Jordy Aparecido Faria de Araújo

Detecção de objetos com Redes Neurais Embarcadas

Brasil

2018

sistema de bibliotecas ufg

PRPG
PRÓ-REITORIA DE PÓS-GRADUAÇÃO

UFG

TERMO DE CIÊNCIA E DE AUTORIZAÇÃO PARA DISPONIBILIZAR VERSÕES ELETRÔNICAS DE TESES E DISSERTAÇÕES NA BIBLIOTECA DIGITAL DA UFG

Na qualidade de titular dos direitos de autor, autorizo a Universidade Federal de Goiás (UFG) a disponibilizar, gratuitamente, por meio da Biblioteca Digital de Teses e Dissertações (BDTD/UFG), regulamentada pela Resolução CEPEC nº 832/2007, sem ressarcimento dos direitos autorais, de acordo com a Lei nº 9610/98, o documento conforme permissões assinaladas abaixo, para fins de leitura, impressão e/ou *download*, a título de divulgação da produção científica brasileira, a partir desta data.

1. Identificação do material bibliográfico: **[ ] Dissertação** **[ ] Tese**

2. Identificação da Tese ou Dissertação:

Nome completo do autor: Jordy Aparecido Faria de Araújo

Título do trabalho: Detecção de Objeto com deep learning embarcada

3. Informações de acesso ao documento:

Concorda com a liberação total do documento [X] SIM [ ] NÃO[1]

Havendo concordância com a disponibilização eletrônica, torna-se imprescindível o envio do(s) arquivo(s) em formato digital PDF da tese ou dissertação.

Jordy A. Faria de Araújo
Assinatura do(a) autor(a)[2]

Ciente e de acordo:

Andrea S Soares
Assinatura do(a) orientador(a)[2]

Data: 30 / 11 / 2018

[1] Neste caso o documento será embargado por até um ano a partir da data de defesa. A extensão deste prazo suscita justificativa junto à coordenação do curso. Os dados do documento não serão disponibilizados durante o período de embargo.
Casos de embargo:
- Solicitação de registro de patente;
- Submissão de artigo em revista científica;
- Publicação como capítulo de livro;
- Publicação da dissertação/tese em livro.

[2] A assinatura deve ser escaneada.

Versão atualizada em setembro de 2017.

Jordy Aparecido Faria de Araújo

Detecção de objetos com Redes Neurais Embarcadas

Trabalho de Conclusão de Curso da Universidade Federal de Goiás na área de Deep Learning. Visando fazer a visão computacional para detecção de objetos usando Redes Neurais Embarcadas.

Universidade Federal de Goiás

EMC

Engenharia de Computação

Orientador: Anderson da Silva Soares

Brasil

2018

Ficha de identificação da obra elaborada pelo autor, através do
Programa de Geração Automática do Sistema de Bibliotecas da UFG.

Aparecido Faria de Araújo, Jordy
Detecção de objetos com redes neurais embarcadas [manuscrito] / Jordy Aparecido Faria de Araújo. - 2018.
xxviii, 28 f.: il.

Orientador: Prof. Dr. Anderson Soares; co-orientador Thyago Marques; co-orientador Sandrerley Ramos.
Trabalho de Conclusão de Curso (Graduação) - Universidade Federal de Goiás, Escola de Engenharia Elétrica, Mecânica e de Computação (EMC), Engenharia de Computação, Goiânia, 2018.
Bibliografia.
Inclui siglas, tabelas, lista de figuras.

1. Deep Learning. 2. Detecção de objeto. 3. Visão computacional. 4. Redes neurais. 5. Embarcado. I. Soares, Anderson, orient. II. Título.

CDU 004

MINISTÉRIO DA EDUCAÇÃO
UNIVERSIDADE FEDERAL DE GOIÁS
ESCOLA DE ENGENHARIA ELÉTRICA, MECÂNICA E DE COMPUTAÇÃO
Coordenação de Estágio e Projeto Final

ATA DE AVALIAÇÃO DE PROJETO FINAL

Aos 30 dias do mês de NOVEMBRO do ano de 2018,
foi apresentado e defendido o Projeto Final intitulado VISÃO COMPUTACIONAL PARA DETECÇÃO DE OBJETOS USANDO REDES NEURAIS EMBARCADAS
perante a banca examinadora composta pelos membros:

1. ANDERSON DA SILVA SOARES, orientador e presidente;
2. Thyago Carvalho Marques; 3. Sandoval Ramos Pires.

Após a exposição do trabalho por parte do(s) autor(es), aluno(s) do curso de Engenharia Elétrica, foram lhe(s) atribuídas as seguintes notas pelos membros da banca:

Aluno(s)	Membro 1	Membro 2	Membro 3
JORDY APARECIDO FARIA DE ARAÚJO	8,9	9,0	9,1

Nada mais havendo a registrar, eu, ANDERSON DA SILVA SOARES, designado secretário "ad hoc" da banca examinadora, lavrei a presente Ata do ocorrido, a qual, lida e considerada conforme, vai assinada por mim e pelos membros da banca.
Goiânia, 30 de NOVEMBRO de 2018.

Anderson S. Soares
T. Marques
Sandoval Ramos P.

Agradecimentos

Agradeço primeiramente ao meu professor orientador Anderson Soares, que apesar de uma agenda apertada, aceitou me orientar para este TCC que me levou à um grande aprendizado na área de Deep Learning.

Agradeço também aos membros do Núcleo de Robótica Pequi Mecânico por me possibilitarem o contato com assuntos mais avançados na área de robótica e com isso pude escolher o tema deste trabalho. O que também me possibilitou ajudar a equipe Humanoid. E pela ajuda com a anotação de imagens do dataset usado para o treinamento da rede.

Agradeço em especial ao membro do Pequi Mecânico Vinícius Araújo Santos, pela ajuda em problemas específicos de codificação em detecção de objetos. Antes dessa ajuda, achava que eu não conseguiria terminar a monografia a tempo.

Agradeço a minha namorada Luana H. I. Solana por me ajudar emocionalmente. Não só durante o TCC, mas durante todo o curso. Sempre me incentivando a aguentar firme e seguir em frente. Me apoiando nos momentos difíceis, nas notas baixas e nas madrugadas viradas programando e/ou estudando.

E por último, e mais importante, agradeço muito aos meus pais, que apesar de uma situação financeira não ser das melhores, eles puderam me sustentar enquanto eu me dedicava exclusivamente à faculdade. Se eu consegui terminar foi graças ao apoio deles.

Resumo

O avanço das técnicas de redes neurais profundas permitiu a evolução de soluções em diversas áreas do conhecimento, principalmente em pesquisas e aplicações de visão computacional. Nesse contexto, este trabalho investiga a aplicação de redes neurais profundas para detecção de objetos. Em particular, o problema de detecção de bola para futebol de robôs humanoides foi explorado. Nesse tipo de aplicação, um dos desafios está no uso de arquiteturas e implementações robustas o suficiente para que sejam embarcadas em hardwares limitados computacionalmente. Os resultados obtidos demonstraram a viabilidade da implementação proposta com uma boa taxa de sucesso na detecção da bola.

Palavras-chaves: Deep Learning, Visão computacional, embarcado, futebol, humanoid.

Lista de ilustrações

Lista de abreviaturas e siglas

NN	Neural Network
CNN	Convolutional Neural Network
bbox	Bouding box
YOLO	You Only Look Once
Rasp	Raspberry Pi 3
FPS	Frames per second
CV	Computer Vision
LARC	Latin American Robots Competition
RNP	Region Proposals Network

Sumário

1 Introdução

A tentativa de extrair informações de imagens já é bem antiga, usando métodos clássicos de aprendizado de máquina como *Suport Vector Machine*, mas os resultados não eram tão bons e confiáveis para problemas mais complexos. Mas os estudos sofreram um grande avanço com Lecun ao conseguir usar Redes Neurais Convolucionais para classificar números escritos a mão.(LECUN et al., 1990)

Tal tecnologia possibilitou muitos avanços em várias áreas como medicina (para diagnósticos a partir de raio-x, ultrassom) barateando o serviço, acelerando o processo e até aumentando a precisão dos diagnósticos. Agricultura, para dizer a saúde da planta, se estava faltando água ou até se tinha alguma doença. Publicidade, podendo remover conteúdo impróprio (sexual, violento) de redes sociais. Segurança, para dizer se uma pessoa estava ou não sobre efeito de narcóticos pela dilatação da pupila. E até mesmo poder organizar as fotos pessoais a partir do conteúdo delas.

Mas apenas classificar não era suficiente para resolver alguns problemas. Era necessário saber onde exatamente estava os objetos em uma imagem, aumentando ainda mais as possibilidades. Com isso além de dizer se uma pessoa está ou não com câncer, mostraria a região em que se encontra o tumor. Mostraria a região da plantação que foi afetada por uma praga.

Também possibilitou o crescimento de outras áreas como a visão computacional para robótica, podendo detectar objetos enquanto o robô se movimenta. Em segurança, identificando a identidade das pessoas em um ambiente. Ou até mesmo para detectar onde estão os carros em uma rua para um sistema de carro autônomo (WU et al., 2017).

Todo ano há uma competição de classificação de imagens conhecida como ImageNet no qual em 2012 Krizhevsky e sua equipe não escreveram nenhum código de visão computacional clássico. Em vez disso, usando o aprendizado profundo, seu computador aprendeu a reconhecer imagens por si mesmo. Eles projetaram uma rede neural chamada AlexNet e a treinaram com um milhão de imagens que requeriam trilhões de operações matemáticas em GPUs. AlexNet, do Krizhevksy, havia batido o melhor software codificado por humanos. Após esse feito no ano seguinte, a grande maioria dos competidores migraram para soluções usando Redes Neurais Profundas (Deep Learning). Em 2015, o aprendizado profundo alcançou níveis de percepção "sobre-humanos".[1] Sendo o estado da arte até o presente momento sendo o Top-1 Accuracy 83.1%. [2]

Enquanto isso para problemas de detecção de objetos existe o dataset do COCO

[1] https://blogs.nvidia.com/blog/2016/01/12/accelerating-ai-artificial-intelligence-gpus/
[2] https://cv-tricks.com/convolutional-neural-networks/comparison-based-on-accuracy/

no qual uma RetinaNet conseguiu um *average precision* de 40.8% geral, com 24.1% para objetos pequenos, 44.2% para objetos de tamanho médio e 51.2% para objetos grandes. [3]

A proposta deste trabalho é treinar uma Rede Neural Convolucional Embarcada para Detecção de Objeto, mais especificamente, para a detecção da bola em um jogo de futebol entre robôs humanoides. Sendo esta uma das categorias do Campeonato Latino Americano de Robótica (LARC). [4] A rede deve ser Embarcada, ou seja, pequena o suficiente para consumir pouco processamento porque será usada em um Raspberry Pi 3 com um processador ARM Cortex-A53 Quad-Core de 1.4 GHz. E que apresente um processamento tal que tenha um FPS (frame por segundo) suficiente para rodar em tempo de jogo.

[3] https://towardsdatascience.com/yolo-v3-object-detection-53fb7d3bfe6b
[4] https://www.robocuphumanoid.org/

2 Motivação

2.1 Latin American Robots Competition

A LARC é a principal competição científica continental no campo da robótica autônoma na América Latina. O evento é composto pelo IEEE *Latin American Robots Competition*. Este evento reúne pesquisadores, estudantes de graduação e pós-graduação de várias Universidades e Institutos de Pesquisa da América Latina e do exterior. Há também competições para estudantes no ensino primário e secundário nas categorias RoboCup Junior.

As equipes normalmente serão compostas por alunos de graduação ou de pós-graduação e pesquisadores de diversas áreas, como Engenharia Elétrica, Engenharia Mecânica, Mecatrônica, Automação e Controle e Engenharia de Computação, Ciência da Computação e Sistemas de Informação entre outros. Os concorrentes têm a oportunidade de desenvolver soluções completas e validá-las de forma prática em áreas que envolvem: Controle, Inteligência Artificial, Sistemas Multiagentes, Sensoriamento, Interface Homem-Robô, Reconhecimento de Objetos e Pessoas, etc.[1]

Há vários desafios para os robôs autônomos que fomentam o desenvolvimento e a produção de novos conhecimentos na área de Robótica Autônoma Inteligente na América Latina. Entre as categorias temos futebol de robô, desafios de resgate, robôs domésticos de serviço, logística e simulação. [2]

2.2 RoboCup Humanoid League

Entre as modalidades presentes na LARC temos a *RoboCup Humanoid League*. O objetivo da iniciativa internacional, RoboCup, é desenvolver uma equipe de robôs humanoides que sejam capazes de vencer os Campeões Mundiais de Futebol até 2050. Em certo sentido, o desafio da RoboCup é o sucessor do desafio do xadrez para a inteligência artificial (um computador batendo o campeão de xadrez mundial) que foi resolvido em 1997, quando Deep Blue venceu contra Garry Kasparow. Atualmente, existem várias ligas de futebol da RoboCup que se concentram em diferentes aspectos deste desafio.

Na *Humanoid League*, robôs autônomos com um plano corporal humano e sentidos humanos jogam futebol um contra o outro. Ao contrário dos robôs humanoides fora da *Humanoid League*, a tarefa de percepção e modelagem mundial não é simplificada pelo

[1] http://www.cbrobotica.org/
[2] http://www.cbrobotica.org/?page_id=2

uso de sensores de alcance não humano. Além de competições de futebol, desafios técnicos acontecem. Andar, correr e chutar dinamicamente a bola, mantendo o equilíbrio; a percepção visual da bola, de outros jogadores e do campo; a auto localização e o jogo em equipe estão entre as muitas questões de pesquisa investigadas na *Humanoid League*. Vários dos melhores robôs humanoides autônomos do mundo competem na Liga Humanoide RoboCup.[3]

2.3 Restrições

Com a restrição de fazer robôs com sentidos (sensores) semelhantes ao dos humanos, o problema da percepção visual da bola se restringe ao uso de câmeras, sensor este que mais se assemelharia à funcionalidade do olho humano. Mas ainda há toda a problemática de detecção de objetos. Sendo que esta detecção deve ser o menos sensível possível à ruído de movimento, já que o robo estará andando. Além de problemas com iluminação, na qual uma mesma cor em iluminações variadas se tornam muito diferentes para a câmera.

E por último, isso tudo deve rodar dentro de um robô humanoid, ou seja, de maneira embarcada. Não é permitido qualquer tipo de comunicação externa com servidor. Por isso o algoritmo de detecção deve ser leve o suficiente para rodar em um dispositivo embarcado e eficiente o suficiente para não consumir toda a bateria antes do final da partida.

[3] https://www.robocuphumanoid.org/

3 Revisão bibliográfica

3.1 Deep Learning

O aprendizado profundo (*Deep Learning*) permite modelos computacionais, compostos de múltiplas camadas de processamento, possibilitando aprender representações de dados com múltiplos níveis de abstração. Esses métodos melhoraram drasticamente o estado-da-arte no reconhecimento de fala, reconhecimento de objetos visuais, detecção de objetos e muitos outros domínios.

As redes neurais profundas aprendem padrões em grandes conjuntos de dados usando o algoritmo *backpropagation*, para indicar como uma máquina deve alterar seus parâmetros internos, que são usados para calcular o impacto no erro de cada neurônio de cada camada e corrigi-los proporcionalmente. Redes convolucionais profundas trouxeram avanços no processamento de imagens, vídeo, fala e áudio, enquanto as redes recorrentes resolvem os problemas com dados sequenciais, como texto e fala.(LECUN; BENGIO; HINTON, 2015)

3.2 Backpropagation

A forma mais comum de aprendizado de máquina, profunda ou não, é o aprendizado supervisionado. Para isso é primeiro coletado um grande conjunto de dados, cada uma rotulada com sua categoria. Durante o treino, é mostrada a entrada da rede e produz uma saída na forma de um vetor de pontuações, uma para cada categoria. Queremos que a categoria desejada tenha a maior pontuação de todas as categorias, mas é improvável que isso aconteça antes do treinamento.

Foi calculado uma função objetiva que mede o erro (ou diferença) entre as pontuações de saída e o padrão desejado de pontuações. A máquina, em seguida, modifica seus parâmetros internos ajustáveis para reduzir esse erro. Esses parâmetros ajustáveis, geralmente chamados de pesos, são números reais que definem a função entrada-saída da máquina. Em um típico sistema de aprendizagem profunda, pode haver centenas de milhões desses pesos ajustáveis e centenas de milhões de exemplos rotulados com os quais treinar a máquina. Para ajustar corretamente o vetor de peso, o algoritmo de aprendizado calcula um vetor gradiente que, para cada peso, indica em qual quantidade o erro aumentaria ou diminuiria se o peso fosse aumentado por uma pequena quantidade. O vetor de peso é então ajustado na direção oposta para o vetor gradiente, ou seja, para a direção que minimiza o erro.

Como cada peso influencia de maneira diferente o erro final, cada peso deve ser corrigido de maneira proporcional a sua contribuição. Tal contribuição pode ser calculada utilizando derivadas parciais. Segue abaixo a equação.

Figura 1: Neurônio (LECUN; BENGIO; HINTON, 2015)

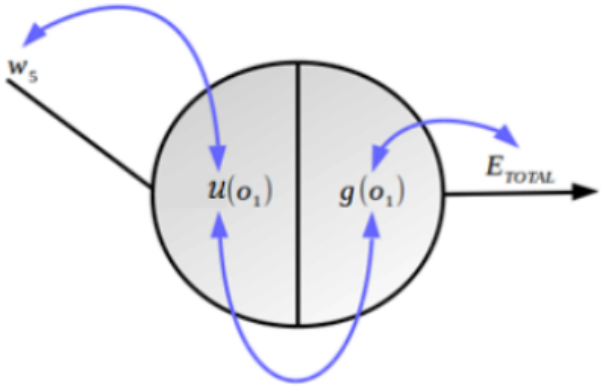


Figura 2: Backpropagation (LECUN; BENGIO; HINTON, 2015)

$$\frac{\partial E_{total}}{\partial w_5} = \frac{\partial E_{total}}{\partial g_{o1}} * \frac{\partial g_{o1}}{\partial u_{o1}} * \frac{\partial u_{o1}}{\partial w_5}$$

A função objetivo, em média sobre todos os exemplos de treinamento, pode ser vista como uma espécie de paisagem montanhosa no espaço de altos valores de peso. O vetor gradiente negativo indica a direção de descida mais íngreme nesta paisagem, levando-a mais perto de um mínimo, onde o erro de saída é baixo, em média. Na prática, a maioria dos praticantes usa um procedimento denominado descida estocástica do gradiente (SGD). Isso consiste em mostrar o vetor de entrada para alguns exemplos, computando as saídas e os erros, computando o gradiente médio para esses exemplos e ajustando os pesos adequadamente. O processo é repetido para muitos pequenos conjuntos de exemplos (*batch*) do conjunto de treinamento até que a média da função objetivo pare de diminuir.

É chamado de estocástico porque cada pequeno conjunto de exemplos fornece uma estimativa ruidosa do gradiente médio em todos os exemplos. Este procedimento simples geralmente encontra um bom conjunto de pesos surpreendentemente rápido quando comparado com técnicas de otimização muito mais elaboradas. Após o treinamento, o desempenho do sistema é medido em um conjunto diferente de exemplos chamado conjunto de testes. Isso serve para testar a capacidade de generalização da máquina - sua capacidade de produzir respostas sobre novas entradas que nunca viu durante o treinamento.(LECUN; BENGIO; HINTON, 2015)

3.3 Redes Convolucionáis

Existem vantagens bem conhecidas na realização de reconhecimento de formas em detectar e combinar características locais. Se um detector de recurso é útil em uma parte da imagem, é provável que seja útil em outras partes da imagem também.

A imagem de entrada é verificada com um único neurônio que tem um campo receptivo local, e armazena os estados deste neurônio em correspondentes locais em uma camada chamada *Feature map* (mapa de características). Esta operação é equivalente para uma convolução com um kernel de tamanho pequeno, seguido por uma função de *Pooling* (para reduzir a dimensão dos filtros). O processo pode ser executado em paralelo, implementando o *feature map* como um plano de neurônios cujos vetores de peso são restritos a serem iguais.

Ou seja, unidades em um *feature map* são restritos para executar a mesma operação em diferentes partes da imagem. Um interessante efeito colateral desta técnica de compartilhamento de peso, já descrita em (Rumelhart, Hinton e Williams, 1986), é reduzir o número de parâmetros livres por uma grande quantidade, uma vez que um grande número de unidades compartilham os mesmos pesos. Além disso, um certo nível de invariância de mudança está presente no sistema: entrada irá deslocar o resultado no *feature map*, mas deixará inalterado caso contrário. Na prática, será necessário ter vários *feature maps*, extraindo diferentes características da mesma imagem.

A ideia de *feature maps* locais e convolucionais pode ser aplicada a subsequentes camadas também, para extrair características de crescente complexidade e abstração. Curiosamente, recursos de nível superior exigem codificação menos precisa de sua localização. Reduzir a dimensão é realmente vantajosa, uma vez que uma ligeira distorção ou tradução da entrada terá efeito reduzido na representação.

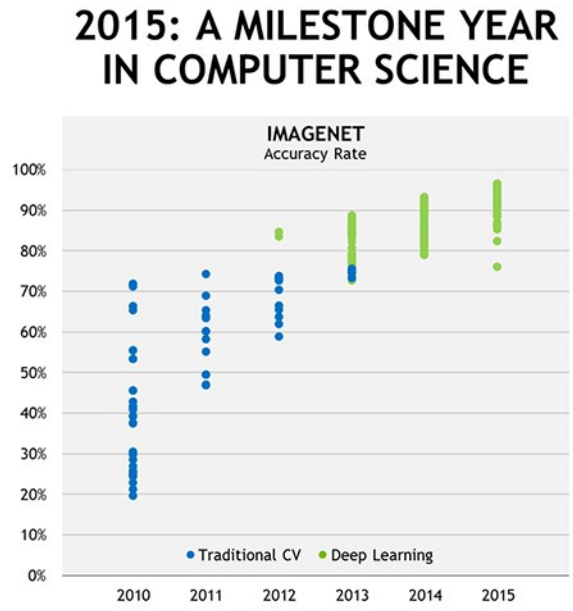
Assim, cada extração de características da rede é seguida por uma camada adicional que executa uma média local e uma subamostragem, reduzindo a resolução do *feature map*. Esta camada introduz um certo nível de invariância às distorções e traduções. Um módulo funcional da nossa rede consiste em uma camada de *feature maps* de peso compartilhado seguido por uma camada de média / subamostragem, mais conhecido como *Pooling*. Isso é uma reminiscência da arquitetura Neocognitron(Fukushima e Miyake, 1982), com a diferença notável de que usamos *backpropagation*(ao invés de aprendizado não supervisionado).(LECUN et al., 1990)

3.4 Detecção de objetos

De 2005 a 2013, várias técnicas foram aplicadas para avançar a precisão da detecção de objetos em conjuntos de dados. Na maior parte destes anos, variantes de HOG

(Histograma de gradientes orientados) + SVM (*Support Vector Machine*) ou DPM (*Deformable Part Models*) usado para definir a precisão do estado da arte nesses conjuntos de dados.(WU et al., 2017)

Figura 3: Deep Learning VS Traditional CV



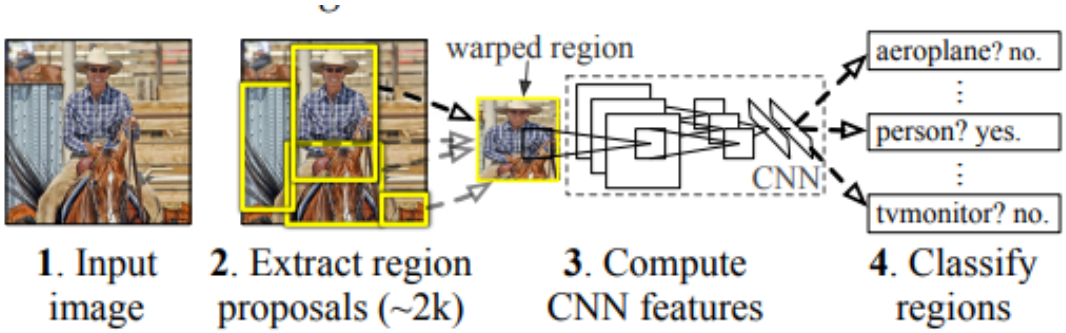
Inicialmente tratavam o problema de detecção como um problema de regressão, no qual queria saber os valores referentes a *bouding box* (caixa de delimitação). Mas tal técnica só funcionava para problemas com um único objeto na imagem. Mas como na maioria dos problemas há mais objetos na imagem partiu-se para outras soluções como um janela detectora que deslizava em toda a imagem para encontrar os objetos. Mas desta forma era necessário que os objetos tivessem um tamanho padronizado nas imagens para poder delimitar esta janela.

No entanto, em 2013, Girshick et al. com a proposta de *Region Based Convolutional Neural Network* (R-CNN), que levaram a ganhos substanciais na precisão da detecção de objetos. A abordagem da R-CNN começa identificando as propostas da região, que tem sido bem sucedido tanto para detecção de objetos quanto para semântica segmentação. No tempo de teste, o método gera cerca de 2000 propostas de regiões independentes da categoria imagem de entrada, extrai um vetor de características de tamanho fixo de cada proposta usando uma CNN (*Convolutional Neural Network*) e, em seguida, classifica cada região com SVMs lineares específicos da categoria. Usa se uma deformação simples (escala de imagem anisotrópica [1]) para computar um tamanho fixo da entrada da CNN de cada proposta de região, independentemente da forma da região. (GIRSHICK et al., 2016)

Uma desvantagem de R-CNN é que calcula a CNN independentemente em cada

[1] Anisotrópico significa que certas propriedades dependem da direção em que são medidas.

Figura 4: Region-based Convolutional Network (GIRSHICK et al., 2016)



proposta de região, levando tempo (<= 1 fps) e consumo ineficiente de energia (>= 200J/frame). (WU et al., 2017)

Embora as CNNs baseadas na região fossem computacionalmente caro como originalmente desenvolvido, seu custo foi drasticamente reduzido graças ao compartilhamento de convoluções. A última encarnação, Fast R-CNN, alcança taxas quase em tempo real usando redes profundas, ao ignorar o tempo gasto na região propostas. Agora, as propostas são o gargalo computacional nos sistemas de detecção de última geração.

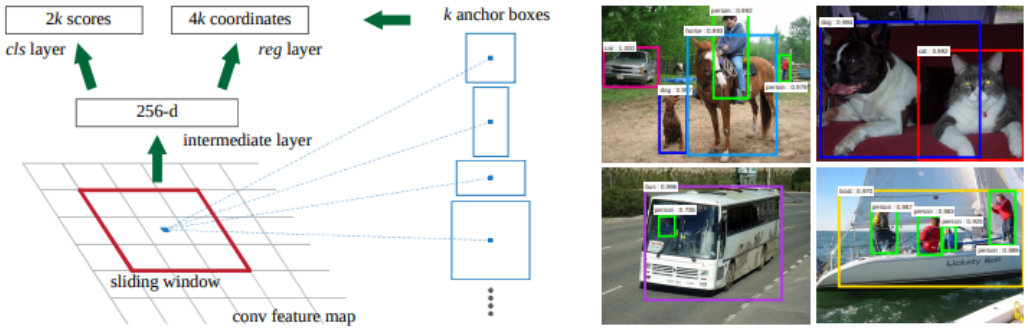
Para esse fim, surgiu as novas *Region Proposals Network* (RPNs) que compartilham camadas convolucionais com redes de detecção de objetos de última geração. Compartilhando convoluções em tempo de teste, o custo marginal para calcular propostas é pequeno (por exemplo, 10 ms por imagem).

Observaram que os mapas de características convolucionais (conv) usados por detectores baseados em regiões, como Fast R-CNN, também pode ser usado para gerar propostas de região. Além dessas características conv, construiram RPNs adicionando duas camadas convolucionais adicionais: uma que codifique cada posição do mapa convolucional em um vetor de características e um segundo que, em cada posição do mapa convolucional as pontuações de objetividade e limites regredidos para propostas de k regiões relativas a várias escalas e aspectos nesse local.

Uma RPN obtém uma imagem (de qualquer tamanho) como entrada e saída de um conjunto de propostas retangulares de objetos, cada uma com uma pontuação de objetividade. Modela-se esse processo com uma abordagem totalmente convencional. Porque o objetivo final é compartilhar computação com uma rede de detecção de objetos Fast R-CNN, assume-se que ambas as redes compartilham um conjunto comum de camadas convolucionais. Essa rede é totalmente conectada a uma janela espacial nn do *feature map* convolucional de entrada. Cada janela deslizante é mapeada para um vetor de menor dimensão. Esse vetor é alimentado em duas camadas totalmente conectadas - uma camada de regressão da *bouding box* (detecção) e uma camada de classificação. (REN et al., 2015)

Estes trabalhos de detecção de objetos reaproveitam classificadores para executar

Figura 5: Fast R-CNN (REN et al., 2015)



detecção. Em vez disso, surgiu uma nova proposta de rede conhecida como *You Only Look Once* (YOLO). A YOLO enquadra a detecção de objetos como um problema de regressão para as *bouding boxes* espacialmente separados e probabilidades de classe associadas. Uma única rede neural prevê as *bouding boxes* e probabilidades de classe diretamente de imagens completas em uma avaliação. Todo o pipeline de detecção é feito por uma rede única, pode ser otimizada de ponta a ponta diretamente no desempenho da detecção. O modelo YOLO processa imagens em tempo real a 45 quadros por segundo. Uma versão menor da rede, Fast YOLO, processa surpreendentes 155 quadros por segundo enquanto ainda alcançando o dobro do mAP de outros detectores em tempo real. Em comparação com sistemas de detecção de última geração, a YOLO faz mais erros de localização, mas é menos provável prever falsos positivos no fundo. Finalmente, a YOLO aprende representações gerais de objetos. Supera outros métodos de detecções ao generalizar imagens naturais para outros domínios.

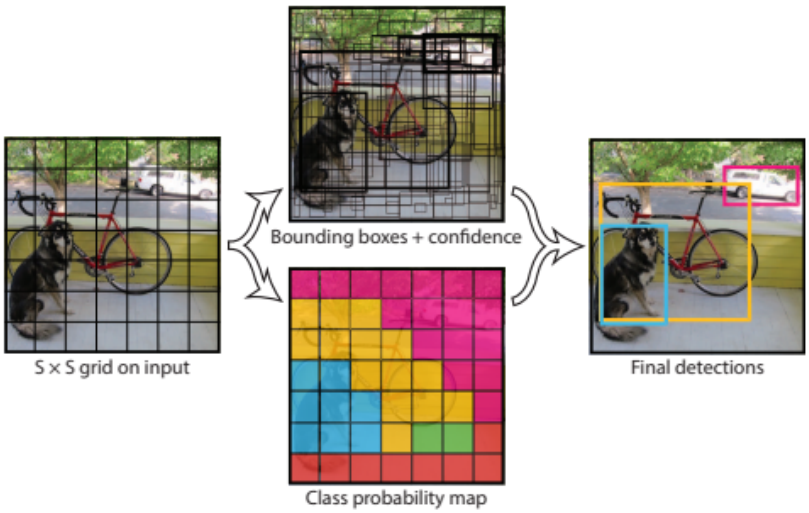
A YOLO divide a imagem de entrada em uma grade SS. Se o centro de um objeto cair em uma célula de grade, essa célula de grade é responsável por detectar esse objeto. Cada célula da grade prevê *bouding boxes* e *confidence score* para essas *bouding boxes*. Esses *confidence score* refletem a confiança que o modelo tem de que a caixa contém um objeto e também quão preciso a rede acha que a *bouding boxe* é o que ela prevê. Formalmente foi definido *confidence score* como $Pr(Objeto)IOU$. Se não existe objeto nessa célula, o *confidence score* deve ser zero. Caso contrário, queremos que a pontuação do *confidence score* seja igual à intersecção sobre união (IOU) das áreas entre a *bouding box* prevista e a verdadeira (da anotação).

Cada *bouding box* consiste em 5 previsões: x, y, w, h e classe. As coordenadas (x, y) representam o centro da *bouding box* em relação aos limites da célula da grade. A largura e altura são previstos em relação à imagem inteira. Finalmente a predição do *confidence score* representa o IOU entre o *bouding box* prevista e a verdadeira.

Cada célula (também conhecida como âncora) da grade também prevê as probabilidades da classe condicional C, $Pr(class_c|Object)$. Essas probabilidades são condicionadas

na célula da grade contendo um objeto. Foi previsto um conjunto de probabilidades de classe por célula de grade, independentemente do número de *bouding box*. No tempo de teste multiplica-se as probabilidades da classe condicional e as previsões individuais do *confidence score*, $Pr(class_c|Object) * Pr(Object) * IOU$. O que dá pontuações de confiança específicas da classe para cada caixa. Essas pontuações codificam a probabilidade dessa classe aparecendo na caixa e quão bem a *bouding box* prevista se encaixa no objeto. (REDMON et al., 2016)

Figura 6: You only look once (REDMON et al., 2016)



3.5 Redes embarcadas

Dado um determinado nível de precisão em uma visão computacional, é natural investigar a questão: o que é o menor tamanho de modelo que pode atingir esse nível de precisão? O SqueezeNet foi o resultado de uma dessas investigações. Alcançou o mesmo nível de precisão que o AlexNet na ImageNet para classificação de imagens com menos de 5MB de parâmetros. Uma redução de 50x em relação ao AlexNet. Depois de SqueezeNet, vários trabalhos continuaram a buscar mais estruturas de rede compactas. ENet explorou a decomposição espacial de núcleos convolucionais.

Juntamente com outras técnicas, ENet alcançou a precisão do nível SegNet para segmentação com 79X menos parâmetros. Recentemente MobileNet explorou a decomposição do canal de kernels convolucionais, e foi aplicado a vários dispositivos móveis para tarefas de visão, incluindo detecção de objetos, classificação de grãos finos, atributos de rosto e reconhecimento de marcas.

Para ser econômico e amplamente implantável, este detector de objetos deve operar em processadores embarcados que dissipam muito menos energia do que as GPUs (*Graphics Processing Unit*) . Tais modelos exigem certos requisitos, como:

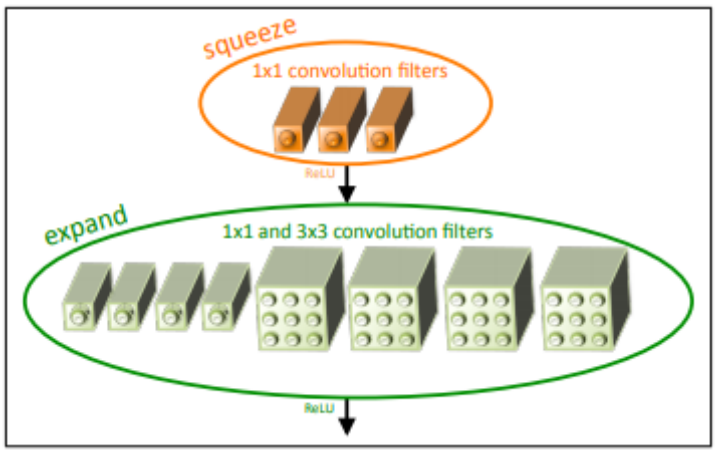
- O tamanho menor do modelo: traz benefícios em treinamento distribuído mais eficiente, menos energia consumida e implantação mais viável em sistemas embarcados.
- Eficiência energética: Sistemas de desktop e racks consomem por volta de 250W de energia para roda uma rede neural. Mas em sistemas embarcados há uma grande limitação de peso e espaço para comportar baterias. Com isso a potência disponível é muito menor. Como o modelo da rede é menor, há menos pesos e com isso menos processamento é necessário, economizando assim energia.
- FPS: Com um modelo menor, há menos cálculos a se fazer para realizar uma predição. Podendo processar mais frames por segundo.

3.5.1 squeezeNet

Pesquisas recentes em Redes Neurais Convolucionais têm focado em aumentar a acurácia em base de dados para visão computacional. Já a arquitetura da squeezeNet busca aproveitar os benefícios de uma rede menor. Com menos parâmetros, mas com uma acurácia equivalente comparada a outros modelos já conhecidos.

A squeezeNet é uma rede de classificação de imagens que introduz um conceito chamado *Fire module*, um novo bloco de construção de redes neurais. O *Fire module* é composto por duas camadas, a camada *squeeze* que possui apenas filtros 1x1, e a saída desta camada se conecta na entrada da camada *expand* que possui filtros 1x1 e 3x3. Sendo que o número de filtros da camada *squeeze* deve ser menor que a soma do número de filtro da camada *expand*. A arquitetura da rede se resume a uma sequência de *Fire modules*, em que o número de filtros por camada vão aumentando gradualmente no decorrer da rede.

Figura 7: Fire module (IANDOLA et al., 2016)



Tal modelo foi capaz de alcançar a mesma acurácia que a AlexNet na base de dados da ImageNet com o tamanho do modelo 510x menor, usando um processo de compressão

de rede chamado *Deep Compression*. De 240 MB da AlexNet para 0.5 MB da squeezeNet comprimida.(IANDOLA et al., 2016)

Figura 8: Tabela de comparação (IANDOLA et al., 2016)

CNN architecture	Compression Approach	Data Type	Original → Compressed Model Size	Reduction in Model Size vs. AlexNet	Top-1 ImageNet Accuracy	Top-5 ImageNet Accuracy
AlexNet	None (baseline)	32 bit	240MB	1x	57.2%	80.3%
AlexNet	SVD (Denton et al., 2014)	32 bit	240MB → 48MB	5x	56.0%	79.4%
AlexNet	Network Pruning (Han et al., 2015b)	32 bit	240MB → 27MB	9x	57.2%	80.3%
AlexNet	Deep Compression (Han et al., 2015a)	5-8 bit	240MB → 6.9MB	35x	57.2%	80.3%
SqueezeNet (ours)	None	32 bit	4.8MB	**50x**	57.5%	80.3%
SqueezeNet (ours)	Deep Compression	8 bit	4.8MB → 0.66MB	**363x**	57.5%	80.3%
SqueezeNet (ours)	Deep Compression	6 bit	4.8MB → 0.47MB	**510x**	57.5%	80.3%

3.5.2 squeezeDet

A rede escolhida neste trabalho foi a SqueezeDet, uma rede neural totalmente convolucional para detecção de objetos. O pipeline de detecção do SqueezeDet é: primeiro, há filtros de convolução empilhados para extrair um *feature map* de alta e baixa resolução para a imagem de entrada. Então, há camadas convolucionais para extrair o *feature map* como entrada e computar uma grande quantidade de *bouding boxes* de objetos e prever sua categorias. Por fim, filtramos essas *bouding boxes* para as detecções finais.

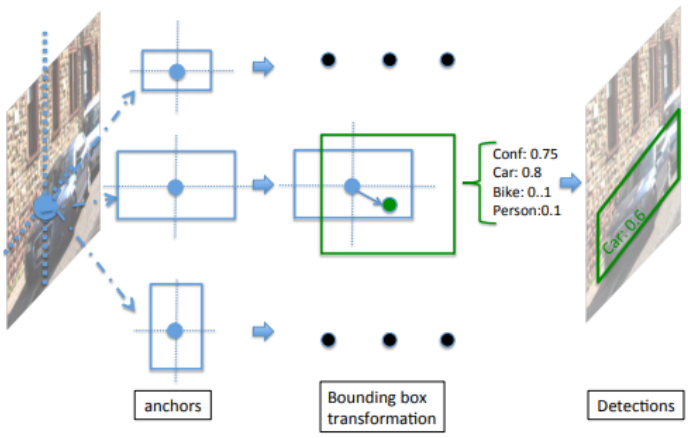
A arquitetura do *backbone* (espinha dorsal) da rede convolucional é a squeezeNet. Depois de fortalecer o modelo SqueezeNet com camadas adicionais seguidas pelo ConvDet. A velocidade de inferência do modelo alcançou 57,2 FPS (quadros por segundo) com resolução de imagem de entrada de 1242x375 para a base de dados do KITTI rodando em uma NVIDIA TITAN X.

Beneficiando do tamanho pequeno do modelo, a SqueezeDet tem custo de memória muito menor e requer menos acessos DRAM, assim consomiu apenas 1,4J de energia por imagem em um TITAN X GPU, que é cerca de 84 vezes menor que um modelo R-CNN mais rápido.

Sendo redes neurais um aprendizado supervisionado, a informação que será usada para treinar a rede (conhecido como anotação ou *label*) é formada por várias *bouding boxes* que representam a localização dos objetos na imagem. Cada *bouding box* possui cinco parâmetros, as coordenadas do ponto central, x e y; a altura e largura da bbox e a classe de objeto à qual a bbox pertence. Sendo estes *bouding box* gerados aleatoriamente na imagem, mas de tamanho previamente definido.

Devido o problema ser mais complexo que uma simples classificação, o cálculo do erro também é mais complexo. Muito semelhante à abordagem da YOLO. Inicialmente é calculado o IOU (*Intersection over Union*) que representa o quão próximo o bbox gerado está do real, sendo este calculado pela divisão entre a intersecção das áreas dos *bouding*

Figura 9: Âncoras (WU et al., 2017)



boxes sobre a união das áreas dos *bouding boxes*. Depois é calculado o *Confidence score* que é a multiplicação do IOU pela probabilidade de conter um objeto de interesse dentro do *bouding box*, dado por $Pr(Object) * IOU$.

Depois é calculado o *Conditional class probability distribution* que representa o quão provável é daquele objeto detectado ser o mesmo objeto da anotação, dado por $Pr(class_{\mathrm{c}}|Object), c \in [1, C]$. E então é calculado o *Bouding box confidence* que é dado por $maxPr(class_{\mathrm{c}}|Object) * Pr(Object) * IOU$. E por último são filtrados os N melhores *bouding box* usando *Non-Maximun Supression*(NMS) para transformar bboxs redundantes em um único.(WU et al., 2017)

4 Metodologia

Foi utilizado como base para esta monografia o trabalho *SqueezeDet: Unified, Small, Low Power Fully Convolutional Neural Networks for Real-Time Object Detection for Autonomous Driving By Bichen Wu, Alvin Wan, Forrest Iandola, Peter H. Jin, Kurt Keutzer (UC Berkeley DeepScale).*[1]

4.1 Transfer Learning

Devido a falta de imagens com anotações específicas para o problema de futebol de humanoid. Será necessário usar uma técnica conhecida como *Transfer Learning*.

Uma suposição importante em muitos algoritmos de aprendizado de máquina e mineração de dados é que o treinamento e os dados futuros devem estar no mesmo espaço de recurso e ter a mesma distribuição. No entanto, em muitas aplicações do mundo real, essa suposição pode não ser válida. Às vezes temos uma tarefa de classificação em um domínio de interesse, mas temos apenas dados de treinamento suficientes em outro domínio de interesse, onde os últimos dados podem estar em um espaço de recurso diferente ou seguir uma distribuição de dados diferente.

Nesses casos, o *Transfer Learning*, se feita com sucesso, melhorara muito o desempenho do aprendizado, evitando esforços de anotação de dados muito dispendiosos. Nos últimos anos, o *Transfer Learning* surgiu como uma nova estrutura de aprendizado para abordar esse problema. (PAN; YANG et al., 2010)

Esta técnica consiste em fazer um treinamento prévio em um dataset fonte que possua uma quantidade vasta de dados anotados. Para possibilitar que a rede aprenda a extrair características, das mais simples como linhas, curvas e cores às mais complexas como formas geométricas, texturas de uma imagem. Características essas que podem ser comuns entre os datasets. Quanto mais próximo forem os domínios dos datasets, mais características terão em comum.

Após tem uma rede boa em extração de características, congela-se as camadas iniciais e retreina as camadas finais com uma taxa de aprendizado baixo usando o dataset alvo do problema em questão, sendo necessário assim menos imagens para que a rede termine de aprender os detalhes do novo domínio. Método esse também conhecido como *Fine tunning*.

[1] https://github.com/BichenWuUCB/squeezeDet

4.2 Dataset

Foi utilizado os pesos de uma squeezeNet pré-treinada no ImageNet (para problemas de classificação de objetos), para servir como extrator de características. Desta forma a rede só precisará aprender a agrupar as características relevantes que compõem uma bola. Com isso será necessário menos imagens para o treinamento.

Para o dataset alvo foi utilizado um dataset colaborativo feito em conjunto por várias equipes competidoras da categoria *Humanoid League*. O dataset pode ser encontrado no site *Image Tagger*. [2] O dataset possui como anotações as classes *ball, goal, robot, obstacle, goalpost, line*. Podendo fazer as anotações pelo próprio site.

Entretanto, há uma quantidade insuficiente de imagens para fazer um treinamento de rede neural para detecção. Grande parte das imagens não estão anotadas ou só possuem anotação da bola ou no máximo dos robôs, sem contar que há muitas imagens parecidas (frames de um vídeo).

Para evitar modificações complexas na estrutura da rede para se adaptar as anotações do novo dataset, foi feito um algoritmo para adaptar as anotações à rede. A partir de uma arquivo JSON, gera-se um arquivo txt para cada imagem com as anotações desta.

4.3 Hiper parâmetros

O trabalho original, que foi feito para o *dataset*(base de dados) KITTI para carros autônomos. Com isso é necessário fazer uma série de ajustes na rede. As principais modificações a se destacar são a dimensão da imagem de entrada, em vez de se usar 1248x384, foi reduzido para 256x256 e por consequência o formato da grid foi reduzido de 78x24 para 16x16. Notem que a proporção de 1/16 deve ser mantida. Tal redução foi necessário por motivos de capacidade de hardware, quanto maior a imagem mais memória é demandada. E como não era necessária uma imagem tão grande para o problema de detecção de bola, foi mais vantajoso reduzir a dimensão da imagem e aumentar o tamanho do batch (quantidade de imagens que juntas alimentam a rede para realizar o backpropagation).

Outra alteração necessária foi nas *anchors seeds* que são uma lista de possíveis dimensões que serão usadas como referência na grid para determinar os *bouding boxes*. Ou seja dimensões maiores são para detectar objetos maiores, e menores para objetos menores. Como o objetivo deste trabalho é detecção de bola, no qual a grande maioria das imagens o diâmetro da bola não ocupa nem 40% da imagem, foi usado dimensões entre 5x5 e 90x90 em uma imagem 256x256.

Devido a natureza do problema, é mais vantajoso não detectar, do que detectar algo errado (falsos positivos), para evitar ruído no sistema de trajetória do robô. Para

[2] https://imagetagger.bit-bots.de/images/imageset/explore/

isso foi aumentado o limite da probabilidade de plotar o *bouding boxes* de 40% para 70%. Além de outras pequenas mudanças nos parâmetros para melhoras a detecção da rede.

5 Resultados

Como treinamento preliminar para testar se a arquitetura da rede tem capacidade de aprender, foi feito um treinamento no dataset do KITII para carros autônomos e apresentou resultados significativamente bons.

Figura 10: Detecção do KITTI

Visto que a arquitetura da rede consegue aprender, foi feito um treinamento usando um dataset do ImageTagger com 9000 imagens quem continham a classe bola. E feito a validação em um conjunto de 1000 imagens novas. Segue abaixo o resultado da avaliação.

Tabela 1: Eval

Número de detecções	10003
Número de objetos	10003
Porcentagem de detecções corretas	0.9840478
Porcentagem de erros de localização	0.0099700
Porcentagem de erros de classificação	0.00

Também apresentou resultados suficientemente bons como podem ser vistos nas imagens detectadas do conjunto de teste abaixo. E seus gráficos de treinamento da bbox loss, da confidence loss e do total loss.

Com um modelo de 16 MB atingiu 2 FPS (Frame per Second) rodando em uma Raspberry Pi 3. Códigos usados neste trabalho se encontram no link. [1]

Foi feito um modelo menor, com 3 camadas a menos e por consequência 2304 filtros a menos o que resultou em uma modelo de 6MB. E os resultados da avaliação na tabela abaixo.

[1] https://github.com/AjJordy/squeezeDet-master

Tabela 2: Eval Small

Número de detecções	10003
Número de objetos	10003
Porcentagem de detecções corretas	0.9531405
Porcentagem de erros de localização	0.0348953
Porcentagem de erros de classificação	0.00

Pode se notar que a redução do tamanho do modelo foi de 62,5% menor e uma difereça de apenas 3% na porcentagem de detecções corretas.

Figura 11: Detecção da bola (0.78, 0.79, 0.81, 0.78, 0.74)

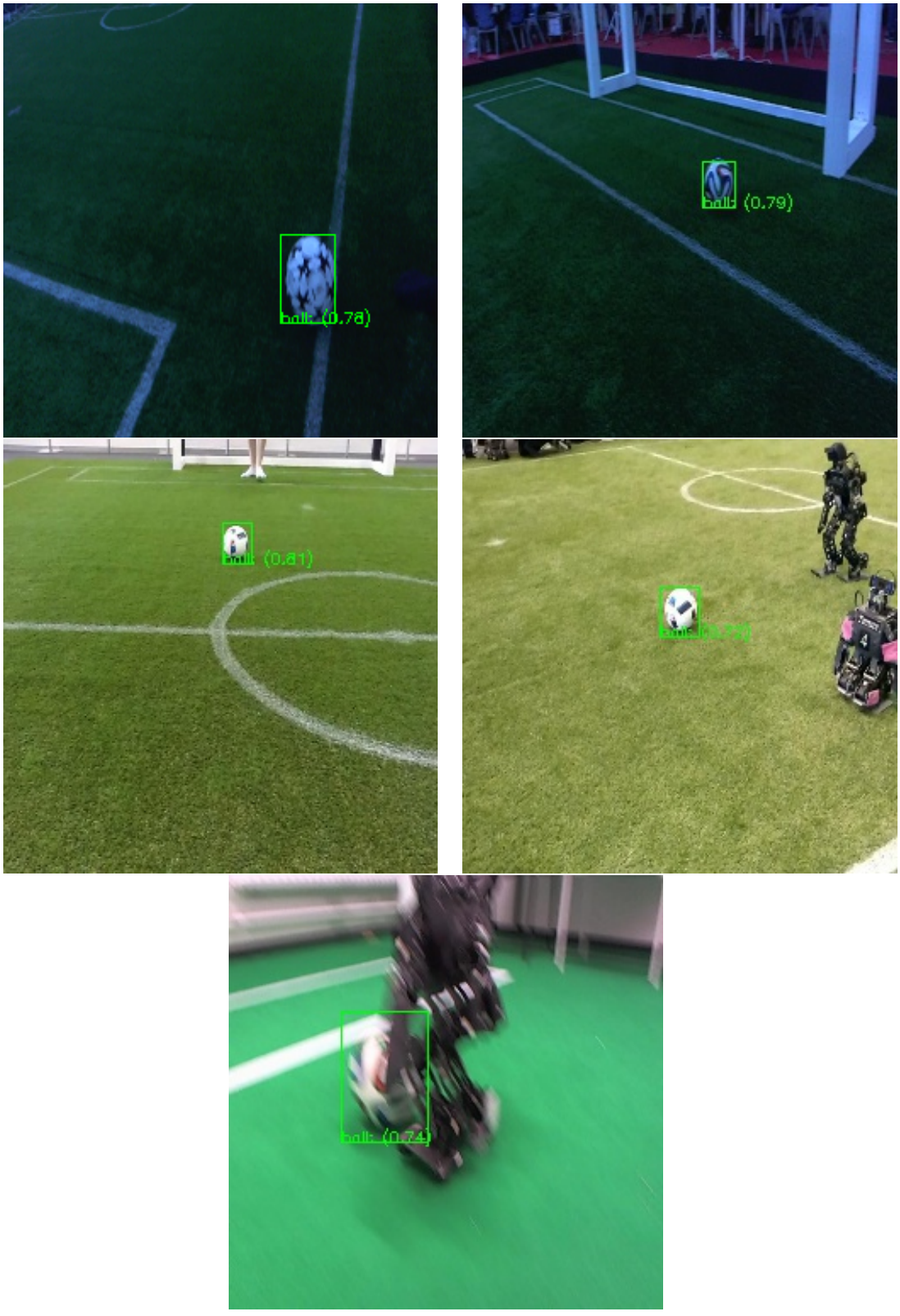


Figura 12: Loss

Conclusão

Pode se notar o quão poderoso são redes neurais como ferramenta, principalmente se tratando de imagem, graças aos filtros convolucionais. E os avanços na área de detecção de imagem já tem demonstrado resultados ótimos com relação a acurácia. A corrida agora tem ido para a simplificação da rede para economizar consumo de energia e processamento.

Os resultados deste trabalho demonstraram que mesmo com uma rede neural tão pequena, 16MB em comparação com os 240MB da AlexNet (classificação), é capaz de fazer uma boa detecção de objeto. Mas a rede atingia apenas 2 FPS rodando numa Raspberry Pi 3. Isso não inviabiliza totalmente a rede para o propósito do futebol.

As sugestões para trabalhos futuros seriam reduzir ainda mais a rede (camadas e/ou filtros). E parametrizar a rede para não perder tanta acurácia com as reduções. Aplicar algoritmos de *Deep Compression* para reduzir ainda mais o tamanho da rede e alcançar melhores FPS. Algoritmos de *pruning*, *quantization*, entre outros. Caso nada disso resulte em um FPS ideal para rodar em uma Raspberry, outra opção viável seria utilizar métodos clássicos de visão computacional para o tracking da imagem depois de já ter encontrado a bola, visando melhoras no FPS.

Outra tarefa muito importante seria anotar mais imagens, para poder além de melhorar a acurácia de detecção da rede mas também para começar a detectar outras classes como gol e robôs. Detecção que será necessária para competir mais seriamente na categoria do futebol de humanoide.

Referências

GIRSHICK, R. et al. Region-based convolutional networks for accurate object detection and segmentation. *IEEE transactions on pattern analysis and machine intelligence*, IEEE, v. 38, n. 1, p. 142–158, 2016. Citado 3 vezes nas páginas 7, 17 e 18.

IANDOLA, F. N. et al. Squeezenet: Alexnet-level accuracy with 50x fewer parameters and< 0.5 mb model size. *arXiv preprint arXiv:1602.07360*, 2016. Citado 3 vezes nas páginas 7, 21 e 22.

LECUN, Y.; BENGIO, Y.; HINTON, G. Deep learning. *nature*, Nature Publishing Group, v. 521, n. 7553, p. 436, 2015. Citado 3 vezes nas páginas 7, 14 e 15.

LECUN, Y. et al. Handwritten digit recognition with a back-propagation network. In: *Advances in neural information processing systems*. [S.l.: s.n.], 1990. p. 396–404. Citado 2 vezes nas páginas 10 e 16.

PAN, S. J.; YANG, Q. et al. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, Institute of Electrical and Electronics Engineers, Inc., 345 E. 47 th St. NY NY 10017-2394 USA, v. 22, n. 10, p. 1345–1359, 2010. Citado na página 24.

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 2 vezes nas páginas 7 e 20.

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 3 vezes nas páginas 7, 18 e 19.

WU, B. et al. Squeezedet: Unified, small, low power fully convolutional neural networks for real-time object detection for autonomous driving. In: *CVPR Workshops*. [S.l.: s.n.], 2017. p. 446–454. Citado 5 vezes nas páginas 7, 10, 17, 18 e 23.