Reconhecimento de *Icebergs* em Imagens Adquiridas por Satélites

Escola Politécnica da Universidade de São Paulo

Trabalho de conclusão de curso da disciplina PSI-5886/2018
Prof. Dr. Emílio Del Moral Hernandez

Rede Neural Convolucional aplicada ao desafio "Statoil Iceberg Recognition"

INTEGRANTES
Alfredo Ferraz - 417996
André Lopes - 9527845
Michel A. L. Vinagreiro - 10579717
William Contesini - 7161462

Conteúdo da Apresentação

1. ***Introdução*** *(William Contesini)*
   - *Definição do Problema*
   - *Imagens Adquiridas pelo Satélite*

2. ***Arquitetura*** *(Michel A. L. Vinagreiro)*
   - *CNNs*
   - *Camadas da CNN*

3. ***Treinamento da Rede Neural*** *(André Lopes)*
   - *Técnicas Adotadas*
   - *Resultados Obtidos*
   -

4. ***Conclusões*** *(Alfredo Ferraz)*

Definição do Problema

Statoil/C-CORE Iceberg Classifier Challenge [1]

- *Icebergs* à deriva constituem um grande perigo para embarcações em alto-mar.
- Em áreas remotas e de clima severo, a única opção viável para reconhecimento de obstáculos às embarcações é via satélite.

Fig. 1: Imagem de um *iceberg* [1]

- Utilizando Inteligência Artificial e imagens obtidas por satélite, o desafio consiste na **implementação de um algoritmo de detecção e discriminação de objetos identificados em alto-mar.**

[1] https://www.kaggle.com/c/statoil-iceberg-classifier-challenge

Sentinel-1 Satellite

Características do satélite *Sentinel-1:*

Fig. 2: Satélite Sentinel-1 [2]

- 600 km acima da Terra;
- *C-Band:* “vê” através da escuridão, chuva, nuvens e neblina;
- Imagens dia e noite;
- Polarização horizontal e vertical;
- Ângulo de incidência.

[2] https://en.wikipedia.org/wiki/Sentinel-1

Imagens Obtidas

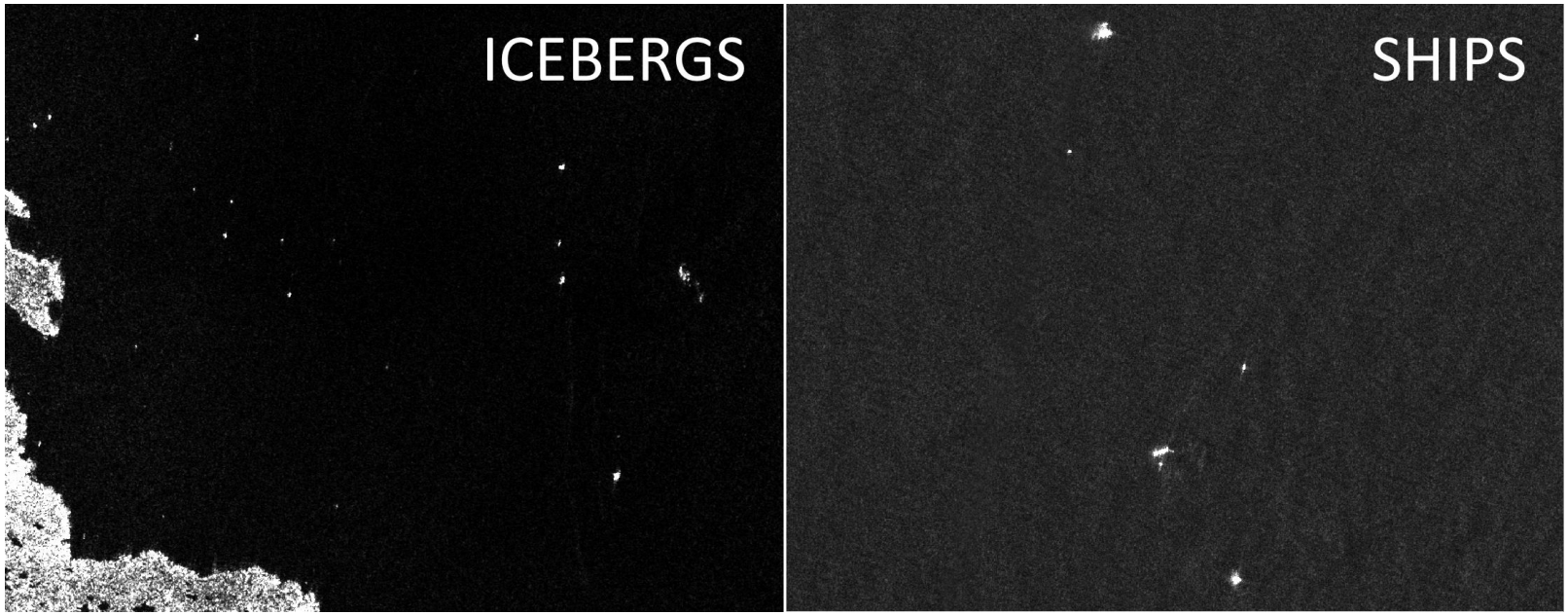

Fig. 3: Imagens obtidas para análise e caracterização [3].

[3] https://www.kaggle.com/c/statoil-iceberg-classifier-challenge#Background

Imagens a Classificar

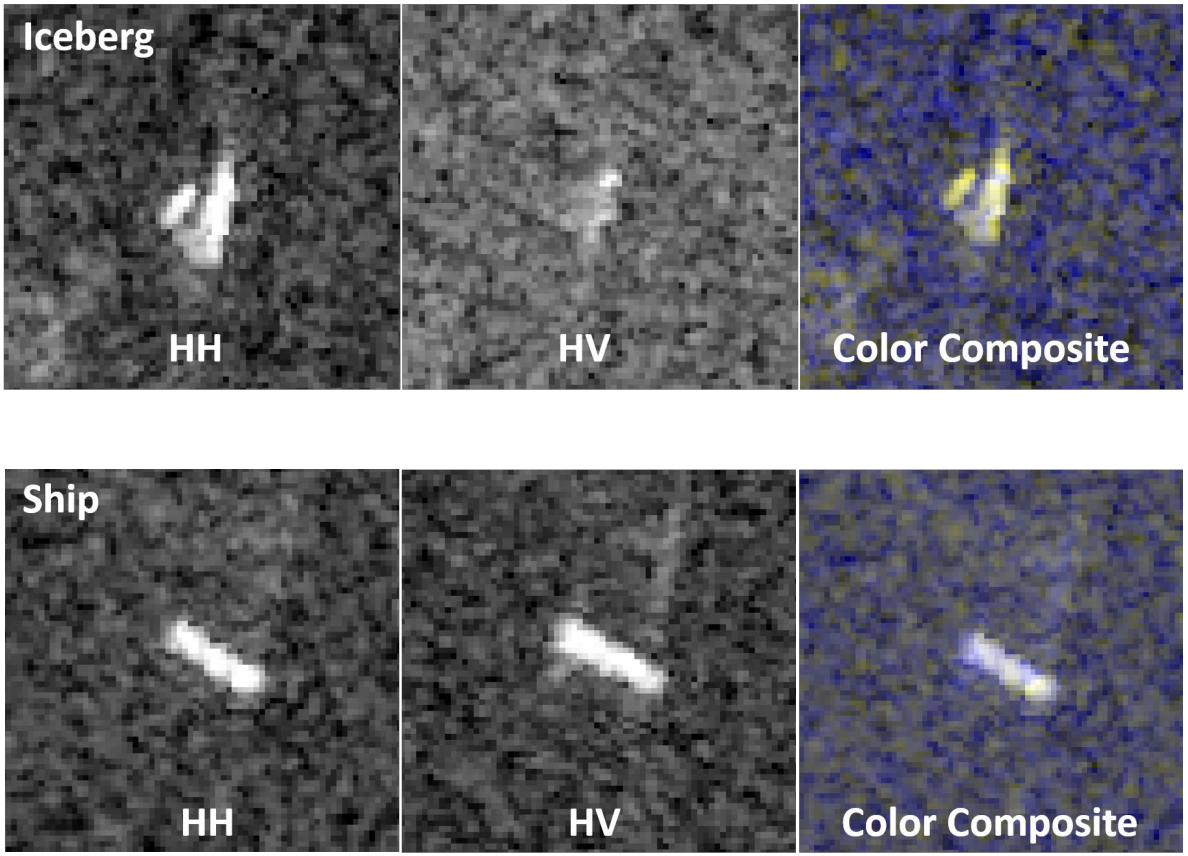

Fig. 4: Imagens adquiridas de *Icebergs* e navios para serem classificadas [3].

[3] https://www.kaggle.com/c/statoil-iceberg-classifier-challenge#Background

Arquitetura da Rede Utilizada: Camada de Convolução

A seguir temos uma animação representando o princípio de atuação da camada de convolução.

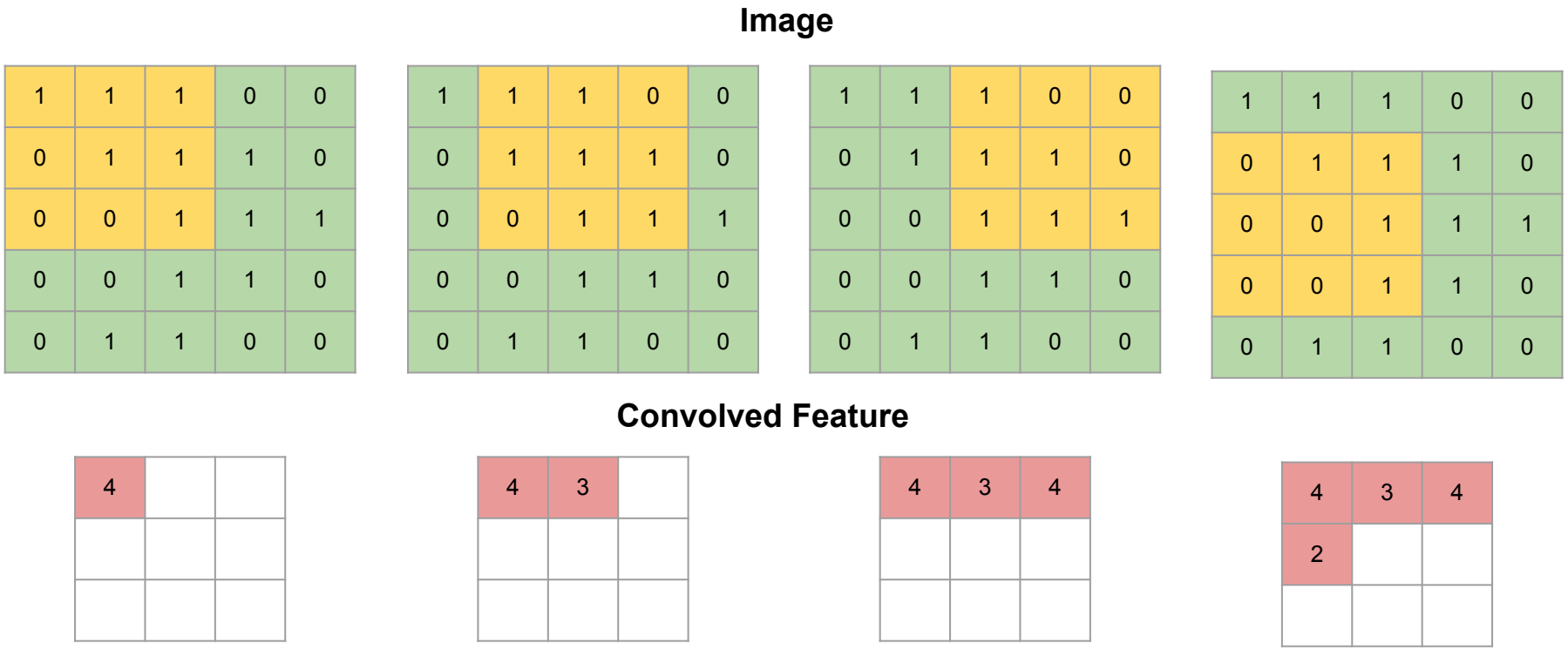

Fig. 5: Animação representando o processo de convolução 2D, elaborada pelo autor.
(a figura original é um GIF, mas foi colocada lado-a-lado para visualização em PDF)

Arquitetura da Rede Utilizada: Max Pooling

Figura mostrando o funcionamento da camada de *Max Pooling* - de cada setor (quadriculado vermelho) é gravada a célula de maior valor, reduzindo a dimensão da matriz de entrada.

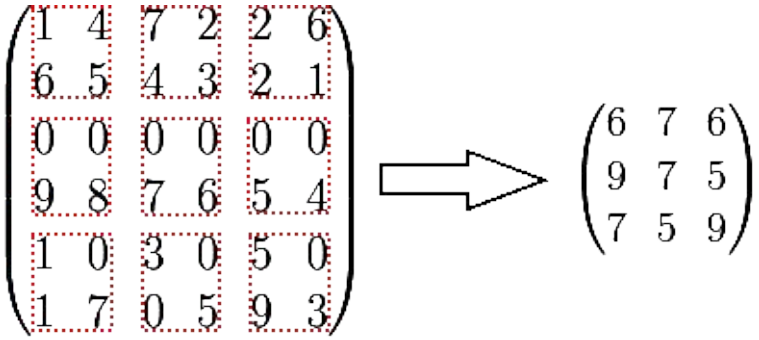

Fig. 6: Processo de *MaxPooling*, elaborado pelo autor.

Arquitetura da Rede Utilizada: Função de Ativação ReLu

Função *ReLu,* utilizada como função de ativação dos neurônios artificiais.

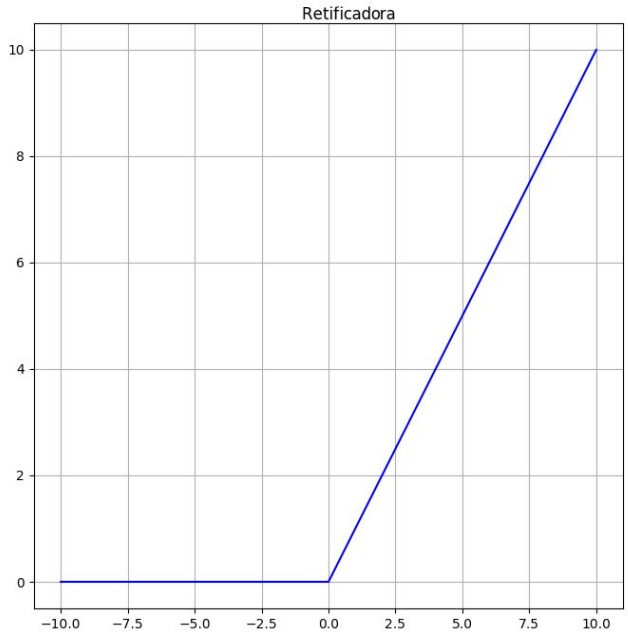

Fig. 7: Comportamento da função Linear Retificada, elaborado pelo autor.

Arquitetura da Rede Utilizada: Batch Normalization

- Técnica para melhorar desempenho e estabilidade das RNAs;
- Fornece qualquer camada com entradas que têm média zero e variação unitária;
- Age como regularizador.

Topologia de uma CNN - *AlexNet*

Topologia de uma CNN genérica do tipo *AlexNet* servida de inspiração para a resolução do problema do grupo.

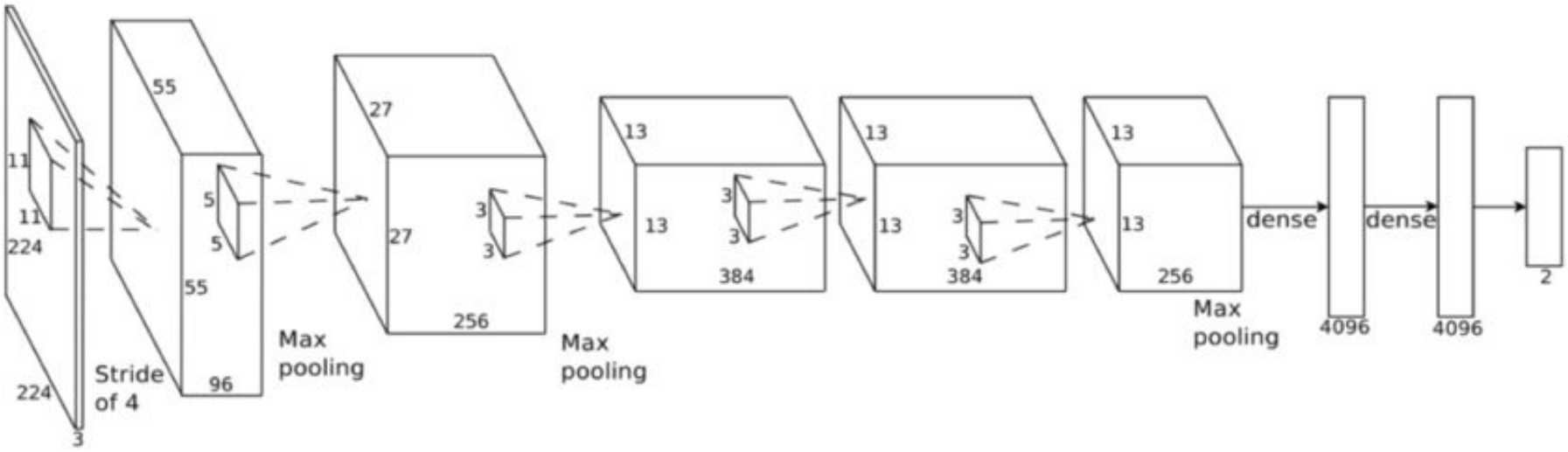

Fig. 8: Rede AlexNet genérica [4].

[4] Pedraza, A.; et al. - Glomerulus Classification with Convolutional Neural Networks

Topologia da Rede Utilizada

Camada (Layer)	Dimensão	Pooling	Tamanho do Filtro	Dropout	Não-Linearidade
Entrada	3x75x75				
Convolucional	64x73x73		3x3		Retificadora linear
Convolucional	64x71x71		3x3		Retificadora linear
Max-pooling	64x35x35	2x2			
Dropout	64x35x35			0.5	
Convolucional	128x33x33		3x3		Retificadora linear
Convolucional	128x31x31		3x3		Retificadora linear
Max-pooling	128x15x15	2x2			
Dropout	128x15x15			0.5	
Convolucional	192x13x13		3x3		Retificadora linear
Convolucional	192x11x11		3x3		Retificadora linear
Max-pooling	192x5x5	2x2			
Dropout	192x5x5			0.5	
Densa	512				Retificadora linear
Dropout	512			0.5	
Densa	256				Retificadora linear
Dropout	256			0.5	
Densa	170				Retificadora linear
Dropout	170			0.5	
Densa	1				Sigmoide

Fig. 9: Configurações das camadas de rede.

Resultados Obtidos - Experimento 1

A seguir, encontram-se os resultados obtidos com o treinamento da rede SEM *Batch Normalization* e SEM *Data Augmentation.*

Hiperparâmetros da rede:

Validation Split: 0.25
Learning Rate: 0.01
Momentum: 0.9

Batch Size: 16
Number of Epochs: 200
Patience: 20

Performance da rede:

Trei. Acu.	Trei. Loss	Val. Acu.	Val. Loss	Test. Loss
0.876	0.412	0.861	0.487	0.534

Tempo de execução: *67,4 minutos*
Tempo de inferência: *77,36 segundos*
Tempo total de treinamento: *64,92 minutos*

Resultados Obtidos - Experimento 2

A seguir, encontram-se os resultados obtidos com o treinamento da rede COM *Batch Normalization* e SEM *Data Augmentation.*

Hiperparâmetros da rede:

Validation Split: 0.25
Learning Rate: 0.01
Momentum: 0.9

Batch Size: 16
Number of Epochs: 200
Patience: 20

Performance da rede:

Trei. Acu.	Trei. Loss	Val. Acu.	Val. Loss	Test. Loss
0.903	0.233	0.879	0.280	0.287

Tempo de execução: *79,87 minutos*
Tempo de inferência: *106,71 segundos*
Tempo total de treinamento: *76,05 minutos*

Resultados Obtidos - Experimento 3

A seguir, encontram-se os resultados obtidos com o treinamento da rede COM *Batch Normalization* e COM *Data Augmentation.*

Hiperparâmetros da rede:

Validation Split: 0.25
Learning Rate: 0.01
Momentum: 0.9

Batch Size: 16
Number of Epochs: 200
Patience: 20

Performance da rede:	**Trei. Acu.**	**Trei. Loss**	**Val. Acu.**	**Val. Loss**	**Test. Loss**
	0.437	0.701	0.478	0.694	0.702

Tempo de execução: *59,37 minutos*
Tempo de inferência: *99,11 segundos*
Tempo total de treinamento: *56,34 minutos*

Resultados Obtidos - Comparações

Podemos visualizar abaixo a tabela comparativa dos resultados obtidos, assim como a pontuação obtida pelos 1º e 2º lugares da competição.

Performance da rede:

Test Loss	**Exp. 1**	**Exp. 2**	**Exp. 3**	**1º Lugar**	**2º Lugar**
	0.534	0.287	0.702	0.08	0.09
Val. Loss	**Exp. 1**	**Exp. 2**	**Exp. 3**	**1º Lugar**	**2º Lugar**
	0.861	0.879	0.478	?	?
Tempo de treinam. [m]	**Exp. 1**	**Exp. 2**	**Exp. 3**	**1º Lugar**	**2º Lugar**
	64,92	79,87	59,37	?	?

Quanto menores são as Perdas do Teste (*Test Loss*), melhor a rede pode ser considerada.

Conclusões

- *Batch Normalization* resultou em melhorias no modelo;
- *Data Augmentation* exige uma escolha cuidadosa dos *hiperparâmetros* - a inclusão desse atributo forneceu os piores resultados*;*
- Baixa qualidade e pouca quantidade das imagens do *Dataset* exigiram pré-processamento e ajuste fino de parâmetros para alcançarmos resultados satisfatórios;

Conclusões

- O melhor resultado obtido, até o momento, foi de *Test Loss = 0.28* no conjunto de teste; o resultado do 1º lugar da competição foi de *Test Loss = 0.08.*

- Apesar de mais baixo do que o do 1ºlugar, os resultados obtidos foram satisfatórios, dado o tempo disponível para a implementação da rede.

- Obtivemos 86% de eficiência no conjunto de validação.

Referências

LECUN, Yann et al. Convolutional networks and applications in vision. In: **ISCAS**. 2010. p. 253-256.

KRIZHEVSKY, Alex; SUTSKEVER, Ilya; HINTON, Geoffrey E. Imagenet classification with deep convolutional neural networks. In: **Advances in neural information processing systems**. 2012. p. 1097-1105.

LOFFE, S., & SZEGEDY, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*.

SRIVASTAVA, Nitish et al. Dropout: a simple way to prevent neural networks from overfitting. **The Journal of Machine Learning Research**, v. 15, n. 1, p. 1929-1958, 2014.

CHOLLET, François et al. Keras. 2015.

HEATON, Jeff. **Artificial intelligence for humans, volume 1: Fundamental algorithms**. CreateSpace Independent Publishing, North Charleston, 2013.

OBRIGADO!

Escola Politécnica da Universidade
de São Paulo

Dúvidas?
Críticas?
Sugestões?

Alfredo Ferraz
André Lopes
Michel André Lima Vinagreiro
William Contesini