

Novas Estratégias de Treinamento e Transferência de Aprendizado

Redes Neurais e Arquiteturas Profundas

Moacir Ponti
ICMC, Universidade de São Paulo

`www.icmc.usp.br/~moacir — moacir@icmc.usp.br`

São Carlos-SP/Brasil – 2021

Agenda

Treinando redes profundas em cenários reais

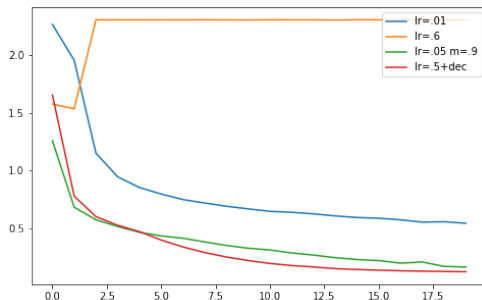
- Suposições para convergência e aprendizado

- Estratégias para melhorar generalização

- Normalização de dados

Transferência de aprendizado

Anteriormente...



Batch size = 2, Erros de Validação MSE = 27.0030, MAE = 3.6657

Batch size = 4, Erros de Validação MSE = 24.0659, MAE = 3.2671

Batch size = 8, Erros de Validação MSE = 24.2604, MAE = 3.2732

Batch size = 16, Erros de Validação MSE = 15.8703, MAE = 2.9850

Batch size = 32, Erros de Validação MSE = 19.7997, MAE = 3.5542

Batch size = 64, Erros de Validação MSE = 26.4097, MAE = 4.1476

Batch size = 128, Erros de Validação MSE = 46.3122, MAE = 5.4005

Batch size = 256, Erros de Validação MSE = 477.0085, MAE = 18.6275

Algumas suposições que fizemos

Dados de treinamento

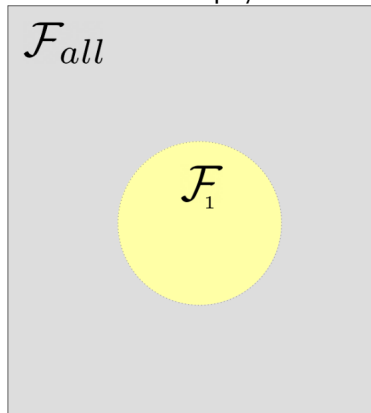
- ▶ Limpos
 - ▶ Representativos e bem definidos com relação à tarefa: classes, valores da regressão, etc.
 - ▶ Baixa taxa de erros de rótulo
 - ▶ Quantidade de dados é suficiente
-
- ▶ E se não for possível?
 - ▶ Riscos: *overfitting*, baixa generalização, maior dificuldade no treinamento.

Complexidade de modelos: "viés" segundo a Teoria do Aprendizado Estatístico

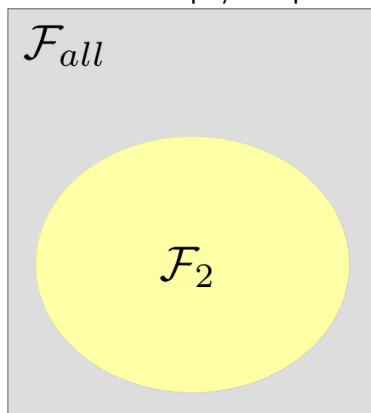
- ▶ Lembrando: Aprendizado de Máquina pode ser formulado como sendo aprender os parâmetros de $f : X \rightarrow Y$
- ▶ Um algoritmo ajusta f a partir de um espaço de funções admissíveis \mathcal{F} :
 - ▶ "muitas" funções: mais graus de liberdade, menor garantia de convergência, possível *overfitting*;
 - ▶ "poucas" funções: menos graus de liberdade, maior garantia de convergência, possível *underfitting*.

Espaço de funções admissíveis

Viés forte: espaço restrito

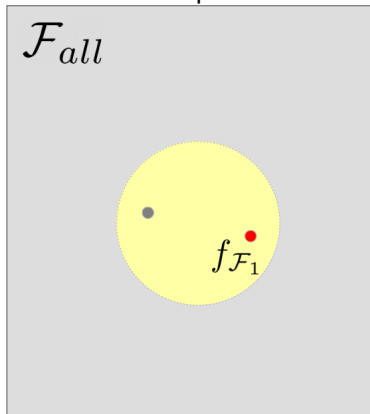


Viés fraco: espaço amplo



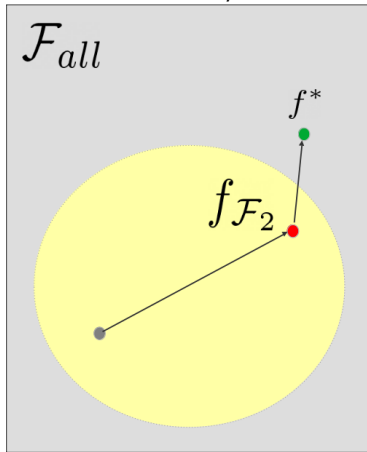
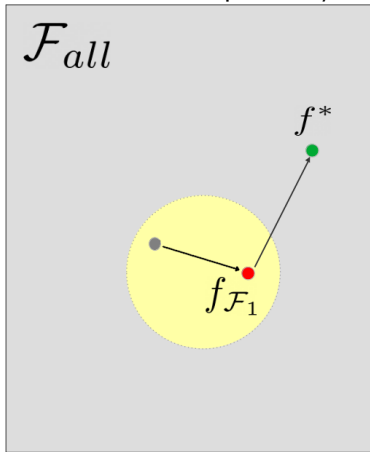
Espaço de funções admissíveis

De um modelo arbitrário para o melhor no espaço

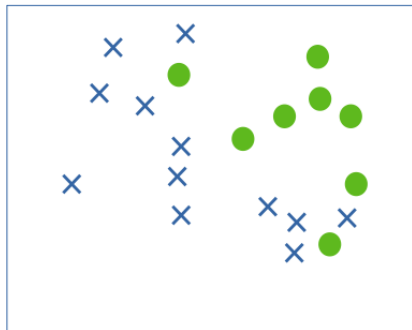


Erros quando definindo o espaço de funções admissíveis

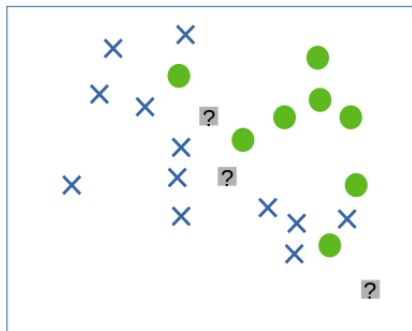
Erro de aproximação vs Erro de estimação



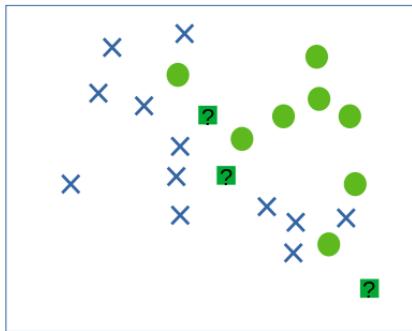
Exemplo com KNN



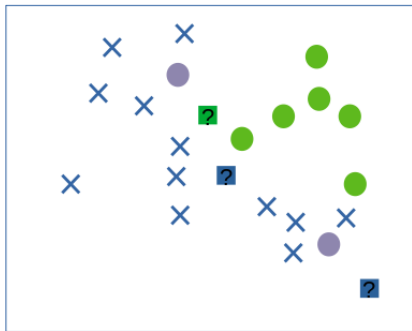
Exemplo com KNN



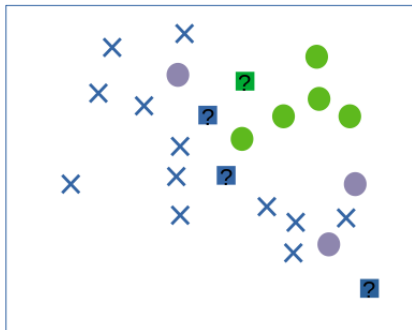
1-NN

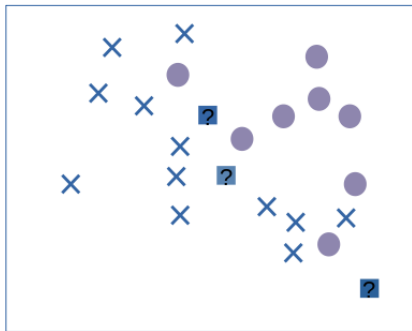


3-NN

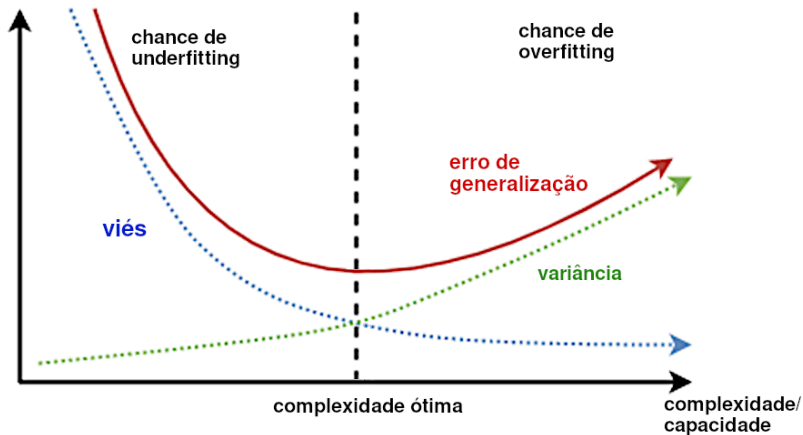


5-NN





Dilema viés e variância



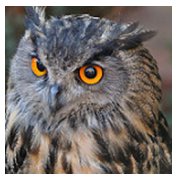
Controvérsias com Deep Learning

Marcus (2018) em "Deep Learning: a critical appraisal":

"... sistemas que se baseiam em Deep Learning frequentemente devem generalizar para além de dados específicos... mas a garantia de performance em alta qualidade nesses cenários é mais limitada."

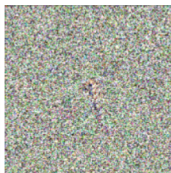
Ataques adversariais

Artigo "Deep Neural Networks are easily fooled"



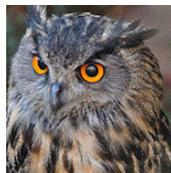
owl
(73% confidence)

+ .05 x



cheetah gradient
features

=



cheetah
(99% confidence)

Ataques adversariais

Pixel attack: assume que há acesso ao conjunto de treinamento



Conjunto de treinamento



CAR (56%)
AIRPLANE (30%)
HORSE (7%)

Ataques adversariais

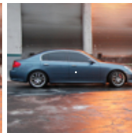
Pixel attack: assume que há acesso ao conjunto de treinamento



Conjunto de treinamento

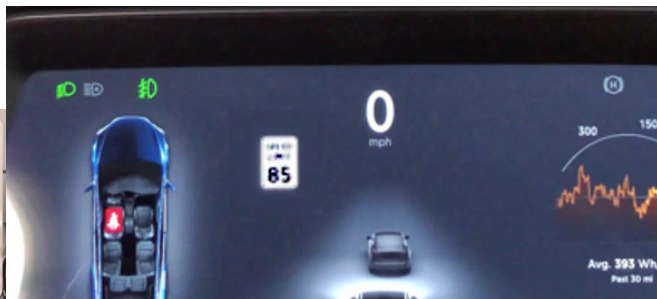


CAR (56%)
AIRPLANE (30%)
HORSE (7%)



AIRPLANE (61%)
CAR (31%)
HORSE (6%)

Ataques adversariais



Zhang et al (2017)

"... nossos experimentos estabeleceram que redes convolucionais profundas do estado da arte (...) *facilmente ajustam rótulos aleatórios* nos dados de treinamento."

UNDERSTANDING DEEP LEARNING REQUIRES RE-THINKING GENERALIZATION

Chiyuan Zhang*
Massachusetts Institute of Technology
chiyuan@mit.edu

Samy Bengio
Google Brain
bengio@google.com

Moritz Hardt
Google Brain
mrtz@google.com

Benjamin Recht†
University of California, Berkeley
brecht@berkeley.edu

Oriol Vinyals
Google DeepMind
vinyals@google.com

ABSTRACT

Despite their massive size, successful deep artificial neural networks can exhibit a remarkably small difference between training and test performance. Conventional

Agenda

Treinando redes profundas em cenários reais

Suposições para convergência e aprendizado

Estratégias para melhorar generalização

Normalização de dados

Transferência de aprendizado

(I) Regularização

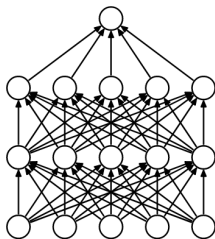
Relembrando a regularização L2 (ou de Tikhonov)

$$\ell(\Theta) = \frac{1}{N} \sum_{i=1}^N \ell_i(x_i, y_i, \Theta) + \lambda \frac{1}{2} \|\Theta\|^2$$

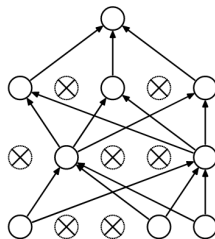
- ▶ Objetivo: limitar a capacidade do modelo de se especializar demais nos dados
- ▶ Formas:
 - ▶ Global: na função de perda ponderada por λ
 - ▶ Definindo λ_l por camada (ou grupos de camadas)
- ▶ Interpretação: vê cada entrada como sendo de maior variância

(II) Dropout

- ▶ Objetivo: limitar a capacidade de certos parâmetros do modelo a memorizarem os dados
- ▶ Implementado na forma de "camada"
- ▶ Em cada iteração, desliga ativações de neurônios aleatoriamente com probabilidade p
- ▶ Interpretação: treinamento com técnica "Bagging"



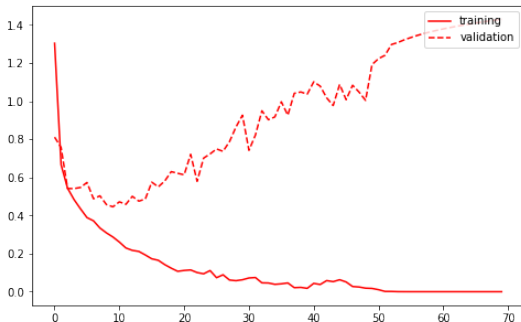
(a) Standard Neural Net



(b) After applying dropout.

(III) Parada precoce

- ▶ Objetivo: evitar que o modelo memorize os dados de treinamento ao treinar por muitas épocas
- ▶ Acompanhar um conjunto de validação e interromper de acordo com a relação do custo no treinamento e validação



(V) Coletar mais dados

- ▶ Objetivo: impedir que o treinamento considere apenas um conjunto limitado de exemplos
- ▶ Baseado na lei dos grandes números, quanto maior a amostra, teremos um melhor estimador

(VI) Aumentação de dados/Data augmentation

- ▶ Objetivo: gerar exemplos artificiais na esperança de que melhore as propriedades de convergência
- ▶ Implementado por meio da manipulação de exemplos existentes, ou sua combinação
- ▶ Exemplos:
 - ▶ Dados estruturados: SMOTE
 - ▶ Não estruturados: rotação, corte, injeção de ruído, e outros que não descaracterizem os dados
 - ▶ Dropout na camada de entrada: eliminando features aleatoriamente a cada iteração.

(VI) Aumentação de dados/Data augmentation (cont.)

Dica para melhoria de performance final

- ▶ Para cada exemplo de teste:
 1. Gerar m exemplos com aumento de dados
 2. Predizer o resultado para os m exemplos
 3. Combinar as predições: por média, maioria ou outro método

Agenda

Treinando redes profundas em cenários reais

Suposições para convergência e aprendizado

Estratégias para melhorar generalização

Normalização de dados

Transferência de aprendizado

Normalização de dados

- ▶ Exemplos de técnicas:
 - ▶ **Normalização (ou padronização) z-score**: valores com média zero e desvio padrão 1;
 - ▶ **Normalização min-max**: valores no intervalo 0-1.
- ▶ Objetivo: facilitar otimização ao normalizar/padronizar a magnitude dos valores utilizados no treinamento:
 - ▶ suaviza as ativações dos neurônios, reduzindo a variância do gradiente;
 - ▶ ataca o problema de "desaparecimento" do gradiente (vanishing gradient) em particular para redes profundas.

Normalização de dados

- ▶ Podemos usar como pré-processamento considerando todos os dados de treinamento
- ▶ Mas durante o treinamento pode também ser aplicado ao batch ou às camadas

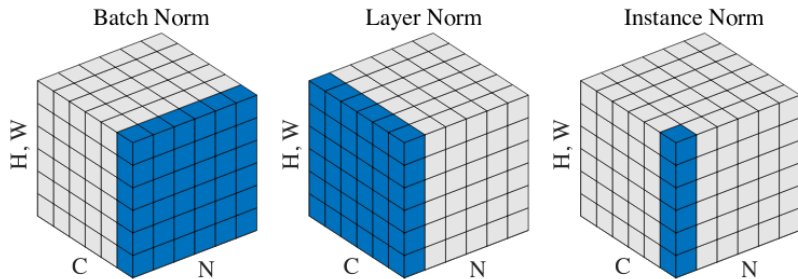
Tipos de normalização baseada em camadas!

Batch

Camada

Instância

Normalização de Dados

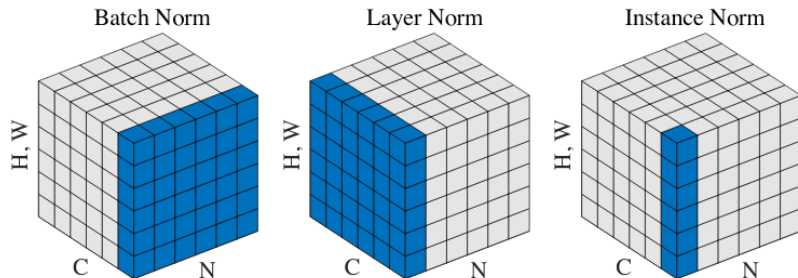


- ▶ C = canais, ou mapas de ativação
- ▶ N = instâncias no batch
- ▶ H, W = dimensões dos mapas de ativação (ex. altura x largura)

A operação aprende ainda os valores γ, β para transformação linear dos dados após normalizados:

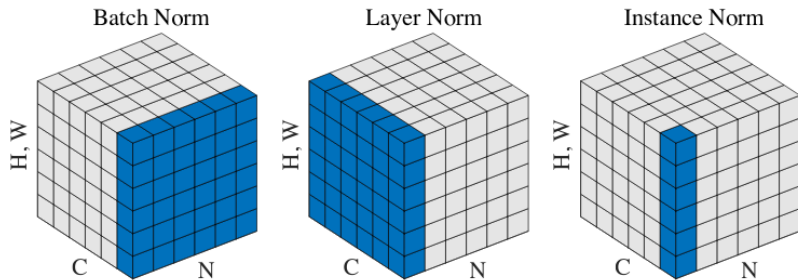
$$\gamma x_i + \beta$$

Normalização de Dados: Batch



- ▶ **Batch normalization (BN):** para cada batch
 - ▶ média e desvio calculados por canal (total C)
 - ▶ normalização com relação ao canal ao longo de N instâncias no batch
- ▶ Funciona melhor com batchsize ≥ 32

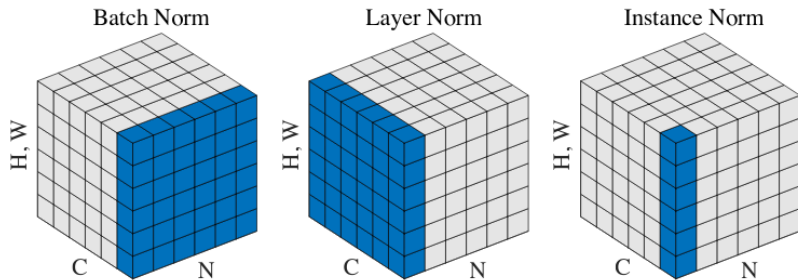
Normalização de Dados: Layer



► Layer normalization (LN):

- média e desvio calculados por instância (total N)
 - normalização com relação à cada instância ao longo de todas as ativações de todos os canais
- Independe do tamanho do batch, mais comum em redes recorrentes e adversariais

Normalização de Dados: Instance



- ▶ **Instance normalization (IN):**
 - ▶ média e desvio calculados por instância e canal (total $N \times C$)
 - ▶ normalização com relação à cada instância ao longo de cada canal individualmente
- ▶ Independe do tamanho do batch, mais comum em redes recorrentes e adversariais

Agenda

Treinando redes profundas em cenários reais

Suposições para convergência e aprendizado

Estratégias para melhorar generalização

Normalização de dados

Transferência de aprendizado

Quando o assunto é volume de dados

Nem sempre...

- ▶ é possível coletar mais
- ▶ aumento é efetiva

Transferência de aprendizado

Utilizar modelo treinado em uma determinada tarefa ou domínio, aproveitando o aprendizado para uma outra tarefa ou domínio alvo.

Transferência de aprendizado

Modos mais comuns

- ▶ Ajuste-fino / adaptação dos parâmetros
- ▶ Extração de características

Ajuste-fino / adaptação dos parâmetros

Exemplo: classificação de imagens

Modelo fonte: CNN treinada na ImageNet

- ▶ Transferência de aprendizado
 - ▶ Redefinir a última camada (inicialização aleatória) com o número de classes desejado
 - ▶ Realizar treinamento a partir dos pesos pré-treinados
 - ▶ Permitir adaptação apenas da últimas camadas, congelando as demais



Ajuste-fino / adaptação dos parâmetros

Exemplo: classificação de imagens

Modelo fonte: CNN treinada na ImageNet

- ▶ Ajuste-fino:
 - ▶ Comumente feito após o anterior, em que já temos o classificador treinado
 - ▶ (Opcionalmente) Re-inicializar ou inserir novas camadas densas ocultas
 - ▶ Permitir adaptação de camadas do meio da rede (sem reinicializar seus pesos), congelando algumas iniciais

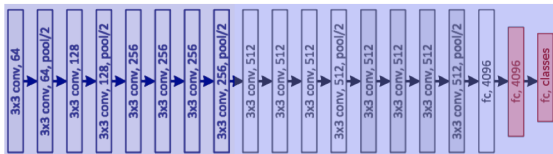


Ajuste-fino / adaptação dos parâmetros

Exemplo: classificação de imagens

Modelo fonte: CNN treinada na ImageNet

- ▶ Ajuste-fino – opção alternativa
 - ▶ Depende de maior quantidade de dados
 - ▶ (Opcionalmente) Re-inicializar ou inserir novas camadas densas ocultas
 - ▶ Permitir adaptação de toda a rede (sem reinicializar seus pesos)



Transferência de aprendizado

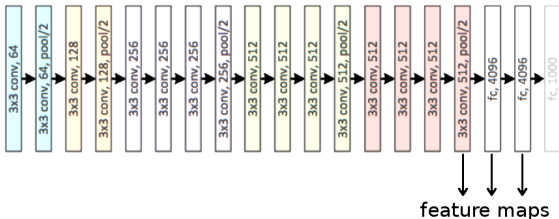
Dicas

- ▶ CNNs com menos parâmetros costumam generalizar melhor para dados muito diferentes do treinamento
- ▶ Exemplos: MobileNet, SqueezeNet, etc. funcionam melhor em imagens médicas do que ResNet e Inception.
- ▶ Ajuste-fino pode não convergir se tivermos poucos dados, ex. menos de 100 instâncias por classe.

Extração de características

Características para dados não estruturados

- ▶ Carregar rede neural treinada em grande base de dados
- ▶ Passar exemplos de sua base de dados pela rede para predição (não treinamento!).
- ▶ Obter os mapas de ativação de alguma camada



Extração de características

Dicas

- ▶ Aplicar redução de dimensionalidade baseada em PCA, Product Quantization ou outra
- ▶ Treinar modelo de aprendizado raso com maiores garantias de aprendizado com poucos dados: SVM, árvore de decisão, etc.
- ▶ Essas características também são efetivas para recuperação baseada em conteúdo
- ▶ Podem ser usados métodos de projeção para as características aprendidas: tSNE, UMAP, PCA.

Mensagem da aula

- ▶ Deep Learning não pode ser tratado como panacéia;
- ▶ Há ainda preocupações sobre sua capacidade de generalização e fragilidade a ataques;
- ▶ Sua grande utilidade está no aprendizado de representações, em particular para dados não estruturados...
 - ▶ representações que parecem ter excelente capacidade de transferência

Bibliography I



Moacir A. Ponti, Fernando dos Santos, Leo Ribeiro, Gabriel Cavallari. **Training Deep Networks from Zero to Hero: avoiding pitfalls and going beyond.**

SIBGRAPI, 2021. Tutorial.

<https://arxiv.org/abs/2109.02752>



Moacir A. Ponti, Leo Ribeiro, Tiago Nazaré, Tu Bui, John Collomosse. **Everything You Wanted to Know About Deep Learning for Computer Vision but were Afraid to Ask.**

SIBGRAPI-T, 2017. Tutorial.



Moacir A. Ponti, **Introduction to Deep Learning (Code)**.

Github Repository:

https://github.com/maponti/deeplearning_intro_datascience

CNN notebook: <https://colab.research.google.com/drive/1EnNjtzdw8ftI07I9xCUhb-ovq1iNy4pf>