RODRIGO FUMIHIRO DE AZEVEDO KANEHISA

DETECÇÃO DE ARMA DE FOGO EM IMAGENS UTILIZANDO REDES NEURAIS CONVOLUCIONAIS

São Luís

2018

RODRIGO FUMIHIRO DE AZEVEDO KANEHISA

DETECÇÃO DE ARMA DE FOGO EM IMAGENS UTILIZANDO REDES NEURAIS CONVOLUCIONAIS

Monografia apresentada ao curso de Ciência da Computação da Universidade Federal do Maranhão, como parte dos requisitos necessários para obtenção do grau de Bacharel em Ciência da Computação.

Orientador: Prof. Dr. Areolino de Almeida Neto

São Luís

2018

Ficha gerada por meio do SIGAA/Biblioteca com dados fornecidos pelo(a) autor(a).
Núcleo Integrado de Bibliotecas/UFMA

Azevedo Kanehisa, Rodrigo Fumihiro de.
Detecção de arma de fogo em imagens utilizando redes neurais convolucionais / Rodrigo Fumihiro de Azevedo Kanehisa. – 2018.
60 f.

Orientador(a): Areolino de Almeida Neto.
Monografia (Graduação) – Curso de Ciência da Computação, Universidade Federal do Maranhão, São Luís, 2018.

1. Deep Learning. 2. Detecção de Armas de Fogo. 3. Redes Neurais Convolucionais. I. de Almeida Neto, Areolino. II. Título.

RODRIGO FUMIHIRO DE AZEVEDO KANEHISA

DETECÇÃO DE ARMA DE FOGO EM IMAGENS UTILIZANDO REDES NEURAIS CONVOLUCIONAIS

Monografia apresentada ao curso de Ciência da Computação da Universidade Federal do Maranhão, como parte dos requisitos necessários para obtenção do grau de Bacharel em Ciência da Computação.

Trabalho aprovado em: São Luís, 04 de Julho de 2018:

Prof. Dr. Areolino de Almeida Neto
(Orientador)
Universidade Federal do Maranhão

Prof. Dr. Geraldo Braz Jr.
Examinador 1

Prof. Dr. Tiago Bonini Borchartt
Examinador 2

São Luís
2018

AGRADECIMENTOS

A Deus por ter me dirigido em todos os momentos de minha vida. A minha mãe, que sempre esteve ao meu lado me apoiando e me estimulando a nunca desistir. Ao meu pai Mario Sirou Kanehisa por sempre ter despertado minha curiosidade e ter me ensinado a buscar conhecimento sempre.

A meu padrasto Marcilio de Almeida Campêlo pela orientação e incentivo diário. A minha avó, Darcy Véras de Azevedo,*in memorium*, pelo grande amor a mim dispensado. Ao meu dedicado avô, Manoel Pereira de Azevedo, sempre pronto a me ajudar. E a toda minha família por estarem comigo sempre que precisei.

Ao meu orientador Areolino de Almeida Neto, pela orientação segura durante todo o curso e principalmente por ocasião da elaboração dessa monografia.

Ao meu professor, Anselmo Cardoso de Paiva, por ter me oferecido a chance de exercitar os ensinamentos teóricos em um laboratório, permitindo-me agregar ainda mais conhecimento e crescer como profissional e pessoa.

Aos "Programadores de Elite", Ricardo Marques, Gabriel Rezende, Alexandro Saraiva, André Felipe, Marcio Franklin, Julia Manayra e aos amigos Alex_Mihawk, Muddy, MadMonkMarkov, PhanThASm, narfi_santos, Gabriel Drummond, Victor Augusto e Albérico de Castro, pela companhia nos momentos de esforço e descontração durante toda essa jornada.

A Universidade Federal do Maranhão por proporcionar ensino de qualidade. E a todos os seus profissionais que me fizeram crescer e aprender em todos esse anos.

Aos sites Stack Overflow e Wolfram Alpha que me permitiram adquirir conhecimento tão necessário em momentos de dúvida. E a todos os desenvolvedores de *software* livre que espalham a liberdade e a cooperação com seu trabalho.

"Se vi mais longe foi por estar de pé sobre ombros de gigantes."
(Isaac Newton)

RESUMO

Esta monografia realiza a detecção automática de armas de fogo em imagens com uso de *deep learning*. Indivíduos portando armas de fogo em ambientes públicos são um forte indicador de situação perigosa. Estudos mostram que a resposta rápida das autoridades é o principal fator na redução do número de vítimas. Uma das formas de garantir a ação rápida das autoridades é a detecção precoce de situações potencialmente perigosas, com ênfase no uso de câmeras de segurança, circuito fechado de televisão (CFTV) e vídeo em tempo real. Porém, o grande número de câmeras a serem observadas levam a sobrecarga dos operadores de CFTV, gerando cansaço e estresse, consequentemente, perda de eficiência na vigilância. Redes neurais do tipo *deep learning* têm mostrado-se eficientes na detecção e identificação de objetos em imagens, tendo as vezes, produzido resultados mais precisos e consistentes que candidatos humanos.

Palavras-chave: Detecção de Armas de Fogo. Deep Learning. Redes Neurais Convolucionais.

ABSTRACT

This monography performs an automatic detection of firearms in images using deep learning. Individuals carrying firearms in public conditions is a strong indicator of dangerous situation. Studies show that a rapid response from the authorities is the main factor in reducing the number of victims. One of the ways to ensure the swift action of the authorities is the early detection of potentially dangerous situations, with emphasis on the use of security cameras, closed circuit television (CCTV) and real-time video. However, the large number of cameras to be observed leads to the overload of the CCTV operators, generating fatigue and stress, consequently, loss of efficiency in surveillance. Neural networks of the deep learning type have been shown to be efficient in the detection and identification of objects in images, having sometimes produced more accurate and consistent results than human candidates.

Keywords: Firearm Detection. Deep Learning. Convolutional Neural Networks.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

ANN	*Artificial Neural Network*
AP	*Average Precision*
CFTV	Circuito Fechado de Televisão
CNN	*Convolutional Neural Network*
COV	Centros de Operação e Vigilância
CPU	*Central Processing Unit*
DNN	*Deep Neural Network*
GPU	*Graphics Processing Unit*
HOG	*Histogram of Oriented Gradients*
IMFDb	*Internet Movie Firearm Database*
IoU	*Intersection over Union*
mAP	*mean Average Precision*
MLP	*Multilayer Perceptron*
ONU	Organização das Nações Unidas
ReLu	*Rectified Linear Unit*
ROC	*Receiver operating characteristic*
ROI	*Regions of Interest*
SALW	*Small Arms and Light Weapons*
SVM	*Support Vector Machine*
YOLO	*You Only Look Once*

SUMÁRIO

1 INTRODUÇÃO

Um indivíduo portando armas em público é um forte indicador de possível situação perigosa. Recentemente, houve o aumento do número de incidentes em que indivíduos ou pequenos grupos, fazem uso de armas de fogo com o objetivo de ferir ou matar o maior número possível de pessoas. Dentre os mais notáveis destes eventos, chamados de tiroteios em massa, estão os de Columbine (EUA, 37 vítimas), o ataque à ilha Uotya (Noruega, 179 vítimas), o massacre de Realengo (Brasil, 13 vítimas), e o ataque contra o jornal Charlie Hebdo (França, 23 vítimas).

De acordo com Archive (2017), nos Estados Unidos, apenas no ano de 2017, ocorreram 347 incidentes de tiroteio em massa em ambientes públicos, levando a 437 mortos e 1803 feridos. No Brasil, entre 1980 e 2014, ocorreram por volta de 1 milhão de óbitos por uso de arma de fogo (WAISELFISZ, 2016). Apenas no ano de 2014, ocorreram por volta de 44 mil óbitos.

Sistemas de vigilância como circuito fechado de televisão (CFTV) e *drones* estão tornando-se cada vez mais comuns, estando presentes em diversos espaços públicos. Kayastha (2016) recomenda a instalação de sistemas de CFTV em ambientes públicos para combater incidentes de tiroteio em massa. A *Scotland Yard* utilizou, em 2009, imagens de câmeras de segurança como evidência em 95% dos casos de homicídio (BARRETT, 2013b). Em algumas cidades no Brasil, a instalação de câmeras de segurança levou a uma queda de 80% no índice de criminalidade (GLOBO; HOJE, 2009).

Na Inglaterra, estima-se que existam por volta de 5 milhões de câmeras de seguranças em todo o país (BARRETT, 2013a). No Brasil, em 2009, existiam por volta de 1 milhão de câmeras de segurança espalhadas pelas ruas (GLOBO; HOJE, 2009). Apesar de auxiliarem no combate a criminalidade, o grande número de câmeras leva a uma enorme sobrecarga para seus operadores.

Sistemas automáticos de vigilância começaram a surgir nos últimos anos, principalmente para o uso em sistemas inteligentes de transporte. Entre eles, estão incluídos, vigilância de tráfego (BRAMBERGER et al., 2003) e reconhecimento de veículos (BARAN; GLOWACZ; MATIOLANSKI, 2015). Tendo essas questões em mente, um sistema capaz de executar detecção automática de armas de fogo em imagens, possibilitaria resposta mais rápida e eficiente dos agentes de segurança pública. Uma das técnicas mais promissoras para a criação de sistemas automáticos de vigilância tem sido o uso de aprendizado de máquina e visão computacional.

1.1 Objetivo

O objetivo deste trabalho é implementar um método automático para a detecção de armas de fogo em imagens, aplicando redes neurais convolucionais (CNN). Este sistema é capaz de identificar a presença de armas de fogo independentemente do modelo da arma e do ambiente.

1.1.1 Objetivos específicos

- Construir uma base de dados de tamanho adequado para aplicações com uso de CNN;
- Investigar o processo de treinamento de uma rede neural para a detecção de armas de fogo;
- Estudar a viabilidade de CNNs para a criação de um detector genérico de armas de fogo;
- Implementar um sistema capaz de detectar armas de fogo em imagens.

1.2 Justificativa

A humanidade enfrenta diversos problemas e dentre esses destaca-se a violência, especialmente com uso de armas de fogo. Atualmente, a emigração de pessoas de países em conflito tem ocasionado também problemas sérios de preconceitos étnicos envolvendo os principais países do mundo, tornando-se motivo de debate até na ONU (Organização das Nações Unidas) (UNHCR, 2017). Assim, nada mais justo que se tentar mitigar essa situação de violência identificando portadores de armas de fogo com potencial de aumentar a violência.

Situações perigosas, como assaltos ou tiroteios em massa, podem muitas vezes ser evitadas através de tecnologias de monitoramento prévio, tais como CFTV, *drones* e raio-x. No entanto, existem muitas situações, como monitoramento de cidades inteiras, onde tais tecnologias têm custo alto de implementação devido a sua escala. Tentar corrigir esta situação com adição de mão de obra, além de custoso, não combate o problema advindo do erro humano, devido a cansaço, negligência, ou má interpretação dos dados. Quando uma única falha pode levar à morte, confiabilidade e consistência tornam-se fatores muito importantes.

Situações de violência, como por exemplo: tiroteios em massa ocorridos em escolas e cinemas norte americanos, em locais de reunião de grande número de pessoas como os que ocorreram em Paris, dentre outros acontecidos nas mais diversas partes do mundo, podem ser evitadas com uso de um sistema automático de detecção prévia de armas de

fogo. Vale destacar que a identificação antecipada de pessoas armadas, constitui-se no principal fator de combate à violência em ambientes de reunião da sociedade.

Estar devidamente preparado para ameaças é de grande importância para equipes de resposta à situações perigosas. Delegacias de polícia, serviços de segurança privada e os COV (Centros de Operação e Vigilância) existentes em recintos alfandegados possuem sistemas de monitoramento, que permitem identificar a presença de pessoas indevidas em certos ambientes, e equipes de pronta reposta, para fazer as abordagens nestas situações. Um sistema capaz de identificar indivíduos armados pode auxiliar as equipes de pronta resposta a estarem devidamente preparadas para responder a situação encontrada, estando cientes da ameaça a ser enfrentada, pois saberão antecipadamente se o intruso está armado ou não.

Sistemas automáticos de vigilância não são capazes de substituir os operadores humanos, entretanto, são muito úteis, especialmente em um situação onde um único operador monitora inúmeras câmeras de CFTV por muitas horas, o que naturalmente entorpece sua consciência e reduz sua capacidade de avaliar a situação.

1.3 Trabalhos relacionados

Verma e Dhillon (2017) propõem o uso de CNN para detecção de porte de armas de fogo. Com treinamento feito a partir de *transfer learning*. Fazendo testes com base de dados extraída do site IMFDb (Internet Movie Firearm Database - Banco de dados de armas em filmes na internet), detectando e classificando três tipos de armas, pistolas, revolveres e espingardas.

Em Bertozzi (2017), foi proposto um método para detecção de armas de fogo e situações potencialmente perigosas, com uso de CNN, além de propor o uso de *hardware* de desenvolvimento próprio para captação das imagens. Este trabalho teve o objetivo de detectar e reagir a presença de arma de fogo ou outros tipos de armamento quando usados de forma ameaçadora.

Já em Ardizzone et al. (2014), utilizam-se técnicas de *bottom-up saliency map* e *top-down saliency map* para criação de modelos probabilísticos para a posição de armas de fogo em imagens, baseados na posição do rosto de uma pessoa portando-a, também com o uso de base extraída do site IMFDb.

Em Grega et al. (2016), foi proposto um método para detecção automática de situações perigosas em sistemas de CFTV, por meio do uso de processamento de imagens e aprendizado de máquina. Foram utilizadas técnicas de janela deslizante, classificadores *fuzzy* e detectores *canny* para detecção de facas e armas de fogo em vídeo. Além do sistema de detecção, os autores construíram e disponibilizaram sua base de dados.

1.4 Estrutura do trabalho

Esta monografia está dividida em cinco capítulos. Além do presente capítulo, apresenta-se a fundamentação teórica necessária para o entendimento deste trabalho no capítulo a seguir. No terceiro capítulo, aborda-se a metodologia utilizada para este trabalho de conclusão de curso, além de descrever as ferramentas utilizadas. No quarto capítulo, são apresentados os resultados obtidos durante este estudo. E por fim, no quinto capítulo, são discutidas as conclusões obtidas com esta monografia.

2 FUNDAMENTAÇÃO TEÓRICA

Detecção automática de arma de fogo tem sido um problema tratado apenas recentemente, dada a sua complexidade. A escolha de CNN para classificação das imagens advém do fato de não ser necessária a extração de características para o classificador. Um classificador convencional teria muita dificuldade em encontrar características que descrevessem com precisão objetos tão variados, dada a variedade em forma e textura dos diversos tipos de armas de fogo e o objetivo de identificá-las independentemente do modelo.

2.1 Aprendizado de máquina

Aprendizado de máquina, do inglês *Machine Learning*, é um subcampo da inteligência artificial que tem como objetivo a criação de sistemas capazes de aprender de forma automática (HOSCH, 2016). Algoritmos de aprendizado de máquina são capazes de construir modelos para tomada de decisões ou para fazer previsões, baseados em informação apresentada previamente, sem que estes modelos sejam explicitamente programados (MITCHELL, 1997). Aprendizado de máquina é utilizado quando a criação de regras explicitas para resolução do problema torna-se excessivamente complicado.

O termo aprendizado de máquina foi cunhado em 1956 por Arthur L. Samuel enquanto esse trabalhava na IBM (SAMUEL, 1959). Sua ideia era mostrar a capacidade de um computador aprender a jogar damas sem ser explicitamente programado para isto. Recentemente, houve um grande crescimento no uso de aprendizado de máquina causado principalmente pelo aumento do poder computacional, com ênfase no uso de unidades de processamento gráfico (GPU) (STEINKRAUS; BUCK; SIMARD, 2005) e grandes bases de dados disponíveis na internet (SCHMIDHUBER, 2015). O processo de aprendizado pode ser supervisionado, não supervisionado ou semi supervisionado.

No processo de aprendizado supervisionado, as classes das instâncias são conhecidas previamente. Pode-se definir aprendizado supervisionado como a tarefa de encontrar uma função que dado uma entrada, esta função a mapeie corretamente para um valor de saída baseado em um exemplo anterior (RUSSELL; NORVIG, 2016). Aprendizado supervisionado é geralmente utilizado para problemas de classificação (ALPAYDIN, 2010).

No aprendizado não supervisionado, as classes das instâncias não são conhecidas previamente. Sendo assim, esta técnica tenta agrupar as instâncias em classes, baseando-se apenas em suas semelhanças. O aprendizado não supervisionado consiste em construir um modelo que agrupe pares de entrada-saída próximos, fazendo com que entradas de valores próximos produzam saídas de valores próximos (DE ALMEIDA NETO, 2003).

Para o aprendizado semi supervisionado, apenas parte do conjunto de dados de treino está classificado. Muitas vezes o custo de classificar previamente o conjunto de dados é muito alto, enquanto a aquisição de dados sem classificação é relativamente simples. Para a utilização de aprendizado semi supervisionado, deve-se assumir que instâncias próximas possuem a mesma classe ou que instâncias próximas criem sub-conjuntos e que seus membros pertençam a mesma classe (CHAPELLE; SCHÖLKOPF; ZIEN, 2006).

Pode-se aplicar aprendizado de máquina para diversas classes de problemas como regressão, classificação, agrupamento, etc. Entre suas aplicações reais então processamento de linguagem natural (COLLOBERT; WESTON, 2008), reconhecimento de fala (HINTON et al., 2012), filtragem de *spam* em correio eletrônico (GUZELLA; CAMINHAS, 2009), reconhecimento de caracteres (SEBASTIANI, 2002), segurança digital (DICKSON, 2017), diagnóstico automático de doenças (YUAN et al., 2011), indicação de vídeos em plataformas de *streaming* (BENNETT et al., 2007), entre outros.

Figura 1 – Tipos de *machine learning*.

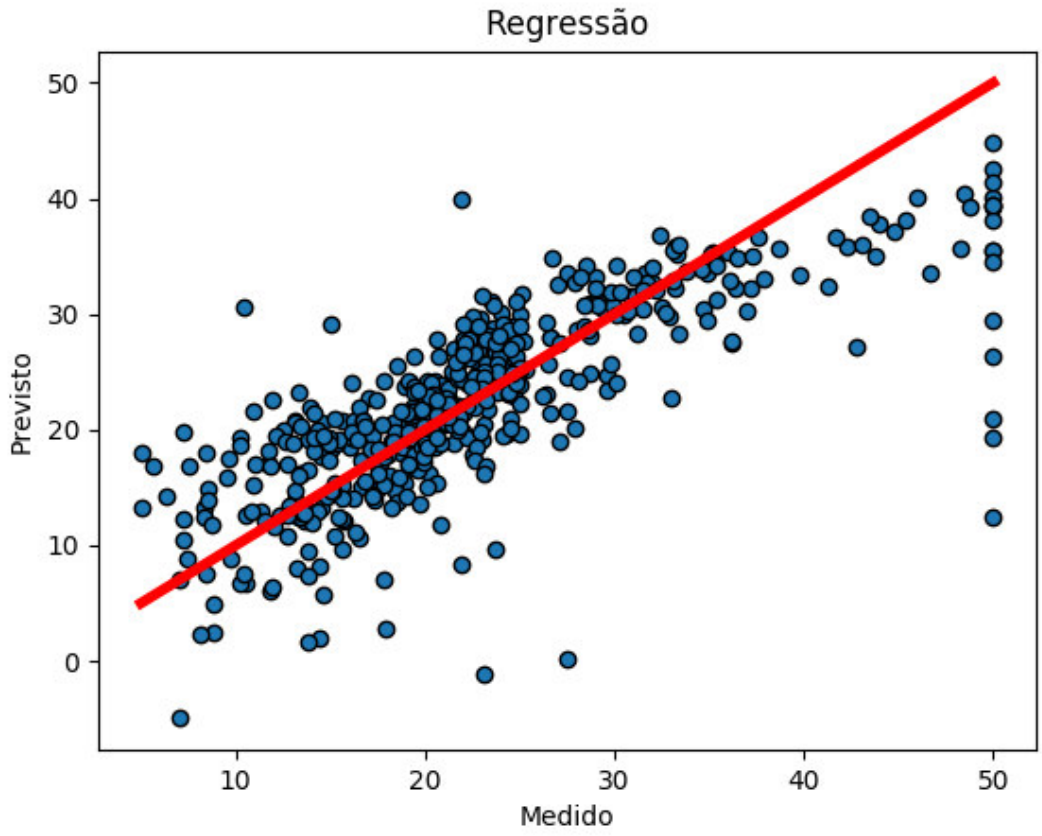
Fonte – Adaptado de (WAHID, 2017)

2.1.1 Regressão

Em estatística, problemas de regressão consistem em modelar e estimar a existência de relação entre variáveis. Regressão permite a análise do comportamento de valores para determinar se ocorre existência de uma relação entre duas variáveis e se esta relação é estatisticamente significativa.

No contexto de aprendizado de máquina, regressão permite a criação de um modelo que possa prever o comportamento de dados desconhecidos, a partir de informação aprendida previamente. Estas características podem ser quantitativas, como o comprimento de pétalas em espécies de flores, categóricas, como tipo sanguíneo (O+, A-, AB+...), ordinais como "pequeno", "médio"ou "grande", entre outros. A figura 2 demonstra uma aplicação de regressão nos preço de moradias em Boston (JR; RUBINFELD, 1978).

Figura 2 – Exemplo de regressão linear.

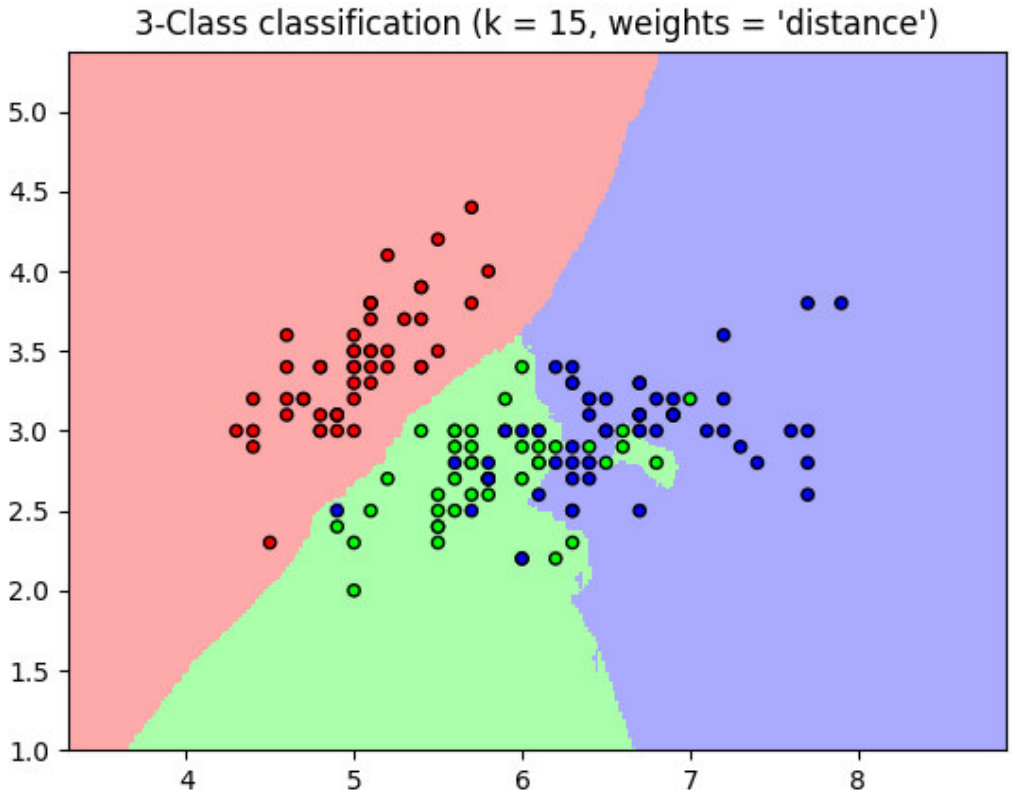


Fonte – Adaptado de (PEDREGOSA et al., 2011)

2.1.2 Classificação

Problemas de classificação são problemas onde deve-se dizer a qual classe uma instância pertence. Em aprendizado de máquina, isto é feito baseando-se em um conjunto de treino contendo instâncias cuja a categoria é conhecida anteriormente. Para o campo de aprendizado de máquina, classificação é considerado um problema de aprendizado supervisionado (ALPAYDIN, 2010). A figura 3 contém um exemplo de classificação, aplicado a base de dados Iris (FISHER, 1936).

Figura 3 – Exemplo de classificação.



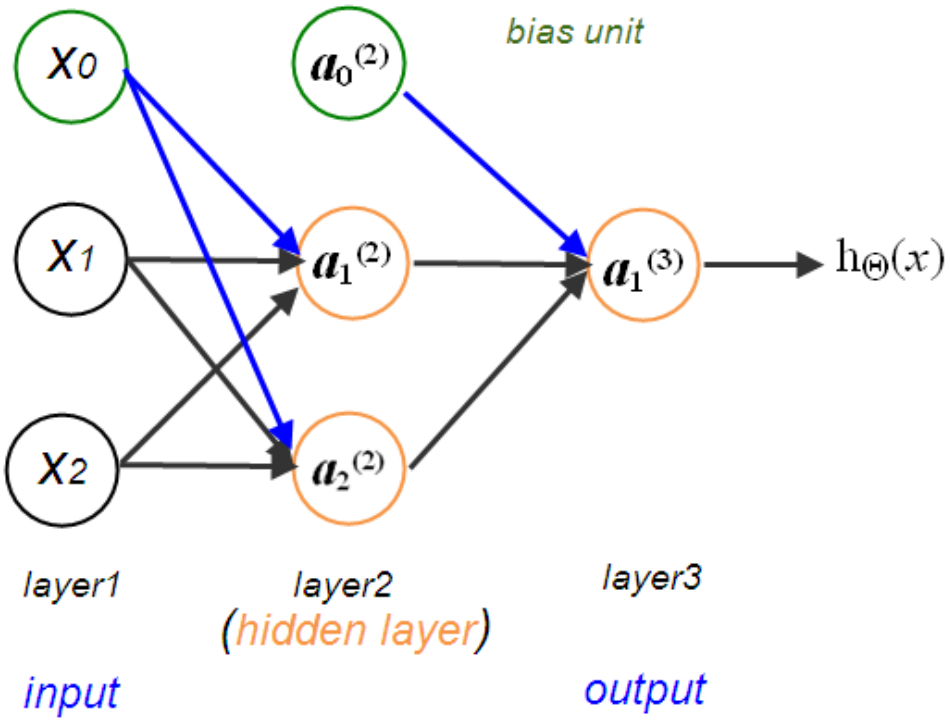
Fonte – (PEDREGOSA et al., 2011)

Um problema de classificação busca por diversas qualidades mensuráveis de uma instância, chamados características. A partir dessas características, cria-se um modelo matemático que tem por objetivo identificar a qual classe uma nova instância pertence. Pode-se pensar em problemas de classificação com um caso específico de problemas de regressão, onde utiliza-se a curva de regressão para dividir subconjuntos contendo características semelhantes.

2.1.3 Redes neurais artificiais

Redes neurais artificiais (ANN), do inglês *Artificial Neural Networks*, são uma técnica de aprendizado de máquina inspirada pelo funcionamento de sistemas nervosos biológicos. Uma rede neural convencional é constituída por vários processadores simples conectados, chamados de neurônios artificiais. Cada neurônio produz uma sequência de ativações. Os neurônios de entrada são ativados através de sensores que percebem o ambiente, outros neurônios, são ativados através de conexões ponderadas com neurônios previamente ativos. A figura 4 traz um exemplo de rede neural.

Figura 4 – Rede neural artificial.



Fonte – Retirado de (MORGAN, 2016)

McCulloch e Pitts (1943) propõem um modelo de neurônio formal baseado no funcionamento dos neurônios biológicos. A ideia consiste em um sistema em que dado um conjunto de entradas e uma função de soma, esse resultaria em uma saída desejada.

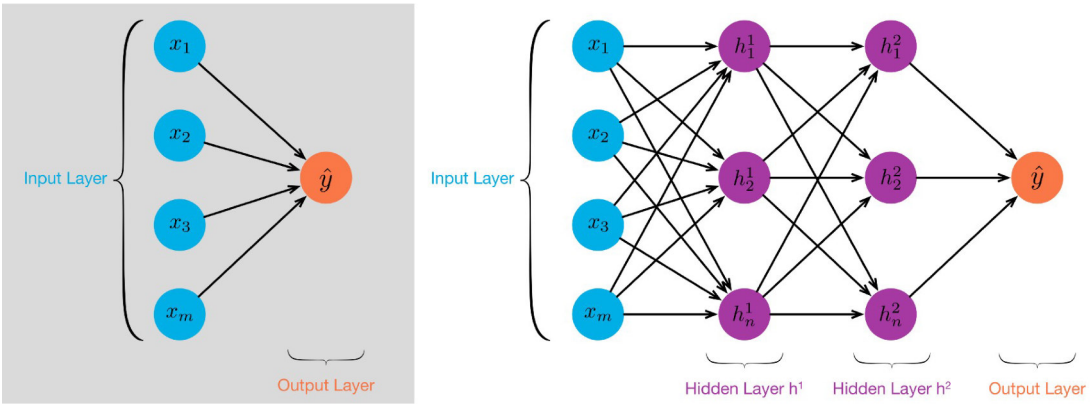
Na década seguinte, um importante avanço foi a invenção do algoritmo *perceptron*. Este algoritmo, além de ser capaz de avaliar valores de entrada, é capaz também de aprender qual resultado gerar a partir de um conjunto de dados de treino. Inicialmente, foi implementado em *software* para o computador IBM 704, porém foi rapidamente construído como *hardware* dedicado chamado de *Perceptron MK1* (ROSENBLATT, 1957).

Mesmo sendo um grade avanço, sua principal limitação era a sua incapacidade de

lidar com problemas não linearmente separáveis, como o operador lógico XOR. Para lidar com isto, foi proposto o uso de múltiplos *perceptrons* encadeados em uma rede conhecida como MLP (*multilayer perceptron* - *perceptron* multicamadas).

O MLP é um tipo de ANN que possui múltiplas camadas de neurônios, além da camada de entrada. A principal vantagem deste tipo de ANN é sua capacidade de resolver problemas não linearmente separáveis (CYBENKO, 1989). Atualmente redes neurais do tipo MLP são consideradas padrão, principalmente quando estas têm apenas uma camada escondida (FRIEDMAN; HASTIE; TIBSHIRANI, 2001). A figura 5 mostra a diferença entre redes de uma camada e de múltiplas camadas.

Figura 5 – Exemplo de ANN, com uma camada e com múltiplas camadas.



Fonte – (KANG, 2017)

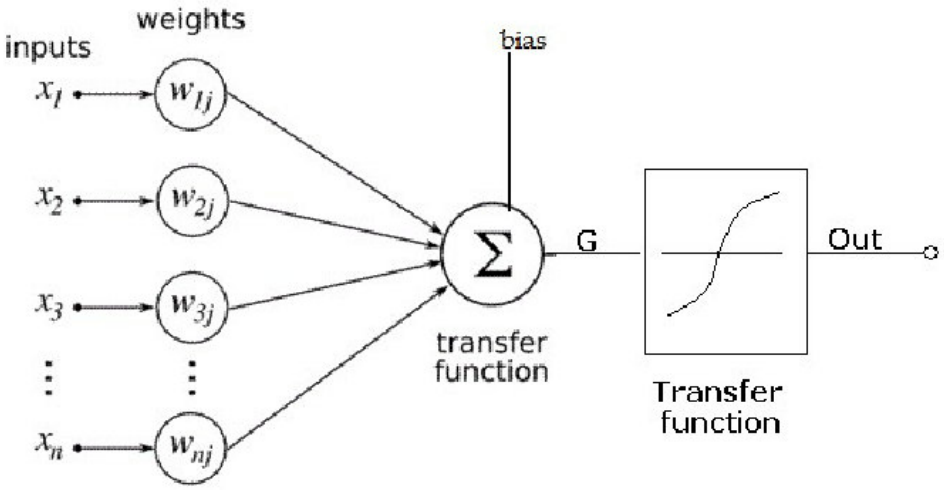
MLPs são aproximadores de função universais (CYBENKO, 1989). Desta forma, são bons algoritmos para regressão. Sendo classificação um caso específico de regressão, MLPs são, por consequência, uma boa técnica para classificação. Entre seus usos principais estão reconhecimento de fala, reconhecimento de imagens e tradução de máquina (WASSERMAN; SCHWARTZ, 1988). ANNs caíram em desuso com o aparecimento de SVM (*support vector machine* - máquina de vetores de suporte), porém sua popularidade tem crescido com o sucesso de técnicas de *deep learning*.

Em uma ANN cada neurônio artificial pode ser descrito matematicamente como uma função. Nesta função há a soma de todas as entradas que este neurônio recebe, multiplicadas pelo peso que cada uma destas possui. Além dos pesos e valores da entrada, o somatório também possui um valor de *bias*. Estes valores por fim, são passados por uma função de ativação que indica se o valor recebido será propagado para a próxima camada, como demonstra a figura 6. Um neurônio artificial pode ser descrito pela equação abaixo:

$$a^m = \sigma(\sum_{i=0}^{n} x_i^m \times w_i^m + b^m) \tag{2.1}$$

onde x_i^m são os valores de entrada, w_i^m são os pesos das entradas, σ é a função de ativação e b^m é o valor de *bias*.

Figura 6 – Neurônio artificial.



Fonte – (THATI et al., 2015)

Diversas funções podem ser utilizadas como função de ativação, algumas mais comuns são descritas pelas equações abaixo e pela figura 7.

Função sigmóide

$$f(x) = \frac{1}{1 + e^{-x}} \tag{2.2}$$

Tangente hiperbólica

$$f(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \tag{2.3}$$

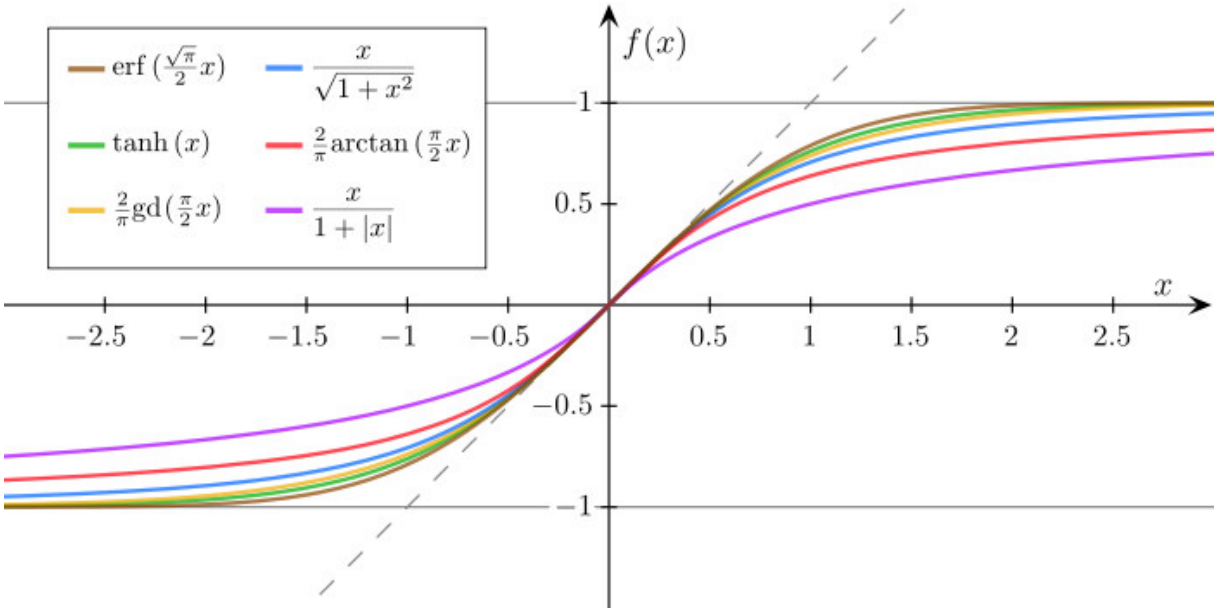
Softsign

$$f(x) = \frac{1}{1 + e^{-x}} \tag{2.4}$$

ReLu

$$f(x) = \begin{cases} 0 & \text{se } x < 0 \\ x & \text{se } x \geq 0 \end{cases} \tag{2.5}$$

Figura 7 – Gráfico de várias funções de ativação.



Fonte – Retirado de (MORGAN, 2016)

Pode-se descrever formalmente uma camada de rede neural como um conjunto de neurônios através da equação abaixo, onde σ representa a função de ativação, a^i representam os valores dos neurônios de uma camada, W os pesos das conexões e b o *bias*. O resultado dessa equação será a saída da camada atual, que pode ser utilizado em uma próxima camada ou representar o resultado final do processamento na rede neural.

$$a^i = \sigma(Wa^i + b) \tag{2.6}$$

O processo de aprendizado em uma rede neural está na procura de pesos para suas ligações que farão a rede produzir o resultado desejado. Dentre os algoritmos de aprendizado mais utilizados, encontra-se o algoritmo chamado de *backpropagation* (RUMELHART; HINTON; WILLIAMS, 1986). Este algoritmo possui uma etapa de *foward propagation*, seguida de uma etapa de *backward propagation*.

Na fase de *foward propagation*, a camada inicial recebe uma entrada e propaga-a por cada conexão a frente até os neurônios na outra extremidade dessas conexões. Estes, por sua vez, realizam um processamento, geralmente não linear, produzindo a saída da camada seguinte. Essa fase pode ser descrita pelas duas equações a seguir.

$$Z^{(n)} = X^{(n-1)} \cdot W^{(n-1)} \tag{2.7}$$

$$\hat{y} = f(Z^n) \tag{2.8}$$

onde X é a matriz de entradas, W é a matriz dos pesos, Z é o resultado da soma ponderada das entradas com os pesos, $f(Z^n)$ é a aplicação da função de ativação sobre a soma ponderada e $\hat{y}$ a entrada dos neurônios da camada seguinte. Esse processo é repetido para cada camada da rede, até atingir a camada final.

Inicialmente, o erro de saída da rede será alto, já que seus pesos não estão ajustados. Para resolver este problema, utiliza-se a fase *backward propagation*. Nesta fase, ocorre o aprendizado de fato.

Na etapa de *backward propagation*, o erro é propagado pela rede. Para que isso ocorra, utiliza-se o método de descida do gradiente para encontrar-se os pesos ótimos. Para cada peso de cada sinapse são realizados os seguintes passos. Primeiro, subtrai-se o resultado conseguido na saída pelo resultado esperado para calcular-se o erro. Após este passo, o erro é multiplicado pela derivada da função de ativação para obter-se o valor de variação. Este valor é então multiplicado pela derivada da função de ativação da camada anterior, de forma análoga a regra da cadeia. Tendo calculado todos os valores de variação, os novos pesos são definidos pela multiplicação da taxa de aprendizado com a variação (RUMELHART; HINTON; WILLIAMS, 1986).

O erro da camada anterior é calculado multiplicando-se o erro da camada atual pela transposta da matriz que representa essa camada. Esse processo é repetido por toda a rede até que a primeira camada seja atingida. Repetem-se todos esses passos até que atinja-se um número de iterações determinado ou o erro desejado. As equações abaixo descrevem esse processo.

$$E^{(n)} = y - \hat{y} \tag{2.9}$$

$$\Delta^{(n)} = f'(W^{(n)}) \cdot E^{(n)} \tag{2.10}$$

$$W^{(n-1)} = LR \cdot \Delta^{(n-1)} \tag{2.11}$$

$\hat{y}$ é a saída atual, y é o valor esperado, LR é a taxa de aprendizado, Δ é o gradiente e E o erro. $E^{(n)}$ é o resultado do da subtração do resultado obtido pelo resultado esperado. $W^{(n)}$ é a transposta da matriz de pesos e $f'(W^{(n)})$ a derivada da função de ativação. $W^{(n)}$ representa o erro na camada anterior, que é o resultado da multiplicação da taxa de aprendizado (LR) pelo gradiente ($\Delta^{(n-1)}$).

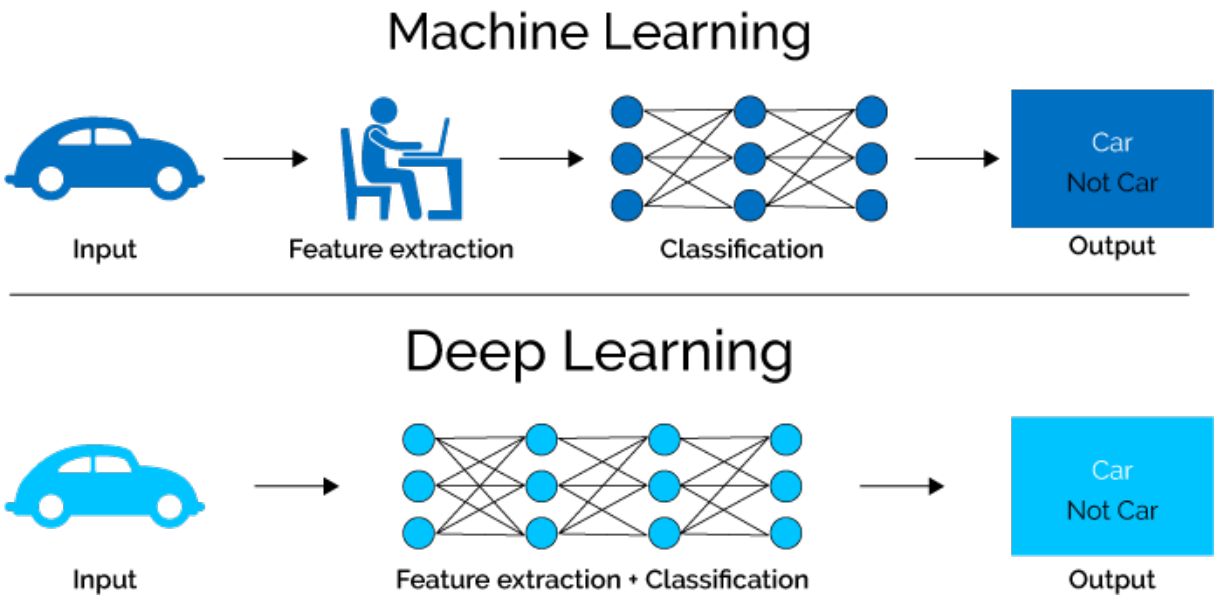
2.1.4 Deep learning

Deep learning, também conhecido como *deep structured learning* ou *hierarchical learning*, é um subcampo do aprendizado de máquina que busca encontrar representações de dados. *Deep learning* é uma aplicação em tarefas de aprendizado de redes neurais artificiais que contêm muitas camadas ocultas (BENGIO et al., 2009).

Uma das características principais de uma DNN (*Deep neural network* - Rede neural profunda) é usar um conjunto de múltiplas camadas de processadores não lineares em cascata. Estes processadores são utilizados para extração de características, de forma que, cada camada subsequente receba informação das camadas anteriores. Assim, aprendem múltiplos níveis de representação que correspondam a múltiplos níveis de abstração da problemática (DENG; YU et al., 2014).

Em contraste com técnicas tradicionais de aprendizado de máquina, em *deep learning* não é necessário a extração manual de características. Este processo é feito pelas próprias camadas da DNN. Desta forma, DNNs podem ser aplicadas em problemas demasiadamente complexos para a extração de características e definição de filtros manuais.

Figura 8 – Comparação entre *deep learning* e técnicas tradicionais de aprendizado de máquina.



Fonte – (SEIF, 2018)

Apesar de existir desde a década de 80 (DECHTER, 1986), o interesse por *deep learning* aumentou em 2012, quando Alex Krizhevsky utilizou redes neurais convolucionais para vencer a competição ImageNet (KRIZHEVSKY; SUTSKEVER; HINTON, 2012), desafio anual para avaliação de algoritmos de detecção de objetos e de classificação de imagens (RUSSAKOVSKY et al., 2015). Este desafio consiste em classificar 14 milhões

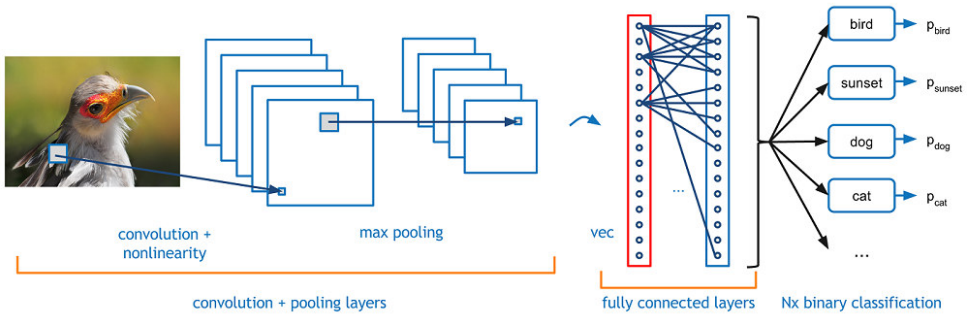
de imagens, entre 20 mil classes. Por seu grande volume de imagens, a ImageNet tem tornado-se um *benchmark* de algoritmos de visão computacional.

Em 2015, o *AlphaGo*, desenvolvido pela Google, foi o primeiro sistema de inteligência artificial a vencer um jogador profissional de "go"(SILVER et al., 2016). Este problema era considerado um dos grandes desafios do campo da inteligência artificial (BOUZY; CAZENAVE, 2001). Esta vitória demostrou o potencial de sistemas de aprendizado de máquina, baseados em *deep learning* para solução de problemas complexos.

2.1.5 Redes neurais convolucionais

Em 1962, Hubel e Wiesel mostraram que alguns neurônios no córtex visual são ativados ao verem arestas alinhadas em certos ângulos (HUBEL; WIESEL, 1962). Esta pesquisa mostrou que o córtex visual possui diversos grupos de neurônios responsáveis pela identificação de certos padrões em imagens. Juntos, estes grupos de neurônios são capazes de identificar elementos contidos na imagem, levando a percepção visual. Tendo como base esta ideia, elaborou-se um tipo de rede neural artificial capaz de executar este tipo de operação em imagens digitais. A este tipo de rede neural é dado o nome de rede neural convolucional (CNN). A figura 9 mostra a arquitetura de uma CNN típica, contendo inicialmente camadas convolucionais e por fim, camadas completamente conectadas.

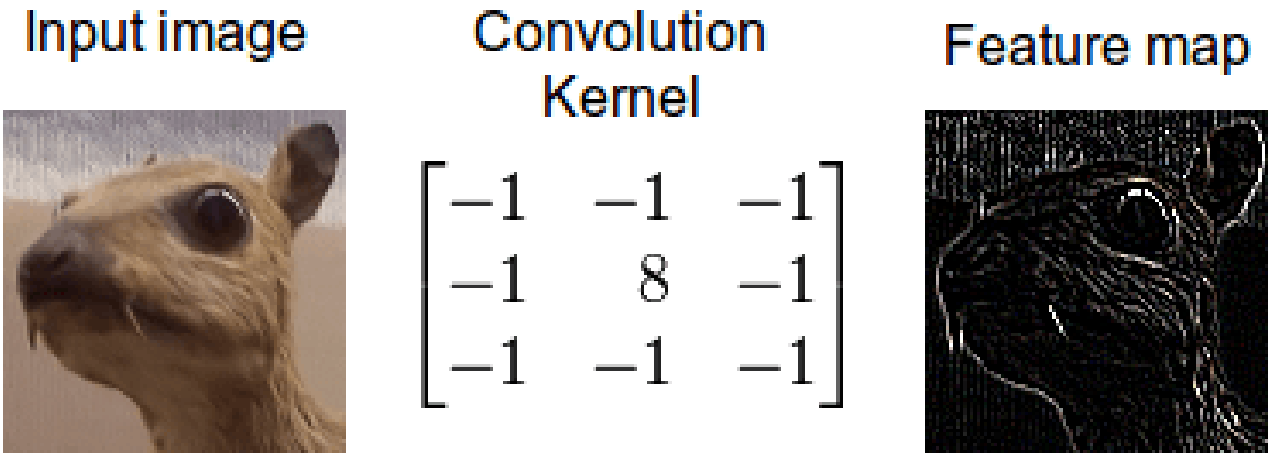
Figura 9 – Exemplo de rede neural convolucional.



Fonte – (DESHPANDE, 2016)

As CNNs recebem este nome devido ao operador de convolução. O objetivo desta operação é extrair características das imagens de entrada. Esta operação preserva a relação espacial entre os *pixels* ao aprender características da imagem usando pequenos quadrados de dados de entrada chamados de *kernels*. Convoluções são operações matemáticas lineares aplicadas em imagens para gerar uma outra imagem destacando as características desejadas, como demonstra a figura 10. Dentro de CNNs ocorrem diversas operações para a extração de características e identificação de imagens.

Figura 10 – Exemplo de kernel.



Fonte – (DETTMERS, 2015)

Uma camada convolucional em uma imagem consiste em um conjunto de *kernels* e *bias* que são aplicados em cima de uma imagem. Dada uma imagem I e um *kernel* K de dimensões $k_1 \times k_2$ a operação de convolução é dada pela equação abaixo.

$$(I * K)_{ij} = \sum_{m=0}^{k_1-1} \sum_{n=0}^{k_2-1} I(i-m, j-n)K(m,n) \tag{2.12}$$

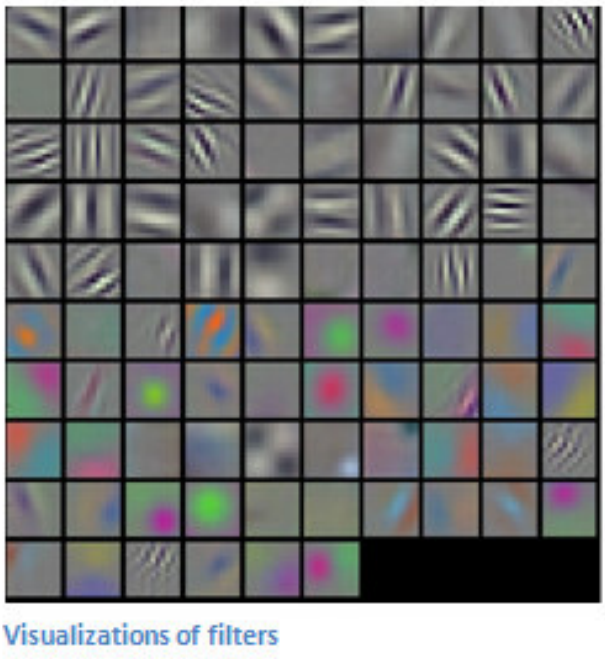
Desta forma, tem-se uma imagem de dimensões $M \times N \times C$, onde M e N são sua altura e largura e C é o número de canais. Sobre esta imagem, aplica-se o *kernel* e o *bias* seguindo a equação abaixo.

$$(I * K)_{ij} = \sum_{m=0}^{k_1-1} \sum_{n=0}^{k_2-1} \sum_{c=1}^{c=1} K_{m,n,c} \times I_{i+m,j+n,c} + b \tag{2.13}$$

O aprendizado de uma CNN está em encontrar valores para estes *kernel*. Quanto mais *kernels* possui-se, mais características serão extraídas, tornando a rede mais robusta para reconhecer imagens novas. Uma das grandes vantagens de CNN sobre técnicas tradicionais é a exigência mínima de pre-processamento das imagens.

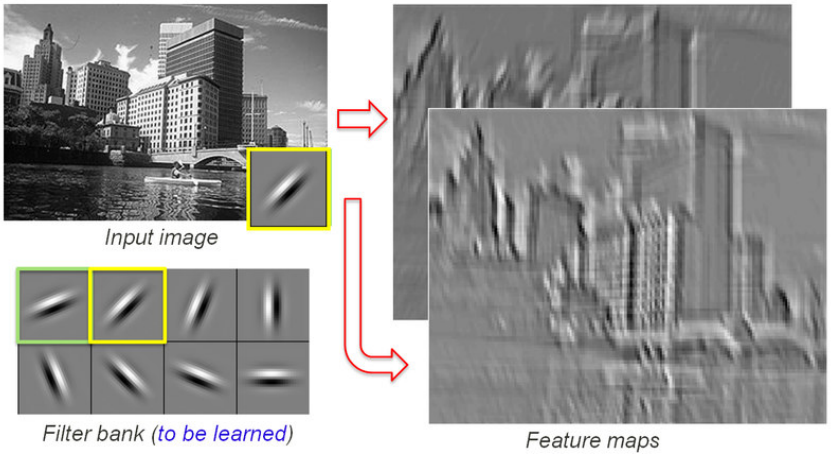
A figura 11 demonstra os *kernels* aprendidos por uma CNN. Ao aplicar esses *kernels* em uma imagem cria-se o *feature map*, uma imagem contendo certas características destacadas, como mostrado na figura 12. A cada convolução são extraídas características cada vez mais complexas para a criação de um modelo que represente o objeto a ser reconhecido, para por fim, executar a classificação (LEE et al., 2009).

Figura 11 – Filtros aprendidos por uma CNN.



Fonte – (DESHPANDE, 2016)

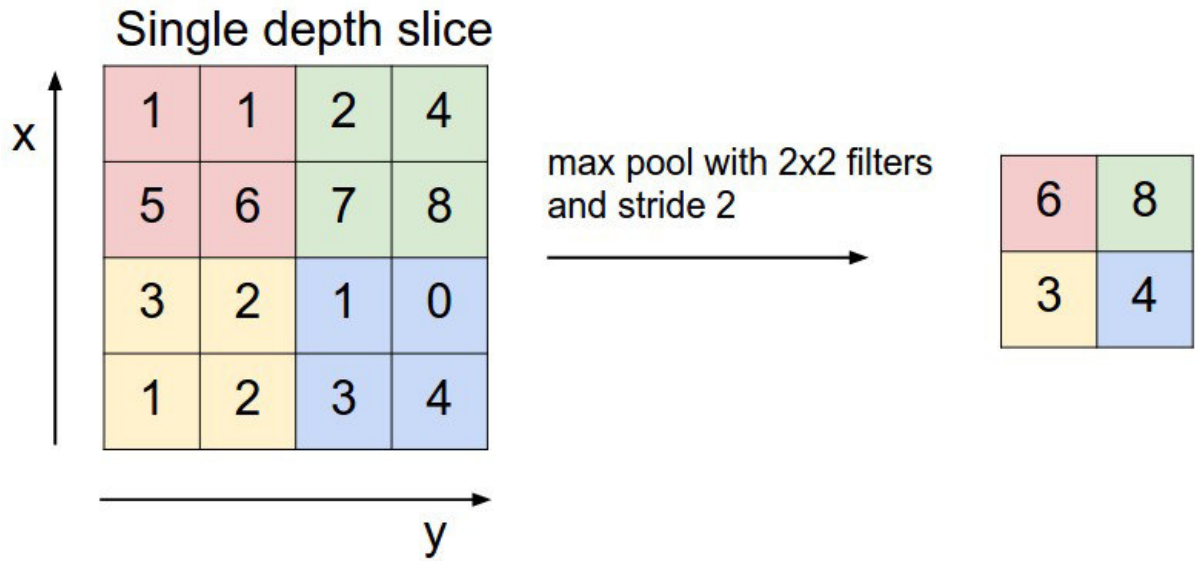
Figura 12 – Exemplo de *feature map*.



Fonte – (ZHOU, 2017)

Também existem as camadas de *pooling*. A função dessas camadas é diminuir progressivamente o tamanho espacial da representação de características, reduzindo a quantidade de computações na rede e auxiliando no controle de *overfitting*. Nesta camada não ocorre aprendizado (LECUN et al., 1989). A figura 13 demonstra um exemplo de uma camada de *pooling*.

Figura 13 – Exemplo de *pooling* 2×2.



Fonte – (KARPATHY, 2016)

Uma operação adicional pode ser realizada após cada convolução ou *pooling*, chamada de ReLU (Rectified linear unit - Unidade linear retificadora), definida como:

$$f(x) = \max(x, 0) \tag{2.14}$$

Essa operação substitui todos os *pixels* negativos no *feature map* por zeros. O propósito do ReLU é introduzir não linearidade na rede neural, pois a convolução é uma operação linear, ao contrário de muitos exemplos reais. Além de introduzir não linearidade a ReLU possui diversas vantagens:

- Redes convolucionais com ReLU's treinam de forma consideravelmente mais veloz do que redes equivalentes com função de ativação tanh (KRIZHEVSKY; SUTSKEVER; HINTON, 2012).
- As funções tanh/sigmóide envolvem computações de custo elevado (exponenciação), enquanto a ReLU é apenas um limiar.

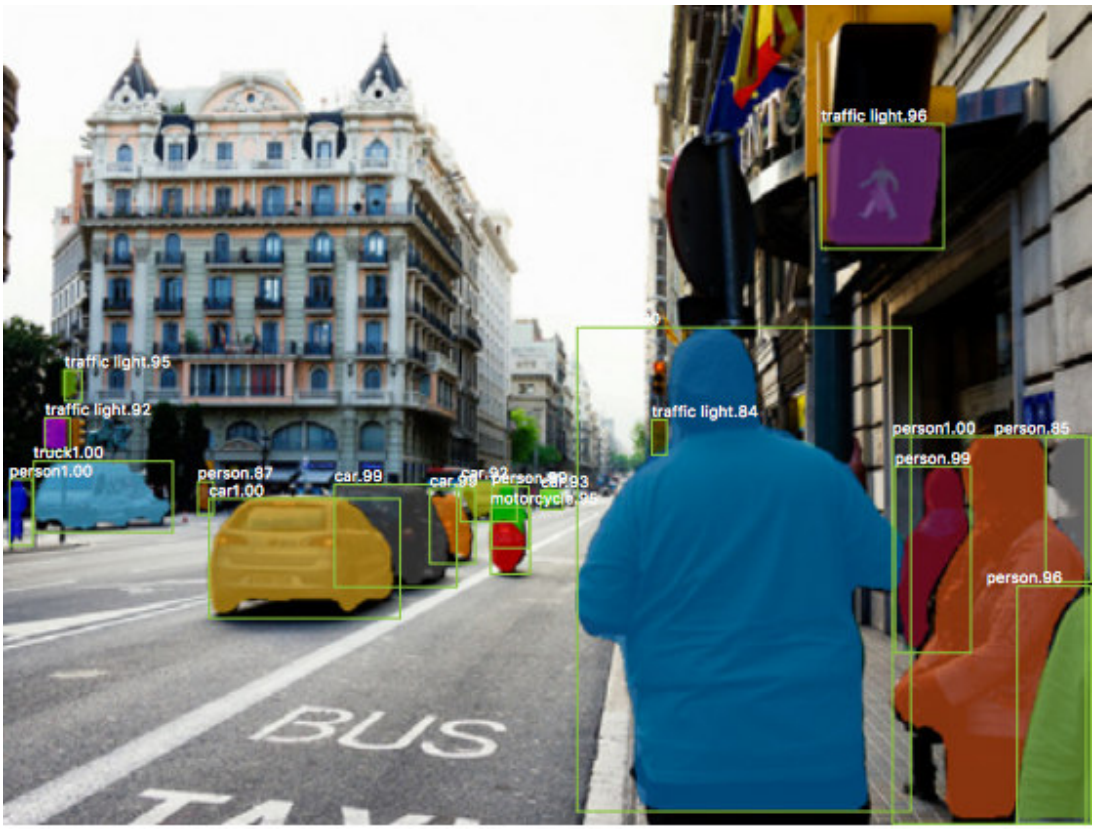
Por fim, há uma camada completamente conectada. Todos os neurônios desta camada estão conectados as neurônios da próxima camada, de forma análoga ao MLP tradicional. Nesta camada ocorre a classificação. O número de neurônios nesta camada é o mesmo número de classes a serem identificadas. Esta camada recebe o resultado das camadas anteriores e tenta predizer a qual classe as características recebidas relacionam-se mais (KRIZHEVSKY; SUTSKEVER; HINTON, 2012).

2.2 Visão computacional

Visão computacional é um campo multidisciplinar com foco em processamento de imagens e inteligência artificial. Este campo visa simular e automatizar o processo de visão humana (BALLARD; BROWN, 1982). Entre suas tarefas estão detecção de objetos, rastreamento de objetos em vídeo, descrição de imagens, entre outros.

Câmeras digitais tem tornado-se cada vez melhores em captar imagens com grande fidelidade. Aliadas ao sistemas baseados em inteligência artificial, permitem que computadores extraiam informação complexa a partir de imagens digitais.

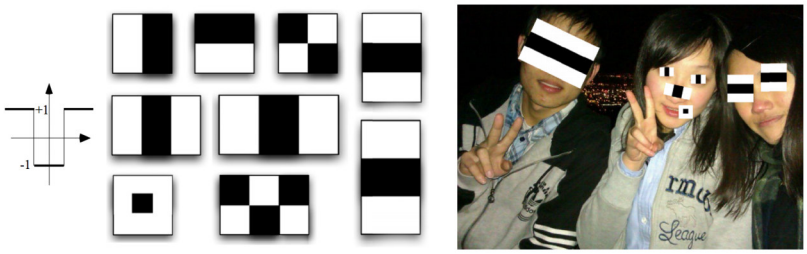
Figura 14 – Exemplo de aplicação de visão computacional.



Fonte – (RAVAL, 2017)

Em 2001, foi criado o primeiro algoritmo prático para detecção de faces por Paul Viola e Michael Jones (VIOLA; JONES, 2001). O algoritmo Viola–Jones utiliza um conjunto de características chamadas *haar features* e a partir delas cria um modelo de classificação com o uso de SVM. Em sua demostração inicial, foi capaz de detectar faces em tempo real por uma *webcam*. Apesar de revolucionário para seu tempo, este algoritmo era sensível a diferenças de ângulo e iluminação. A figura 15 demonstra uma aplicação de *haar features* para detecção de faces.

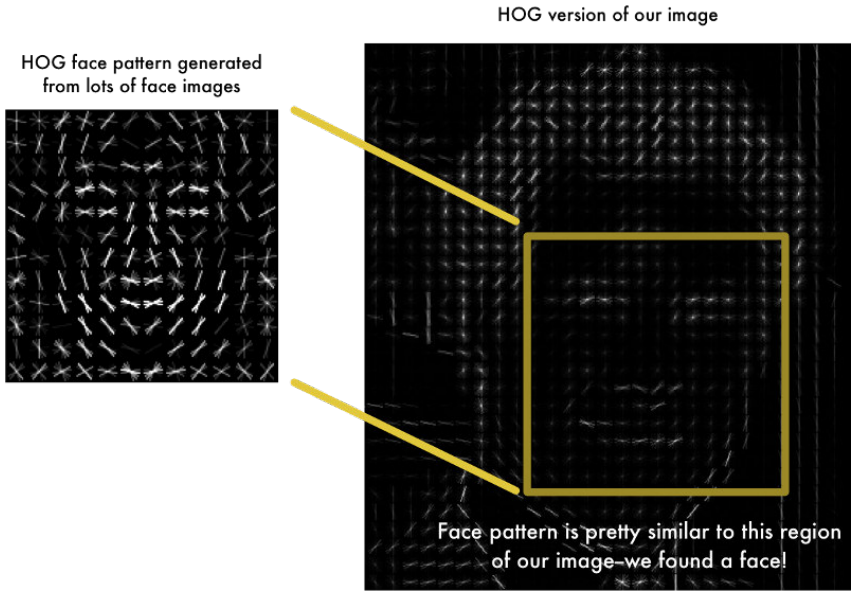
Figura 15 – Exemplo de *haar features*.



Fonte – (JALAL, 2017)

Em 2005, foi inventado um algoritmo que mostrou-se muito mais eficiente, o HOG (*Histogram of oriented gradients* - Histograma de gradientes orientados). Este algoritmo calcula as direções em que os *pixels* tornam-se mais escuros. Para cada *pixel* é calculada quais *pixels* ao seu redor tornam-se mais escuros e a partir disto calcula-se em qual direção, a região tende a escurecer. A imagem é dividida então em regiões de tamanhos iguais, onde mede-se qual direção geral este conjunto de *pixels* escurece. A essa direção geral é dado o nome de gradiente. Com a construção dos gradientes, pode-se criar um modelo que descreva de forma simples e robusta o objeto a ser detectado. Este conjunto de características então é aplicado em um classificador como uma ANN ou SVM. A figura 16 traz um exemplo de HOG.

Figura 16 – HOG aplicado a detecção de faces.

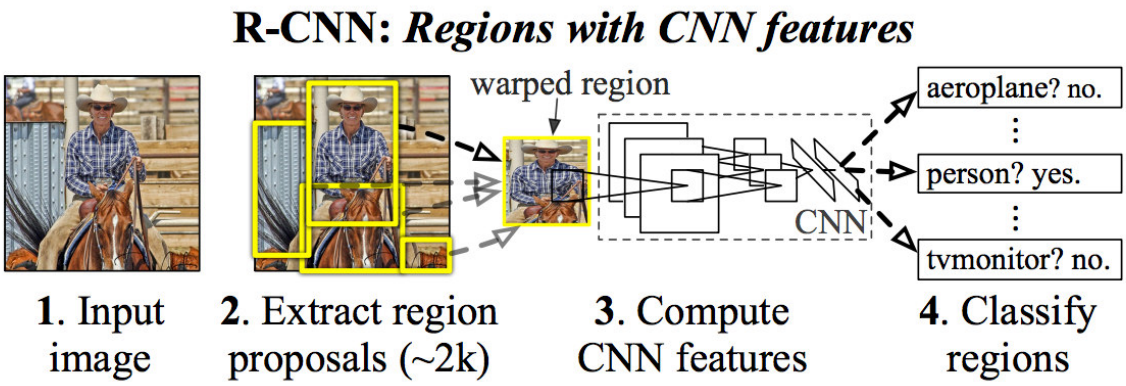


Fonte – (GEITGEY, 2016)

O grande avanço em visão computacional ocorreu em 2012, com a aplicação de redes neurais convolucionais. Inicialmente, CNNs apenas determinavam se a imagem continha o objeto a ser detectado. Porém, com técnicas como janela deslizante, ou seja, uma área que percorre a imagem saltando um *pixel* por vez e a cada salto reaplicando o classificador, foi possível seu uso em detecção de objetos. Entretanto, esta aproximação é lenta, já que o classificador deve ser executado muitas vezes (GIRSHICK et al., 2014).

Uma aproximação mais eficiente para este problema é a R-CNN. Este método utiliza um técnica chamada *selective search*, que consiste na utilização de janelas de diversos tamanhos para tentar agrupar *pixels* por semelhança de cor, textura e intensidade. Desta forma, criam-se regiões candidatas a presença de objetos, reduzindo o espaço de busca , como demonstrado na figura 17 (GIRSHICK et al., 2014).

Figura 17 – Exemplo de R-CNN.



Fonte – (RAVAL, 2017)

Além do R-CNN existem otimizações por cima desta mesma ideia como o Fast R-CNN (GIRSHICK, 2015), Faster R-CNN (REN et al., 2015), Mask R-CNN (HE et al., 2017). Porém, uma técnica utilizando uma aproximação diferente tem mostrado-se extremamente eficiente. O Darknet YOLO (You Only Look Once - Você só olha uma vez) utiliza uma aproximação que permite em apenas uma passada identificar os objetos contidos na imagem.

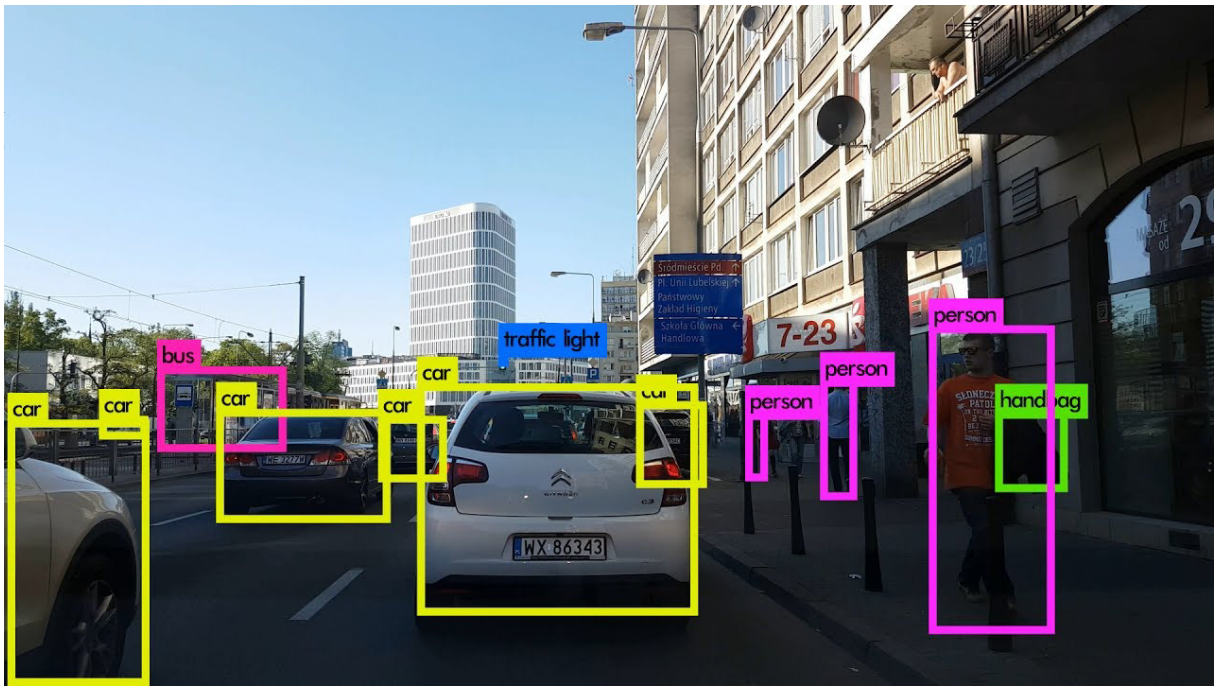
2.2.1 Darknet - YOLO

Detectores de objetos genéricos devem ser velozes, precisos e serem capazes de lidar com uma grande variedade de objetos. Desde sua introdução CNNs e *frameworks* de detecção tem tornado-se cada vez mais velozes e precisos. Entretanto a maioria dos detectores estão limitados há um pequeno conjunto de objetos (REDMON; FARHADI, 2016).

Criação de bases de dados para detecção de objetos é consideravelmente mais custoso e complexo que a criação de bases para classificação. Desta forma, a proposta do Darknet YOLO é a criação de um sistema que possa aproveitar-se da grande quantidade de dados existentes para classificação e utilizá-los para detecção (REDMON; FARHADI, 2016).

Darknet YOLO é um sistema de detecção de objetos de estado de arte baseado em redes neurais convolucionais (REDMON; FARHADI, 2016). Implementado inicialmente para Darknet, um *framework* de redes neurais de código aberto escrita em C e CUDA (REDMON, 2013–2016). A figura 18 demonstra a utilização do YOLO um carro autônomo.

Figura 18 – Darknet YOLO aplicado em um carro autônomo.



Fonte – (MAJEK, 2017)

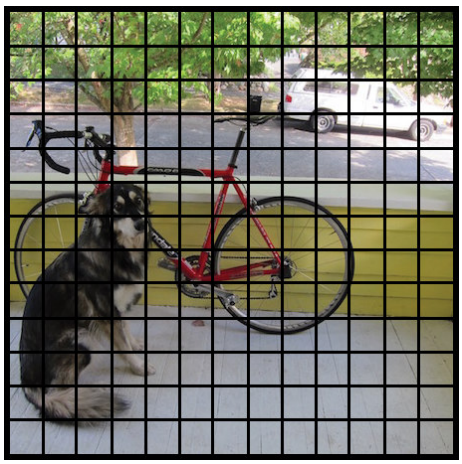
Classificadores tradicionais utilizam técnicas de janela deslizante ou *selective search* para encontrar regiões candidatas e identificar o objeto a ser procurado. Desta forma, regiões com probabilidades altas são consideradas detecções (REDMON; FARHADI, 2016).

Ao contrário de outros métodos, o YOLO não reutiliza um classificador como detector. Este algoritmo observa a imagem apenas uma vez. Para realizar a detecção, a imagem é subdividida em múltiplas sub-regiões. Para cada sub-regiões são calculadas diversas células e as probabilidades de cada uma dessas conter um objeto.

As células então são ordenadas pela probabilidade de um objeto estar contido nesta. Esta ordenação não informa nada sobre a classe, apenas informa se a célula possui probabilidade de conter um objeto. Por fim, células abaixo de um limiar são eliminadas, restando apenas as células com maior probabilidade de conter objetos (REDMON; FARHADI, 2016). Ao observar a imagem uma única vez, o YOLO mostra-se bem mais eficiente que técnicas concorrentes, mostrando-se 1000 vezes mais veloz que a R-CNN e 100 vezes mais veloz que a *Fast R-CNN* (REDMON; FARHADI, 2018). A figura 19 demonstra esse processo.

Figura 19 – Funcionamento do Darknet YOLO.

(a) Divisão inicial em sub-regiões.

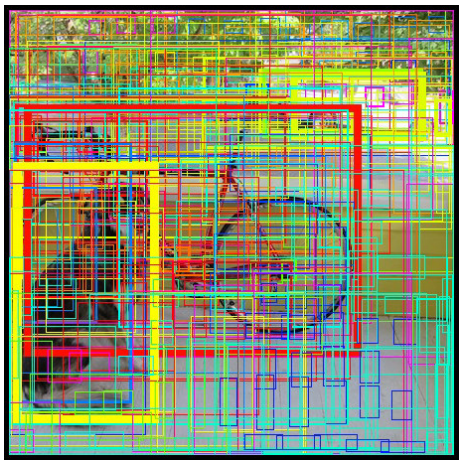


Fonte (RAVAL, 2017)

(b) Regiões candidatas.

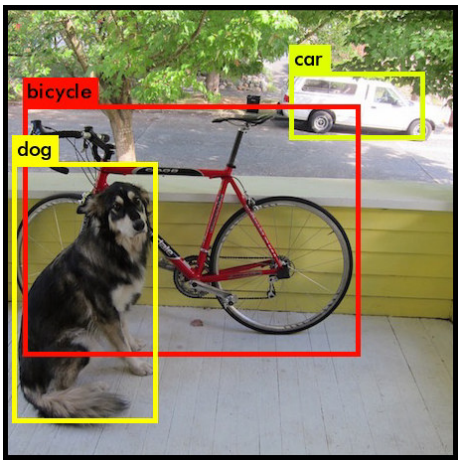
Fonte (RAVAL, 2017)

(c) Classificação das regiões.



Fonte (RAVAL, 2017)

(d) Resultado final.



Fonte (RAVAL, 2017)

Apesar de sua eficiência, o YOLO possui algumas limitações. O YOLO impõe fortes restrições espaciais a predições de *bounding box*, já que cada célula prevê um número limitado de *bounding box*, sendo que, cada uma só pode ter uma classe. Esta limitação espacial limita o número de objetos próximos que o modelo pode detectar. Este modelo tem dificuldades com pequenos objetos que aparecem em grupos, como bandos de pássaros (REDMON et al., 2016).

3 DETECÇÃO DE ARMAS DE FOGO

3.1 Ferramentas

Nesta sessão, são descritas as ferramentas adotadas por este trabalho. Estas ferramentas foram usadas para a aquisição de imagens utilizadas e para a criação da base de dados. São descritas a base de imagens, as ferramentas de *web crawling* e *scraping*, e a ferramenta para marcação de ROIs (*Regions of Interest* - Regiões de Interesse).

3.1.1 Banco de Imagens

O IMFDb é um banco de dados *on-line* de imagens de armas de fogo usadas ou exibidas em filmes, programas de televisão, videogames e animes. Lá, estão incluídos, artigos relacionados à atores, e alguns personagens famosos, como James Bond, listando as armas específicas associadas em suas obras (BOURJAILY, 2009). A figura 20 contém um exemplo de artigo do IMFDb.

Figura 20 – Exemplo de uma imagem de um artigo do IMFDb.

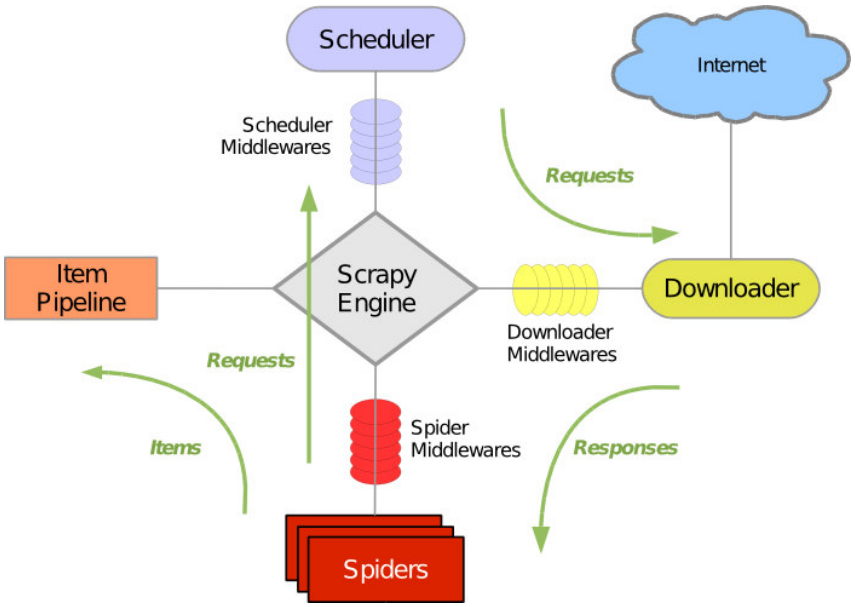
Fonte – (BUNNI, 2007)

Integrado no site, há uma seção de hospedagem de imagens semelhante à *Wikimedia Commons*, que inclui fotos de armas de fogo, logotipos de fabricantes, capturas de tela e arte relacionada (BOURJAILY, 2009). Desta sessão foi extraída a base de dados utilizada para casos positivos. Foram extraídas por volta de 20 mil imagens, das quais foram utilizadas aproximadamente 4600, uma vez que o processo de marcação foi manual.

3.1.2 Scrapy

Para obtenção das imagens, foi implementado um *webcrawler*, programa de computador que navega automaticamente na internet seguindo regras pré-definidas, a partir da biblioteca Scrapy (KOROBOV, 2015). Scrapy é uma biblioteca de código aberto, usada para raspagem de sites, extração de dados estruturados das páginas e *web crawling*. Pode ser usada para um grande número de aplicações, como mineração de dados, processamento de informação e arquivamento. Atualmente, é mantida pela Scrapinghub Ltda., uma empresa de desenvolvimento e serviços de correção web. A figura 21 contém um diagrama da arquitetura da biblioteca.

Figura 21 – Arquitetura do Scrapy.



Fonte – (KOROBOV, 2015)

Sua arquitetura baseia-se na construção de *spiders*, que são entidades responsáveis pela navegação dentro das URLs fornecidas. Além de navegar e acessar links, uma *spider* também processa as informações encontradas, podendo filtrá-las e baixá-las. Para este trabalho foi construída uma *spider* que busca dentro do IMFDb, imagens e faz o *download* dessas, buscando por páginas de modelos específicos de armas de fogo.

O critério para as imagens foi a busca de modelos mais populares de armas, como fuzis AK-47 ou M-16 (UN. SECRETARY-GENERAL et al., 1997), pistolas Colt M1911, Glock ou Sig, espingardas como Remington M870, Mossberg 500 (BRAUER, 2013). Muitas vezes, obras contendo estes modelos também continham outros modelos não especificados durante as buscas. Desta forma, foram adquiridas imagens contendo grande variedade de modelos.

3.1.3 Google Images Download

Para a criação de base de imagens negativas, foi utilizado o *script* em Python, Google Images Download. Esta ferramenta é um programa de linha de comando para pesquisar palavras-chave ou frases-chave no Google Imagens e opcionalmente, baixá-las, podendo também ser invocado outro arquivo Python (VASA, 2018).

Figura 22 – Exemplos de casos negativos.

(a) Exemplo de caso negativo 1.

Fonte (HAYNES, 2011)

(b) Exemplo de caso negativo 2.

Fonte (DENNIS, 2018)

(c) Exemplo de caso negativo 3.

Fonte (BLACK; DECKER, 2018)

(d) Exemplo de caso negativo 4.

Fonte (BOSH, 2018)

As imagens buscadas foram de objetos alusivos a armas de fogo, tanto em forma quanto em textura, como secadores de cabelo e furadeiras. Também foram buscados, objetos classificados incorretamente durante testes preliminares como guarda chuvas ou corrimãos metálicos. A base de casos negativos construída continha 1939 imagens. Desta forma, obteve-se uma base de contra exemplos para os quais a aplicação não deveria classificar como arma de fogo.

3.2 Métodos

Nesta sessão, é descrita a metodologia adotada para o desenvolvimento deste trabalho. Primeiramente, é descrito o processo de preparo da base de dados para o treinamento da CNN. Por fim, descreve-se o processo de treinamento do sistema.

3.2.1 Marcação da Base

Após a aquisição da base de dados, foi realizado o preparo desta. Esse preparo consistiu em utilizar a ferramenta BBox-Label-Tool (QIU, 2016) para marcar as ROIs nas imagens. Este processo foi realizado apenas para o conjunto de imagens positivas, visto que, no conjunto de imagens negativas não há regiões de interesse.

Foram consideradas, apenas imagens contendo armas de pequeno porte (*small arms*), pela definição da ONU, no protocolo SALW (*Small arms and light weapons* - Armas portáteis e armas leves). Dentro desta categoria, estão incluídas armas curtas (pistolas e revólveres) e armas longas (submetralhadoras, carabinas, fuzis, espingardas e metralhadoras leves) (UN. SECRETARY-GENERAL et al., 1997) (SURVEY; (GENEVA, 2007). Foram ignorados lançadores de granadas, lançadores de foguetes, armas montadas em veículos e explosivos.

Para cada imagem, foi gerado um arquivo com o mesmo nome desta. Este arquivo continha a quantidade de *labels* na imagem e as coordenadas das *bounding boxes* de cada uma dessas. Em alguns casos, não foram preenchidas *labels* de uma imagem, por não obedecerem a um critério mínimo de qualidade, tendo baixa resolução ou iluminação ruim. Estas imagens foram salvas com zero *labels* (nenhuma marcação) e foram depois eliminadas da base positiva.

Das 20 mil imagens obtidas, apenas 4646 foram marcadas. O processo de marcação de ROI nas imagens foi executado de forma manual. Cada imagem foi analisada individualmente e as regiões contendo armas de fogo foram marcadas manualmente. Por esta razão, apesar de terem sido baixadas 20 mil imagens, foram utilizadas. A figura 23 demonstra um exemplo da ferramenta marcação.

Figura 23 – *Screenshot* da ferramenta de marcação.

Fonte – Autor

3.2.2 Treinamento

Tendo preparado as bases de dados, a etapa de treinamento pôde começar. Nesta etapa, foi utilizado o *framework* Darknet para a CNN YOLO. A arquitetura utilizada foi a mesma empregada para a competição Pascal VOC, uma vez que esta se mostrou extremamente eficiente na classificação de mais de 9 mil classes de objetos. Muitas vezes,

as imagens continham objetos de mesma classe com grande variedade de elementos, como cachorros e pássaros, além de possuir objetos mecânicos, também, com grande variedade, como carros e aviões (EVERINGHAM et al., 2015).

A base de dados foi dividida aleatoriamente em dois grupos, treino e teste. Utilizou-se 90% (5916 imagens) das imagens para treino e 10% (669 imagens) para teste. A taxa de aprendizado utilizada foi de 0,0001, o *momentum* de 0,9 e o *decay* de 0,0005. O *batch size* foi de 64 imagens e o tamanho da camada de entrada foi 416×416. Por fim, foram geradas âncoras para o treinamento em cima da base de casos positivos, com uso do algoritmo de agrupamento k-means (LLOYD, 1982).

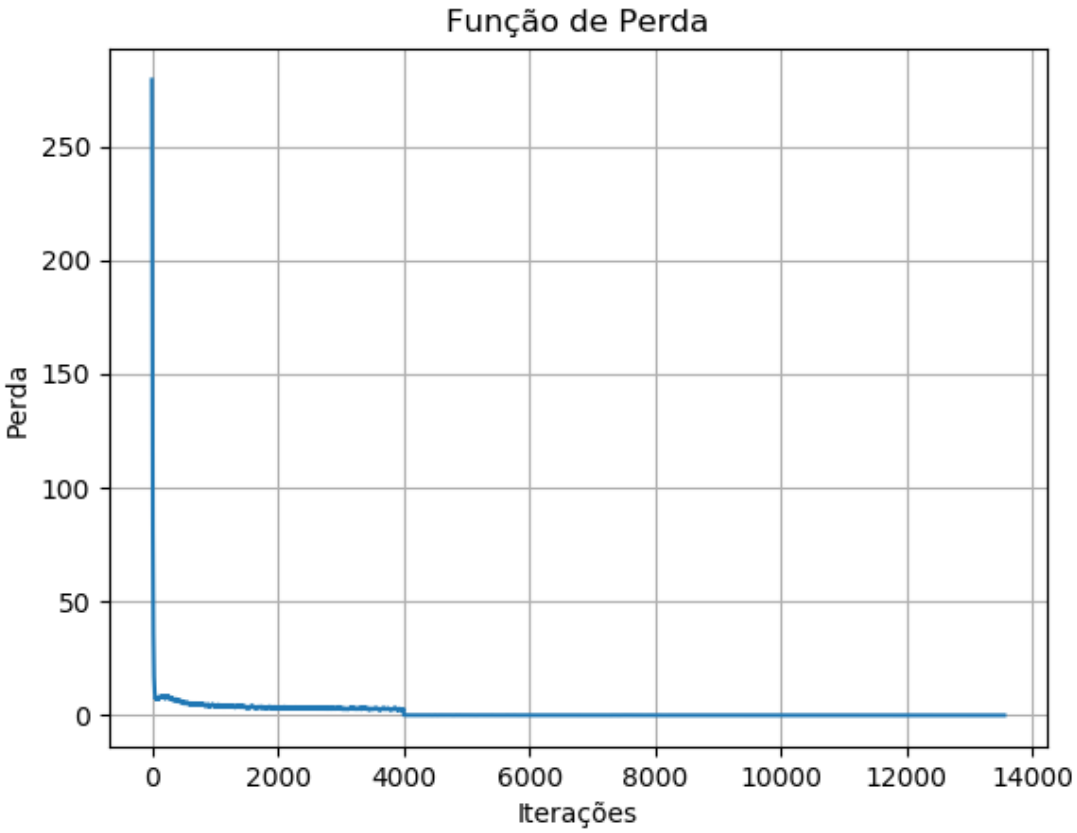
A arquitetura utilizada pelo YOLO é a arquitetura de uma CNN convencional. Ela é baseada na arquitetura VGG, utilizando filtros 3×3 e dobrando o número de canais após cada camada de *max pooling*.

O processo de treinamento foi executado utilizando um computador contendo uma GPU modelo NVidia GTX 1050. Este equipamento foi necessário já que o tempo de treinamento e execução em CPU (*Central Processing Unit* - Unidade Central de Processamento) seria demasiadamente elevado. Cada época levou em média 5,5 segundos. Até as primeiras 1000 épocas, foi salvo um *checkpoint* da rede a cada 100 épocas, após as 1000 primeiras, este intervalo foi aumentado para 1000 épocas. O gráfico na figura 24 mostra a função de perda da rede durante o treinamento.

Pode-se perceber que a perda cai rapidamente com poucas dezenas de iterações, mostrando a eficiência da ferramenta em aprender a detectar os objetos desejados. Os *checkpoints* gerados durante o treinamento permitem avaliar o aprendizado da rede durante várias épocas, garantindo que *overfitting* seja evitado.

A arquitetura da rede utilizada neste projeto foi baseada na arquitetura utilizada por (REDMON; FARHADI, 2016) no desafio PASCAL VOC (EVERINGHAM et al., 2015). Esta arquitetura foi escolhida por ter mostrado-se eficiente para a classificação e detecção de objetos durante o desafio mencionado. A descrição da arquitetura encontra-se na tabela 1.

Figura 24 – Função de perda do treinamento após 13 mil iterações.



Fonte – Autor

Tabela 1 – Arquitetura utilizada neste trabalho

Type	Filter	Size/Stride	Input	Output
Convolutional	32	3 X 3 / 1	416 X 416 X 3	416 X 416 X 32
Max Pooling		2 X 2 / 2	416 X 416 X 32	208 X 208 X 32
Convolutional	64	3 X 3 / 1	208 X 208 X 32	208 X 208 X 64
Max Pooling		2 X 2 / 2	208 X 208 X 64	104 X 104 X 64
Convolutional	128	3 X 3 / 1	104 X 104 X 64	104 X 104 X 128
Convolutional	64	1 X 1 / 1	104 X 104 X 128	104 X 104 X 64
Convolutional	128	3 X 3 / 1	104 X 104 X 64	104 X 104 X 128
Max Pooling		2 X 2 / 2	104 X 104 X 128	52 X 52 X 128
Convolutional	256	3 X 3 / 1	52 X 52 X 128	52 X 52 X 256
Convolutional	128	1 X 1 / 1	52 X 52 X 256	52 X 52 X 128
Convolutional	256	3 X 3 / 1	52 X 52 X 128	52 X 52 X 256
Max Pooling		2 X 2 / 2	52 X 52 X 256	26 X 26 X 256
Convolutional	512	3 X 3 / 1	26 X 26 X 256	26 X 26 X 512
Convolutional	256	1 X 1 / 1	26 X 26 X 512	26 X 26 X 256
Convolutional	512	3 X 3 / 1	26 X 26 X 256	26 X 26 X 512
Convolutional	256	1 X 1 / 1	26 X 26 X 512	26 X 26 X 256
Convolutional	512	3 X 3 / 1	26 X 26 X 256	26 X 26 X 512
Max Pooling		2 X 2 / 2	26 X 26 X 512	13 X 13 X 512
Convolutional	1024	3 X 3 / 1	13 X 13 X 512	13 X 13 X 1024
Convolutional	512	1 X 1 / 1	13 X 13 X 1024	13 X 13 X 512
Convolutional	1024	3 X 3 / 1	13 X 13 X 512	13 X 13 X 1024
Convolutional	512	1 X 1 / 1	13 X 13 X 1024	13 X 13 X 512
Convolutional	1024	3 X 3 / 1	13 X 13 X 512	13 X 13 X 1024
Convolutional	1024	3 X 3 / 1	13 X 13 X 1024	13 X 13 X 1024
Convolutional	1024	3 X 3 / 1	13 X 13 X 1024	13 X 13 X 1024
Route	16			
Reorg		/2	26 X 26 X 512	13 X 13 X 2048
Route	26 24			
Convolutional	1024	3 X 3 / 1	13 X 13 X3072	13 X 13 X 1024
Convolutional	30	1 X 1 / 1	13 X 13 X1024	13 X 13 X 30
Detection				

4 TESTES E AVALIAÇÕES

Este capítulo trata da apresentação dos resultados obtidos durante este trabalho. Estes testes foram realizados para analisar o desempenho da metodologia. O sistema tem como objetivo executar duas tarefas principais, classificação e detecção. O problema de classificação consiste em avaliar presença do objeto procurado na imagem apresentada. Já o problema de detecção trata de identificar a localização do objeto, além de sua presença na imagem.

4.1 Métricas

O problema de classificação foi avaliado de maneira tradicional. Para cada imagem de teste, o sistema teve como objetivo identificar a presença ou ausência de arma de fogo. Para a avaliação da classificação, foram usadas a sensibilidade, especificidade, acurácia e foi traçada a curva ROC (*Receiver Operating Characteristic* - característica de operação do receptor).

Para o problema de detecção, os testes padrões usados no desafio PASCAL VOC (EVERINGHAM et al., 2008) foram utilizados como métricas. Estas métricas foram escolhidas para gerar um critério de comparação objetivo da técnica utilizada neste trabalho com técnicas futuras.

Para validação dos resultados de detecção, foram considerados verdadeiros positivos detecções onde a *bounding box* gerada pelo sistema, sobrepunha-se a *bounding box* demarcada durante o prepara da base em caso de teste positivo ou ausência de *bounding box* em imagens negativos. Para este teste foram considerados falsos positivos a presença de *bounding box* em regiões que não continham arma de fogo e falsos negativos, regiões que continham arma de fogo, porém não foram marcadas.

4.1.1 Classificação e Validação dos Resultados

Para avaliação dos resultados, foram utilizadas algumas medidas de estatística descritiva, como sensibilidade, especificidade e acurácia. Estes valores são determinados seguindo as equações abaixo (FAWCETT, 2006).

A sensibilidade indica a porcentagem de casos em que o objeto foi identificado, sendo calculada pela seguinte equação:

$$S = \frac{VP}{VP + FN} \tag{4.1}$$

onde S é a sensibilidade, VP indica o número de verdadeiros positivos e FN o número de falsos negativos.

A especificidade indica o número de casos positivos que não foram corretamente classificados. Este valor é dado pela equação abaixo:

$$E = \frac{VN}{VN + FP} \tag{4.2}$$

onde E é a especificidade, VN indica o número de verdadeiros negativos e FP o número de falsos positivos.

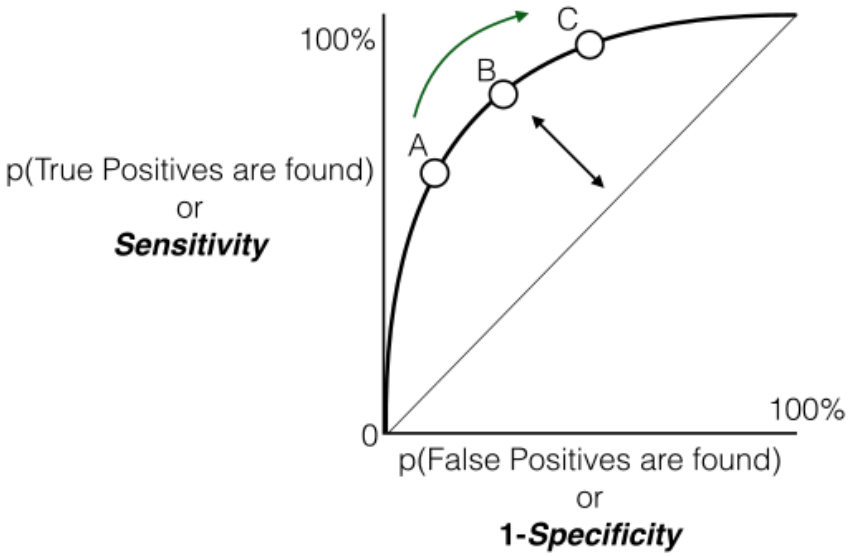
A acurácia indica a porcentagem de casos classificados corretamente dentro de todos os casos de teste. A acurácia é calculada a partir da seguinte equação:

$$A = \frac{VP + VN}{VP + VN + FP + FN} \tag{4.3}$$

4.1.2 Curva ROC

Com os resultados obtidos anteriormente, foi traçada a curva ROC. Esta curva trata-se da representação gráfica do desempenho de um classificador binário com limiares variados (FAWCETT, 2006). O espaço ROC é definido pela taxa de falsos positivos (FPR) dada por $1 - sensibilidade$, no eixo x e pela taxa de verdadeiros positivos (TPR) dada pela sensibilidade, no eixo y. Um classificador ideal encontra-se no canto superior esquerdo, onde todas as instâncias positivas foram corretamente identificadas e não ocorreu nenhum falso positivo. A figura 25 traz um exemplo de curva ROC.

Figura 25 – Exemplo de curva ROC.



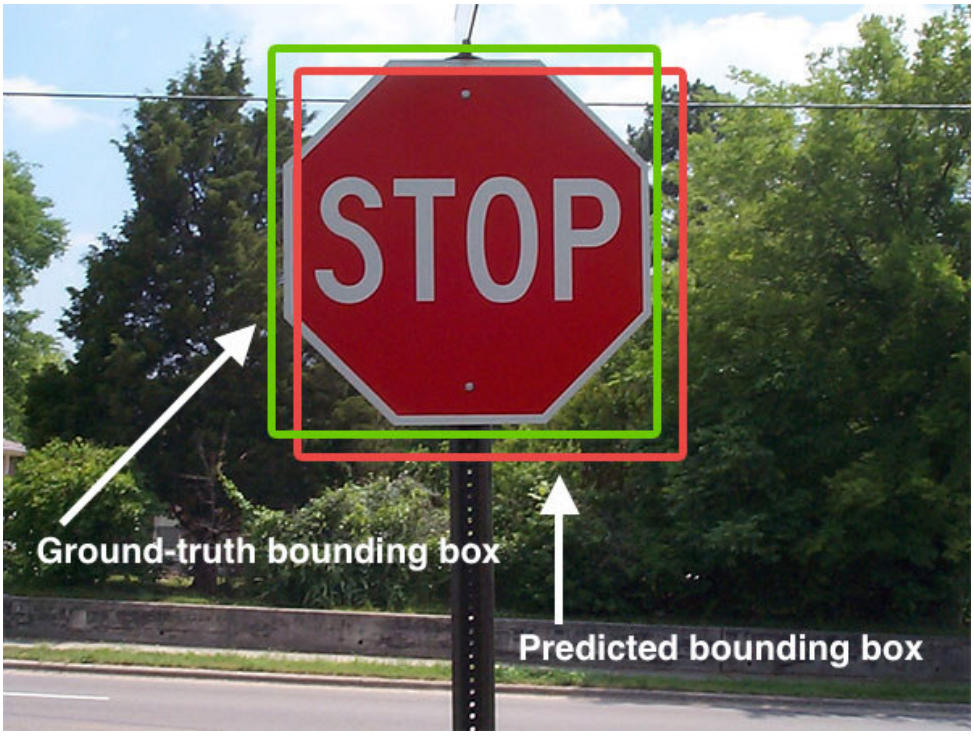
Fonte – (COMMONS, 2016)

4.1.3 Interseção sobre União

A interseção sobre união (IoU), do inglês *Intersection over Union*, também conhecido como índice de Jaccard, é o cálculo da interseção da *bounding box* de detecção pela *bounding box* de *ground truth*. Esta métrica é muito utilizada em desafios de detecção de objetos como a PASCAL VOC (EVERINGHAM et al., 2010) e para avaliação de algoritmos como HOG + SVM e CNNs (ROSEBROCK, 2016).

Para avaliar um detector de objetos com IoU, são necessários os valores de *ground truth*, ou seja, os valores das *bounding boxes* marcadas manualmente, e os valores das *bounding boxes* obtidas na detecção. Qualquer algoritmo que gere *bounding box* pode ser avaliado com uso de IoU.

Figura 26 – Exemplo de *bounding box*. Em vermelho a detecção, em verde o *ground truth*. O IoU calcula a interseção entre as duas *bounding boxes*.



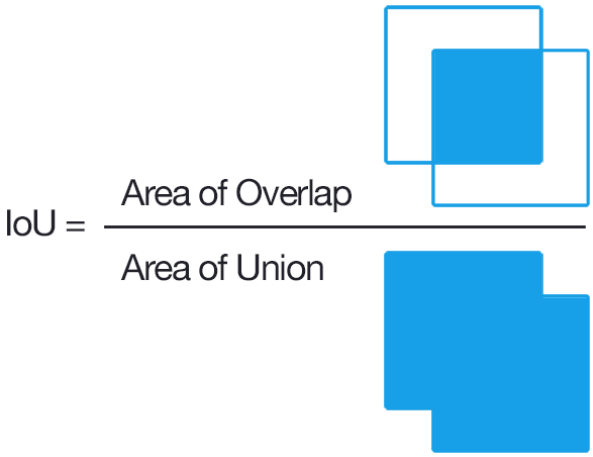
Fonte – (ROSEBROCK, 2016)

Com os valores de detecção e de *ground truth*, são calculadas as interseções, ou seja, a área encoberta por ambas as *bounding boxes*, e a uniões das *bounding boxes*, que consistem nas áreas de ambas as *bounding boxes* somadas. Com estes resultados divide-se a área de interseção pela área de união, obtendo-se o IoU.

Formalmente pode-se definir o cálculo da IoU com a equação abaixo, onde IoU é o valor de *Intersection over Union* e A e B são os valores das *bounding box*:

$$IoU = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|} \tag{4.4}$$

Figura 27 – Equação de interseção sobre união.



Fonte – (ROSEBROCK, 2016)

No desafio PASCAL VOC, utiliza-se um valor de IoU de 50%, com este valor considera-se um *match*, ou seja, uma classificação positiva (EVERINGHAM et al., 2015). Valores com IoU inferior a 50% são considerados erros, mesmo que contenham o objeto a ser detectado. A *bounding box* detectada deve possuir a mesma *label* da *ground truth* e a região deve ser detectada apenas uma vez para ser considerado um verdadeiro positivo (EVERINGHAM et al., 2010).

Figura 28 – Exemplo de avaliação de múltiplas IoU sobre múltiplas *bounding box*.

Fonte – (ROSEBROCK, 2016)

4.1.4 Mean Average Precision

A mAP (*mean Average Precision* - média da precisão média) é a medida padrão para comparar algoritmos de busca. Umas de suas principais vantagens é a geração de um único valor numérico como resultado, facilitando a comparação objetiva entre técnicas diferentes. É calculada pelo valor médio da precisão média. A precisão média (AP) é a média dos valores de precisão em todas as classificações em que os valores relevantes são encontrados.

Para todas as classes, calcula-se a média da precisão. Após isto, calcula-se o valor médio entre o resultado das médias de precisão de cada classe, resultando no mAP. Este valor é muito utilizado para comparação de algoritmos de busca e mais recentemente de detecção de objetos, sendo uma das métricas padrões no desafio PASCAL VOC (EVERINGHAM et al., 2015). Estes cálculos são realizados seguindo as equações abaixo.

$$AP = \int_0^1 p(r)dr \tag{4.5}$$

$$mAP = \frac{\sum_{i=1}^{Q} AP(i)}{Q} \tag{4.6}$$

Precisão e revocação são métricas de valor único com base em toda a lista de instâncias identificadas pelo sistema. Ao calcular-se a precisão e revocação em todas as posições na sequência classificada de instâncias, pode-se traçar a curva de precisão por revocação. A AP é definida pela curva de precisão e revocação como a média ponderada das precisões obtidas em cada limiar, com o aumento da revocação do limiar anterior usado como o peso. A AP é dada pela área sob a curva de revocação de precisão. Para cada classe calcula-se individualmente seu valor de AP (ZHU, 2004).

Por fim a mAP é calculada a partir dos resultado obtidos na soma anterior, para isto são somados todos os valores de AP e esses são divididos pelo número total de classes. Para problemas com apenas uma classe, o valor de mAP é igual ao valor de AP.

4.2 Resultados

Nesta sessão, são apresentados os resultados obtidos em cada um dos experimentos realizados. Estes testes foram realizados para analisar o desempenho da metodologia utilizada. Na base de dados utilizada para testes, havia um total de 669 imagens, 446 positivas e 223 negativas, contendo um total de 554 objetos marcados.

Inicialmente, foram realizados testes com o treinamento feito utilizando-se apenas imagens com casos positivos. Esta rede foi treinada durante seis mil épocas. Estes testes foram feitos para avaliar a eficiência de detecção da ferramenta e verificar quais objetos eram detectados incorretamente. Após isto, foi construída a base de dados de casos negativos, baseada nos resultados obtidos nos testes preliminares e um treinamento subsequente foi realizado. Este treinamento foi feito durante 13 mil épocas.

A aplicação mostrou-se eficiente em detectar armas de pequeno porte de todos os tipos e modelos. Demonstrando a capacidade de localizar objetos mesmo quando ocorre oclusão parcial. Além de ser capaz de detectar todo tipo de arma de pequeno porte, o

sistema foi capaz de detectar armas leves, como metralhadoras antiaéreas e armas montadas em veículos, assim como lançadores de foguetes e lançadores de granadas.

O sistema é capaz de entender generalização dos objetos. Isto o permite detectar objetos que remetem propositalmente a armas de fogo, como armas de energia utilizadas em obras de ficção científica e funcionar em trabalhos de arte, mesmo tendo sido treinada com imagens reais. Estes resultados mostram que o sistema é capaz de lidar com novos modelos e tecnologias sendo introduzidos pela indústria de defesa e lidar com formas diferentes de capitação de imagens.

A aplicação também é capaz de distinguir objetos com forma de 'L', como furadeiras e secadores de cabelo, comumente comparados à armas de fogo. Isso ocorre mesmo quando são empunhados de forma a fazer referência ao porte de armas.

Os resultados obtidos durante os testes de classificação das imagens mostraram-se promissores. Das 669 imagens de testes, 644 foram corretamente classificadas, sendo 427 verdadeiros positivos e 217 verdadeiros negativos. Das imagens classificadas incorretamente, houve seis falsos positivos e 19 falsos negativos. Os valores obtidos foram $95,73\%$ de sensibilidade, $97,30\%$ de especificidade e $96,26\%$ de acurácia. A curva ROC traçada a partir desses resultado pode ser vista na figura 29.

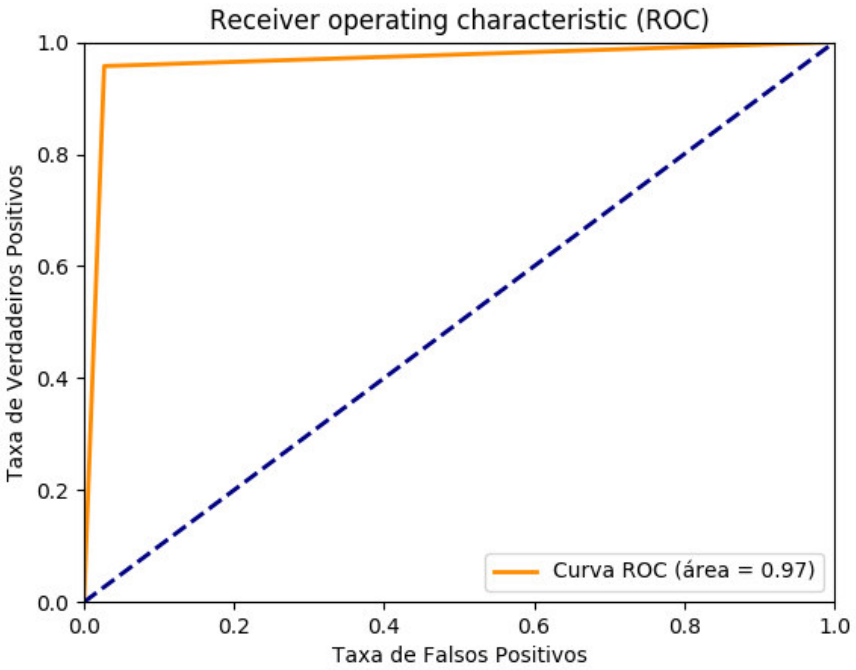
Tabela 2 – Matriz de Confusão

Previsto /Real	P	N
P'	427	6
N'	19	217

O valor de mAP foi de $70,72\%$, comparável a aplicações de estado de arte para detecção de objetos para uma única classe, este resultado pode ser visto na figura 30. Pode-se ressaltar que em alguns casos, o erro ocorreu por diferenças de IoU entre a *bounding box* e o *ground truth*, porém como pode ser visto nas imagens abaixo, o sistema é capaz de realizar as detecções úteis mesmo nesta situação. Outros casos considerados erros foram casos em que ocorreu a detecção de objetos não marcados previamente na base, porém sendo armas, como lançadores de foguetes e armamento montado em veículos. Estes resultados, embora corretos, reduzem o mAP dentro da métrica utilizada.

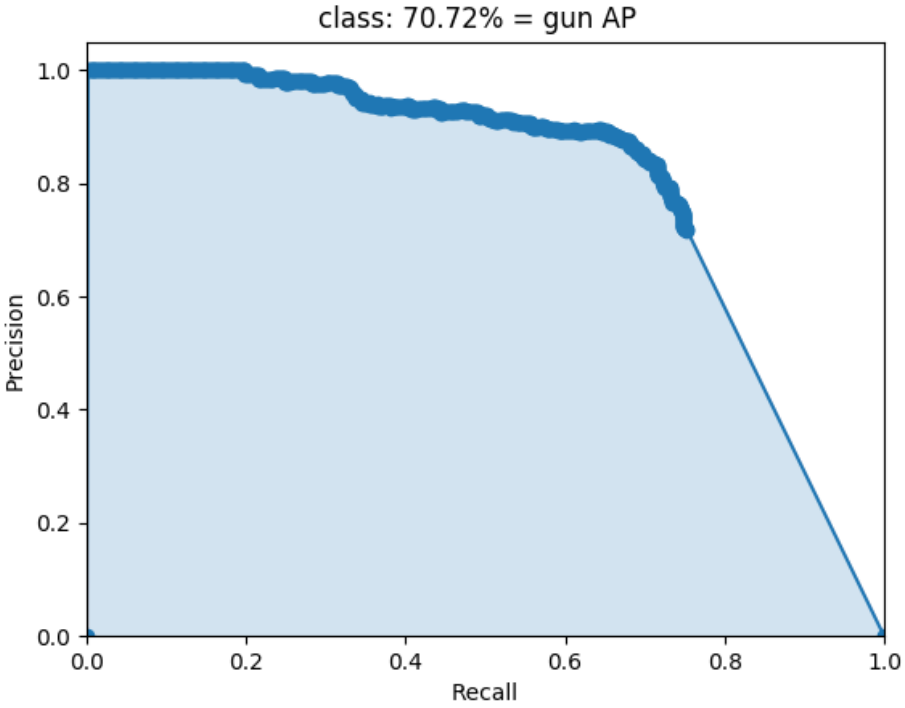
As falhas de detecção ocorreram em situações onde há objetos metálicos ou foscos, geralmente próximos de mãos, principalmente se estes objetos forem alongados. As falhas também ocorreram com objetos em forma de 'L', metálicos, polidos ou de cor escura, quando em contraste com fundo claro. Entre os objetos que geram falsos positivos, estão corrimãos metálicos quando próximos ou cobertos por mãos, postes e estruturas metálicas contendo protusões.

Figura 29 – Curva ROC.



Fonte – Autor

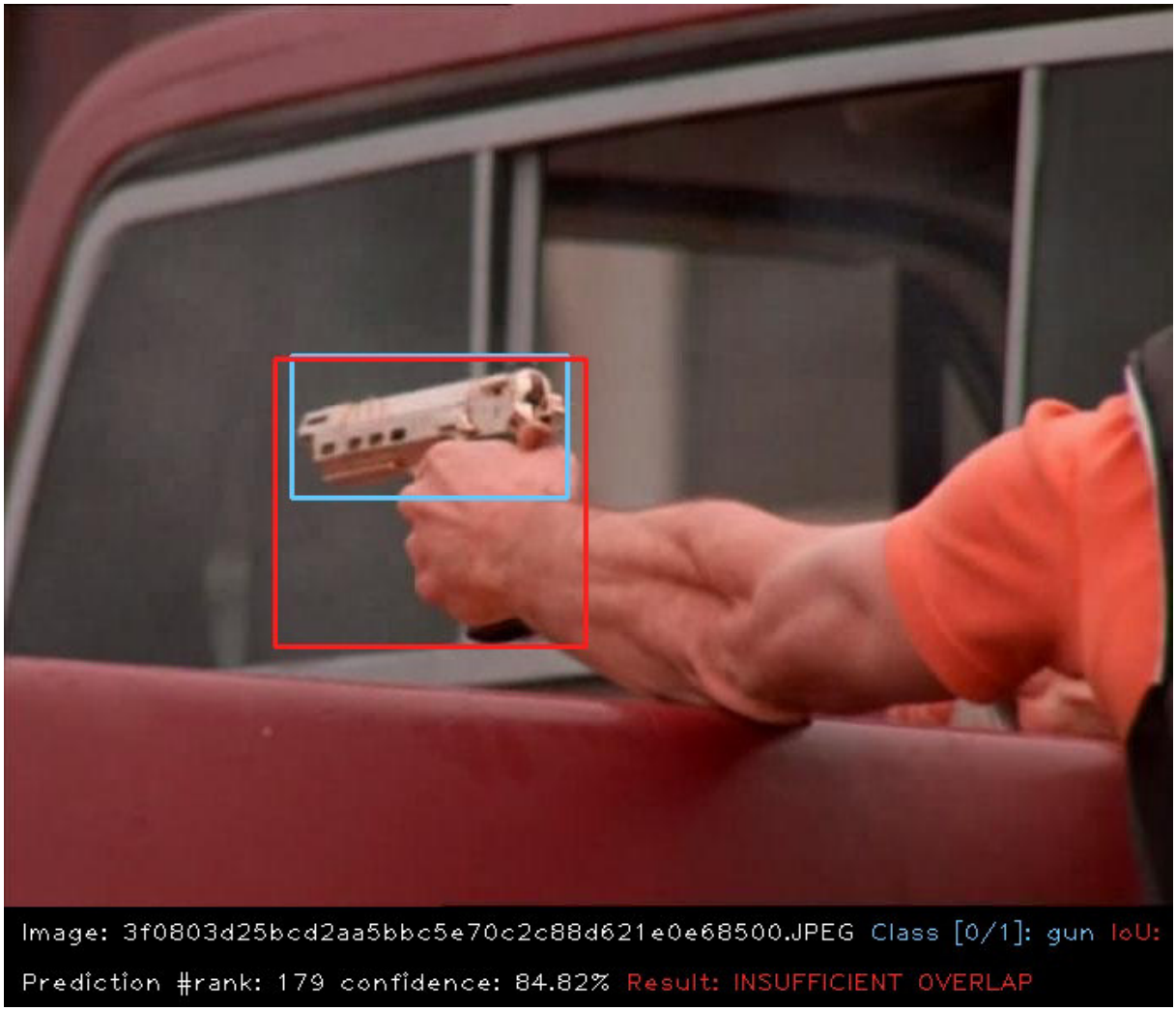
Figura 30 – Precisão e revocação.



Fonte – Autor

Nas imagens abaixo, alguns desses casos são demonstrados. Na primeira imagem há o exemplo de uma detecção válida, porém, com IoU inferior a 50%. A região predita pelo sistema é indicada pela caixa em vermelho e a região de *ground truth* a caixa em azul. Essas imagens tratam apenas de analisar detecções individuais, desta forma as apenas uma das regiões detectadas está destacada em cada imagem.

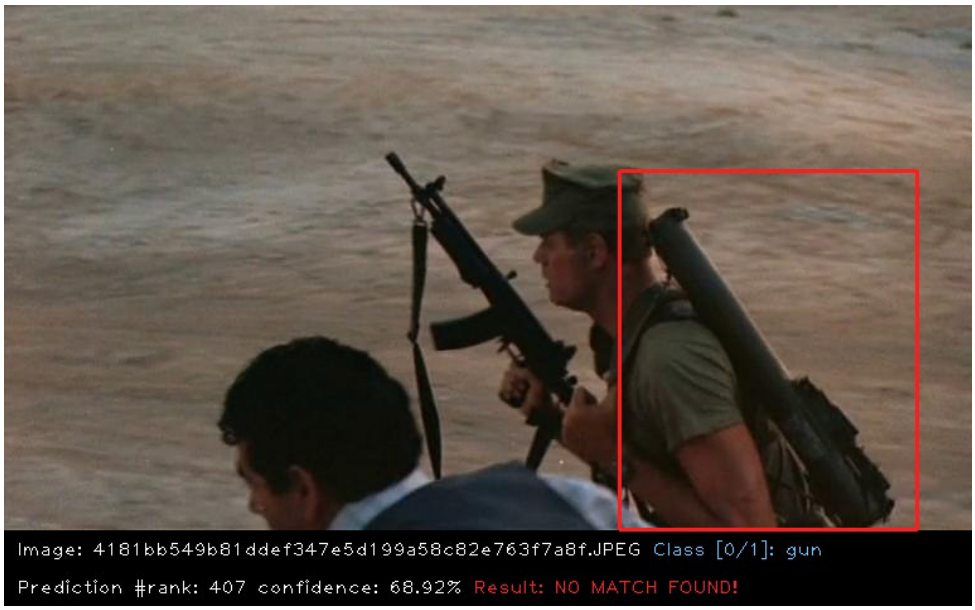
Figura 31 – Demonstração de erro em mAP, porém com detecção correta e válida.



Fonte – Autor

A figura 31 demostra um exemplo de detecção em que houve sobreposição insuficiente entre o *ground truth* (azul) e o valor predito (vermelho), mesmo assim esta detecção seria uma detecção válida em uma aplicação real. As figuras 32 e 33 demonstram casos de objetos não marcados por não serem classificados como *small arms*, que foram detectados pelo sistema.

Figura 32 – Demonstração de detecção de objeto não marcado, porém com detecção correta e válida.



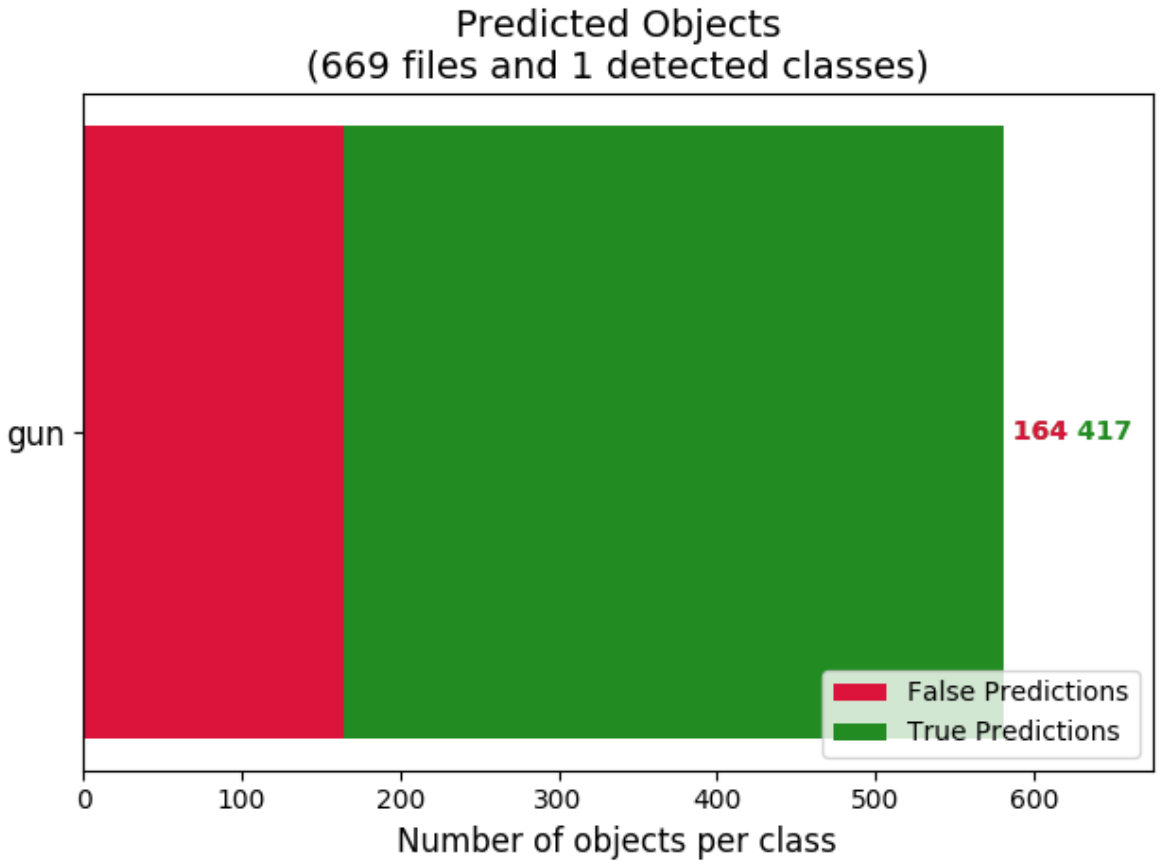
Fonte – Autor

Figura 33 – Demonstração de detecção de objeto não marcado, porém com detecção correta e válida.

Fonte – Autor

O resultado final da detecção é mostrado pela figura 34. Dos 554 objetos marcados 417 foram identificados corretamente. Dentre os 164 erros estão falsos positivos e falsos negativos, detecções com IoU ou confiança abaixo de 50% e detecções de objetos não marcados durante o preparo da base.

Figura 34 – Gráfico de detecção de objetos.



Fonte – Autor

4.3 Demonstração de Resultados

As imagens abaixo contém demonstrações dos resultados obtidos com a ferramenta implementada. A figura 35 mostra uma parede com múltiplas réplicas de armas de fogo. Dos 17 objetos presentes, nove foram corretamente detectados. As falhas ocorrem por limitações da técnica com objetos próximos.

Na figura 36, há dois secadores de cabelo fazendo alusão a armas de fogo e uma arma sob oclusão parcial. Este teste demonstra a capacidade do sistema de identificar objetos com forma remetente a armas de fogo. Além de mostrar robustez da técnica a oclusão parcial.

Figura 35 – Demonstração dos resultados de casos positivos.

Fonte – Autor

Figura 36 – Demonstração dos resultados de casos positivos.

Fonte – Autor

Na figura 37, demonstra-se a detecção de um revolver quase que totalmente ocluso. Apenas parte do cano, cabo e guarda do gatilho são visíveis. Esta imagem mostra a capacidade do sistema de identificar objetos quase que totalmente oclusos.

A figura 38 mostra uma arma ficcional da série *Star Trek: Discovery*. Mesmo não sendo uma arma real, essa traz a ideia de arma de fogo. Este resultado mostra a capacidade do sistema de reconhecer armas independente do modelo, mesmo quando estas são objetos de ficção.

Figura 37 – Demonstração dos resultados de casos positivos.

Fonte – Autor

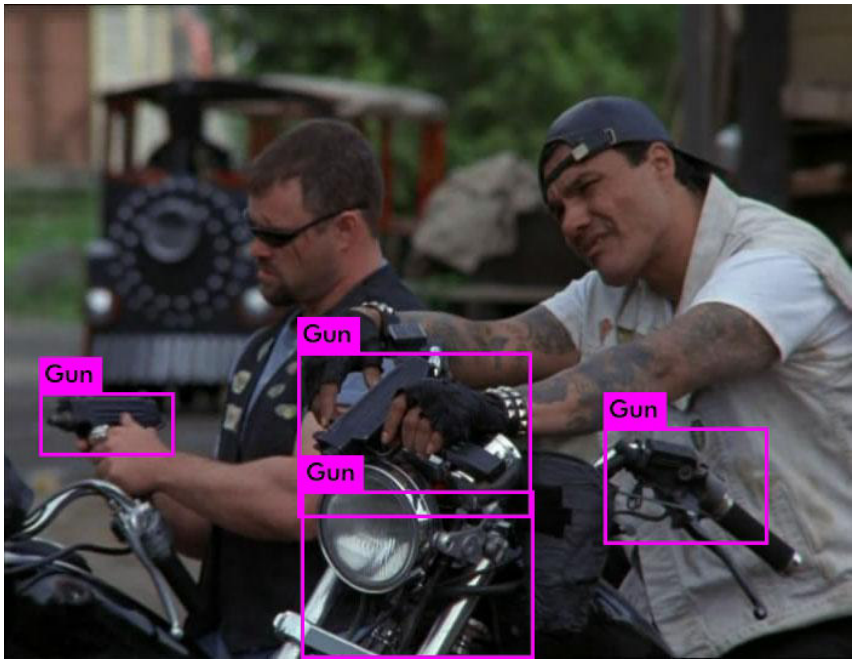
Figura 38 – Demonstração dos resultados de casos positivos.

Fonte – Autor

As imagens a seguir demonstram casos de erro. Na figura 39, pode-se perceber que as duas armas foram detectadas corretamente, porém, parte do guidão e da suspensão da motocicleta foram incorretamente classificados como arma de fogo. Este é um caso de erro comum, onde há um objeto metálico e outro com forma de 'L', ambos próximos a mãos.

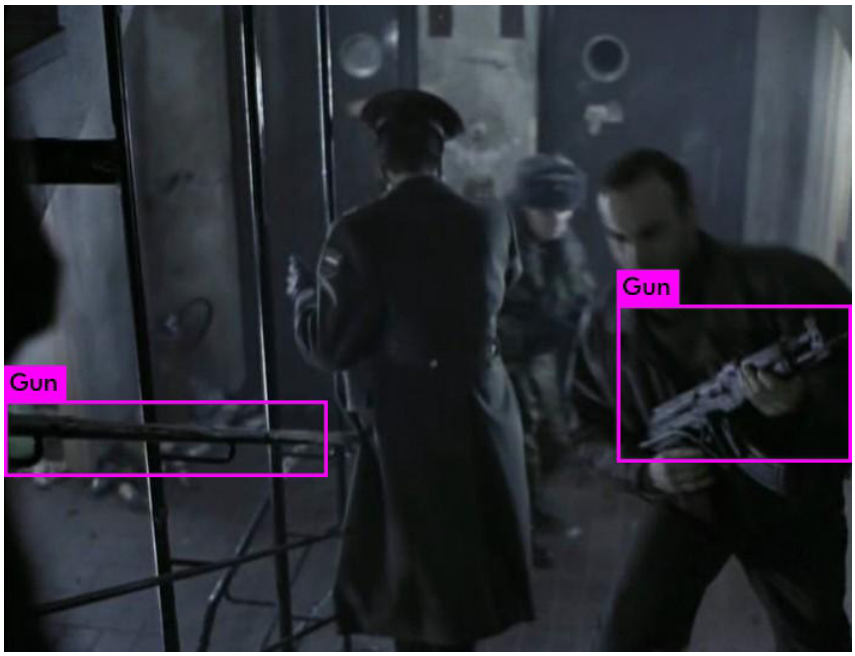
A figura 40 mostra que o fuzil foi corretamente detectado, contudo o corrimão ao lado também foi identificado. Mais uma vez a detecção ocorreu com um objeto metálico alongado.

Figura 39 – Demonstração dos resultados de casos erro.



Fonte – Autor

Figura 40 – Demonstração dos resultados de casos erro.



Fonte – Autor

Embora alguns casos de erro tenham ocorrido, esses são mínimos e não inviabilizam o sistema, tendo em vista que as taxas de sensibilidade, especificidade e acurácia, atingiram respectivamente $95,73\%$, $97,30\%$, $96,26\%$, evidenciando a capacidade do sistema de identificar imagens contendo arma de fogo com grande eficiência, assim como, a taxa de mAP, de 70%, mostra a eficiência do sistema em localizar os objetos.

5 CONCLUSÃO

Este trabalho demonstrou a viabilidade da utilização de CNNs para criação de um detector genérico de armas de fogo. O método utilizado mostrou-se robusto, sendo capaz de detectar corretamente armas que não foram apresentadas durante o treinamento, sendo estas de qualquer modelo e em grande variedade de ambientes. A base de dados construída para este trabalho mostrou-se suficientemente variada para permitir que o sistema seja capaz de entender o conceito de arma de fogo.

Os resultados obtidos, $95,73\%$ de sensibilidade, $97,30\%$ de especificidade, $96,26\%$ de acurácia e 70% de mAP, mostram a eficiência da técnica. A ferramenta utilizada destaca-se por sua velocidade de detecção, podendo ser executada em tempo real. Outra contribuição é a aplicação pela primeira vez desta ferramenta em detecção de arma de fogo.

Apesar das limitações do sistema, ao serem estudados os casos de erros, pôde-se perceber que estes são fruto de uma base de dados relativamente homogênea em termos de qualidade de imagem. Esta situação pode ser contornada com a inclusão de imagens de qualidade menor ou adquiridas com equipamento para o uso em aplicação real.

Violência com arma de fogo é um sério problema de segurança pública. As ferramentas computacionais desenvolvidas neste trabalho poderão contribuir para aumentar a possibilidade de detecção precoce de situações potencialmente perigosas, aumentando a eficiência dos agentes de segurança pública e auxiliando no combate a violência.

5.1 Trabalhos Futuros

A partir dos resultados obtidos com esta técnica, abre-se a possibilidade para a implementação deste sistema, com aplicação dessa mesma técnica para o fluxo de entrada de vídeo gerada por câmeras de segurança, permitindo uso em ambientes reais.

De forma similar, pode-se usar a mesma abordagem em redes sem fio e de dados móveis para a proteção de ambientes abertos através de *drones*. O sistema deve ser testado para analisar a eficiência da comunicação, verificando a capacidade de transmissão de vídeo de alta qualidade e da latência. Pode-se, ainda, realizar a adaptação do sistema para funcionamento em *hardware* embarcado, visando criar dispositivos que operem de acordo com o conceito de *SmartCameras*, reduzindo o envio de dados.

Planeja-se também aplicar o detector em estado atual no resto da base para marcação automatizada desta, aumentando seu tamanho, além de incluir imagens de outras fontes como câmeras de segurança e outros métodos de captura como infravermelho,

intensificadores de imagens para visão noturna e raio-x.

REFERÊNCIAS

ALPAYDIN, E. *Introduction to machine learning. sl*. [S.l.]: The MIT Press, 2010. Citado 2 vezes nas páginas 18 e 21.

ARCHIVE, G. V. *Mass Shootings*. 2017. <http://www.shootingtracker.com/>. (Acessado em 06/03/2018). Citado na página 14.

ARDIZZONE, E. et al. Combining top-down and bottom-up visual saliency for firearms localization. In: IEEE. *Signal Processing and Multimedia Applications (SIGMAP), 2014 International Conference on*. [S.l.], 2014. p. 25–32. Citado na página 16.

BALLARD, D. H.; BROWN, C. M. Computer vision, article, 4 pages prentice-hall. *Englewood Cliffs, New Jersey, believed to be published more than one year prior to the filing date of the present application*, 1982. Citado na página 32.

BARAN, R.; GLOWACZ, A.; MATIOLANSKI, A. The efficient real-and non-real-time make and model recognition of cars. *Multimedia Tools and Applications*, Springer, v. 74, n. 12, p. 4269–4288, 2015. Citado na página 14.

BARRETT, D. One surveillance camera for every 11 people in britain, says cctv survey. *The Telegraph*, v. 10, 2013. Citado na página 14.

BARRETT, D. *One surveillance camera for every 11 people in Britain, says CCTV survey - Telegraph*. 2013. <https://www.telegraph.co.uk/technology/10172298/One-surveillance-camera-for-every-11-people-in-Britain-says-CCTV-survey.html>. (Acessado em 06/03/2018). Citado na página 14.

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. Citado na página 27.

BENNETT, J. et al. The netflix prize. In: NEW YORK, NY, USA. *Proceedings of KDD cup and workshop*. [S.l.], 2007. v. 2007, p. 35. Citado na página 19.

BERTOZZI, N. *Advisors: Susan Jarvis*. Tese (Doutorado) — Worcester Polytechnic Institute, 2017. Citado na página 16.

BLACK; DECKER. *Black and Decker Hammer Drill KR504RE - 500W | Lazada*. 2018. <https://www.lazada.com.my/products/black-and-decker-hammer-drill-kr504re-500w-i217770278-s275833084.html>. (Acessado em 06/04/2018). Citado na página 39.

BOSH. *Furadeira de Impacto Bosh Profissional de 550W - BOSCH-GSB550RE - R$ 239.99 na LojadoMecanico*. 2018. <https://www.lojadomecanico.com.br/produto/92104/21/221/200/Furadeira-de-Impacto-Bosh-Profissional-de-550W-110V?utm_source=zoom&utm_campaign=xmlzoom&utm_medium=comparadores&utm_content=92104>. (Acessado em 06/04/2018). Citado na página 39.

BOURJAILY, P. *Bourjaily: The Internet Movie Firearms Database | Field & Stream*. 2009. <https://www.fieldandstream.com/blogs/gun-nut/2009/04/bourjaily-internet-movie-firearms-database>. (Accessado em 04/05/2018). Citado na página 37.

BOUZY, B.; CAZENAVE, T. Computer go: an ai oriented survey. *Artificial Intelligence*, Elsevier, v. 132, n. 1, p. 39–103, 2001. Citado na página 28.

BRAMBERGER, M. et al. A smart camera for traffic surveillance. In: *Proceedings of the first workshop on Intelligent Solutions in Embedded Systems*. [S.l.: s.n.], 2003. p. 153–164. Citado na página 14.

BRAUER, J. *The US Firearms Industry*. [S.l.]: Graduate Institute of International and Development Studies, Geneva. http://www. smallarmssurvey. org/fileadmin/docs/F-Working-papers/SAS-WP14-US-Firearms-Industry. pdf Accessed, 2013. Citado na página 38.

BUNNI. *IMFDb - Internet Movie Firearms Database - Guns in Movies, TV and Video Games*. 2007. <http://www.imfdb.org/wiki/Main_Page>. (Acessado em 04/05/2018). Citado na página 37.

CHAPELLE, O.; SCHÖLKOPF, B.; ZIEN, A. *Semi-Supervised Learning. Adaptive Computation and Machine Learning series*. [S.l.]: MIT Press, Cambridge, 2006. Citado na página 19.

COLLOBERT, R.; WESTON, J. A unified architecture for natural language processing: Deep neural networks with multitask learning. In: ACM. *Proceedings of the 25th international conference on Machine learning*. [S.l.], 2008. p. 160–167. Citado na página 19.

COMMONS, W. *File:ROC curve.svg — Wikimedia Commons, the free media repository*. 2016. [Online; acessado 3-Junho-2018]. Disponível em: <https://commons.wikimedia.org/w/index.php?title=File:ROC_curve.svg&oldid=212240481>. Citado na página 46.

CYBENKO, G. Approximation by superpositions of a sigmoidal function. *Mathematics of control, signals and systems*, Springer, v. 2, n. 4, p. 303–314, 1989. Citado na página 23.

DE ALMEIDA NETO, A. *Aplicações de Múltiplas Redes Neurais em Sistemas Mecatrônicos*. Tese (Doutorado) — Instituto Tecnológico de Aeronáutica, 2003. Citado na página 18.

DECHTER, R. *Learning while searching in constraint-satisfaction problems*. [S.l.]: University of California, Computer Science Department, Cognitive Systems Laboratory, 1986. Citado na página 27.

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. Citado na página 27.

DENNIS, K. *Online » Top Tech Apps that Can Help Streamline Your Business*. 2018. <https://www.nace.net/blog_home.asp?display=56>. (Acessado em 06/04/2018). Citado na página 39.

DESHPANDE, A. *A Beginner's Guide To Understanding Convolutional Neural Networks – Adit Deshpande – CS Undergrad at UCLA ('19)*. 2016. <https://adeshpande3.github.io/A-Beginner%27s-Guide-To-Understanding-Convolutional-Neural-Networks>. (Acessado em 04/12/2018). Citado 2 vezes nas páginas 28 e 30.

DETTMERS, T. *Understanding Convolution in Deep Learning - Tim Dettmers*. 2015. <http://timdettmers.com/2015/03/26/convolution-deep-learning/>. (Acessado em 10/07/2018). Citado na página 29.

DICKSON, B. Exploiting machine learning in cybersecurity. *TechCrunch. Retrieved*, p. 05–23, 2017. Citado na página 19.

EVERINGHAM, M. et al. The pascal visual object classes challenge: A retrospective. *International Journal of Computer Vision*, v. 111, n. 1, p. 98–136, jan. 2015. Citado 3 vezes nas páginas 42, 48 e 49.

EVERINGHAM, M. et al. *The pascal visual object classes challenge 2007 (voc 2007) results (2007)*. 2008. Citado na página 45.

EVERINGHAM, M. et al. The pascal visual object classes (voc) challenge. *International journal of computer vision*, Springer, v. 88, n. 2, p. 303–338, 2010. Citado 2 vezes nas páginas 47 e 48.

FAWCETT, T. An introduction to roc analysis. *Pattern recognition letters*, Elsevier, v. 27, n. 8, p. 861–874, 2006. Citado 2 vezes nas páginas 45 e 46.

FISHER, R. A. The use of multiple measurements in taxonomic problems. *Annals of eugenics*, Wiley Online Library, v. 7, n. 2, p. 179–188, 1936. Citado na página 21.

FRIEDMAN, J.; HASTIE, T.; TIBSHIRANI, R. *The elements of statistical learning*. [S.l.]: Springer series in statistics New York, 2001. v. 1. Citado na página 23.

GEITGEY, A. *Machine Learning is Fun! Part 4: Modern Face Recognition with Deep Learning*. 2016. <https://medium.com/@ageitgey/machine-learning-is-fun-part-4-modern-face-recognition-with-deep-learning-c3cffc121d78>. (Acessado em on 18/04/2018). Citado na página 33.

GIRSHICK, R. Fast r-cnn. *arXiv preprint arXiv:1504.08083*, 2015. Citado na página 34.

GIRSHICK, R. et al. Rich feature hierarchies for accurate object detection and semantic segmentation. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. [S.l.: s.n.], 2014. p. 580–587. Citado na página 34.

GLOBO, G.; HOJE, J. *G1 > Brasil - NOTÍCIAS - Câmeras ajudam a reduzir índice de criminalidade*. 2009. <http://g1.globo.com/Noticias/Brasil/0,,MUL1023939-5598,00-CAMERAS+AJUDAM+A+REDUZIR+INDICE+DE+CRIMINALIDADE.html>. (Acessado em 06/22/2018). Citado na página 14.

GREGA, M. et al. Automated detection of firearms and knives in a cctv image. *Sensors*, Multidisciplinary Digital Publishing Institute, v. 16, n. 1, p. 47, 2016. Citado na página 16.

GUZELLA, T. S.; CAMINHAS, W. M. A review of machine learning approaches to spam filtering. *Expert Systems with Applications*, Elsevier, v. 36, n. 7, p. 10206–10222, 2009. Citado na página 19.

HAYNES, R. *50 worst suburbs for speed in NSW | Daily Telegraph*. 2011. <https://www.dailytelegraph.com.au/worst-suburbs-for-speed-in-nsw/news-story/202e84b5b923c9937ad069498389b73a?sv=c36675bebca23e77b1fefb1fd80e50e>. (Acessado em 06/04/2018). Citado na página 39.

HE, K. et al. Mask r-cnn. In: IEEE. *Computer Vision (ICCV), 2017 IEEE International Conference on*. [S.l.], 2017. p. 2980–2988. Citado na página 34.

HINTON, G. et al. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine*, IEEE, v. 29, n. 6, p. 82–97, 2012. Citado na página 19.

HOSCH, W. L. *Machine learning*. Encyclopædia Britannica, inc., 2016. Disponível em: <https://www.britannica.com/technology/machine-learning>. Citado na página 18.

HUBEL, D. H.; WIESEL, T. N. Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *The Journal of physiology*, Wiley Online Library, v. 160, n. 1, p. 106–154, 1962. Citado na página 28.

JALAL, M. *How to understand Haar-like feature for face detection - Quora*. 2017. <https://www.quora.com/How-can-I-understand-Haar-like-feature-for-face-detection>. (Acessado em 17/04/2018). Citado na página 33.

JR, D. H.; RUBINFELD, D. L. Hedonic housing prices and the demand for clean air. *Journal of environmental economics and management*, Elsevier, v. 5, n. 1, p. 81–102, 1978. Citado na página 20.

KANG, N. *Multi-Layer Neural Networks with Sigmoid Function— Deep Learning for Rookies*. 2017. <https://towardsdatascience.com/multi-layer-neural-networks-with-sigmoid-function-deep-learning-for-rookies-2-bf464f09eb7f>. (Acessado em 11/04/2018). Citado na página 23.

KARPATHY, A. Cs231n convolutional neural networks for visual recognition. *Neural networks*, v. 1, 2016. Citado na página 31.

KAYASTHA, R. Preventing mass shooting through cooperation of mental health services, campus security, and institutional technology. 2016. Citado na página 14.

KOROBOV, M. *Scrapy, a fast high-level web crawling & scraping framework for Python*. [S.l.]: GitHub, 2015. <https://github.com/scrapy/scrapy>. Commit = 5aebdac45d0563a17b7cd0d0da0078585d950605, (Acessado em 26/06/2017). Citado na página 38.

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. Citado 2 vezes nas páginas 27 e 31.

LECUN, Y. et al. Backpropagation applied to handwritten zip code recognition. *Neural computation*, MIT Press, v. 1, n. 4, p. 541–551, 1989. Citado na página 30.

LEE, H. et al. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In: ACM. *Proceedings of the 26th annual international conference on machine learning*. [S.l.], 2009. p. 609–616. Citado na página 29.

LLOYD, S. Least squares quantization in pcm. *IEEE transactions on information theory*, IEEE, v. 28, n. 2, p. 129–137, 1982. Citado na página 42.

MAJEK, K. *4K YOLO COCO Object Detection #2*. 2017. <https://www.youtube.com/watch?v=EhcpGpFHCrw>. (Acessado em 18/06/2018). Citado na página 35.

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. Citado na página 22.

MITCHELL, T. M. Machine learning (mcgraw-hill international editions computer science series). McGraw-Hill, 1997. Citado na página 18.

MORGAN, S. *Python Tutorial: Neural Networks with backpropagation for XOR using one hidden layer*. 2016. <http://www.bogotobogo.com/python/python_Neural_Networks_Backpropagation_for_XOR_using_one_hidden_layer.php>. (Acessado em 10/04/2018). Citado 2 vezes nas páginas 22 e 25.

PEDREGOSA, F. et al. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, v. 12, p. 2825–2830, 2011. Citado 2 vezes nas páginas 20 e 21.

QIU, p. S. *BBox-Label-Tool*. [S.l.]: GitHub, 2016. <https://github.com/puzzledqs/BBox-Label-Tool>. Commit = eb88f25abceb75b3077002f87953788319705ebf, (Acessado em 26/06/2017). Citado na página 40.

RAVAL, S. *YOLO Object Detection·*. 2017. <https://github.com/llSourcell/YOLO_Object_Detection/blob/master/YOLO%20Object%20Detection.ipynb>. (Acessado em 19/04/2018). Citado 3 vezes nas páginas 32, 34 e 36.

REDMON, J. *Darknet: Open Source Neural Networks in C*. 2013–2016. <http://pjreddie.com/darknet/>. Citado na página 35.

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. Citado na página 36.

REDMON, J.; FARHADI, A. Yolo9000: Better, faster, stronger. *arXiv preprint arXiv:1612.08242*, 2016. Citado 4 vezes nas páginas 34, 35, 36 e 42.

REDMON, J.; FARHADI, A. Yolov3: An incremental improvement. *arXiv*, 2018. Citado na página 36.

REN, S. et al. Faster r-cnn: Towards real-time object detection with region proposal networks. In: *Advances in neural information processing systems*. [S.l.: s.n.], 2015. p. 91–99. Citado na página 34.

ROSEBROCK, A. *Intersection over Union (IoU) for object detection - PyImageSearch*. 2016. <https://www.pyimagesearch.com/2016/11/07/intersection-over-union-iou-for-object-detection/>. (Acessado em 05/09/2018). Citado 2 vezes nas páginas 47 e 48.

ROSENBLATT, F. *The perceptron, a perceiving and recognizing automaton Project Para*. [S.l.]: Cornell Aeronautical Laboratory, 1957. Citado na página 22.

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. Citado 2 vezes nas páginas 25 e 26.

RUSSAKOVSKY, O. et al. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, v. 115, n. 3, p. 211–252, 2015. Citado na página 27.

RUSSELL, S. J.; NORVIG, P. *Artificial intelligence: a modern approach*. [S.l.]: Malaysia; Pearson Education Limited,, 2016. Citado na página 18.

SAMUEL, A. L. Some studies in machine learning using the game of checkers. *IBM Journal of research and development*, IBM, v. 3, n. 3, p. 210–229, 1959. Citado na página 18.

SCHMIDHUBER, J. Deep learning in neural networks: An overview. *Neural Networks*, v. 61, p. 85–117, 2015. Published online 2014; based on TR arXiv:1404.7828 [cs.NE]. Citado na página 18.

SEBASTIANI, F. Machine learning in automated text categorization. *ACM computing surveys (CSUR)*, ACM, v. 34, n. 1, p. 1–47, 2002. Citado na página 19.

SEIF, G. *I'll tell you why Deep Learning is so popular and in demand*. 2018. <https://medium.com/swlh/ill-tell-you-why-deep-learning-is-so-popular-and-in-demand-5aca72628780>. (Acessado em 04/13/2018). Citado na página 27.

SILVER, D. et al. Mastering the game of go with deep neural networks and tree search. *nature*, Nature Research, v. 529, n. 7587, p. 484–489, 2016. Citado na página 28.

STEINKRAUS, D.; BUCK, I.; SIMARD, P. Using gpus for machine learning algorithms. In: IEEE. *Document Analysis and Recognition, 2005. Proceedings. Eighth International Conference on*. [S.l.], 2005. p. 1115–1120. Citado na página 18.

SURVEY, G. S. A.; (GENEVA, S. Graduate Institute of I. S. *Small Arms Survey 2007: Guns and the City*. Cambridge University Press, 2007. (Small Arms Survey). ISBN 9780521880398. Disponível em: <https://books.google.com.br/books?id=Lmy9e_LGwdoC>. Citado na página 40.

THATI, A. et al. Breath acetone-based non-invasive detection of blood glucose levels. *International Journal on Smart Sensing & Intelligent Systems*, v. 8, n. 2, 2015. Citado na página 24.

UN. SECRETARY-GENERAL et al. *GENERAL AND COMPLETE DISARMAMENT: SMALL ARMS*. [S.l.]: United Nations, 1997. <https://www.un.org/Depts/ddar/Firstcom/SGreport52/a52298.html>. (Acessado em 14/05/2018). Citado 2 vezes nas páginas 38 e 40.

UNHCR, O. o. t. U. N. H. C. f. R. *Global Trends: Forced Displacement in 2016*. [S.l.]: United Nations High Commissioner for Refugees, 2017. Citado na página 15.

VASA, H. *Google Images Download*. [S.l.]: GitHub, 2018. <https://github.com/hardikvasa/google-images-download>. Commit = eccc6acc4c644cb5fb3c5a1add95f5d7aac27bad, (Acessado em 21/04/2018). Citado na página 39.

VERMA, G. K.; DHILLON, A. A handheld gun detection using faster r-cnn deep learning. In: ACM. *Proceedings of the 7th International Conference on Computer and Communication Technology*. [S.l.], 2017. p. 84–88. Citado na página 16.

VIOLA, P.; JONES, M. Rapid object detection using a boosted cascade of simple features. In: IEEE. *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*. [S.l.], 2001. v. 1, p. I–I. Citado na página 32.

WAHID, A. *Big data and machine learning for Businesses*. 2017. <https://www.slideshare.net/awahid/big-data-and-machine-learning-for-businesses>. (Acessado em 06/04/2018). Citado na página 19.

WAISELFISZ, J. J. Mapa da violência 2016: homicídios por armas de fogo. Secretaria Nacional de Juventude, 2016. Citado na página 14.

WASSERMAN, P. D.; SCHWARTZ, T. Neural networks. ii. what are they and why is everybody so interested in them now? *IEEE expert*, IEEE, v. 3, n. 1, p. 10–15, 1988. Citado na página 23.

YUAN, M. J. et al. Watson and healthcare, how natural language processing and semantic search could revolutionize clinical decision support. *IBM developerWorks, IBM Corporation*, 2011. Citado na página 19.

ZHOU, J. *(5) Genre classification on movie posters | LinkedIn*. 2017. <https://www.linkedin.com/pulse/genre-classification-movie-posters-jianda-zhou/>. (Acessado em 10/07/2018). Citado na página 30.

ZHU, M. Recall, precision and average precision. *Department of Statistics and Actuarial Science, University of Waterloo, Waterloo*, v. 2, p. 30, 2004. Citado na página 49.