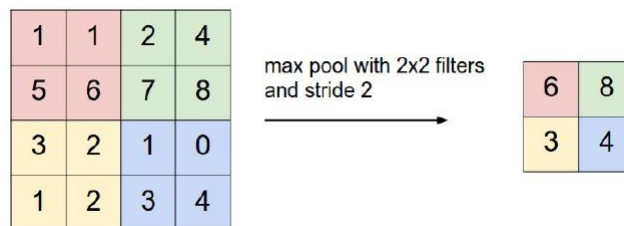
50

- Pooling mais utilizado: *max-pooling*
  - ▣ Seleciona-se a maior ativação para propagar na região de interesse
  - ▣ Permite capturar semelhanças em imagens mesmo que elas estejam um pouco deslocadas

# Camada de Pooling

51

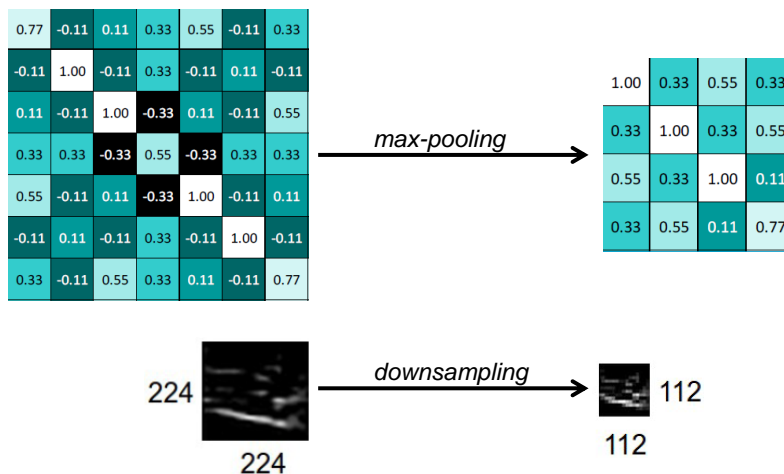
- Algoritmo
  - ▣ Selecione o tamanho da janela
    - Normalmente 2 x 2 ou 3 x 3
  - ▣ Deslize sua janela sobre a imagem
  - ▣ Para cada janela, selecione o valor máximo (*max-pooling*)



# Camada de Pooling

52

- Um conjunto de entradas gera saídas menores



## Camada de Pooling

53

- Reconstrução: camada de *pooling* da AlexNet



[https://github.com/lnFoCusp/tf\\_cnnvis](https://github.com/lnFoCusp/tf_cnnvis)

## Camada Fully Connected

54

- Também chamada de camada *Dense*
  - ▣ Trabalha como uma *Rede Multilayer Perceptron*
    - Os neurônios atuam sobre todos os dados fornecidos pelas camadas anteriores
    - Cada neurônio da camada de saída representa uma classe do problema

# Camada Fully Connected

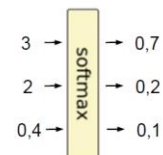
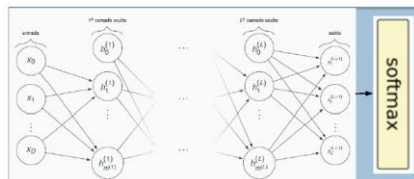
55

- Camada Convolutacional
  - ▣ Extração de atributos de alto nível da imagem
- Camada Fully Connected
  - ▣ Busca aprender como classificar esses atributos em um conjunto de classes

# Camada Fully Connected

56

- Função softmax
  - ▣ Utilizada quando a rede é voltada para tarefas de classificação
  - ▣ Permite interpretar os valores da camada de saída como uma distribuição de probabilidades



# Treinamento

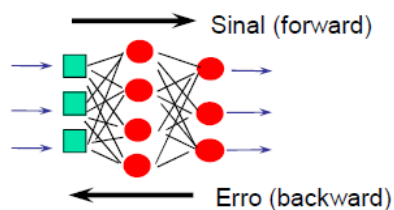
57

- *Error back-propagation* ou Retropropagação do erro
- Funcionamento
  - ▣ Saída produzida pela rede é diferente do resultado esperado
  - ▣ Determina o grau de responsabilidade de cada parâmetro da rede
  - ▣ Valor do parâmetro é alterado com o propósito de reduzir o erro produzido

# Treinamento

58

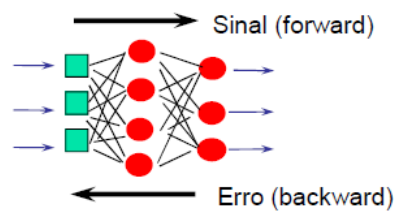
- *Error back-propagation* ou Retropropagação do erro
  - ▣ Sentido direto (*forward*)
    - Cálculo da saída e do erro
  - ▣ Sentido inverso (*backward*)
    - Propagação do erro
    - $\text{Erro} = (\text{resposta obtida}) - (\text{resposta correta})$



# Treinamento

59

- *Error back-propagation* ou Retropropagação do erro
  - ▣ Utiliza a regra da cadeia para calcular o quão rápido o erro muda quando mudamos a ativação de uma unidade oculta
  - ▣ Pesos são ajustados para que as previsões fiquem mais precisas



# Dropout

60

- Desligamento de neurônios
  - ▣ Consiste em eliminar aleatoriamente (e temporariamente) alguns dos neurônios ocultos na rede
  - ▣ Os neurônios de entrada e saída não são afetados
  - ▣ Permite aprender descritores mais robustos dos dados

# Dropout

61

## □ Desligamento de neurônios

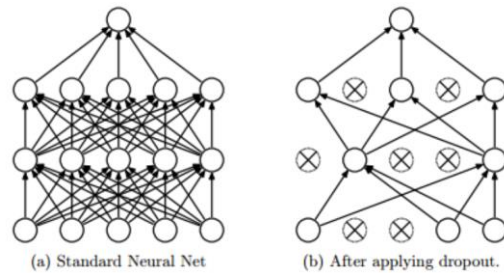


Figure 1: Dropout Neural Net Model. **Left:** A standard neural net with 2 hidden layers. **Right:** An example of a thinned net produced by applying dropout to the network on the left. Crossed units have been dropped.

Dropout - A Simple Way to Prevent Neural Networks from Overfitting (JMLR, 2014)

# Dropout

62

## □ Desligamento de neurônios = rede mais robusta

- Eliminar neurônio equivale a treinar redes neurais diferentes
  - Diminui o *overfitting*
- Reduz co-adaptações complexas de neurônios
  - Um neurônio não pode confiar na presença de outros neurônios em particular, pois podem estar desligados
  - O neurônio é forçado a aprender atributos mais robustos e úteis em conjunto com outros neurônios aleatoriamente selecionados

## Batch Normalization

63

- Normalização em Lote
- Durante o treinamento, os dados são normalizados
  - ▣ No entanto, os dados são transformados a medida que passam pelas diferentes camadas da rede
  - ▣ Isso pode fazer com que os dados voltem a ficar desnormalizados

## Batch Normalization

64

- Esse problema é conhecido como **mudança de covariável interna** (*internal covariate shift*)
  - ▣ Aumenta o tempo necessário para treinar a rede
  - ▣ Aumenta as chances de dissipação do gradiente
  - ▣ Se torna pior a medida que a rede fica mais profunda



## Batch Normalization

65

- Consiste em normalizar os dados fornecidos a cada camada oculta
- Vantagens
  - ▣ Reduz a dependência da inicialização
  - ▣ Melhora o fluxo de gradiente em redes profundas
  - ▣ Diminui o tempo de treinamento
    - Diminuição de 14 vezes na quantidade de épocas necessárias para treinar a rede

## Projetando a Rede

66

- Como deve ser a estrutura da rede?
  - ▣ Quantas camadas?
  - ▣ Quais tipos de camadas usar?
  - ▣ Qual a ordem das camadas?
  - ▣ Qual tipo de filtro?
  - ▣ Qual tipo de normalização?
  - ▣ Taxa de Dropout?

# Projetando a Rede

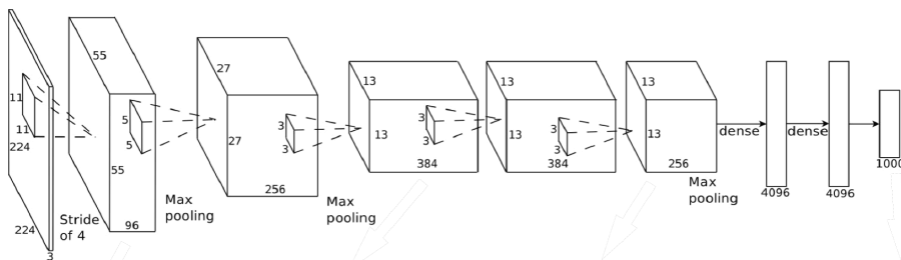
67

- AlexNet (Krizhevsky, 2012)
  - ▣ 60 milhões de parâmetros
    - Entrada: 224 x 224
    - conv1: 96 filtros de 11 x 11 x 3, *stride* 4
    - conv2: 256 filtros de 5 x 5 x 48
    - conv3: 384 filtros de 3 x 3 x 256
    - conv4: 384 filtros de 3 x 3 x 192
    - conv5: 256 filtros de 3 x 3 x 192
    - fc1 e fc2: 4096 neurônios

# Projetando a Rede

68

- AlexNet (Krizhevsky, 2012)



# Projetando a Rede

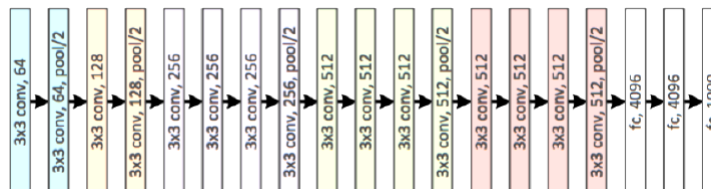
69

- VGG 19 (Simonyan 2014)
  - ▣ +camadas, -filtros = menos parâmetros
  - ▣ Entrada: 224 x 224
  - ▣ Filtros: todos 3 x 3
  - ▣ conv 1-2: 64 filtros + maxpool
  - ▣ conv 3-4: 128 filtros + maxpool
  - ▣ conv 5-6-7-8: 256 filtros + maxpool
  - ▣ conv 9-10-11-12: 512 filtros + maxpool
  - ▣ conv 13-14-15-16: 512 filtros + maxpool
  - ▣ fc1 fc2: 4096 neurônios

# Projetando a Rede

70

- VGG 19 (Simonyan 2014)



# Projetando a Rede

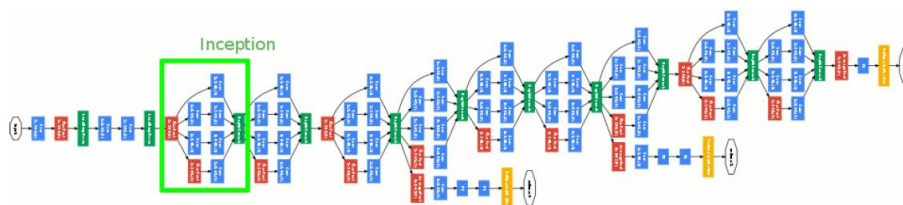
71

- GoogLeNet (Szegedy, 2014)
  - ▣ 22 camadas
  - ▣ Inicia com 2 camadas convolucionais
  - ▣ Camada Inception (“filter bank”)
    - Filtros 1 x 1, 3 x 3, 5 x 5 + max pooling 3 x 3
    - Reduz a dimensionalidade usando filtros 1 x 1
    - 3 classificadores em diferentes partes

# Projetando a Rede

72

- GoogLeNet (Szegedy, 2014)
  - ▣ Blue = convolução
  - ▣ Red = pooling,
  - ▣ Yellow = camada Softmax loss fully connected
  - ▣ Green = normalização ou concatenação





## Utilizando uma rede pré-treinada

75

### Finetuning

#### Dados similares a ImageNet

- Manter o treinamento das camadas convolucionais
- Treinar apenas as camadas FC



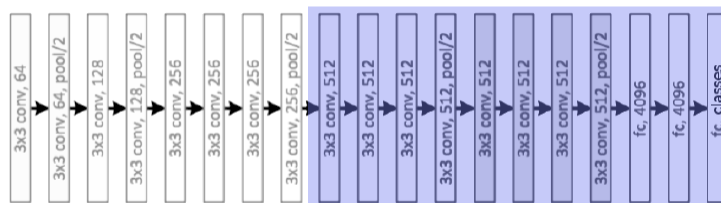
## Utilizando uma rede pré-treinada

76

### Finetuning

#### Dados diferentes da ImageNet

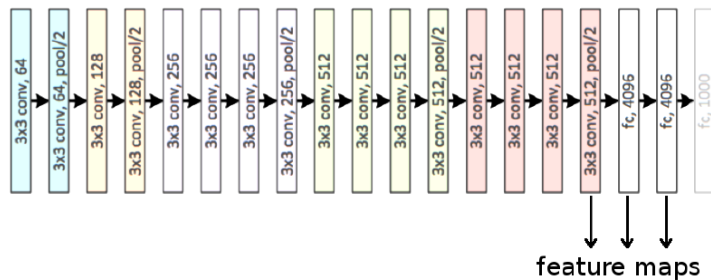
- Manter o treinamento das camadas convolucionais inferiores
- Treinar as outras camadas convolucionais e as camadas FC



## Utilizando uma rede pré-treinada

77

- Extração de atributos
  - ▣ Pegar os valores de ativação das últimas camadas de convolução ou FC
    - Utilizar um classificador externo: SVM, k-NN, Naive Bayes etc



## Aplicações

78

- Reconhecimento de fala e processamento de sinais
  - ▣ Microsoft, 2012: nova versão do MAVIS (*Microsoft Audio Video Indexing Service*), sistema de voz baseado em Deep Learning
  - ▣ ~30% menos erros nos 4 principais conjuntos de *benchmark*

# Aplicações

79

## □ Reconhecimento de objetos

### ▣ Classificação de objetos MNIST

- Banco de dados de dígitos manuscritos
- Taxa de erro: 0,27%

```

0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4
5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5
6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6
7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7
8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8
9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9

```

### ▣ ImageNet

- Base de dados de reconhecimento de imagens naturais
- No primeiro uso, reduziu a taxa de erro de 26,1% para 15,3%
- Atualmente, a taxa de erro é menor que 5%

# Agradecimentos

80

## □ Agradeço aos professores

- ▣ Prof. Anderson Soares – Universidade Federal de Goiás (UFG)
- ▣ Anderson Tenório - Universidade Federal de Pernambuco (UFPE)
- ▣ Prof. Moacir Ponti - Universidade de São Paulo (USP)
- ▣ Prof. Eduardo Bezerra (CEFET/RJ)

## □ pelo material disponibilizado