APRENDIZADO DE MÁQUINA

AUTOENCODERS E TRANSFER LEARNING

PROF. RONALDO CRISTIANO PRATI

ronaldo.prati@ufabc.edu.br

Bloco A, sala 513-2

REDES PROFUNDAS

- A "profundidade" das redes profundas é apenas parte do seu sucesso.
- O excelente desempenho em algumas tarefas se deve a outras duas características:
  - Autoencoders
  - Transferência de aprendizado

REDES AUTOCODIFICADORAS

- Redes autocodificadoras (autoencoders) criam uma representação adjacente ao conjunto de dados de forma não supervisionada (ou auto-supervisionada)
- Permite o aprendizado de representações mais concisas

REDES AUTOCODIFICADORAS

- Características identificadas durante o aprendizado são úteis para uso posterior em tarefas de aprendizado supervisionado.
- Transformações são aplicadas na entrada de acordo com dois tipos de funções:
  - Função de extração de características
  - Função de reconstrução

ENCODER

- **Função de extração de características (encoder)** mapeia o conjunto de treinamento para uma representação latente.
- Mapeia o espaço de dimensão m para um espaço de dimensão k

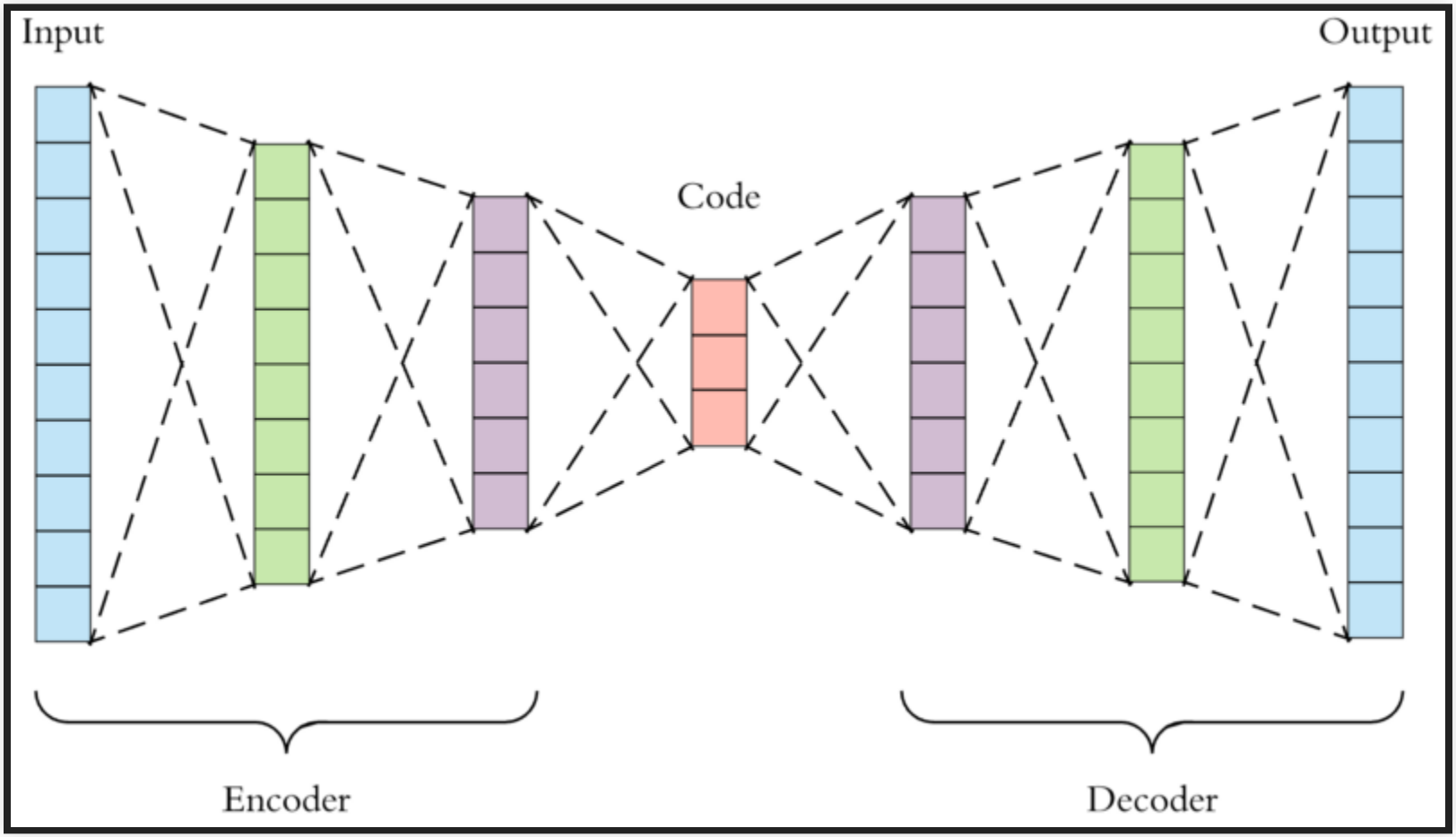
$$h = R^m \rightarrow R^k$$

DECODER

- **Função de reconstrução (decoder)** mapeia a representação produzida por h de volta para o espaço original.
- Mapeia o espaço de dimensão k de volta para o espaço de dimensão m

$$r = R^k \rightarrow R^m$$

AUTOENCODERS



TREINAMENTO

- Um autoencoder é treinado para reproduzir na sua saída a própria entrada.
- O objetivo do treinamento é definir parâmetros em h e r tal que o erro de reconstrução seja minimizado.
- Pode ser treinada usando backpropagation com gradiente descendente, com o cuidado de substituir os valores-alvo desejados pela própria entrada

TREINAMENTO

- Possibilidade: pesos atados (tied weights)
- Há pares de matrizes de pesos em posições simétricas na rede, uma é a transposta da outra!
  - menos parâmetros para otimizar;
  - previne soluções degeneradas.

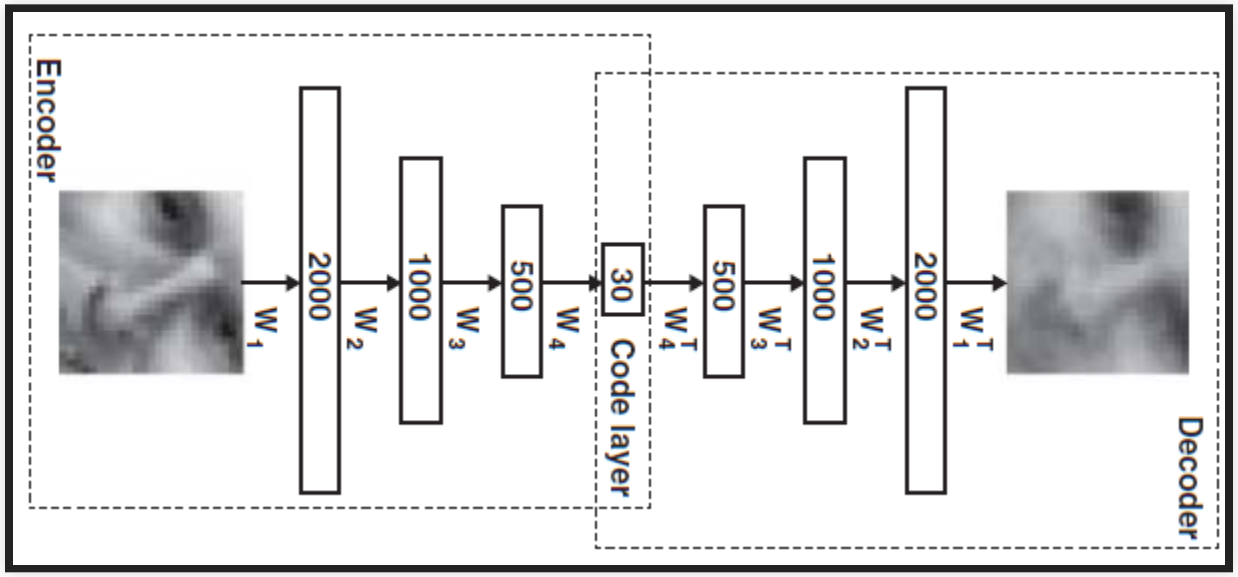
SUBCOMPLETAS X SUPERCOMPLETAS

- Se a representação latente em uma autocodificadora tem dimensão K:
  - $k < m$: undercomplete autoencoder;
  - $k > m$: overcomplete autoencoder.
- A escolha de K determina a quantidade de unidades da camada intermediária central, que tipo de informação o autoencoder pode aprender acerca da distribuição de entrada.

CASO $K < M$ (BOTTLENECK)

- **Objetivo:** aprender uma representação compacta do conjunto de dados.
- Exemplo:
  - Considere como entrada o conjunto de disciplinas do BC&T, podemos criar uma camada K com o número de cursos pós-BC&T que tentaria codificar qual é o curso que o aluno pretende seguir.

CASO $K < D$ (BOTTLENECK)



CASO $K > M$

- **Objetivo:** encontrar características da entrada para posteriormente serem apresentadas a um classificador linear (SVM, k-NN, etc).
- Problema potencial no treinamento: autocodificadora apenas copia os m bits da entrada para m unidades na camada central.
- aprende a função identidade, i.e. $f(x) = x$, deixando de usar $k - m$ unidades nessa camada.

AUTOENCODER COM FILTRAGEM DE RUÍDO

- **Ideia básica:** fazer com que a representação aprendida seja robusta a ruídos nos dados.
- Aplicar alterações aleatórias em cada exemplo de treinamento antes de apresentá-lo à rede.
- Alternativas:
  - com probabilidade p, atribuir zero a um atributo do exemplo.
  - perturbar cada componente de x por meio de um ruído gaussiano aditivo.

AUTOENCODER COM FILTRAGEM DE RUÍDO

CONTRASTIVE AUTOEMCODER

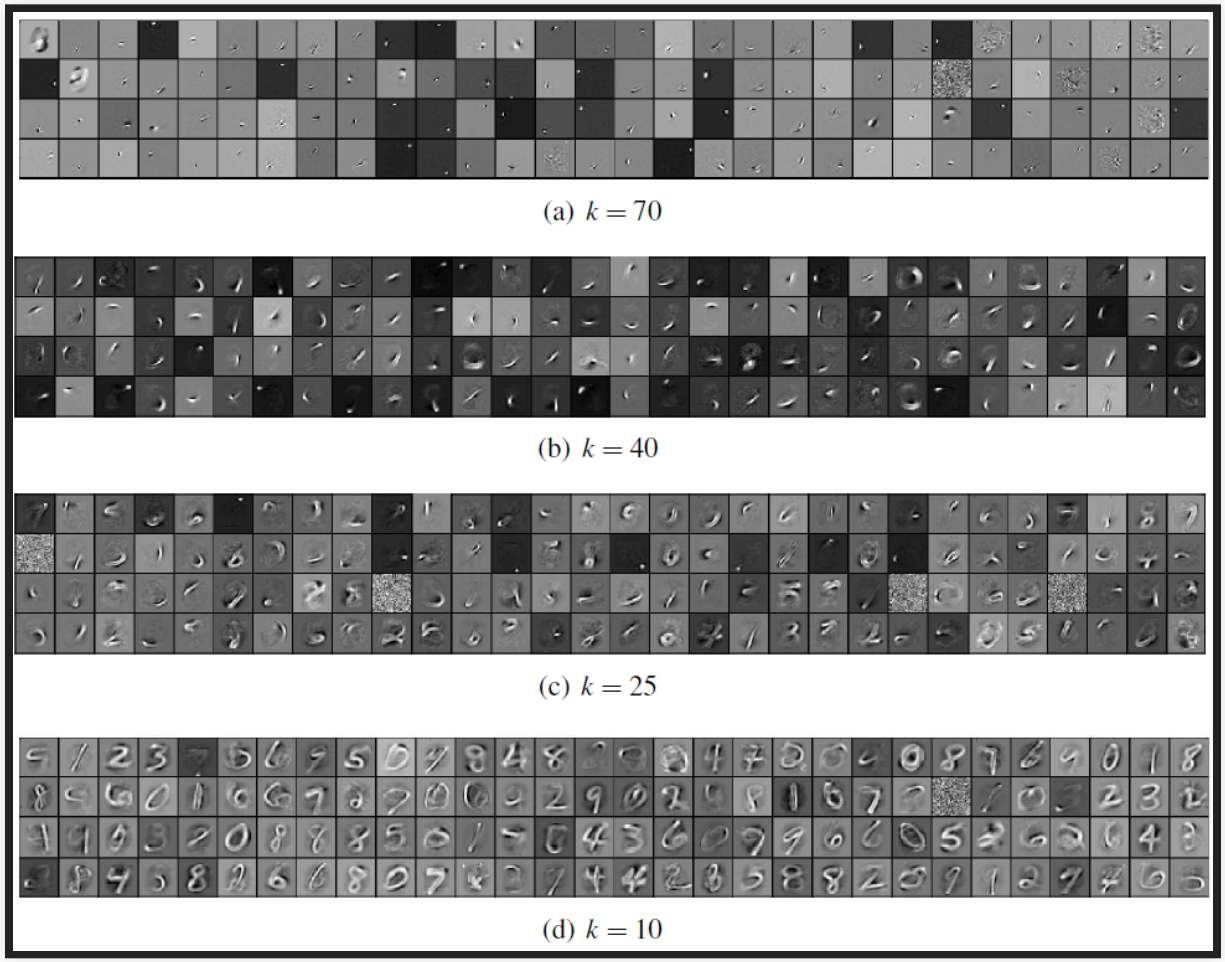
- **Ideia básica:** Ideia básica: adicionar uma penalização à função de perda para penalizar representações indesejadas.
- similar a regularização

$$J(\theta) = \frac{1}{2m} \sum_{i=i}^{m} (h_\theta(x^i) - x^i)^2 + \lambda \| \frac{\partial h_\theta(x)}{\partial x} \|$$

SPARSE AUTOENCODER

- **Objetivo:** fazer com que apenas uma pequena quantidade de unidades da camada oculta seja ativada para cada padrão de entrada.
- A esparsidade pode ser obtida por meio de "*dropout*": mantendo apenas as k unidades mais ativas e tornando todas as demais unidades iguais a zero.

SPARSE AUTOENCODER



(a) $k = 70$

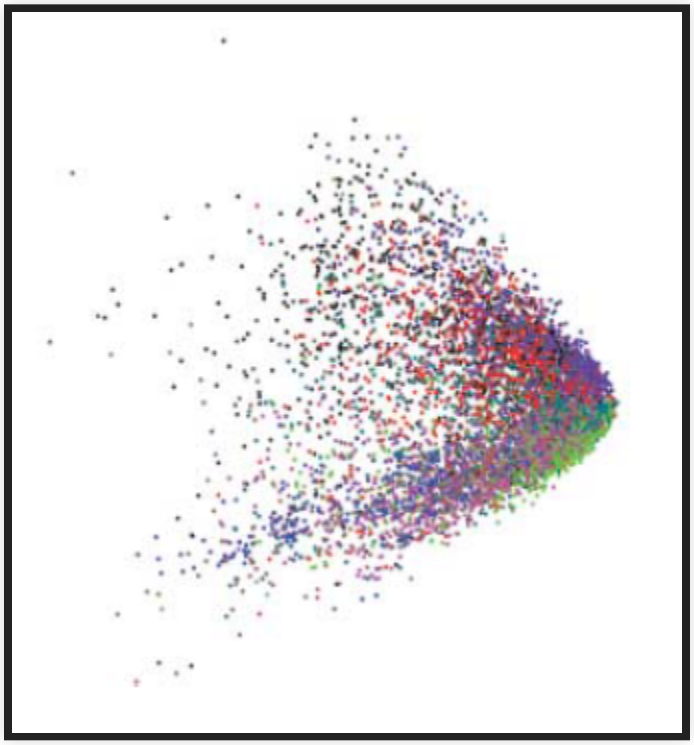
(b) $k = 40$

(c) $k = 25$

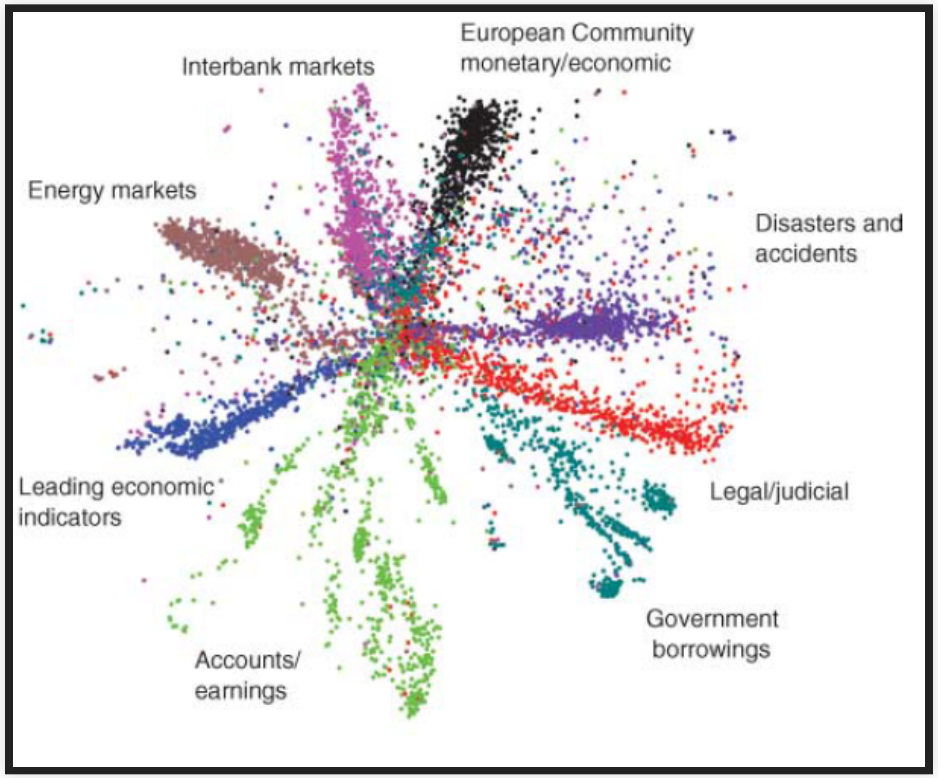
(d) $k = 10$

AUTOENCODER - APLICAÇÕES

Redução da dimensionalidade



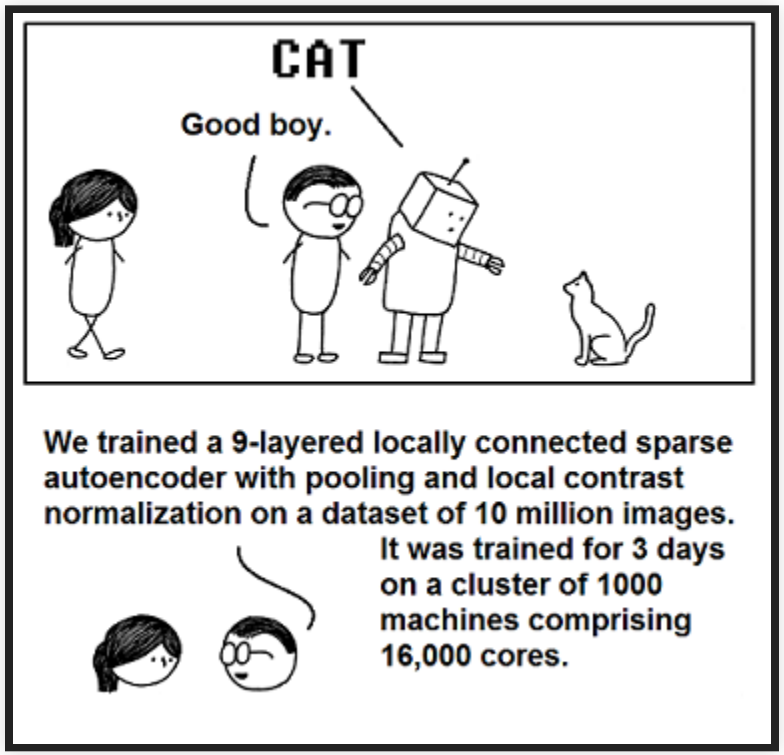
PCA



Autoencoder

AUTOENCODER - APLICAÇÕES

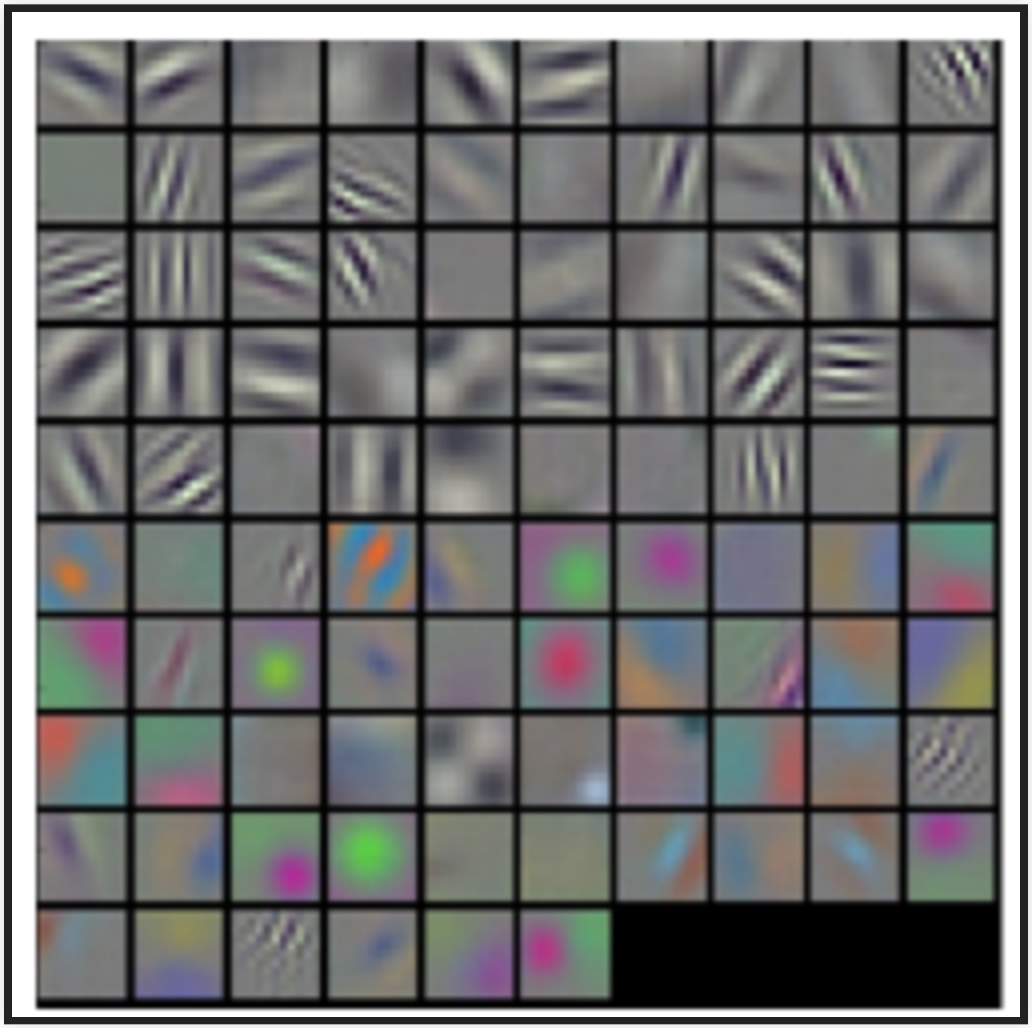
- Geração de conceitos
- 10M vídeos do YouTube, 1 frame por vídeo (200x200)
- A rede aprendeu os conceitos de face humana e de face de gatos.



TRANSFER LEARNING

- Em redes profundas, as primeiras camadas de uma rede neural são as mais difícies (demoradas) de treinar
- Normalmente requerem muitos exemplos
- Entretanto, elas são geralmente responsáveis por aprender características simples.

- primeira camada de uma convnet para classificação de imagens



TRANSFER LEARNING

- Camadas mais profundas devem capturar características que são mais particulares ao problema específico
- Essas camadas são mais simples (rápidas) de treinar, uma vez que ajustar o peso delas tem um impacto maior no resultado final

TRANSFER LEARNING

- Configuraçẽos famosas e vencedoras de competições são difíceis de treinar do zero
  - Datasets grandes
  - Muitas iterações
  - Necessitam de alto poder computacional
  - E experimentação para ajustar os parâmetros

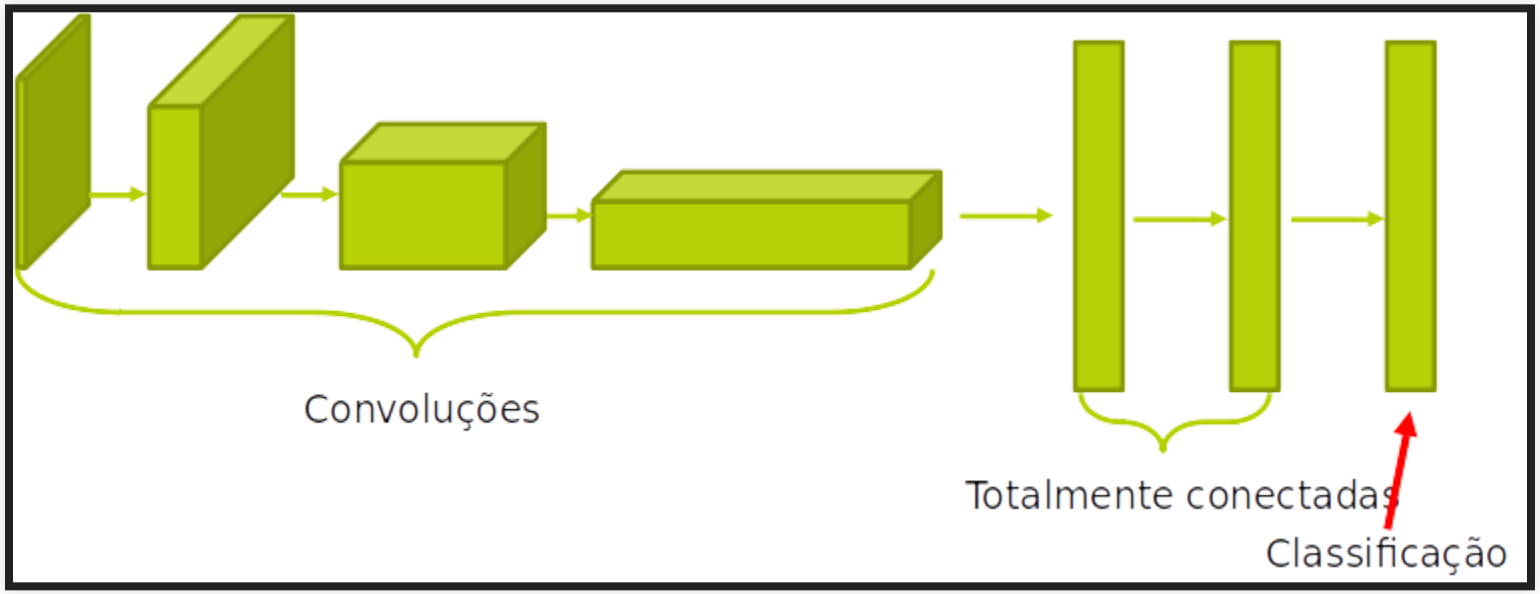
TRANSFER LEARNING

- Entretanto, os atributos básicos (linhas, formas) aprendidas em estágios anteriores podem ser úteis em outros contextos
- O resultado do treinamento são as matrizes de pesos (números) que são facilmente armazenados

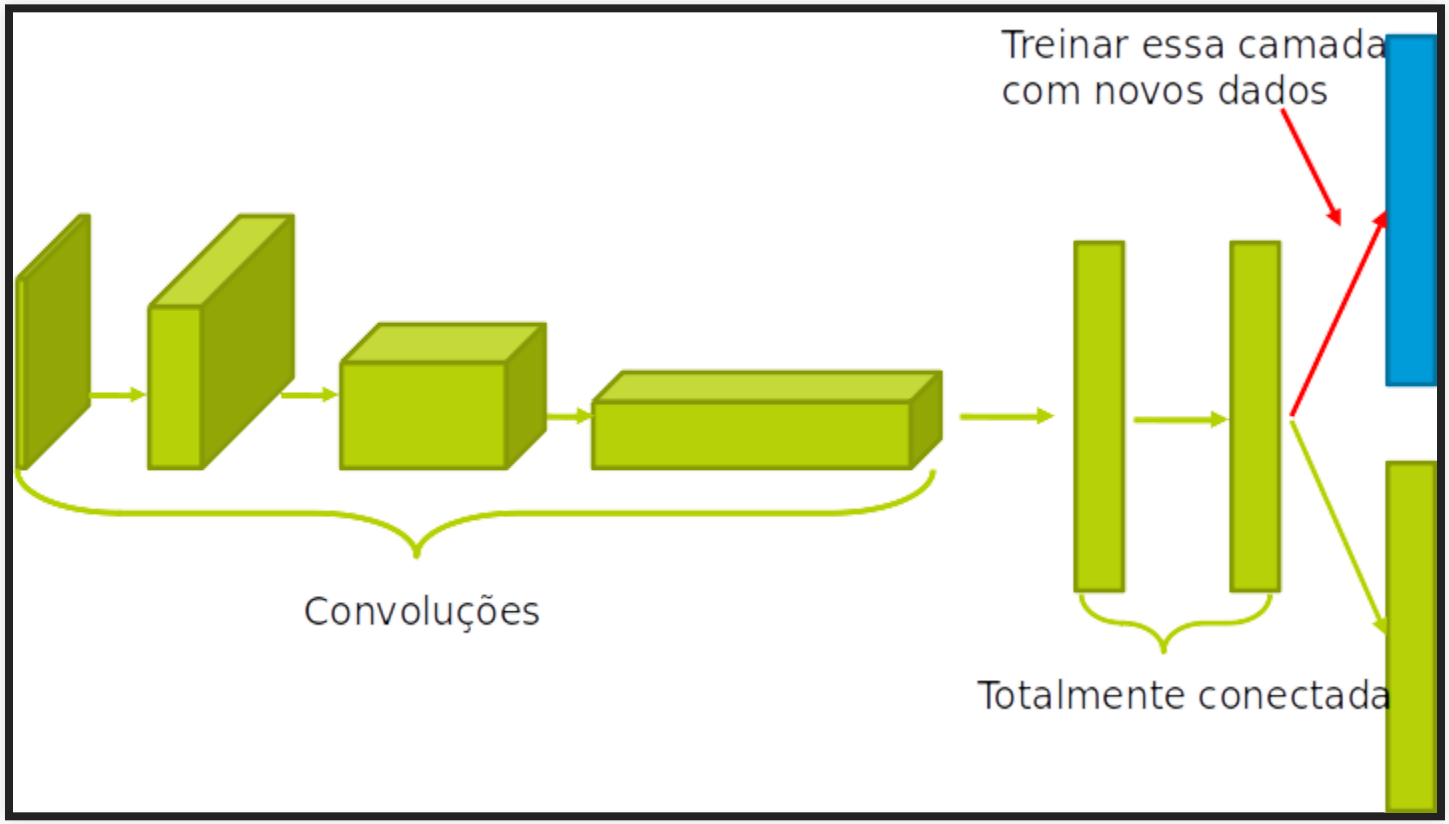
TRANSFER LEARNING

- **Ideia:** armazenar o peso de redes pré-treinadas, e re-treinar apenas as camadas específicas para a aplicação
- Isso é chamado de **transferência de aprendizado**

TRANSFER LEARNING



TRANSFER LEARNING



TRANSFER LEARNING

- A adição de camadas pré-treinadas em um data set específico também é chamada de "ajuste-fino" (fine-tunning).
- Existem várias opções em "quanto" e "até que camada" fazer o ajuste fino.
  - Devemos treinar só a última camada?
  - Mais de uma?
  - Ajustar os pesos da rede inteira ou só das camadas retreinadas?

GUIDELINES

- Apesar de não existir um regra única, existem alguns guidelines:
- Quanto mais similir os novos dados são do problema, menos ajuste fino é necessário.
  - Usar uma parte de uma rede treinada no ImageNet para classificar entre pessoas e gatos pode requerer pouco ajuste, uma vez que o features relevantes já teriam sido aprendidas

GUIDELINES

- Quanto mais dados houver sobre o problema específico, mais a rede se benificiará de um ajuste mais longo e profundo.
  - Se você tiver somente 100 gatos e 100 pessoas nos dados específicos, você provavelemente faria um peque ajusto. Se tivermos 10.000 de cada, um maior ajuste poderia ser utilizado.

GUIDELINES

- Se os dados são de domínio substancialmente diferentes que os dados originais, transfer learning pode ser de pouca valia.
  - Por exemplo, uma rede treinada para reconhecimento de escrita em pode ser pouco útil para diferenciar gatos de humanos.