

# Estimação Inteligente de Idade de Telespectadores para Aplicações de Sugestão de Conteúdo em Smart TVs

Nicoli P. Araújo, Elloá B. Guedes

<sup>1</sup> Escola Superior de Tecnologia  
Universidade do Estado do Amazonas  
Av. Darcy Vargas, 1200 – Manaus – Amazonas

{npda.eng, ebgcosta}@uea.edu.br

**Abstract.** This work presents a proposal for estimating the age of viewers for content suggestion applications on Smart TVs using machine learning techniques. Such a tool can be used in a variety of ways, including to facilitate the collection of information that contributes to a better content delivery experience, to the creation and control of custom settings, and to the implementation of more efficient parental control strategies.

**Resumo.** Este trabalho apresenta uma proposta para estimação de idade de telespectadores para aplicações de sugestão de conteúdo em Smart TVs utilizando técnicas de machine learning. Tal ferramenta pode ser utilizada de diversas maneiras, incluindo para facilitar a coleta de informações que contribuem para melhor experiência de provimento de conteúdo, criação e controle de configurações personalizadas e para a implementação de estratégias de controle parental mais eficientes.

## 1. Introdução

As *Smart TVs*, dispositivos resultantes evolução tecnológica dos aparelhos de televisão domésticos, destacam-se por sua capacidade de conexão à internet e de transmissão de conteúdos advindos de *smartphones* e outros dispositivos (NEWSROOM, 2011; PERAKAKIS; GHINEA, 2015). Segundo a Pesquisa Nacional por Amostra de Domicílios (PNAD) realizada pelo IBGE em 2015, existem 16 milhões de *Smart TVs* em residências e pontos comerciais no Brasil, cujos 94% foram adquiridos entre 2014 e 2015. Estes aparelhos foram responsáveis por 68,2% do total de televisores vendidos no primeiro semestre de 2017 (IBGE, 2015).

Igual à seção  
smart tvs.  
Mudar?

Este aumento de vendas tem várias causas, das quais destacam-se os muitos benefícios resultantes do uso de *Smart TVs* quando comparadas aos aparelhos convencionais (SHIN; HWANG; CHOO, 2013; BETWEEN, 2017). Em especial, cita-se o aumento da qualidade na transmissão, utilização de aplicativos diversos, a capacidade de conexão com dispositivos a exemplo de *smartphones* e *players* de mídia digital, a possibilidade de acesso à conteúdo *online* e *on demand*, gratuitos ou mediante assinaturas. Além destes benefícios, cuja maioria é resultante da conectividade com a internet, outros fatores têm justificado o aumento das vendas e do interesse do público consumidor pelas *Smart TVs*, tais como o encerramento da transmissão de sinal analógico da televisão aberta, a Copa do Mundo 2018 e a tecnologia 4K (GUIMARÃES, 2017; BRAZILIENSE, 2018; CAPELAS, 2017).

Considerando a grande difusão das *Smart TVs* nos lares brasileiros, é essencial que estes aparelhos sejam capazes de capturar o perfil e o interesse dos seus telespectadores a fim de oferecer uma experiência mais rica. A recomendação de conteúdo, por exemplo,

pode levar em conta características individuais, tais como idade e gênero. Porém, se fornecidos de maneira habitual, via preenchimento de formulários, além de ser uma tarefa massante, pode não refletir de maneira realística o perfil individual dos vários usuários que podem estar à frente de uma *Smart TV* em um determinado momento.

Apesar das dificuldades práticas mencionadas, é interessante notar que muitas *Smart TVs* possuem dispositivos para captura de imagens, como câmeras, pois também costumam dispor de aplicações para troca de mensagens de vídeo (SCHOFIELD, 2017). Respeitadas as preferências de privacidade de cada usuário, se estas câmeras forem habilidades para aquisição de imagens daqueles que estão à frente do televisor, então é possível usá-las como entrada para sistemas inteligentes de identificação de características, cujas previsões podem ser usadas, por exemplo, para recomendação de conteúdo. No caso da idade, em particular, é possível usar estas informações para realizar um controle parental mais eficiente, protegendo crianças e adolescentes de conteúdos inadequados à sua faixa etária.

Diante do que foi exposto, esta proposta de trabalho de conclusão de curso considera o desenvolvimento de estratégias inteligentes, baseadas na utilização de técnicas de *Deep Learning*, para estimação da idade de telespectadores a partir de fotografias faciais. Embora a estimativa de outras características também pudesse ser realizada mediante a análise de fotografias faciais, desde gênero até a presença de doenças, optou-se pela idade por ser um atributo comum a todos os telespectadores, pelo potencial de aplicações, pela existência de bases de dados adequadamente rotuladas com este atributo e pelo menor potencial de infringência das searas privadas dos usuários.

## 1.1. Objetivos

O objetivo geral deste trabalho consiste em elaborar estratégias inteligentes para estimativa de idade de telespectadores de *Smart TVs* a partir de suas respectivas fotografias faciais. Para alcançar esta meta, alguns objetivos específicos precisam ser contemplados, a citar:

1. Formular um referencial teórico sobre redes neurais convolucionais, contemplando seu arcabouço matemático, suas características, principais arquiteturas, métodos de treinamento e teste;
2. Consolidar uma base de dados com exemplos realísticos para treinamento dos modelos, tendo em vista a captura de padrões representativos ao domínio do problema;
3. Identificar tecnologias adequadas para implementação dos estimadores;
4. Propor, treinar e testar diferentes estimadores de idade baseados em redes neurais convolucionais para a tarefa em questão;
5. Avaliar comparativamente os estimadores propostos.

## 1.2. Justificativa

A realização de um trabalho de conclusão de curso desta natureza é justificada por várias razões. No contexto da interação entre telespectador e *Smart TV*, um estimador de idade pode ser utilizado para facilitar a coleta de informações que contribuam para melhor experiência de provimento de conteúdo e de configurações personalizadas. Em particular, a estimativa de idade dos telespectadores pode ser especialmente empregada na implementação de um controle parental mais eficiente, protegendo crianças e adolescentes de conteúdos inadequados à sua faixa etária.

Um outro aspecto que ressalta a importância da realização de um trabalho desta natureza é a prática e a proposição de soluções envolvendo *Machine Learning*. Esta é uma área de vanguarda na Computação e seu potencial para resolução de problemas práticos está em franco desenvolvimento. Ao considerar a elaboração do estimador proposto, será necessário dominar conhecimentos de ferramental tecnológico atual, o que pode colaborar na minimização da distância entre o profissional em formação e os anseios do mercado de trabalho da área.

Por fim, há que se mencionar a relação entre a área de pesquisa considerada neste trabalho de conclusão de curso e o Laboratório de Sistemas Inteligentes (LSI). Este trabalho alinha-se com os objetivos desta iniciativa do Núcleo de Computação (NUCOMP), motivando o desenvolvimento de uma solução inovadora que utiliza técnicas da Inteligência Artificial.

### 1.3. Metodologia

A metodologia para o desenvolvimento deste trabalho consiste na realização da *fundamentação teórica sobre Machine Learning*, em especial contemplando os conceitos relativos às redes neurais convolucionais. Para tanto, considerar-se-á a literatura desta área para que haja o entendimento das bases matemáticas deste modelo computacional, como funcionam, quais as características e as arquiteturas mais importantes. Neste estudo, além dos aspectos teóricos, serão considerados os ambientes de desenvolvimento, bibliotecas e outras tecnologias para implementação dos conceitos contemplados.

Os demais passos que compõem a metodologia deste trabalho baseiam-se no *fluxo de atividades de machine learning* (MARSLAND, 2015). Inicialmente, haverá a aquisição e o pré-processamento de imagens para *consolidar uma base de dados* para esta tarefa de aprendizado. Nesta etapa, será considerada a literatura e, se possível, bases de dados já disponíveis e apropriadamente anotadas, com licença livre de utilização.

A seguir, há a *proposição de diferentes modelos de redes neurais convolucionais* para a tarefa de aprendizado considerada. Nesta etapa, serão elencados diferentes parâmetros e hiperparâmetros de configuração, bem como arquiteturas. Estes procedimentos visam consolidar um espaço de busca de modelos que possam endereçar a tarefa de maneira mais eficiente.

O próximo estágio consiste no *treinamento das redes neurais convolucionais* para o problema em questão. Durante este processo, uma parte da base de dados será apresentada aos modelos para que haja o ajuste de pesos, compreendendo o aprendizado das características relevantes. O treinamento das redes ocorrerá utilizando computação em nuvem, tendo em vista a infra-estrutura de hardware necessária para realizar este procedimento.

Segue-se então o *teste das redes*, respeitando uma abordagem de validação cruzada e utilizando métricas de desempenho apropriadas. O objetivo desta fase consiste em aferir os modelos propostos e treinados quanto à sua capacidade de generalização.

Por fim, para identificação de um modelo mais adequado à esta tarefa, as *métricas de desempenho serão comparadas* e os melhores modelos elencados a partir destes valores, apontando assim um estimador apropriado para o problema inicialmente considerado.

Alem destas atividades, há que se considerar a escrita da proposta e do projeto final do trabalho de conclusão de curso, bem como as defesas parcial e final.

Tabela 1: Cronograma de atividades levando em consideração os dez meses (de 02/2018 a 12/2018) para a realização do TCC.

	2018										
	02	03	04	05	06	07	08	09	10	11	12
<b>Escrita da Proposta</b>	X	X	X	X	X						
<b>Fundamentação Teórica sobre Machine Learning</b>	X	X	X	X							
<b>Consolidação da Base de Dados</b>			X	X							
<b>Proposição de Modelos de Redes Neurais Convolucionais</b>				X	X	X	X	X			
<b>Defesa da Proposta</b>					X						
<b>Escrita do Trabalho Final</b>						X	X	X	X	X	X
<b>Treinamento das Redes Neurais Convolucionais</b>					X	X	X	X	X	X	
<b>Teste das Redes Neurais Convolucionais</b>				X	X	X	X	X	X	X	
<b>Comparação de Metricas de Desempenho</b>					X	X	X	X	X	X	
<b>Defesa do Trabalho Final</b>											X

#### 1.4. Cronograma

O cronograma de realização das atividades pode ser visto na Tabela 1. As atividades listadas possuem relação com a metodologia detalhada na seção anterior, compreendendo os requisitos elementares para a realização deste trabalho.

#### 1.5. Organização do Documento

Para a apresentação desta proposta de trabalho de conclusão de curso, o presente documento está organizado como segue. Inicialmente, uma fundamentação teórica pode ser vista na Seção 2. Uma análise dos trabalhos relacionados encontra-se na Seção 3. Na Seção 4 detalha-se uma solução proposta para a tarefa endereçada. Na seção 5 estão os resultados obtidos. Finalmente, as considerações finais podem ser encontradas na Seção 6.

coloco que  
são resultados  
parciais?

## 2. Fundamentação Teórica

A fundamentação teórica para a realização deste trabalho compreende conceitos ligados às *Smart TVs* e ao *Machine Learning*. Quanto ao primeiro tópico, uma caracterização das *Smart TVs* é apresentada na Subseção 2.1, e uma visão geral dos conceitos ligados à classificação indicativa é apresentada na Subseção 2.2. Quanto ao segundo tópico, a Subseção 2.3 comprehende os conceitos essenciais de *Machine Learning*, em que as redes neurais são particularmente detalhadas na Subseção 2.4. Os conceitos mais emergentes desta área, envolvendo *Deep Learning*, são descritos na Subseção 2.5.

### 2.1. Smart TVs

As *Smart TVs* são o resultado da evolução tecnológica junto aos aparelhos de televisão domésticos. Possuem capacidades interativas ligadas à internet, acesso a conteúdo online, *e-commerce* de conteúdo televisivo, navegação web e acesso a redes sociais. Estes

aparelhos podem ser equipados com câmeras e microfones e são capazes de transmitir conteúdo 2D e 3D (NEWSROOM, 2011; PERAKAKIS; GHINEA, 2015).

O principal diferencial no tocante ao hardware entre *Smart TVs* e as antigas tecnologias LED e LCD TV reside na conexão com a internet, que pode ser realizada via módulo Wi-Fi ou Ethernet (BETWEEN, 2017; QUAIN, 2018). Para promover esta conexão e posterior interação com o usuário, estas televisões utilizam os mesmos sistemas operacionais e conjuntos de aplicativos que computadores ou *smartphones* convencionais, em especial mencionam-se navegador web e diversos aplicativos.

É possível também que *Smart TVs* exibam conteúdo de mídia transmitido a partir de *smartphones*, *players* de mídia ou computadores conectados na mesma rede Wi-Fi, conforme o padrão de compartilhamento de mídia DLNA (*Digital Living Network Alliance*) (MICHELE; KARPOW, 2014; SHIN; HWANG; CHOO, 2013; PERAKAKIS; GHINEA, 2015; KOVACH, 2010). Muitos modelos destes televisores também possuem ferramentas para o reconhecimento de comandos de voz, possibilitando funcionalidades como troca e busca de canais, controle de volume, etc. Este controle de voz costuma também estar integrado com funções das casas inteligentes, tendência da Internet das Coisas (QUAIN, 2018).

A Figura 1 exibe um diagrama representativo dos elementos que compõem uma *Smart TV*. As legendas para os números apresentados na imagem estão na Tabela 2. Dentro os diversos fabricantes destes dispositivos, em nível mundial destacam-se as marcas Hisence, LG, Panasonic, Phillips, Samsung, Sharp, Sony, TCL, Toshiba e Vizio (QUAIN, 2018).



Figura 1: Diagrama representativo de uma *Smart TV* e seus componentes (NEWSROOM, 2011). Ver legenda dos componentes na Tabela 2.

As aplicações disponíveis para *Smart TVs* são diversas, permitindo, por exemplo, o acesso a conteúdo de programas e também a informações esportivas, como é comum no

Tabela 2: Legenda dos componentes citados na Figura 1.

Número	Descrição	Número	Descrição
1	Moldura	13	Sintonizador, 4 portas HDMI e 3 portas USB
2	Painel de cristal negro (célula)	14	3D <i>Hyper Real Engine</i>
3	Molde da moldura do meio	15	Placa de Alimentação
4	Folha óptica	16	Sensor de luz ambiente
5	LGP – <i>Light Guide Plate</i>	17	Módulo <i>bluetooth</i>
6	LED	18	Módulo Wi-Fi
7	Chassi traseiro	19	Auto-falantes
8	Cobertura intermediária	20	Suporte quadrangular
9	Cobertura traseira	21	Botão <i>touch</i> operacional
10	Placa de circuito principal (Placa mãe)	22	Câmera de video de telefone
11	<i>Smart Real Engine</i>	23	Suporte de parede
12	<i>Speed Backlite Engine</i>	24	Controle remoto QWERTY
		25	Óculos 3D

caso do futebol. Um exemplo de aplicação disponível para *Smart TVs* é a disponibilizada desde 2016 pela emissora aberta SBT, vide Figura 2. Este aplicativo contém novelas, programas e outras atrações disponibilizadas pela emissora que podem ser assistidos *on demand* (SBT, 2015). Outros exemplos compreendem os aplicativos de *streaming*, tais como Netflix, Amazon Prime Video, Hulu e Pandora (CIRIACO, ).



Figura 2: Aplicativo SBT. Fonte: (SBT, 2015)

Segundo a Pesquisa Nacional por Amostra de Domicílios (PNAD) realizada pelo IBGE em 2015, foi observado um total de 103 milhões de aparelhos de televisões em residências e pontos comerciais, das quais 16 milhões são de *Smart TVs*. A pesquisa detalha que 94% destas *Smart TVs* foram adquiridas entre 2014 e 2015. Os números mostram um posterior aumento nas vendas de aparelhos televisores deste tipo, representando 68,2% do total de televisores vendidos no primeiro semestre de 2017 (IBGE, 2015).

Há muitos benefícios resultantes do uso de *Smart TVs* quando comparadas aos aparelhos convencionais. Em especial, cita-se o aumento da qualidade na transmissão,

a utilização de aplicativos diversos e a possibilidade de acesso à conteúdo *online* e *on demand*, gratuitos ou mediante assinaturas. Além destes benefícios, cuja maioria é resultante da conectividade com a internet, outros fatores têm justificado o aumento das vendas e do interesse do público consumidor pelas *Smart TVs*, tais como o encerramento da transmissão de sinal analógico da televisão aberta, a Copa do Mundo 2018 e a tecnologia 4K (GUIMARÃES, 2017; BRAZILIENSE, 2018; CAPELAS, 2017).

Apesar da grande disponibilidade de conteúdo nas *Smart TVs*, é imprescindível levar em conta as restrições e recomendações deste conteúdo para o público alvo a que se destina. Neste sentido, a próxima seção detalha as políticas vigentes de classificação indicativa de conteúdo televisivo.

## **2.2. Classificação Indicativa para Conteúdo Televisivo**

O processo de classificação indicativa integra o sistema de garantias dos direitos da criança e do adolescente quanto a promover, defender e garantir o acesso a espetáculos e diversões públicas adequados à condição de seu desenvolvimento, mas reserva-se o direito final aos pais e responsáveis quanto à escolha do conteúdo adequado a estes (DEPUTADOS, 1995).

No Brasil, a *Coordenação de Classificação Indicativa* (Cocind), vinculada ao Ministério da Justiça, é o órgão responsável pela classificação indicativa de obras destinadas à televisão e outros meios, incluindo até mesmo aplicativos. A análise da classificação indicativa realizada pelo Cocind considera o grau de incidência de conteúdos de sexo e nudez, violência e drogas nas obras a serem avaliadas, como sintetizado na Tabela 3. O processo envolve o exame do conteúdo das obras a serem classificadas, a atribuição de classificação indicativa, verificação do cumprimento das normas associadas e advertência por descumprimento destas normas (JUSTIÇA, 2014).

No mundo, conteúdos televisivos são comumente classificados quanto ao grau de incidência de assuntos como linguagem vulgar, conteúdo sexual, drogas e violências, além de temas como conteúdo perturbador e discriminação, a exemplo dos Países Baixos. É frequente a aplicação de restrições de horários para a transmissão de conteúdos restritivos. As classes podem incluir restrição de idade e/ou supervisão de responsáveis, como ocorre nos Estados Unidos, Chile, Equador, Hong Kong, entre outros. Em países como a Austrália e Nova Zelândia, há um sistema de classificação indicativa para televisão aberta e outro para fechada, e um sistema de classificação especial para programas direcionados ao público infantil, na Austrália. Na Colômbia, é proibida a transmissão aérea de pornografia, mesmo em canais adultos. O ícone da classificação indicativa frequentemente deve ser exibido antes do início do programa, antes do início de cada bloco, a exemplo do Brasil, ou durante toda a transmissão do programa, como é o caso da França. Na Alemanha, apenas o aviso “O programa a seguir não é recomendado para espectadores abaixo de 16/18 anos” é mostrado na tela caso haja conteúdo potencialmente ofensivo. Em países como Portugal, Polônia e Singapura, a implantação de sistemas de classificação indicativa é posterior ao ano de 2000 (WIKIPEDIA, 2018).

## **2.3. Machine Learning**

*Machine Learning* (ML), também chamado de Aprendizado de Máquina, é uma subárea da Inteligência Artificial que trata do estudo sistemático de algoritmos e sistemas que são capazes de melhorar seu desempenho com a experiência. Um algoritmo neste domínio é capaz de aprender a partir de dados, capturando padrões e efetuando inferências. Esses algoritmos podem ser entendidos em uma analogia com humanos e outros animais

Tabela 3: Categorias de classificação indicativa propostas pela Portaria No. 368, de 11 de Fevereiro de 2014. Fonte: (JUSTIçA, 2012)

Categoría	Símbolo	Descrição do Conteúdo
Livre		Conteúdo predominantemente positivos ou que contenham imagens de violência fantasiosa, armas sem violência, mortes sem violência, ossadas e esqueletos sem violência, nudez não erótica e consumo moderado ou inusitado de drogas lícitas.
Não recomendado para menores de dez anos		Presença de armas com violência; medo ou tensão; angústia; ossadas e esqueletos com resquícios de ato de violência; atos criminosos sme violência; linguagem depreciativa; conteúdos educativos sobre sexo; descrições verbais do consumo de drogas lícitas; discussão sobre o tráfico de drogas; e o uso medicinal de drogas ilícitas.
Não recomendado para menores de doze anos		Ato violento; lesão corporal; descrição de violência; presença de sangue; sofrimento da vítima; morte natural ou acidental com violência; ato violento contra animais; exposição ao perigo; exposição de pessoas em situações constrangedoras ou degradantes; agressão verbal; obscenidade; bullying; exposição de cadáver; assédio sexual; supervvalorização de beleza física; supervvalorização do consumo; nudez velada; insinuação sexual; carícias sexuais; masturbação não explícita; linguagem chula; linguagem de conteúdo sexual; simulações de sexo; apelo sexual; consumo de drogas lícitas; indução ao uso de drogas lícitas; consumo irregular de medicamentos; menção a drogas ilícitas.
Não recomendado para menores de catorze anos		Morte intencional; estigma ou preconceito; nudez; erotização; vulgaridade; relação sexual não explícita; prostituição; insinuação do consumo de drogas ilícitas; descrições verbais do consumo de drogas ilícitas; e discussão sobre a des-criminalização de drogas ilícitas.
Não recomendado para menores de dezesseis anos		Estupro; exploração sexual; coação sexual; tortura; mutilação; suicídio; violência gratuita ou banalização da violência; aborto, pena de morte ou eutanásia; relação sexual intensa não explícita; produção ou tráfico de qualquer droga ilícita, consumo de drogas ilícitas; indução ao consumo de drogas ilícitas.
Não recomendado para menores de dezoito anos		Violência de forte impacto; elogio; glamourização e/ou apologia à violência; crueldade; crimes de ódio; pedofilia; sexo explícito; situações sexuais complexas ou de forte impacto; apologia ao uso de drogas ilícitas.

que, ao se depararem com determinada situação, costumam procurar lembranças de situações similares, de como agiram, e se o comportamento adotado foi vantajoso, e deve ser repetido, ou prejudicial, devendo ser evitado (MARSLAND, 2015; GOODFELLOW; BENGIO; COURVILLE, 2016; FLACH, 2012).

Para consolidar o aprendizado, os algoritmos de *machine learning* precisam passar por um processo de aquisição da experiência, comumente chamado de treinamento. De acordo Mitchell (MITCHELL, 1997), um algoritmo que aprende a partir da experiência  $E$  quanto a um conjunto de tarefas  $T$  e medida de performance  $P$ , se sua performance nas tarefas em  $T$ , medida por  $P$ , melhora com a experiência  $E$ .

Ao preparar um algoritmo de *machine learning* para desenvolver determinada tarefa, busca-se um modelo, ou seja, uma função, que mapeie as instâncias do espaço de entrada para o de saída (FLACH, 2012). Estes modelos podem ser agrupados em diferen-

tes categorias ao se considerar o tipo de aprendizado e também a saída desejada para o algoritmo. A Figura 3 apresenta uma visão geral do estado da arte acerca dos modelos de ML e suas subdivisões.

Quanto ao tipo de aprendizado, as tarefas de ML podem ser agrupadas em três tipos diferentes, a depender da presença e do tipo de resposta dada ao algoritmo quanto ao desempenho de suas saídas. No *aprendizado supervisionado*, o algoritmo deve aprender a inferir valores a partir de atributos preditores e do respectivo atributo alvo fornecido como exemplo, ou seja, de cenários em que se têm seus valores de saída conhecidos. Os modelos mais comumente utilizados neste tipo de aprendizado são as máquinas de vetores de suporte, redes neurais artificiais *feed-forward*, regressão linear e logística, etc. Já no *aprendizado não-supervisionado*, o algoritmo deve inferir padrões e estruturas a partir de dados não rotulados, buscando alguma estrutura interna que caracterize os dados. Exemplos de modelos aplicáveis a este cenário são os algoritmos *k-means* e *k-medoids*. Por fim, no *aprendizado por reforço* o algoritmo não recebe dados nem tampouco rótulos, e deve aprender a partir das recompensas positivas ou negativas dadas por ações que modifiquem o ambiente de maneira satisfatória ou não (FLACH, 2012).

Quanto ao tipo de saída desejada, os problemas que podem ser endereçados segundo ML são de classificação, regressão, transcrição, tradução automática, detecção de anomalia, síntese e amostragem. No caso do aprendizado supervisionado, em particular, as principais tarefas realizadas são de classificação e regressão (FLACH, 2012).

Um algoritmo proposto a uma tarefa de classificação deve especificar cada entrada  $x$  como pertencente a uma dentre  $k$  categorias pré-determinadas, produzindo uma saída  $y = f(x)$  tal que a função  $f$  é definida como  $f : \mathbb{R}^n \rightarrow \{1, \dots, k\}$ , ou seja,  $f$  mapeia sequências de números reais  $x$  de dimensão  $n$  para um valor inteiro  $y$  dentre  $k$  possibilidades (GOODFELLOW; BENGIO; COURVILLE, 2016). Dentre as tarefas de classificação estão, por exemplo, o reconhecimento de objetos em uma imagem, determinar se um indivíduo será ou não vítima de determinada doença, se sobreviverá ou não a determinado acidente, etc.

Numa tarefa de regressão, por sua vez, objetiva-se aprender uma função de valor real a partir de uma entrada (FLACH, 2012). Assim, a saída  $y = f(x)$  é dada pela função  $f : \mathbb{R}^n \rightarrow \mathbb{R}$ , ou seja,  $f$  mapeia uma entrada multidimensional  $x$  para um valor  $y$  real (GOODFELLOW; BENGIO; COURVILLE, 2016). Algumas tarefas de regressão envolvem a previsão de preços de um mercado de ações, a determinação do risco do seguro para um carro, do volume diário de precipitação em determinada cidade, etc.

Os modelos de ML são organizados em dois grandes grupos, dos tipos paramétricos ou não paramétricos. Segundo Russel e Norvig, um modelo de aprendizado que resume dados utilizando um conjunto de parâmetros de tamanho definidos independente do número de exemplos de treinamento é chamado de *modelo paramétrico*. Dentre os modelos paramétricos está a regressão logística. Já um *modelo não-paramétrico*, por sua vez, é aquele que não pode ser caracterizado por um conjunto limitado de parâmetros. Alguns exemplos de modelos não-paramétricos são máquinas de vetores de suporte, redes neurais artificiais, *k* vizinhos mais próximos e árvores de decisão CART e C4.5 (RUSSELL; NORVIG, 2016).

Dentre os modelos não paramétricos, as redes neurais artificiais têm demonstrado resultados satisfatórios em tarefas de classificação e regressão quando aplicadas em diversas áreas. Em especial, aplicações de *Deep Learning* no reconhecimento de objetos e

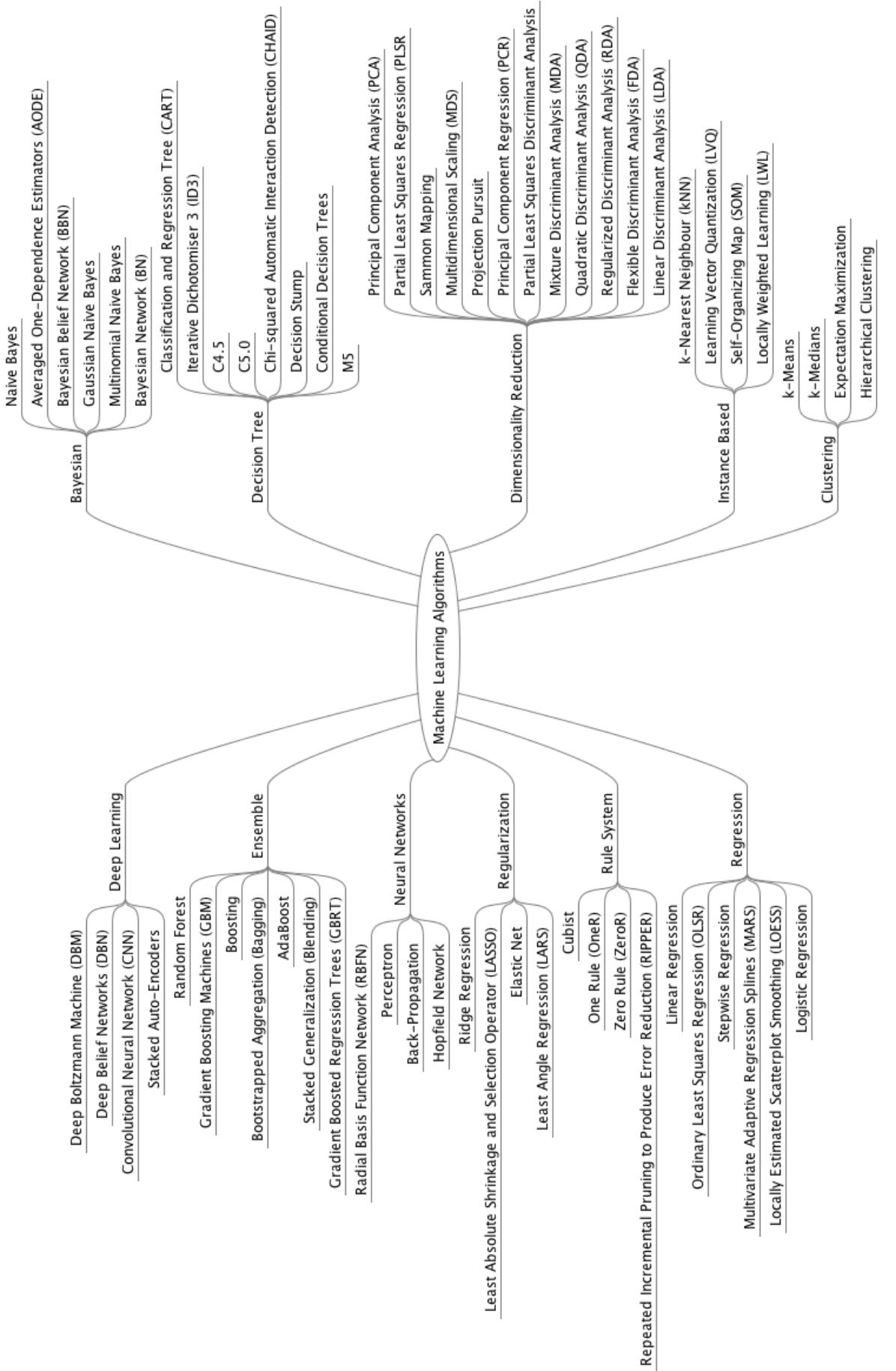
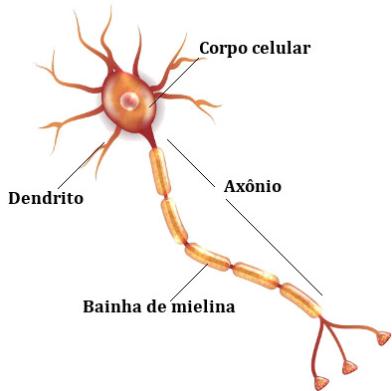


Figura 3: Mapa mental dos algoritmos de *Machine Learning* organizados por área e sub-área (BROWNLÉE, 2016).

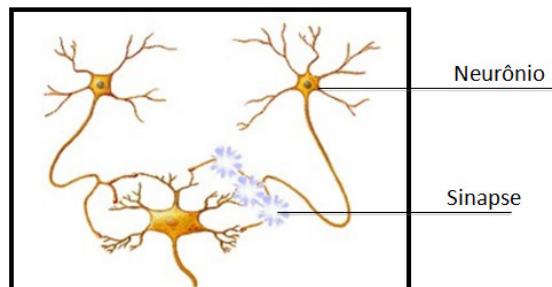
Figura 4: Redes neurais biológicas.

(a) Neurônio biológico e seus componentes.

Fonte: (SANTOS, )



(b) Sinapse entre neurônios. Fonte: (GIOVANNELLI; PASQUALIN, )



no processamento de linguagem natural, por exemplo, têm trazido ainda mais atenção a este modelo. Diante desta importância e da utilização no contexto deste trabalho, estes conceitos serão abordados com mais profundidade nas seções a seguir.

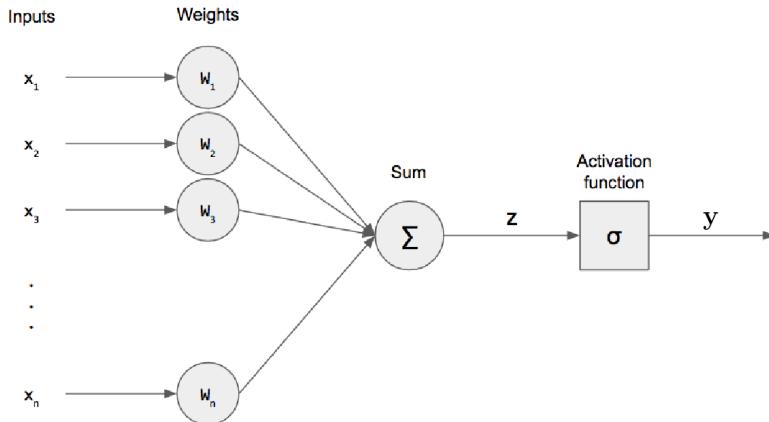
#### 2.4. Redes Neurais Artificiais

As *Redes Neurais Artificiais* (RNAs) são um modelo de computação caracterizado por sistemas que, em algum nível, lembram a estrutura do cérebro humano. São sistemas paralelos e distribuídos, compostos por unidades de processamento simples, os *neurônios artificiais*, que calculam funções matemáticas, normalmente não-lineares. Estes neurônios são dispostos em uma ou mais camadas e interligados por um grande número de conexões normalmente unidirecionais e comumente associadas a pesos, que armazenam o conhecimento representado no modelo e ponderam a entrada recebida por cada neurônio da rede. Os principais atrativos das RNAs envolvem a capacidade de capturar tendências a partir de um conjunto de exemplos e dar respostas coerentes para dados não-conhecidos, ou seja, de generalizar a informação aprendida (BRAGA; CARVALHO; LUDERMIR, 2007).

A motivação para a criação deste modelo vem do funcionamento do cérebro biológico, que é formado por neurônios interligados e que se comunicam entre si de modo contínuo e paralelo através de impulsos nervosos. Esta complexa rede neural biológica é capaz de reconhecer padrões e relacioná-los, produzir emoções, pensamentos, percepção e cognição. Cada neurônio biológico é composto de um corpo, dendritos e um axônio, como ilustrado na Figura 4a. Os dendritos são responsáveis pela recepção de impulsos nervosos vindos de outros neurônios; o corpo combina os sinais recebidos pelos dendritos e caso o resultado ultrapasse determinado limiar de excitação do neurônio, são gerados novos impulsos nervosos, que são transmitidos pelo axônio até os dendritos dos neurônios seguintes. Esta conexão unilateral entre neurônios biológicos, denominada sinapse, encontra-se ilustrada na Figura 4b.

Com base no modelo biológico, McCulloch e Pitts propuseram em (MCCULLOCH; PITTS, 1943) um neurônio artificial. Como mostrado na Figura 5, o modelo de McCulloch e Pitts de neurônio artificial contém  $n$  terminais de entrada, denotados por  $x = x_1, \dots, x_n$ , e um terminal de saída  $y$ . Esta organização faz uma alusão aos dendritos, centro e axônio de um neurônio biológico.

Figura 5: Representação de um neurônio artificial. Fonte: (DESHPANDE, 2017)



A saída  $y$  é o resultado do mapeamento da função de ativação  $\sigma$ , conforme expresso na Equação 1, aplicada à soma ponderada do vetor de entrada  $x$  pelo conjunto de pesos  $w = w_1, \dots, w_n$ . No caso de um neurônio mais simples, como o de McCulloch e Pitts, a ativação do neurônio é obtida através da aplicação de uma função degrau deslocada, a depender da comparação da entrada com um limiar de ativação  $\theta$ , conforme Equação 3 (MCCULLOCH; PITTS, 1943).

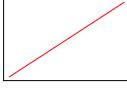
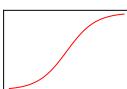
$$y = \sigma(z), \quad (1)$$

$$z = \sum_{i=1}^n x_i w_i \quad (2)$$

$$\sigma(z) = \begin{cases} 1, & \text{se } z \geq \theta \\ 0, & \text{caso contrário.} \end{cases} \quad (3)$$

Considerando desenvolvimentos posteriores, sabe-se atualmente que a escolha da função de ativação  $\sigma$  das camadas ocultas e da camada de saída deve considerar funções contínuas e deriváveis (HORNIK, 1991), em que comumente são optadas pelas funções apresentadas na Tabela 4.

Tabela 4: Exemplos de funções de ativação (GOODFELLOW; BENGIO; COURVILLE, 2016)

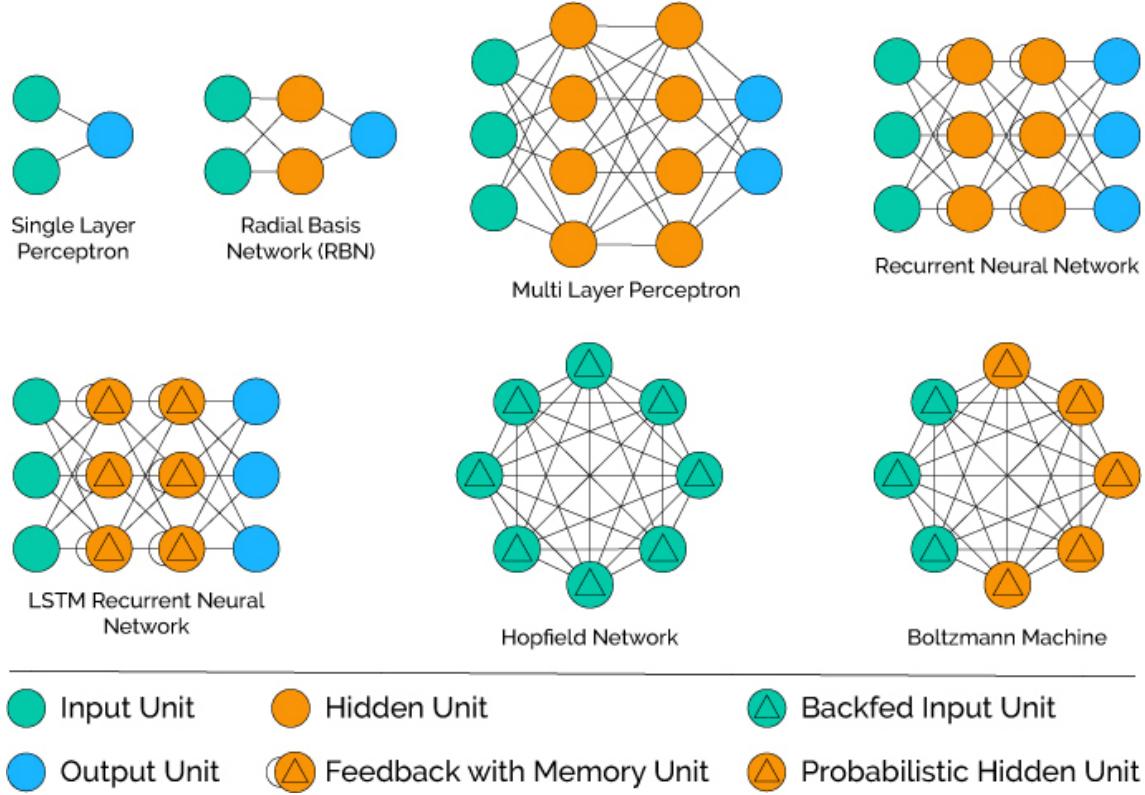
Nome	Gráfico	Equação	Intervalo
Identidade ou Linear		$\sigma(z) = z$	$(-\infty, +\infty)$
Tangente Hiperbólica		$\sigma(z) = \tanh(z) = \frac{(e^z - e^{-z})}{(e^z + e^{-z})}$	$(-1, 1)$
Sigmoide ou Logística		$\sigma(z) = \frac{1}{1 + e^{-x}}$	$(0, 1)$
Unidade Linear Retificada		$\sigma(z) = \max(0, z)$	$[0, \infty)$
Softmax		$g(z_j) = \frac{e^{z_j}}{\sum_{j=1}^K e^{z_k}} \quad j = 1, \dots, K$	$(-\infty, \infty)$

Em 1958, Frank Rosenblatt desenvolveu o neurônio *Perceptron* (ROSENBLATT, 1958), que mais tarde seria empregado como a unidade de processamento das RNA e de outros modelos de ML, a exemplo das máquinas de vetores de suporte. O Perceptron de Rosenblatt agregou ao neurônio de McCulloch e Pitts conceitos cruciais para a caracterização das RNAs como são conhecidas hoje, como a não obrigatoriedade de igualdade dos pesos e limiares de ativação, a possibilidade de os pesos serem positivos ou negativos, a diversidade de funções de ativação, entre outros. Além desta caracterização, uma contribuição relevante deste trabalho contempla a proposição de um algoritmo de aprendizado que permite a adaptação dos pesos de uma RNA através da otimização do desempenho da rede. Isto atribuiu ao modelo Perceptron a capacidade de aprender tarefas que contenham dados linearmente separáveis (BRAGA; CARVALHO; LUDERMIR, 2007).

O modelo Perceptron apresentava algumas limitações, atribuídas principalmente à sua linearidade e simplicidade, características que possibilitam resolver apenas problemas linearmente separáveis (BRAGA; CARVALHO; LUDERMIR, 2007). A disposição de neurônios em camadas e a utilização de funções de ativação nas saídas dos neurônios caracterizou as RNAs, capazes de serem aproximadas universiais de qualquer função contínua graças à otimização por minimização da dissimilaridade entre o valor previsto pela rede  $\hat{y}$  e o valor real  $y$ . Atualmente, as RNAs podem apresentar diversos tipos de arquitetura, ao variar-se parâmetros como o número de camadas, quantidade de neurônios em cada camada, os tipos de conexões entre neurônios e topologia de rede. Alguns exemplos de arquiteturas podem ser encontrados na Figura 6.

Quanto aos tipos de conexão possíveis entre os neurônios, tem-se que as RNAs podem ser do tipo *feedforward* ou recorrente. As RNAs *feedforward*, tais como a exemplificada na Figura 7a, são comumente associadas à um grafo acíclico em que as saídas de uma camada servem de entrada à camada seguinte, e assim sucessivamente, até que seja produzida uma saída. As RNAs recorrentes, tais como a rede exemplificada na Figura

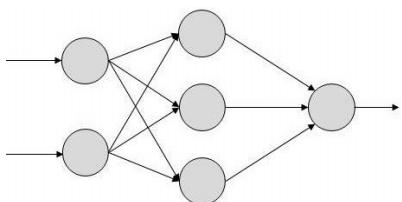
Figura 6: Arquiteturas populares de RNAs. Fonte: (GILL, 2017)



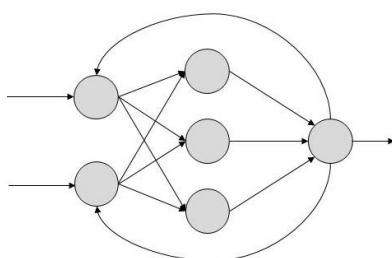
7b, contém conexões entre neurônios de modo a formar um grafo direcionado cíclico, o que permite que o modelo capture sequências de comportamentos organizados em séries temporais.

Figura 7: Exemplos de RNA com diferentes tipos de conexões entre neurônios (POINT, 2018).

(a) Exemplo de RNA *feedforward*.



(b) Exemplo de RNA recorrente.



Um dos parâmetros relacionados à arquitetura de uma RNA é a quantidade de camadas ocultas. Pode-se ter redes de camada única, compostas por um neurônio que conecta todos os parâmetros de entrada às saídas do modelo, a exemplo das redes Perceptron. Há também as redes de múltiplas camadas, que consistem de mais de um neurônio entre entrada e saída da rede, como retratado na Figura 8. Redes com múltiplas camadas, as chamadas Redes Neurais *Feedforward Multilayer Perceptron* (MLP), são capazes de aproximar diversas funções. Quando a Equação 1 é aplicada a uma camada oculta  $c$ , de modo que a entrada da função nesta camada é a saída da camada anterior  $c - 1$ , o resultado

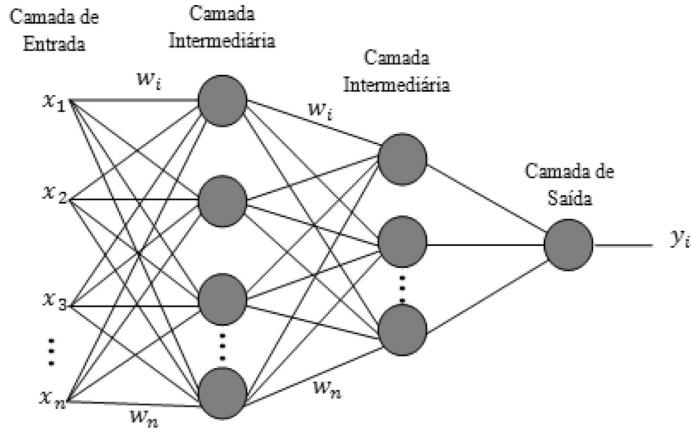
$a^c$  é construído como mostrado na Equação 4 e chamado de mapa de características ou *feature map*. Para uma RNA com profundidade  $C$ , a saída prevista  $\hat{y}$  é dada pelo mapa de características  $a^C$  (HORNIK, 1991; BRAGA; CARVALHO; LUDELMIR, 2007).

$$z^c = \sum_{i=1}^n a_i^{c-1} w_i^c + b_i^c \quad (4)$$

$$a^c = \sigma(z^c) \quad (5)$$

Segundo o Teorema da Aproximação Universal definido por Hornik em 1991, se a ativação de uma rede neural MLP for uma função limitada e não-constante, então dada uma entrada  $x$ , a rede é capaz de aproximar qualquer função contínua, provida uma quantidade adequada de camadas ocultas. Esta característica atribui às redes neurais artificiais o potencial de se tornarem máquinas de aprendizado universal (HORNIK, 1991).

Figura 8: Rede Neural MLP com duas camadas ocultas.



O objetivo das RNAs é aproximar funções que mapeiem entradas  $X$  às suas respectivas saídas  $Y$ . Para atingir este objetivo, é necessário minimizar a disparidade entre as saídas previstas  $\hat{Y}$  e as saídas desejadas  $Y$  através da atualização dos pesos  $W$  e do bias  $b$  dos neurônios das camadas. A função que calcula tal disparidade é chamada *função custo*, dada por  $J$  e tida como a soma das funções de perda,  $L(\hat{y}_i, y_i)$ , entre cada saída esperada  $y_i$  e obtida pela RNA  $\hat{y}_i$ , que vai sendo acumulada à medida que o modelo é apresentado a  $m$  exemplos do evento que se deseja aprender. A função custo está representada na Equação 6, e a perda ou erro pode ser calculada de diversas maneiras, a depender do tipo de tarefa de aprendizado, função de ativação e algoritmo de otimização escolhidos.

$$J = \frac{1}{m} \sum_{i=1}^m L(\hat{y}_i, y_i). \quad (6)$$

No contexto de maximizar as previsões corretas de uma RNA, deve-se atualizar gradualmente os pesos e bias do modelo de maneira a encontrar uma função que melhor represente o comportamento do fenômeno apresentado através dos dados, de maneira a minimizar a função custo para que haja a otimização do desempenho da RNA. Este procedimento é descrito pelo algoritmo de *back-propagation*, que consiste em duas fases que se alternam: a fase *forward* e a fase *backwards* (HAYKIN, 2009).

A fase *forward*, também chamada *forward propagation*, consiste na inferência das saídas da rede perante um conjunto de  $m$  entradas, em que uma é fornecida por vez. Neste processo, dada uma entrada  $x_i$ , a informação flui para frente pela rede, isto é, as informações iniciais que são propagadas até as camadas ocultas, e de lá até que a saída  $\hat{y}_i$  seja produzida. Ao final de uma fase *forward*, a função custo  $J$  é calculada. Modificações presentes na literatura também consideram algumas execuções da fase *forward* para obtenção do custo médio nestas iterações. A representação da sequência de passos realizados nesta fase *forward* está detalhada no Algoritmo 1 (HAYKIN, 2009; GOODFELLOW; BENGIO; COURVILLE, 2016).

---

**Algoritmo 1:** Fase *forward*


---

**Entrada:**  $i$ -ésimo par de exemplos de entrada  $x_i$  e saída  $y_i$  do conjunto de treinamento

**Saída:** Perda  $L(\hat{y}, y)$

**início**

A entrada  $x_i$  é apresentada à primeira camada da rede, como o primeiro vetor de características  $a^0$

**para** cada camada  $c=1, \dots, C$  **faça**

| Calcular a saída da camada  $a^c = \sigma^c(z^c)$ ,  $z^c = w^c \times a^{c-1}$

**fim**

Tomar como valor de saída do modelo a saída da última camada  $\hat{y} = h^C$

Calcular a perda  $L(\hat{y}, y)$ .

**fim**

---

A fase *backwards* é responsável por permitir que a informação referente à diferença entre os valores de saída obtidos  $\hat{y}$  e os esperados  $y$  calculada através da função custo na fase *forward* flua para trás. Isto ocorre por meio da atualização dos pesos dos neurônios, a começar por aqueles localizados na camada de saída, passando pelas camadas ocultas, até atingir a camada de entrada. Ao percorrer este caminho contrário, para cada camada é calculado o gradiente  $\delta$  da perda  $L$  em função dos pesos e bias, dado pela fórmula mostrada na Equação 7.

$$\delta_w = \nabla_w J = \left[ \frac{\partial J}{\partial w_1}, \frac{\partial J}{\partial w_2}, \dots, \frac{\partial J}{\partial w_n} \right]^T \quad (7)$$

$$\delta_b = \nabla_b J = \left[ \frac{\partial J}{\partial b_1}, \frac{\partial J}{\partial b_2}, \dots, \frac{\partial J}{\partial b_n} \right]^T \quad (8)$$

No contexto do cálculo, o gradiente indica o sentido e a direção para as quais se devem mover os valores dos pesos e bias das camadas de maneira a se obter o maior incremento possível de perda. Considerando que o objetivo consiste na minimização *gradual* dos parâmetros da rede na iteração  $t + 1$ , o ajuste dos pesos e bias dos neurônios na iteração  $t$  é comumente realizado por meio do método gradiente descendente indicado na Equação 9, no qual o valor do gradiente  $\delta$  é multiplicado por uma taxa de aprendizado  $\eta$  e então subtraído dos valores de  $w$  e  $b$  (HAYKIN, 2009; GOODFELLOW; BENGIO; COURVILLE, 2016).

$$w^c(t+1) = w(t) - \eta \nabla_{w^c} \quad (9)$$

$$b^c(t+1) = b(t) - \eta \nabla_{b^c} \quad (10)$$

A fase *backwards*, denotada no Algoritmo 2, lança mão de sequências de operações de regras da cadeia para calcular os gradientes. Inicialmente, o gradiente  $\delta$  da função de ativação  $\sigma$  da camada  $l$  é obtido ao realizar o produto interno do gradiente calculado sob a função de custo da RNA pela derivada da função de saída  $\sigma^c$ . Esta combinação de derivações encadeadas é altamente eficiente, o que faz com que o custo operacional da computação do gradiente seja  $O(m)$ , sendo  $m$  o número de exemplos no conjunto de treinamento. Assim, o custo computacional cresce de maneira proporcional e linear à quantidade de exemplos presente no conjunto de treinamento (HAYKIN, 2009; GOODFELLOW; BENGIO; COURVILLE, 2016).

---

**Algoritmo 2:** Fase *backwards*.

---

**Entrada:** Custo  $J$

**Saída:** Gradientes dos parâmetros da RNA MLP atualizados

**íncio**

Calcular gradiente do custo, ou seja, da perda da camada de saída

$$\delta = \nabla_y J = \nabla_y L(\hat{y}, y)$$

**para** cada camada  $c=C, \dots, 1$  **faz**

Calcular o gradiente da camada  $c$ , dado por  $\delta^c = \delta^{c-1} \cdot \sigma^c(z^c)$

Atualizar variação em  $w$ , dada por  $\nabla_{w^c} = \delta^c a^{(c-1)}$

Atualizar variação em  $b$ , dada por  $\nabla_{b^c} = \delta^c$

**fim**

**fim**

---

Além dos parâmetros  $w$  e  $b$ , as RNAs MLP têm hiperparâmetros, responsáveis por controlar as mudanças ocorridas nos parâmetros. Bengio define hiperparâmetros como variáveis cujos valores devem ser definidos antes que o algoritmo de treinamento se inicie. Alguns hiperparâmetros já discutidos neste texto são a taxa de aprendizado  $\eta$ , número de camadas ocultas, quantidade de neurônios e tipos de funções de ativação escolhidos para cada camada. O número de vezes que o conjunto de dados é apresentado ao modelo no treinamento, conhecido como número de épocas, também é um hiperparâmetro, assim como a quantidade de exemplos de treinamento apresentados à rede de uma só vez na fase *forward*, chamado de *batch size* (BENGIO, 2012).

Sumarizando os conceitos apresentados, o algoritmo para treinamento supervisionado de uma RNA MLP se dá como mostrado no Algoritmo 3.

Ao lidar com técnicas que visam acelerar ou potencializar o processo de aprendizado RNAs, o número de hiperparâmetros aumenta. Alguns exemplos destas técnicas incluem os algoritmos que realizam a otimização do gradiente descendente através da regularização dos pesos, como a Estimação Adaptativa do Momento, ou *Adam* e o algoritmo de otimização da família dos métodos quase Newtonianos *L-BFGS*. Outros hábitos comuns incluem a adoção de métodos de inicialização dos pesos que evitem o gradiente de convergir para pontos de sela, e de uma taxa de aprendizado que diminui conforme o número de épocas executadas aumenta para que o gradiente converja para um ponto mínimo mais rapidamente (GOODFELLOW; BENGIO; COURVILLE, 2016).

---

**Algoritmo 3:** Algoritmo de treinamento de uma RNA (BRAGA; CARVALHO; LUDELMIR, 2007).

---

**Entrada:** Conjuntos de exemplos e respectivos rótulos ( $X, Y$ ), rede neural a ser treinada, número de épocas  $e$ , taxa de aprendizado  $\eta$  e *batch size*  $b$ .

**Saída:** Rede neural treinada.

**início**

Inicialização aleatoriamente os vetores de pesos  $W$  e bias  $b$

**para** cada batch =  $1, \dots, b$  do conjunto de dados **faça**

Fase forward: Calcular previsões  $\hat{y}$  e custos  $J$ .

Fase backwards: Calcular gradientes dos pesos  $\nabla_{w^c}$  e bias  $\nabla_{b^c}$

Atualizar valores dos pesos e bias a partir do gradiente descendente.

**fim**

**fim**

---

As RNAs *feedforward* MLP são amplamente utilizadas em aplicações de diversos domínios. Inicialmente, destacaram-se as aplicações voltadas para o mercado financeiro visando, por exemplo, otimizar estratégias de marketing. Aplicações posteriores consideraram a alocação de assentos em aviões, aprovação de empréstimo, controle de qualidade em processos industriais, dentre outros (WIDROW; E; LEHR, 1994). O escopo de aplicações deste modelo continua a crescer nos dias atuais, especialmente diante do desenvolvimento de variantes, a exemplo das redes neurais convolucionais, com grande capacidade de detecção de padrões e pouco esforço de pré-processamento. Reconhecimento de caracteres e dígitos (LECUN et al., 1998), processamento de imagens médicas para reconhecimento de características associadas à doenças cardíacas (OKTAY et al., 2018), pulmonares (GAO et al., 2018) e mamárias (DUBROVINA et al., 2018) são alguns exemplos de aplicações de vanguarda destes modelos compreendidos dentro da sub-área de *Deep Learning*, que será caracterizada na seção a seguir.

## 2.5. Deep Learning

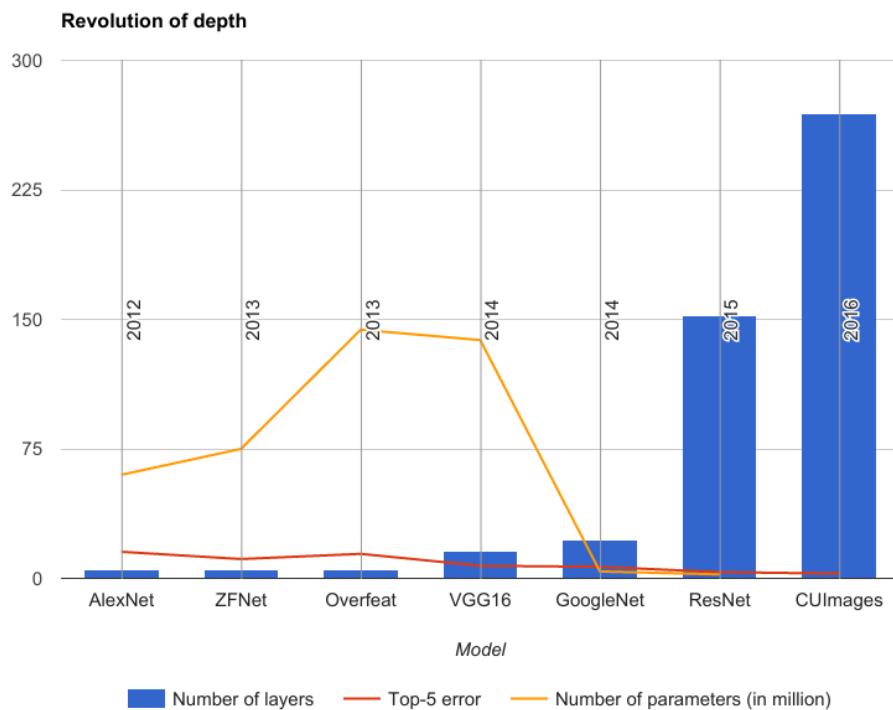
*Deep Learning* (DL), também conhecido como Aprendizado Profundo, compreende um conjunto de técnicas de ML que podem ser aplicadas em problemas de aprendizado supervisionado e não-supervisionado. A principal característica dos modelos neste domínio é a capacidade de representar e reconhecer características sucessivamente complexas, por meio da adição de níveis ou camadas de operações não-lineares em sua arquiteturas, a exemplo das redes neurais profundas, máquinas de Boltzmann profundas e fórmulas proposicionais. Modelos deste tipo ganharam popularidade ao se mostrarem capazes de resolver problemas complexos com um desempenho cada vez maior (BENGIO et al., 2009).

A melhoria do desempenho de modelos de DL é decorrente do aumento recente da quantidade de dados disponíveis sobre temas complexos, aliado ao aumento da disponibilidade de recursos computacionais para executar modelos mais robustos (GOODFELLOW; BENGIO; COURVILLE, 2016; DENG; YU et al., 2014). Alguns dados fornecidos pela IBM reforçam esta afirmação: em 2017 foram gerados 2,5 quintilhões de bytes de dados por dia, e 90% do volume total de dados gerados até 2017 no mundo foi criado somente nos últimos dois anos (IBM, 2017). Estes fatores possibilitaram a implementação de modelos que apresentaram uma melhoria significativa na eficiência de generalização

frente a modelos existentes até então, especialmente em virtude da capacidade de organizar a computação como uma composição de várias operações não-lineares (funções de ativação) e uma hierarquia de características re-utilizadas (adição de camadas) (GOODFELLOW; BENGIO; COURVILLE, 2016).

Para exemplificar o efeito da adição de camadas aos modelos de DL, a Figura 9 mostra uma visão geral do aumento da profundidade das camadas de redes neurais e o desempenho destas em problemas de detecção de objetos em imagens. Nota-se que, à medida que a profundidade aumenta, há uma diminuição no erro. Mais recentemente, isto também tem implicado na redução do número de parâmetros treináveis, por meio da implementação de técnicas de subamostragem (HAYKIN, 2009). Este panorama reforça a hipótese de que o aumento da profundidade das redes neurais impacta positivamente na captura de características e que estes avanços têm tornado as tarefas mais factíveis, com uma diminuição do esforço computacional associado, em comparação com modelos mais rasos (GOODFELLOW; BENGIO; COURVILLE, 2016).

Figura 9: Evolução de profundidade, taxa de erro e número de parâmetros das redes neurais profundas com o passar dos anos. Fonte: (ECARLAT, 2017).



### 2.5.1. Breve Histórico

O termo *Deep Learning* não é recente, foi utilizado pela primeira vez por Dechter, no contexto da descoberta de todas as configurações de conflitos mínimas a fim de resolver um problema de satisfação de restrições (DECHTER, 1986). Porém, ganhou força a partir de pesquisas sobre RNAs *feedforward* com muitas camadas ocultas, também conhecidas por redes neurais profundas (DENG; YU et al., 2014).

Considera-se que o desenvolvimento de DL pode ser dividido em três momentos. No primeiro momento, houve a proposição de modelos lineares simples, compostos ape-

nas por um neurônio, a exemplo dos neurônios de McCulloch e Pitts (MCCULLOCH; PITTS, 1943) e *Perceptron* de Rosenblatt (ROSENBLATT, 1958). No segundo instante, iniciado nos anos 1980, teve-se como eixo central a interconexão entre vários neurônios e a proposição do algoritmo *back-propagation* para ajuste de pesos no treinamento das RNAs (RUMELHART; MCCLELLAND, 1986; RUMELHART; HINTON; WILLIAMS, 1986). Com estas contribuições, houve muita aplicação das RNAs em diversos domínios. Ainda no final deste segundo momento, duas contribuições relevantes foram feitas: os modelos *Long Short-Term Memory* (LSTM) e LeNet, que utiliza o algoritmo de *back-propagation* para treinar uma rede neural convolucional profunda para reconhecer dígitos escritos à mão (LECUN et al., 1998).

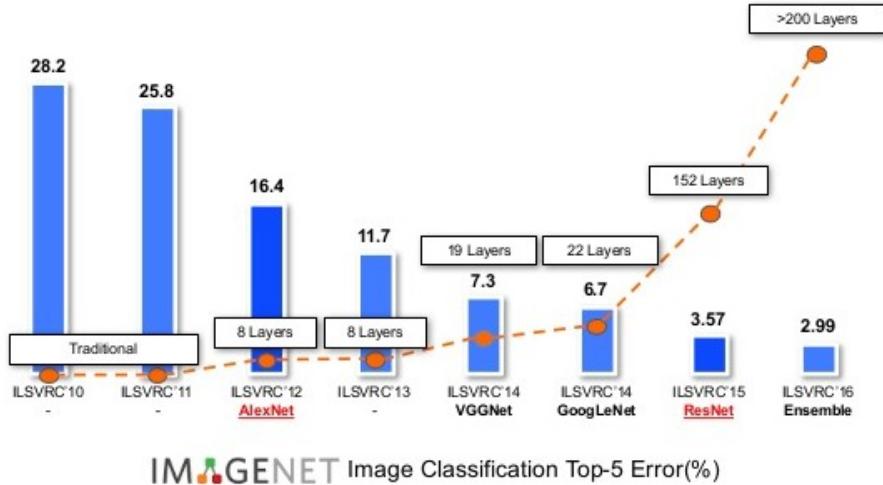
A terceira fase têm um marco inicial mais definido: compreende o ano de 2006, quando Hinton utilizou o termo *deep belief network* para designar um tipo de modelo de RNA MLP cujo treinamento adota uma estratégia gulosa e orientada a camadas. Cada camada é pré-treinada individualmente como uma máquina de Boltzmann restrita, e o modelo inteiro é então ajustado utilizando técnicas de treinamento supervisionado, incluindo o algoritmo de *backpropagation*. A partir deste marco, outros pesquisadores passaram a investigar a técnica de Hinton e, com o tempo, o termo passou a designar modelos compostos de várias camadas sucessivas de operações não lineares utilizados para o aprendizado de determinada tarefa (HINTON; OSINDERO; TEH, 2006; HINTON, 2007; GOODFELLOW; BENGIO; COURVILLE, 2016; DENG; YU et al., 2014).

Na conjectura atual, modelos de DL têm superado significativamente o estado da arte de modelos inteligentes em diversas competições em todo o mundo. A *ImageNet Large Scale Visual Recognition Challenge* (ILSVRC) (IMAGENET, 2015) é uma competição em que equipes de pesquisa avaliam seus algoritmos em um conjunto de dados fornecido, e competem para chegar à melhor acurácia em várias tarefas de reconhecimento visual automático. O conjunto de dados utilizado, denominado ImageNet (IMAGENET, 2016), consiste de um conjunto de aproximadamente 14 milhões de imagens de 21 mil itens organizados hierarquicamente, em que cada item contém algumas centenas de imagens de exemplo. A performance dos modelos submetidos para a competição é chamada taxa de erro top-5, e representa o erro computado quando a classe alvo não se encontra entre as 5 apontadas pelo modelo como as que têm maior probabilidade de estarem na imagem. Em 2011, os melhores resultados de classificação no ILSVRC tinham por volta de 25% de erro top-5 nas tarefas propostas. Em 2012, o modelo AlexNet, uma rede neural convolucional proposta segundo as ideias de DL, atingiu apenas 16,4% de erro, propondo um ganho até então nunca visto entre duas edições sucessivas da competição (IMAGENET, 2012).

O gráfico da Figura 9 sintetiza o histórico da competição ILSVRC, em que a partir do ano de 2012 houve a introdução de modelos baseados em DL. O histograma mostra a diminuição do erro na tarefa de aprendizado proposta e a linha laranja enfatiza o número de camadas ocultas utilizadas nos modelos vencedores.

Apesar do foco inicial de DL ter sido concentrado no desenvolvimento de técnicas de aprendizado não-supervisionado e na habilidade de modelos profundos de boa generalização a partir de conjuntos de dados pequenos, o cenário atual das pesquisas nesta área consideram o uso de técnicas de aprendizado supervisionado visando o endereçamento de conjuntos de dados massivos e categorizados, e também redes neurais profundas híbridas, que misturem técnicas e conceitos de diferentes origens (DENG; YU et al., 2014; GOODFELLOW; BENGIO; COURVILLE, 2016).

Figura 10: Evolução do erro dos modelos vencedores da competição ILSVRC pela profundidade das redes neurais (BORTH, 2017; IMAGENET, 2015)



Dentre estes resultados de vanguarda no ILSVRC destacam-se os modelos de DL construídos com redes neurais convolucionais, que têm impulsionado os mais recentes avanços na área de Visão Computacional desde a proposição na literatura da rede LeNet (LECUN et al., 1998). Em particular, desde a proposição e vitória do modelo AlexNet (KRIZHEVSKY; SUTSKEVER; HINTON, 2012) no ano de 2012, as redes neurais convolucionais têm dominado o ranking dos anos seguintes da competição, conforme ilustrado na Figura 10 (DENG; YU et al., 2014). Dada a importância deste tipo de rede neural, os conceitos sobre as mesmas e os modelos canônicos serão apresentados a seguir.

### 2.5.2. Redes Neurais Convolucionais

*Redes Neurais Convolucionais* (CNNs, do inglês *Convolutional Neural Networks*) são uma classe de redes neurais *feedforward* com topologia bem definida e estrutura em grade, com o uso de operações de convolução em pelo menos uma de suas camadas (GOODFELLOW; BENGIO; COURVILLE, 2016). Aplicadas em tarefas de classificação, regressão, localização, detecção e outras, este tipo de modelo se destaca no reconhecimento de padrões em dados de alta dimensionalidade, a exemplo de séries temporais, imagens e vídeos (KHAN et al., 2018).

A operação de convolução possui um papel central nas CNNs. Esta operação descreve a média ponderada de uma determinada função  $x_1(t)$  sob um intervalo fixo de uma variável, enquanto os pesos da média ponderada considerada pertencem à função  $x_2(t)$  amostrados em intervalos  $a$  (BRACEWELL, 1986). Assim, a convolução  $s(t)$  de duas funções  $x_1(t)$  e  $x_2(t)$  é uma função  $s : \mathbb{Z} \rightarrow \mathbb{R}$ , denotada  $s(t) = x_1(t) * x_2(t)$ , e definida conforme Equação 11 (LATHI, 2006):

$$s(t) = x_1(t) * x_2(t) = \int_{-\infty}^{\infty} x_1(a)x_2(t-a)da. \quad (11)$$

No contexto de ML, a função  $x_1(t)$  é chamada de *input*, a função  $x_2(t)$  é o *kernel*, e a saída  $s(t)$  consiste no *feature map*, ou mapa de características. No contexto prático, o

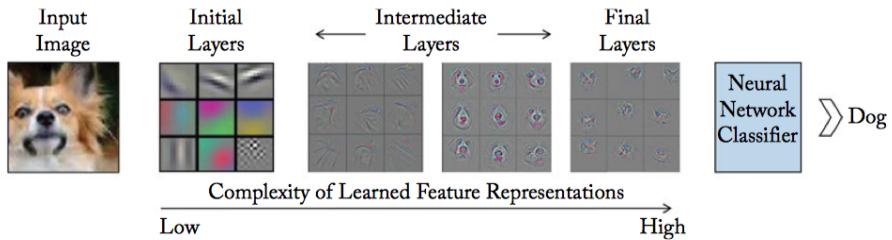
*input* normalmente é um vetor multidimensional de dados e o *kernel* é um vetor multidimensional de pesos que devem ser ajustados para aprendizado das CNN. Considerando, por exemplo, uma imagem  $I$  de dimensões  $(m, n)$  como *input* e a aplicação de um *kernel*  $K$ , a versão discreta da convolução, passível de implementação computacional e equivalente à Equação 11, é mostrada na Equação 12:

$$S(i, j) = I(i, j) * K(i, j) = \sum_m \sum_n I(m, n)K(i - m, j - n), \quad (12)$$

em que  $S$  é o *feature map* resultante e  $(i, j)$  é a posição correspondente nesse mapa. Para otimizar os aspectos de implementação, os valores resultantes da operação de convolução são armazenados apenas nas posições  $(i, j)$  explicitamente declaradas (GOODFELLOW; BENGIO; COURVILLE, 2016).

Os *feature maps*, resultantes das operações de convolução, compreendem a noção de filtros, responsáveis por capturarem características relativas à entrada, tais como contornos, linhas, texturas, etc. Quando combinados de maneira sequencial, como proposto pelas CNNs, as características capturadas pelas camadas convolucionais vão se tornando mais complexas à medida que se aumenta a profundidade da rede. Assim, um primeiro *feature map* de uma camada convolucional captura um simples contorno, enquanto um *feature map* em uma camada mais profunda da rede pode capturar uma forma, um rosto ou até um objeto inteiro (BUDUMA, 2017). Esta noção é ilustrada na Figura 11.

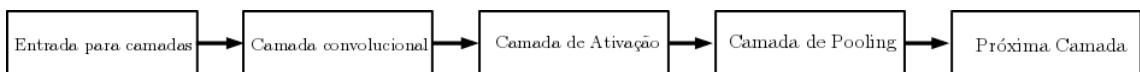
Figura 11: Papel das camadas convolucionais e *feature maps* nas CNNs. Fonte: (KHAN et al., 2018).



As camadas convolucionais, que capturam os *feature maps* e contém os pesos da rede, normalmente são seguidas por funções de ativação como as exemplificadas na Seção 2.4, mais especificamente na Tabela 4. Via de regra, a toda camada convolucional em uma CNN, segue-se uma função de ativação, finalizando em uma operação de *pooling*, como mostra a Figura 12.

Uma função de *pooling* substitui a saída da rede em determinada localização por uma síntese estatística das saídas vizinhas. Por exemplo, a operação *max pooling* retorna o valor máximo em uma área retangular, enquanto a *average pooling* retorna a média das saídas de um retângulo. O objetivo desta operação é fazer com que o *feature map* seja

Figura 12: Componentes de uma camada de uma rede neural convolucional (GOODFELLOW; BENGIO; COURVILLE, 2016).



invariante a pequenas mudanças na entrada. Esta invariância à pequenas mudanças locais é uma propriedade útil quando o mais importante for a existência da característica e não exatamente a sua posição, o que aumenta a eficiência geral da CNN ao reduzir drasticamente o número de valores a serem passados entre duas camadas quaisquer (GOODFELLOW; BENGIO; COURVILLE, 2016).

Outros parâmetros, como *padding* e *strides*, são importantes para a captura de características. O parâmetro *padding* consiste em adicionar um número de linhas e colunas em cada lado da entrada de maneira a controlar o tamanho do *feature map* resultante da operação de convolução. Já a distância entre duas janelas da convolução, ou da operação de *pooling*, é medida através do número de *strides*. Ambos parâmetros manipulam as dimensões das saídas das camadas de uma CNN (CHOLLET, 2017).

Embora se tenha uma noção clara das camadas individuais e de suas respectivas funções, a combinação das mesmas em uma rede neural convolucional não é uma tarefa trivial, podendo resultar em um número arbitrariamente grande de redes com milhares de parâmetros ajustáveis, cujo desempenho acerca de um problema ainda precisará ser aferido. Considerando os esforços computacionais para isto, a maioria das soluções atuais baseadas em DL fazem uso de CNNs canônicas já propostas na literatura, as quais são apresentadas a seguir.

### 2.5.3. Modelos Canônicos de Redes Neurais Convolucionais

Os modelos canônicos de CNNs são arquiteturas que trouxeram contribuições importantes, pioneiras na aplicação de técnicas que são comuns ainda hoje no cenário de DL, comumente utilizadas em diversas tarefas de aprendizado (DESHPANDE, 2016).

A LeNet é o primeiro modelo de rede neural a aplicar convoluções ao invés das camadas totalmente conectadas convencionais. Foi proposta por LeCun em 1998 e é composta de 7 camadas, sendo uma camada de entrada, duas camadas convolucionais, duas camadas de *pooling*, uma camada totalmente conectada e a camada de saída. A tarefa de aprendizado endereçada por esta rede na ocasião de sua proposição foi o reconhecimento de dígitos manuscritos. Para o treinamento e teste desta rede foi utilizado o conjunto de dados *Modified National Institute of Standards and Technology* (MNIST), composto de 60000 imagens de treinamento e 10000 de teste dos dígitos de 0 a 9 escritos à mão (LECUN; CORTES; BURGES, ). A LeNet foi amplamente utilizada por bancos para o reconhecimento de números escritos à mão em cheques digitalizado em imagens em escala de cinza de tamanho  $32 \times 32$  (LECUN et al., 1998). Apesar de pesquisas nesta área continuarem no decorrer dos anos, a quantidade insuficiente de bases de imagens catalogadas e o baixo poder computacional da época fizeram com que as CNNs permanecessem sem grandes destaques até o ano de 2012 (DESHPANDE, 2016).

AlexNet foi a primeira CNN ganhadora do desafio ILSVRC, em 2012, ao atingir um erro top-5 igual a 15.4%. O segundo melhor modelo daquele ano atingiu um erro de 26.2%. Esta rede, treinada para uma tarefa de classificação utilizando imagens de 1000 categorias da ImageNet, é formada por 5 camadas convolucionais com filtros de tamanho  $11 \times 11$ , intercaladas com camadas de *max-pooling* e *dropout* e 3 camadas totalmente conectadas. Utilizava a função de ativação *ReLU* ao invés da tradicional tangente hiperbólica. Na ocasião, para obter uma quantidade de exemplos razoável mediante o número de parâmetros ajustáveis, foi realizado um aumento artificial nas imagens de entrada, modificando-as segundo translações, reflexões horizontais e cortes. A AlexNet foi então

treinada utilizando duas GPU GTX 580 por 5 a 6 dias, com o algoritmo de *backpropagation* utilizando gradiente descendente estocástico para *batch* e técnicas como *momentum* e *weight decay*. Estas técnicas garantiram à rede um desempenho significativamente melhor que o dos modelos tradicionais que estavam sendo aplicados para a ILSVRC daquele ano (KRIZHEVSKY; SUTSKEVER; HINTON, 2012).

Uma CNN que com um amplo destaque pela simplicidade e profundidade é a VGG. Apesar de não ter ganhado a ILSVRC 2014, alcançou um erro top-5 de 7.3%. Foi concebida na Universidade de Oxford, a VGG-19 contendo 19 camadas que utilizam estritamente filtros de  $3 \times 3$  com *stride* e *pad* de 1, juntamente com *max-pooling* de tamanho  $2 \times 2$  e *stride* 2. O número de filtros dobra após cada camada de *max-pooling*, o que reforça a idéia de diminuir dimensões espaciais de largura e altura e aumentar a profundidade. A VGG-19 foi treinada em parte da base ImageNet utilizando 4 GPUs Nvidia Titan Black por duas a três semanas (SIMONYAN; ZISSERMAN, 2014).

Uma das primeiras arquiteturas de CNNs que se desviou do caminho normal de simplesmente empilhar camadas convolucionais e de pooling em uma estrutura sequencial foi a GoogLeNet, também chamada de Inception. Dotada de 22 camadas convolucionais, a rede ganhou o ILSVRC 2014 com um erro top-5 de 6.7%. Esta performance se deve aos chamados de módulos Inception, compostos de camadas da rede que ocorrem em paralelo. Ao invés de realizar uma operação de convolução ou *max-pooling* de cada vez, várias operações diferentes são realizadas e os mapas de características obtidos são condensados e conectados ao próximo bloco Inception. A Figura 13 mostra que há 4 conjuntos de operações a serem realizadas, todas contendo convoluções com filtros  $1 \times 1$ , podendo haver também a convolução com filtros de tamanho  $3 \times 3$ ,  $5 \times 5$ , ou uma operação de *max-pooling* de  $3 \times 3$ . A convolução com filtro  $1 \times 1$  é aplicada para reduzir a dimensionalidade do problema reduzindo a profundidade do mapa de características, além de retirar da entrada informações bem detalhadas em volume. As convoluções que utilizam filtros de  $3 \times 3$  e  $5 \times 5$  dão ao modelo a capacidade de extrair informações relevantes em larga escala. A camada de *max-pooling* é aplicada a fim de reduzir a largura e altura do mapa de características e combater *overfitting*. Na GoogLeNet, as camadas totalmente conectadas são substituídas por *average pooling*, o que economiza um grande número de parâmetros. A rede, que foi treinada em algumas GPUs de alta performance por uma semana, utiliza ReLU como função de ativação e dispõe de 9 módulos Inception com mais de 100 camadas no total. Todas estas técnicas utilizadas para reduzir a dimensionalidade do problema surtiram efeito pois, como mostrado na Figura 9, nota-se que esta rede tem 12 vezes menos parâmetros que a sua predecessora AlexNet (SZEGEDY et al., 2015).

A Microsoft ResNet foi a CNN vencedora do ILSVRC 2015, com uma taxa de erro top-5 de 3.6%. Composta de um total de 152 camadas, esta rede neural deve o sucesso de sua profundidade ao bloco residual, representado na Figura 14 que, de maneira resumida, soma à saída de um certo bloco de convoluções a saída de um bloco anterior, para que ambos *feature maps* sejam alimentados à função de ativação ReLU.

Em CNNs tradicionais, a partir de uma imagem de entrada, obtém-se o *feature map* resultante de uma operação de convolução e então aplica-se o *max-pooling*. Este resultado produz uma nova representação, que possui pouca relação com a entrada original. No caso da ResNet, o bloco residual faz com que a entrada original persista em camadas mais profundas da rede, o que garante que as características determinantes para a tarefa de aprendizado sejam disseminadas para camadas mais profundas, ao mesmo tempo em que mantém a dimensionalidade reduzida. Outra razão para o bloco residual ser efetivo

Figura 13: Bloco Inception da CNN GoogLeNet. Fonte: (DESHPANDE, 2016).

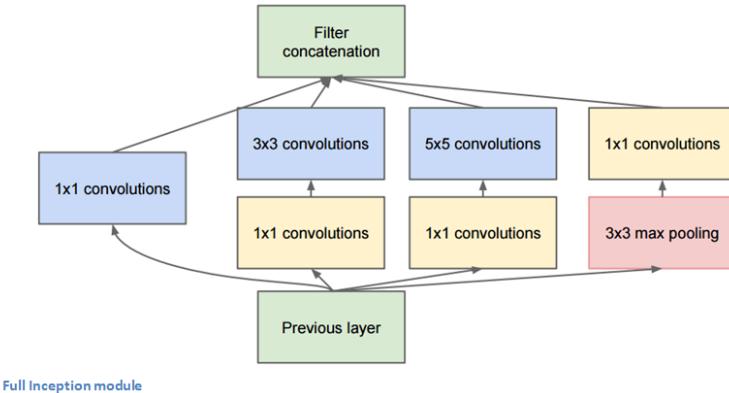
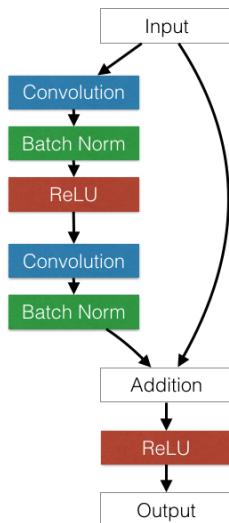


Figura 14: Bloco Residual da CNN ResNet. Fonte: (GROSS; WILBER, 2016).



é que durante a fase *backwards* o gradiente vai fluir mais facilmente pela rede, pois os blocos residuais estão diretamente conectados às camadas mais rasas na arquitetura, o que facilita a distribuição do gradiente. Na ocasião da sua proposição, a ResNet foi treinada em 8 GPUs por duas a três semanas (HE et al., 2016).

Observada a importância das CNNs apresentadas nesta seção, vale notar o esforço computacional para o treino das mesmas, que demora dias mesmo com um hardware específico para este fim. Todo o resultado do treinamento destas redes com o conjunto de dados ImageNet encontram-se disponíveis em *frameworks* para implementação de CNNs, como o *Keras* (KERAS, 2018a) e *Tensorflow* (TENSORFLOW, 2018b). Disponibilizar estas versões de tais CNNs favorece o aproveitamento das mesmas em contextos análogos, como mostrado a seguir.

#### 2.5.4. Transfer Learning

As várias redes convolucionais treinadas com milhares de exemplos da base de dados ImageNet e validadas na competição ILSVRC, a exemplo das que foram mencionadas na seção anterior, registram alto desempenho nas tarefas de aprendizado para as quais foram

originalmente projetadas. Considerando a arquitetura das mesmas, percebe-se como resultado do treinamento efetuado que uma grande quantidade de parâmetros foi ajustada mediante os milhares de exemplos apresentados, processo este que demorou vários dias mesmo mediante uso de hardware altamente especializado.

Segundo Oquab, representações de imagens aprendidas por CNNs a partir de conjuntos de dados com grande número de exemplos podem ser transferidas eficientemente para outras tarefas de reconhecimento visual que tenham uma quantidade limitada de dados de treinamento (OQUAB et al., 2014). Para tanto, faz-se uso de uma técnica denominada *transfer learning*, que consiste em transferir os conhecimentos entre domínios relacionados. No contexto das CNNs aplicadas em reconhecimento de objetos em imagens, as camadas internas podem agir como detectores de características em médio nível. Estas podem ser pré-treinados na tarefa fonte, na qual utiliza-se um conjunto de dados vasto como o ImageNet, e então re-utilizados na tarefa alvo, que pode conter um conjunto de dados mais restrito.

Após o treinamento para a tarefa original, a rede aprende a identificar características mais elementares, como linhas, contornos e objetos, que podem ser redirecionadas na tarefa alvo, que possui um objetivo mais específico. O trabalho de Zeiler e Rob, por exemplo, ilustra uma aplicação de *transfer learning* em uma tarefa de classificação, em que os autores transferem o conhecimento de uma CNN AlexNet treinada originalmente para o conjunto de dados ImageNet adaptando-a para o conjunto de dados Caltech-256, cuja base de dados possui apenas cerca de 30 mil imagens (ZEILER; FERGUS, 2014).

Na prática, o *transfer learning* é feito ao transferir os parâmetros de peso  $w$  e de bias  $b$  de um modelo já consolidado na literatura e pré-treinado com um conjunto de dados mais robusto, para um modelo similar ainda não treinado. Com o objetivo aproveitar os parâmetros pré-treinados, remove-se a última camada, ou seja, a camada de saída, e em seguida adiciona-se uma ou mais camadas novas, sem treinamento. Neste ponto, outros hiperparâmetros são considerados (número de camadas removidas, quantidade e tipo de camadas adicionadas, por exemplo) e o modelo passa por um processo de ajuste, chamado *fine tuning*, para ajustar os novos parâmetros adicionados em conformidade com os pré-existentes, permitindo a extração das características do conjunto de dados da tarefa alvo de maneira satisfatória (OQUAB et al., 2014).

## 2.6. Tecnologias Utilizadas

Para a realização deste trabalho serão utilizadas tecnologias comumente relacionadas às práticas de *Machine Learning*. A linguagem de programação adotada é o Python 3, em conjunto com bibliotecas como a *Python Data Analysis Library* (Pandas) (NUMFOCUS, 2018b) utilizada para análise e consolidação de conjuntos de dados, as bibliotecas de visualização de dados Seaborn (WASKOM, 2018) e Matplotlib (NUMFOCUS, 2018a), as bibliotecas de *Machine Learning* com suporte a *Deep Learning* Keras (KERAS, 2018b) e TensorFlow (TENSORFLOW, 2018c). A fim de executar o treinamento e teste dos modelos propostos, utilizou-se uma instância de máquina virtual disponibilizada através da *Google Compute Engine* (GCE) (GOOGLE, 2018), dotada de 4 núcleos de processamento com 4 GHz cada e 16 GB de memória RAM. A GCE é parte da *Google Cloud Platform*, uma suíte de computação em nuvem oferecida pelo *Google*.

## 3. Trabalhos Relacionados

A proposta apresentada está relacionada com inúmeros trabalhos envolvendo a aplicação de redes neurais convolucionais e outros modelos de *machine learning* para a estimativa

de idade de indivíduos.

Segundo (FU; GUO; HUANG, 2010), a idade pode ser inferida a partir de padrões distintos que emergem através da aparência da face. Técnicas comuns para a estimativa da idade envolvem a dedução de modelos matemáticos a partir do estudo do crescimento de medidas da face e do crânio (KWON; LOBO, 1999), da textura do rosto (LANITIS; TAYLOR; COOTES, 2002), da captura de tendências de envelhecimento a partir de várias imagens de indivíduos de mesma idade (FU; XU; HUANG, 2007) e a extração de características específicas relacionadas à idade (SUO et al., 2008; LOU et al., 2018). Modelos de *machine learning* também são utilizados para a tarefa, em especial as redes neurais artificiais, K-vizinhos mais próximos e máquinas de vetores de suporte.

Recentemente, a aplicação de redes neurais convolucionais em problemas de classificação e detecção de objetos em imagens têm obtido resultados significativamente positivos. Em (SIMONYAN; ZISSERMAN, 2014; HE et al., 2016; SZEGEDY et al., 2015; REDMON et al., 2016; LIU et al., 2016) dentre outros trabalhos, são descritas arquiteturas robustas capazes de detectar dezenas de objetos em várias situações. Treinadas com conjuntos de dados visuais que contam com milhares de exemplos como a ImageNet (IMAGENET, 2015), Pascal VOC (EVERINGHAM et al., 2014) e COCO (LIN et al., 2014), estas redes são conhecidas por seu bom desempenho. Algumas destas redes foram afinadas utilizando conjuntos de dados menores e especializados para a tarefa de estimativa de idade.

O trabalho de Rothe em (ROTHE; TIMOFTE; GOOL, 2015a) relata um método para estimativa de idade aparente em imagens de faces imóveis utilizando *deep learning*. O método proposto consiste de detectar uma face em uma imagem, para em seguida estimar sua idade. Para esta última tarefa, Propõe-se um conjunto de 20 redes neurais convolucionais classificadoras com arquiteturas VGG-16 pré-treinadas com a base de dados visuais ImageNet, e ajustadas utilizando imagens disponibilizadas pelos sites do IMDb, da Wikipedia, e o conjunto de dados *Looking At People*–LAP para anotação de idade aparente. Cada modelo tem como saída um número discreto entre 0 e 100, representando a idade prevista. A saída final do modelo consiste na localização do rosto e na média entre as idades previstas pelos 20 redes para o rosto detectado. A solução atingiu um MAE (*Mean Average Error*) de 3.221 na fase de testes.

Em (LIU et al., 2015) cria-se um estimador de idade composto pela fusão de um modelo regressor e outro classificador. Realiza-se um pré-processamento das imagens de entrada, que envolve a detecção das faces presentes em cada imagem, seguida pela etapa de localização de pontos de referência, como olhos, nariz e boca, e por fim há a normalização das faces. Dois métodos de normalização de face são testados, a normalização exterior e interior. Após este pré-processamento, as imagens resultantes são alimentadas a modelos de redes neurais convolucionais profundas inspiradas na *GoogLeNet* (SZE-GEDY et al., 2015). O modelo sofreu modificações em sua arquitetura, como adição de normalização do *batch*, remoção de camadas de *dropout* e perda. Foram treinados e testados diversos modelos com variações no tipo de normalização da face, tamanho do corte dos rostos, tipo de tarefa preditiva, etc. Os modelos resultantes destas variações foram unidos em um conjunto, que conseguiu prever idades com MAE de 3.3345.

Ademais, é possível encontrar resultados satisfatórios para a tarefa de aprendizado proposta utilizando modelos menos complexos. Com o objetivo de consolidar um método de classificação de idade e gênero, (LEVI; HASSNER, 2015) propõe uma rede neural

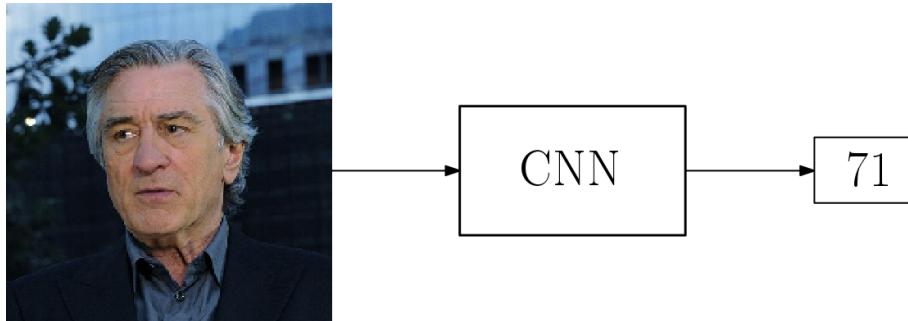
convolucionais de natureza mais simples, se comparada com as citadas acima. Sua arquitetura consiste em três camadas convolucionais com *dropout* e funções de ativação *ReLU*, seguidas por três camadas totalmente conectadas. A camada de saída tem como função de ativação a *Softmax*. A escolha por um design de rede menor é motivado pelo desejo de reduzir o risco de *overfitting* e pela natureza do problema, que contém somente 8 classes de idade. O modelo é treinado utilizando apenas o conjunto de referência *Adience*, composto por imagens não filtradas para classificação de idade e gênero. Considerando uma margem de erro de uma classe de idade vizinha, a melhor rede obteve acurácia de  $84.7\% \pm 2.2$  ao empregar a técnica de sobre-amostragem.

## 4. Solução Proposta

### 4.1. Tarefa de Aprendizado

A tarefa de aprendizado considerada para a estimativa de idade de telespectadores é a regressão. Neste contexto, uma imagem em cores RGB de dimensões  $224 \times 224$  pixels contendo uma face humana centralizada será fornecida como entrada. A saída desejada é a estimativa de idade, em anos, da pessoa correspondente, conforme exemplificado na Figura 15. Esta tarefa será abordada segundo o paradigma de aprendizado supervisionado.

Figura 15: Tarefa de aprendizado



Os dados disponíveis para este contexto serão particionados em três conjuntos disjuntos, sendo 70% reservados para o treino, 10% para validação e 20% para teste. Esta partição obedece à técnica *Holdout* de validação cruzada (BRINK; RICHARDS; FETHEROLF, 2016).

Os modelos propostos para esta tarefa terão seu desempenho aferido perante os dados do conjunto de testes de acordo com a métrica de desempenho *Root Mean Squared Error* (RMSE). Esta métrica considera a diferença entre cada um dos valores previstos  $\hat{y}$  e os reais  $y$ , e posteriormente quantifica uma média imune à variação positiva ou negativa desta diferença. A Equação 13 denota o cálculo do RMSE (BRINK; RICHARDS; FETHEROLF, 2016).

$$\text{RMSE} = \sqrt{\frac{1}{n} \sum_{i=1}^n (y_i - \hat{y})^2}. \quad (13)$$

### 4.2. Conjunto de Dados

Para a tarefa de aprendizado apresentada, dispõe-se da base de dados experimentais IMDb, composta de 452.132 exemplares contendo imagens e outras informações de 20.284 dos

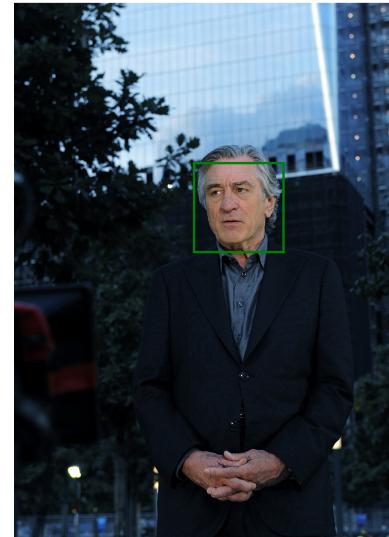
atores mais populares listados no site IMDb. O conjunto de dados foi construído utilizando técnicas de *web crawling* aplicadas aos perfis de atores do site, em que foram coletadas todas as imagens relacionadas à celebridade, além de informações como data de nascimento, nome e gênero (ROTHE; TIMOFTE; GOOL, 2015b).

A base de imagens oriunda do site IMDb foi organizada por Rothe et al. considerando uma tarefa de aprendizado análoga à deste trabalho, conforme mencionado na Seção 3 (ROTHE; TIMOFTE; GOOL, 2015a). Nesta base, há também as coordenadas da localização de um rosto detectado na imagem, além de uma pontuação atribuída ao rosto produzida pelo detector de face, quantificando o grau de certeza na detecção do rosto. Partindo da possibilidade de haver mais de um rosto por imagem, uma segunda pontuação é atribuída pelo detector, referente ao grau de certeza de que há outro rosto na mesma imagem.

Neste contexto, cada exemplo deste conjunto de dados é referente a uma imagem, cujos meta-dados estão descritos em seus atributos, que compreendem o nome, gênero, data de nascimento e um número de identificação da celebridade cujo perfil estava atrelado à imagem, o endereço da foto em disco, a suposta localização da face da celebridade, e pontuações referentes a duas possíveis faces encontradas. Assim, há exemplos de imagens em que há apenas um rosto, como mostrado na Figura 16. Já na Figura 17 está o exemplo de uma imagem onde há mais de um rosto, porém a localização do rosto está correta. Por fim, na Figura 18 há uma imagem com mais de um rosto, porém o rosto identificado neste item não é o da celebridade cujos dados estão referenciados.

Figura 16: Exemplo de imagem do conjunto de dados contendo apenas um rosto.

Meta-dado	Valor
ID Celebridade	16349
Nome	Robert De Niro
Endereço da imagem	imdb/34/nm0000134_rm334009 0368_1943-8-17_2011.jpg
Pontuação da Face	5.21396
Pontuação da Segunda Face	NaN
Localização da Face	(663.65, 992.475, 590.134, 918.959)
Data de Nascimento	1943 – 08 – 17
Ano da Foto	2011
Gênero	Masculino



A versão original das imagens do conjunto de dados IMDb ocupava 267 GB em disco. Porém, uma versão pré-processada dessas imagens está disponível, contendo as faces recortadas com 40% da largura e altura da imagem original, totalizando 7,1 GB de dados. Esta versão foi considerada neste trabalho.

#### 4.3. Limpeza e Pré-processamento dos dados

A fim de adequar melhor o conjunto de dados para os modelos de CNNs utilizados, realizou-se uma limpeza e pré-processamento dos meta-dados e das imagens da base

Figura 17: Exemplo de imagem do conjunto de dados contendo mais de um rosto com a classificação correta.

Meta-dado	Valor
ID Celebridade	16349
Nome	Robert De Niro
Endereço da imagem	imdb/34/nm0000134_rm17663 60064_1943-8-17_2010.jpg
Pontuação da Face	5.12527
Pontuação da Segunda Face	5.08887
Localização da Face	(914.886, 1426.31, 287.31, 798.734)
Data de Nascimento	1943 – 08 – 17
Ano da Foto	2010
Gênero	Masculino

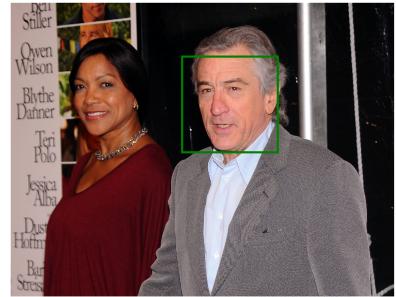


Figura 18: Exemplo de imagem do conjunto de dados contendo mais de um rosto com a classificação errônea.

Meta-dado	Valor
ID Celebridade	16349
Nome	Robert De Niro
Endereço da imagem	imdb/34/nm0000134_rm14800 44288_1943-8-17_2012.jpg
Pontuação da Face	5.51656
Pontuação da Segunda Face	4.55379
Localização da Face	(1392.72, 1614.18, 225.55, 447.003)
Data de Nascimento	1943 – 08 – 17
Ano da Foto	2012
Gênero	Masculino



IMDb, que se iniciou com o cálculo do atributo alvo, a idade, a partir dos atributos originais fornecidos. A idade foi aferida através da data de nascimento da celebridade e do ano em que a fotografia em questão foi capturada.

Uma análise do conjunto de dados revelou a presença de itens com idade e gênero apresentando valores nulos, inválidos ou negativos, que foram descartados. Observou-se também a presença de múltiplos exemplos referentes à mesma pessoa com a mesma idade. Houve a remoção de tais exemplos, a fim de evitar que a apresentação de um mesmo rosto com a mesma idade provocasse *overfitting* nos modelos. Exemplos atípicos, possivelmente resultados de rotulação incorreta, como idade maior que 100 anos ou não compatível com os dados da celebridade referida nos meta-dados também foram descartados. Os atributos de pontuação de rostos foram úteis para identificar e remover exemplos em que não havia nenhum rosto identificado, ou em que havia mais de uma face na imagem. Este descarte foi realizado com o objetivo de eliminar rotulações errôneas, como a mostrada na Tabela 18.

A última etapa consistiu na padronização das dimensões das imagens. Considere-

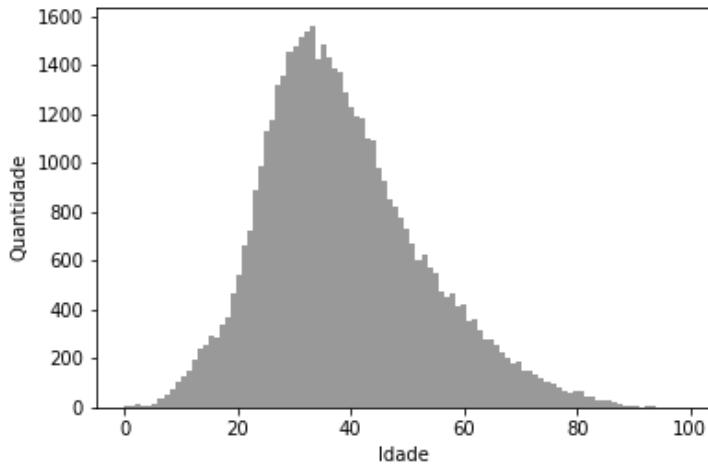


Figura 19: Histograma de frequência da idade do conjunto de dados utilizado.

rando a literatura, definiu-se o tamanho para  $224 \times 224$  pixels e o modo RGB como padrões. Por fim, após a padronização das imagens de entrada, o cálculo do atributo alvo idade, a adequação do caminho para as imagens em disco e a remoção de exemplos impróprios, seguiu-se o descarte dos outros meta-dados irrelevantes para a tarefa de estimativa de idade de um indivíduo a partir de imagem. A data em que a foto foi tirada, nome, número de identificação, gênero, data de nascimento, localização do rosto da celebridade e pontuações de rostos nas imagens foram removidos.

Por fim, o conjunto de dados consolidado consiste de 47.950 exemplos contendo imagens e idades de 14.607 celebridades distintas, ocupando 1,2GB. O histograma de frequência da distribuição de idades de 0 a 100 anos presente nos exemplos da base de dados pode ser visualizado na Figura 19. Este total foi então dividido como proposto: conjunto de treinamento, contendo 70% dos exemplos, ou seja, 33.565 amostras; conjunto de validação, referente a 10% dos dados, ou seja, 4.795 itens; e, por fim, conjunto de testes, contendo os 20% restantes, ou seja, 9.590 exemplos.

#### 4.4. Modelos de CNN Considerados

Levando em conta a adoção de CNNs como o modelo de Aprendizado de Máquina a ser usado neste trabalho, considerou-se a utilização das arquiteturas LeNet e AlexNet. A implementação da AlexNet seguiu a prática atual de utilizar apenas uma GPU em seu treinamento, então as camadas divididas no trabalho original foram unificadas (TENSORFLOW, 2018a). Todas as funções de ativação tangente hiperbólica disponíveis nas versões originais destas redes foram substituídas pela função *ReLU*, por ser mais eficiente computacionalmente, evitar que o gradiente descendente tenda a zero e por promover uma convergência mais rápida (MAAS; HANNUN; NG, 2013). Adotou-se um *batch size* igual a 64 para o treinamento, e o método de otimização do gradiente descendente foi o *Adam*. O número de épocas e a taxa de aprendizado foram obtidas de maneira experimental, observando a perda obtida ao final de cada época.

A fim de caracterizar a tarefa de regressão proposta, as camadas de saída da LeNet e AlexNet com múltiplos neurônios voltados à classificação foram substituídas por apenas um neurônio com função de ativação *ReLU*. Após análise dos resultados preliminares obtidos para estes modelos iniciais, substituiu-se a *ReLU* da camada de saída por uma de

Tabela 5: Resultados preliminares do treino e teste dos modelos propostos utilizando *ReLU* na camada de saída.

Modelo	Épocas	RMSE
LeNet	95	41.08
AlexNet	55	41.96

suas variantes, chamada *Leaky ReLU*, e expressa na Figura 20. A taxa de aprendizado inicial foi padronizada em um valor de  $10^{-3}$  com decaimento de  $10^{-10}$  para ambas as redes.

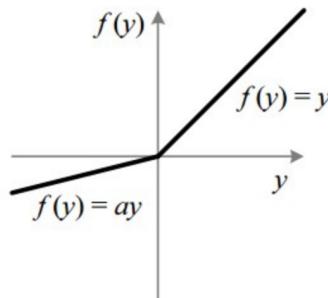


Figura 20: Função de Ativação *Leaky ReLU*

Na seção a seguir estão os resultados preliminares obtidos do treino dos modelos, hiperparâmetros e estratégias supracitados.

## 5. Resultados Parciais

Inicialmente, fez-se uso da função de ativação *ReLU* para as camadas de saída dos modelos propostos. Os resultados obtidos estão expostos na Tabela 5 a seguir:

Os RMSEs obtidos na fase de testes para as redes que utilizam *Leaky ReLU* como função de ativação na saída e taxa de aprendizado inicial de  $10^{-3}$  podem ser observados na Tabela 6.

Tabela 6: Resultados preliminares do treino e teste dos modelos propostos utilizando *Leaky ReLU* na camada de saída.

Modelo	Épocas	RMSE
LeNet	12	41.55
AlexNet	6	14.38

## 6. Considerações Parciais

O objetivo deste trabalho consiste em elaborar estratégias inteligentes para estimação de idade de telespectadores de *Smart TVs* a partir de suas respectivas fotografias faciais. Para este fim, foram propostos, treinados e testados em caráter preliminar dois modelos de CNNs já bem estabelecidos na literatura, a LeNet e AlexNet, com dois perfis de hiperparâmetros cada um.

Ao observar as previsões realizadas pelas primeiras configurações de LeNet e AlexNet propostas, notou-se que ambas exibiam saídas iguais a zero para quaisquer imagens de entrada. Concluiu-se que as redes estavam sofrendo do chamado *ReLU dying problem*, ou problema da morte da *ReLU*. Considerando o gráfico desta função de ativação exibido na Tabela 4, nota-se que a *ReLU* exibe saída 0 para entradas com valores negativos, e saída linear para entradas positivas. Apesar dos benefícios da utilização desta função de ativação, os valores de entrada negativos geram saídas e gradientes nulos, o que significa que os parâmetros correspondentes a tais entradas não serão ativados nem atualizados. Isto pode levar a valores nulos na camada de saída. As maneiras de contornar este problema incluem utilizar variantes da *ReLU* que não exibam saídas nulas, diferentes estratégias de inicialização e regularização de pesos e *batches*, entre outras. Assim, substituiu-se apenas na camada de saída a *ReLU* por uma de suas variantes, chamada *Leaky ReLU*. Esta função de ativação produz saídas negativas frente a estímulos negativos, como é possível observar na Figura 20 (CLEVERT; UNTERTHINER; HOCHREITER, 2015; PEDAMONTI, 2018).

Com isto, observa-se uma melhora significativa na performance da AlexNet, enquanto o RMSE da LeNet não sofreu grandes mudanças. Quanto às saídas das redes, a LeNet exibe valores positivos e negativos próximos de zero, e a AlexNet prevê a mesma idade, 36, 72 anos, para qualquer face apresentada. Nota-se que este valor é próximo da média de idade do conjunto de dados, de 38.63 anos.

Nos próximos meses, os esforços estarão concentrados em pesquisar e adotar estratégias que resolvam os problemas identificados, como substituir as funções de ativação das camadas ocultas por outras variantes da *ReLU*, adotar métodos específicos de inicialização de pesos, normalização de *batch*, entre outros. Planeja-se também a proposição, o treinamento e teste de outras redes inspiradas em modelos canônicos mais robustos já aplicados em tarefas de aprendizado similares, utilizando técnicas de *transfer learning*.

## Referências

- BENGIO, Y. Practical recommendations for gradient-based training of deep architectures. In: *Neural networks: Tricks of the trade*. [S.I.]: Springer, 2012. p. 437–478.
- BENGIO, Y. et al. Learning deep architectures for ai. *Foundations and trends® in Machine Learning*, Now Publishers, Inc., v. 2, n. 1, p. 1–127, 2009.
- BETWEEN, D. *Difference between Smart TV and Normal TV*. 2017. <<http://www.differencebetween.info/difference-between-smart-tv-and-normal-tv>>. Acessado em 21 de Março de 2018.
- BORTH, D. D. *Deep Learning – Future of AI*. [S.I.]: SlideShare, 2017. <<https://www.slideshare.net/GroupeT2i/deep-learning-the-future-of-ai>>. Acessado em 23 de Abril de 2018.
- BRACEWELL, R. N. *The Fourier transform and its applications*. [S.I.]: McGraw-Hill New York, 1986. v. 31999.
- BRAGA, A. de P.; CARVALHO, A. P. de Leon F. de; LUDELMIR, T. B. *Redes Neurais Artificiais: Teoria e Aplicações*. 2. ed. Rio de Janeiro: LTC, 2007.
- BRAZILIENSE, C. *Copa e novas tecnologias prometem aumentar venda de TVs no Brasil em 2018*. 2018. <<http://www.correobraziliense.com.br>>.

com.br/app/noticia/economia/2018/01/23/internas\_economia,654966/copa-e-novas-tecnologias-prometem-aumentar-venda-de-tvs-no-brasil.shtml>. Acessado em 21 de Março de 2018.

BRINK, H.; RICHARDS, J.; FETHEROLF, M. *Real-world machine learning*. [S.l.]: Manning Publications Co., 2016.

BROWNLEE, J. *Parametric and Nonparametric Machine Learning Algorithms*. [S.l.]: Machine Learning Mastery, 2016. <<https://machinelearningmastery.com/parametric-and-nonparametric-machine-learning-algorithms/>>. Acessado em 13 de Junho de 2018.

BUDUMA, N. *Fundamentals of Deep Learning*. Estados Unidos: Editora O'Reilly, 2017.

CAPELAS, B. *Explosão no consumo de vídeos online coloca em xeque o futuro da televisão*. 2017. O Estado de S. Paulo. Acessado em 20 de Março de 2018. Disponível em: <<http://link.estadao.com.br/noticias/geral/explosao-no-consumo-de-videos-online-coloca-em-xeque-o-futuro-da-television,70001695828>>.

CHOLLET, F. *Deep learning with python*. [S.l.]: Manning Publications Co., 2017.

CIRIACO, D. *Os melhores serviços de streaming de vídeo disponíveis no Brasil*. <<https://canaltech.com.br/internet/os-melhores-servicos-de-streaming-de-video-disponiveis-no-brasil/>>. Acessado em 20 de Março de 2018.

CLEVERT, D.; UNTERTHINER, T.; HOCHREITER, S. Fast and accurate deep network learning by exponential linear units (elus). *CoRR*, abs/1511.07289, 2015. Disponível em: <<http://arxiv.org/abs/1511.07289>>.

DECHTER, R. *Learning while searching in constraint-satisfaction problems*. [S.l.]: University of California, Computer Science Department, Cognitive Systems Laboratory, 1986.

DENG, L.; YU, D. et al. Deep learning: methods and applications. *Foundations and Trends® in Signal Processing*, Now Publishers, Inc., v. 7, n. 3–4, p. 197–387, 2014.

DEPUTADOS, C. dos. *Estatuto da Criança e do Adolescente*. BRASIL: [s.n.], 1995.

DESHPANDE, A. *The 9 Deep Learning Papers You Need To Know About (Understanding CNNs Part 3)*. [S.l.]: Adeshpande, 2016. <<https://adeshpande3.github.io/The-9-Deep-Learning-Papers-You-Need-To-Know-About.html>>. Acessado em 30 de Maio de 2018.

DESHPANDE, M. *Perceptrons: The First Neural Networks*. [S.l.]: Zenva Academy, 2017. <<https://pythonmachinelearning.pro/perceptrons-the-first-neural-networks/>>. Acessado em 11 de Junho de 2018.

DUBROVINA, A. et al. Computational mammography using deep neural networks. *Computer Methods in Biomechanics and Biomedical Engineering: Imaging & Visualization*, Taylor & Francis, v. 6, n. 3, p. 243–247, 2018. Disponível em: <<https://doi.org/10.1080/21681163.2015.1131197>>.

ECARLAT, P. *CNN – Do we need to go deeper?* [S.l.]: Medium, 2017. <<https://medium.com/finc-engineering/cnn-do-we-need-to-go-deeper-afe1041e263e>>. Acessado em 23 de Abril de 2018.

EVERINGHAM, M. et al. *The PASCAL Visual Object Classes Homepage*. [S.l.]: The PASCAL VOC project, 2014. <<http://host.robots.ox.ac.uk/pascal/VOC/>>. Acessado em 13 de Junho de 2018.

FLACH, P. *Machine learning: the art and science of algorithms that make sense of data*. [S.l.]: Cambridge University Press, 2012.

FU, Y.; GUO, G.; HUANG, T. S. Age synthesis and estimation via faces: A survey. *IEEE transactions on pattern analysis and machine intelligence*, IEEE, v. 32, n. 11, p. 1955–1976, 2010.

FU, Y.; XU, Y.; HUANG, T. S. Estimating human age by manifold analysis of face pictures and regression on aging features. In: *IEEE. Multimedia and Expo, 2007 IEEE International Conference on*. [S.l.], 2007. p. 1383–1386.

GAO, M. et al. Holistic classification of ct attenuation patterns for interstitial lung diseases via deep convolutional neural networks. *Computer Methods in Biomechanics and Biomedical Engineering: Imaging & Visualization*, Taylor & Francis, v. 6, n. 1, p. 1–6, 2018. Disponível em: <<https://doi.org/10.1080/21681163.2015.1124249>>.

GILL, N. S. *Artificial Neural Networks, Neural Networks Applications and Algorithms*. [S.l.]: XenonStack, 2017. <<https://www.xenonstack.com/blog/data-science/artificial-neural-networks-applications-algorithms/>>. Acessado em 11 de Junho de 2018.

GIOVANELLI, D. E. de O.; PASQUALIN, D. L. M. de A. *Por que acontecem as crises epilépticas?* [S.l.]: Neuro Vitta. <[http://neurovitta.com.br/ctd\\_noticias\\_detalhes.php?id=OQ==](http://neurovitta.com.br/ctd_noticias_detalhes.php?id=OQ==)>. Acessado em 30 de Maio de 2018.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. *Deep learning*. [S.l.]: MIT press Cambridge, 2016. v. 1.

GOOGLE. *Compute Engine*. [S.l.]: Google, 2018. <<https://cloud.google.com/compute/>>. Acessado em 14 de Junho de 2018.

GROSS, S.; WILBER, M. *Training and investigating Residual Nets*. [S.l.]: Torch, 2016. <<http://torch.ch/blog/2016/02/04/resnets.html>>. Acessado em 01 de Junho de 2018.

GUIMARÃES, N. *Com fim do sinal analógico, busca por smart TVs cresce 11%*. 2017. <<http://www.leiaja.com/tecnologia/2017/07/17/com-fim-do-sinal-analogico-busca-por-smart-tvs-cresce-11/>>. Acessado em 22 de Março de 2018.

HAYKIN, S. S. *Neural networks and learning machines*. [S.l.]: Pearson Upper Saddle River, NJ, USA:, 2009. v. 3.

HE, K. et al. Deep residual learning for image recognition. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. [S.l.: s.n.], 2016. p. 770–778.

HINTON, G. E. Learning multiple layers of representation. *Trends in cognitive sciences*, Elsevier, v. 11, n. 10, p. 428–434, 2007.

HINTON, G. E.; OSINDERO, S.; TEH, Y.-W. A fast learning algorithm for deep belief nets. *Neural computation*, MIT Press, v. 18, n. 7, p. 1527–1554, 2006.

HORNIK, K. Approximation capabilities of multilayer feedforward networks. *Neural networks*, Elsevier, v. 4, n. 2, p. 251–257, 1991.

IBGE. *Pesquisa Nacional por Amostra de Domicílios: Acesso à Internet e à Telvisão e Posse de Telefone Móvel Celular para Uso Pessoal*. 2015. <<https://biblioteca.ibge.gov.br/visualizacao/livros/liv99054.pdf>>. Acessado em 16 de Março de 2018.

IBM, M. C. *10 Key Marketing Trends for 2017 and Ideas for Exceeding Customer Expectations*. 2017. <<https://www-01.ibm.com/common/ssi/cgi-bin/ssialias?htmlfid=WRL12345USEN>>. Acessado em 23 de Março de 2018.

IMAGENET. *Large Scale Visual Recognition Challenge 2012 (ILSVRC2012)*. 2012. <<http://image-net.org/challenges/LSVRC/2012/index>>. Acessado em 31 de Maio de 2018.

IMAGENET. *Large Scale Visual Recognition Challenge (ILSVRC)*. 2015. <<http://image-net.org/challenges/LSVRC/>>. Acessado em 31 de Maio de 2018.

IMAGENET. *ImageNet*. 2016. <<http://www.image-net.org/>>. Acessado em 31 de Maio de 2018.

JUSTIÇA, M. da. *Política Pública de Classificação Indicativa*. BRASIL: [s.n.], 2014.

JUSTIÇA, S. N. de. *Classificação Indicativa Guia Pratico*. BRASIL: [s.n.], 2012.

KERAS. *Applications*. [S.l.]: Keras Documentation, 2018. <<https://keras.io/applications/>>. Acessado em 01 de Junho de 2018.

KERAS. *Keras: The Python Deep Learning library*. [S.l.]: Keras Documentation, 2018. <<https://keras.io/>>. Acessado em 14 de Junho de 2018.

KHAN, S. et al. *A Guide to Convolutional Neural Networks for Computer Vision*. Austrália: Morgan & Claypool, 2018.

KOVACH, S. *What Is A Smart TV?* 2010. <<http://www.businessinsider.com/what-is-a-smart-tv-2010-12>>. Acessado em 15 de Março de 2018.

KRIZHEVSKY, A.; SUTSKEVER, I.; HINTON, G. E. Imagenet classification with deep convolutional neural networks. In: *Advances in neural information processing systems*. [S.l.: s.n.], 2012. p. 1097–1105.

KWON, Y. H.; LOBO, N. da V. Age classification from facial images. *Computer vision and image understanding*, Elsevier, v. 74, n. 1, p. 1–21, 1999.

LANITIS, A.; TAYLOR, C. J.; COOTES, T. F. Toward automatic simulation of aging effects on face images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, IEEE, v. 24, n. 4, p. 442–455, 2002.

LATHI, B. P. *Sinais e Sistemas Lineares-2*. [S.l.]: Bookman, 2006.

LECUN, Y. et al. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, IEEE, v. 86, n. 11, p. 2278–2324, 1998.

LECUN, Y.; CORTES, C.; BURGES, C. J. *THE MNIST DATABASE of handwritten digits*. <<http://yann.lecun.com/exdb/mnist/>>. Acessado em 01 de Junho de 2018.

LEVI, G.; HASSNER, T. Age and gender classification using convolutional neural networks. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*. [S.l.: s.n.], 2015. p. 34–42.

- LIN, T. et al. Microsoft COCO: common objects in context. *CoRR*, abs/1405.0312, 2014. Disponível em: <<http://arxiv.org/abs/1405.0312>>.
- LIU, W. et al. Ssd: Single shot multibox detector. In: SPRINGER. *European conference on computer vision*. [S.l.], 2016. p. 21–37.
- LIU, X. et al. Agenet: Deeply learned regressor and classifier for robust apparent age estimation. In: *Proceedings of the IEEE International Conference on Computer Vision Workshops*. [S.l.: s.n.], 2015. p. 16–24.
- LOU, Z. et al. Expression-invariant age estimation using structured learning. *IEEE transactions on pattern analysis and machine intelligence*, IEEE, v. 40, n. 2, p. 365–375, 2018.
- MAAS, A. L.; HANNUN, A. Y.; NG, A. Y. Rectifier nonlinearities improve neural network acoustic models. In: *Proc. icml*. [S.l.: s.n.], 2013. v. 30, n. 1, p. 3.
- MARSLAND, S. *Machine learning: an algorithmic perspective*. [S.l.]: CRC press, 2015.
- MCCULLOCH, W. S.; PITTS, W. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, Springer, v. 5, n. 4, p. 115–133, 1943.
- MICHÉLE, B.; KARPOW, A. Watch and be watched: Compromising all smart tv generations. In: IEEE. *Consumer Communications and Networking Conference (CCNC), 2014 IEEE 11th*. [S.l.], 2014. p. 351–356.
- MITCHELL, T. *Machine Learning*. McGraw-Hill Education, 1997. (McGraw-Hill international editions - computer science series). ISBN 9780070428072. Disponível em: <<https://books.google.com.br/books?id=xOGAngEACAAJ>>.
- NEWSROOM, S. *Smart TV: Piece by Piece*. 2011. <<https://news.samsung.com/global/smart-tv-piece-by-piece>>. Acessado em 15 de Março de 2018.
- NUMFOCUS. *MatPlotLib*. 2018. <<https://matplotlib.org/>>. Acessado em 14 de Junho de 2018.
- NUMFOCUS. *Python Data Analysis Library*. [S.l.]: NUMFOCUS, 2018. <<https://pandas.pydata.org/>>. Acessado em 14 de Junho de 2018.
- OKTAY, O. et al. Anatomically constrained neural networks (acnns): application to cardiac image enhancement and segmentation. *IEEE transactions on medical imaging*, IEEE, v. 37, n. 2, p. 384–395, 2018.
- OQUAB, M. et al. Learning and transferring mid-level image representations using convolutional neural networks. In: IEEE. *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*. [S.l.], 2014. p. 1717–1724.
- PEDAMONTI, D. Comparison of non-linear activation functions for deep neural networks on MNIST classification task. *CoRR*, abs/1804.02763, 2018. Disponível em: <<http://arxiv.org/abs/1804.02763>>.
- PERAKAKIS, E.; GHINEA, G. A proposed model for cross-platform web 3d applications on smart tv systems. In: ACM. *Proceedings of the 20th International Conference on 3D Web Technology*. [S.l.], 2015. p. 165–166.
- POINT, T. *Artificial Intelligence - Neural Networks*. 2018. <[https://www.tutorialspoint.com/artificial\\_intelligence/artificial\\_intelligence\\_neural\\_networks.htm](https://www.tutorialspoint.com/artificial_intelligence/artificial_intelligence_neural_networks.htm)>. Acessado em 11 de Junho de 2018.

- QUAIN, J. R. *Smart TVs: Everything You Need to Know*. 2018. <<https://www.tomsguide.com/us/smart-tv-faq,review-2111.html>>. Acessado em 23 de Março de 2018.
- REDMON, J. et al. You only look once: Unified, real-time object detection. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. [S.l.: s.n.], 2016. p. 779–788.
- ROSENBLATT, F. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, American Psychological Association, v. 65, n. 6, p. 386, 1958.
- ROTHER, R.; TIMOFTE, R.; GOOL, L. V. Dex: Deep expectation of apparent age from a single image. In: *Proceedings of the IEEE International Conference on Computer Vision Workshops*. [S.l.: s.n.], 2015. p. 10–15.
- ROTHER, R.; TIMOFTE, R.; GOOL, L. V. *IMDB-WIKI – 500k+ face images with age and gender labels*. 2015. <<https://data.vision.ee.ethz.ch/cvl/rrothe/imdb-wiki/>>. Acessado em 14 de junho de 2018.
- RUMELHART, D. E.; HINTON, G. E.; WILLIAMS, R. J. Learning representations by back-propagating errors. *nature*, Nature Publishing Group, v. 323, n. 6088, p. 533, 1986.
- RUMELHART, D. E.; MCCLELLAND, J. L. Parallel distribution processing: exploration in the microstructure of cognition. MA: MIT Press, Cambridge, 1986.
- RUSSELL, S. J.; NORVIG, P. *Artificial intelligence: a modern approach*. [S.l.]: Malaysia; Pearson Education Limited,, 2016.
- SANTOS, V. S. D. *O que é neurônio?* [S.l.]: Brasil Escola. <<https://brasilescola.uol.com.br/o-que-e/biologia/o-que-e-neuronio.htm>>. Acessado em 30 de Maio de 2018.
- SBT. *Smart TV – TV Conectada*. 2015. <<http://www.sbt.com.br/tvconectada/>>. Acessado em 23 de Março de 2018.
- SCHOFIELD, J. *How can I make video calls from my TV set?* 2017. <<https://goo.gl/eCynUh>>. Acessado em 14 de junho de 2018.
- SHIN, D.-H.; HWANG, Y.; CHOO, H. Smart tv: are they really smart in interacting with people? understanding the interactivity of korean smart tv. *Behaviour & information technology*, Taylor & Francis, v. 32, n. 2, p. 156–172, 2013.
- SIMONYAN, K.; ZISSERMAN, A. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.
- SUO, J. et al. Design sparse features for age estimation using hierarchical face model. In: IEEE. *Automatic Face & Gesture Recognition, 2008. FG'08. 8th IEEE International Conference on*. [S.l.], 2008. p. 1–6.
- SZEGEDY, C. et al. Going deeper with convolutions. In: CVPR. [S.l.], 2015.
- TENSORFLOW. *AlexNet*. 2018. <<https://github.com/tensorflow/models/blob/master/research/slim/nets/alexnet.py>>. Acessado em 11 de Junho de 2018.
- TENSORFLOW. *Models and examples built with TensorFlow*. 2018. <<https://github.com/tensorflow/models>>. Acessado em 01 de Junho de 2018.
- TENSORFLOW. *TensorFlow*. 2018. <<https://www.tensorflow.org/>>. Acessado em 14 de Junho de 2018.

WASKOM, M. *seaborn: statistical data visualization*. 2018. <<https://seaborn.pydata.org/>>. Acessado em 14 de Junho de 2018.

WIDROW, B.; E, D. R.; LEHR, M. A. Neural networks: applications in industry, business and science. *Communications of the ACM*, ACM, v. 37, n. 3, p. 93–105, 1994.

WIKIPEDIA. *Television content rating system*. 2018. <[https://en.wikipedia.org/wiki/Television\\_content\\_rating\\_system#Countries\\_without\\_TV\\_rating\\_systems](https://en.wikipedia.org/wiki/Television_content_rating_system#Countries_without_TV_rating_systems)>. Acessado em 21 de Março de 2018.

ZEILER, M. D.; FERGUS, R. Visualizing and understanding convolutional networks. In: SPRINGER. *European conference on computer vision*. [S.l.], 2014. p. 818–833.