FLODNet - Detecção e reconhecimento de objetos em dispositivos de baixa especificação: um estudo de caso em classificação de alimentos

B. A. G. de Oliveira[1], **F. M. F. Ferreira** [1] *, **and C. A. P. S. Martins** [1] *

[1]Instituto de Ciências Exatas e Informática (ICEI) –
Pontifícia Universidade Católica de Minas Gerais (PUC Minas)

bernardo.godinho.oliveira@gmail.com, flaviamagfreitas@pucminas.br,

capsm@pucminas.br

Abstract. *The intrinsic ability of humans to rapidly detect, differentiate and classify objects allows us to make quick decisions in regards to what we see. Several appliances can make use of a fast and lightweight automated object detection for images or videos. Throughout the last 5 years, the technology industry has constantly introduced computational and hardware solutions, such as devices with impressive processing and storage capabilities. However, object detection and recognition methods usually require high processing power and/or large storage availability, making it hard for resource constrained devices to perform the detection and recognition in real-time without a connection to a powerful server. The model presented in this paper requires only 95 megabytes of storage and took 113 ms in average per image running on a laptop CPU, making it suitable for standalone devices that can be used on the go.*

Resumo. *A capacidade intrínseca dos humanos de detectar, diferenciar e classificar rapidamente os objetos nos permite tomar decisões rápidas em relação ao que é visto. Aplicações podem se beneficiar de detecção rápida e leve de objetos para imagens ou vídeos. Embora, nos últimos 5 anos, o setor de tecnologia tenha apresentado dispositivos com recursos de processamento e armazenamento impressionantes, os métodos de detecção e reconhecimento de objetos geralmente requerem alto poder de processamento e/ou grande disponibilidade de armazenamento, tornando difícil para os dispositivos com recursos restritos realizar a detecção e reconhecimento em tempo real sem uma conexão com um servidor. O modelo apresentado neste documento requer apenas 95 megabytes de armazenamento e a execução requer 113 ms em média por imagem em CPU de um laptop, tornando-o adequado para dispositivos que podem ser usados em qualquer lugar.*

1. Introdução

A detecção de objetos é um passo crucial para um bom reconhecimento de objetos, especialmente se houver diversos objetos em uma mesma imagem [Endres and Hoiem 2010]. As técnicas de detecção de objetos localizam instâncias de objetos em uma imagem ou

*Orientadores do trabalho de conclusão de curso do primeiro autor.

um vídeo, geralmente exibindo coordenadas dos pontos de uma caixa delimitadora para cada objeto [de Sande et al. 2011]. Objetos, ao contrário do fundo, são instâncias com limites e centro bem definidos [Alexe et al. 2010]. Uma técnica comum para realizar a detecção de objetos é formulá-la como um problema de classificação, de tal forma que janelas deslizantes ou máscaras de convolução em posições, escalas e aspectos diferentes são usadas sobre representações, ou sobre a própria imagem, para classificar pequenas regiões [Hosang et al. 2014].

Na classificação de objetos, tanto a detecção como o reconhecimento são modelados de forma que: (i) cada tipo de objeto seja uma classe, (ii) uma imagem pode conter uma ou várias instâncias dessas classes (objetos) e (iii) existe um conjunto finito de tipos de objeto. Normalmente, um classificador avançado e complexo, tal como uma rede neural convolucional (CNN - *Convolutional Neural Network*) profunda, é necessário para obter uma alta acurácia nas detecções. No entanto, o uso desses modelos diminui consideravelmente a velocidade da avaliação, devido ao aumento do custo computacional [Felzenszwalb et al. 2010]. Métodos para detectar e reconhecer objetos que requerem menos recursos são importantes para conseguir a detecção rápida de objetos em dispositivos de baixo desempenho, baixo custo ou baixo consumo [Zhang et al. 2011]. Esse tipo de sistema pode beneficiar áreas como: eletrônicos de uso pessoal, transporte, saúde, sistemas de segurança, entre outros.

Este trabalho foca no desenvolvimento da *Fast and Lightweight Object Detection Network* (FLODNet), apresentada em [de Oliveira et al. 2018]. Trata-se de um modelo de CNN que fornece detecção e reconhecimento de objetos rapidamente, enquanto economiza recursos como espaço de armazenamento, processamento e memória. Essas restrições permitem que dispositivos portáteis alcancem uma detecção de objeto em vídeos com alta taxa de quadros, sem o uso de um *graphic processing unit* (GPU). Mesmo quando executado em hardware de maior capacidade, um modelo mais simples e menor é mais eficiente em energia e irá dissipar menos calor [Albers 2010]. Como um exemplo de aplicação do mundo real, o estudo de caso apresentado neste artigo mostra o uso da FLODNet para detectar e reconhecer o conteúdo de refeições em um prato. O objetivo desta tarefa é detectar o conteúdo do prato e classificar cada alimento. A exatidão do posicionamento das caixas de delimitação propostas pode ser relaxada em favor de uma detecção mais rápida, sem prejudicar a acurácia do reconhecimento do objeto.

2. Trabalhos Relacionados

Recentemente diversos trabalhos mostraram o potencial das CNNs para detecção e reconhecimento de objetos. Em 2014, foi proposta a R-CNN [Girshick et al. 2014], que utilizava técnicas de processamento de imagens para propor regiões que seriam enviadas ao classificador. Devido ao alto custo computacional e à detecção lenta da R-CNN, vários métodos foram propostos para melhorar a tarefa de propor regiões para detecção de objetos. Alexe et al. [Alexe et al. 2012] apresentam uma técnica que, usando janelas deslizantes, analisa todas as regiões da imagem à procura de objetos. Uma evolução direta da R-CNN é a Fast R-CNN [Girshick 2015], que usa uma camada de agrupamento de regiões de interesse (ROI - *Regions of Interest*) para reduzir a quantidade de vezes que a mesma imagem é analisada. A Faster R-CNN [Ren et al. 2015], estado-da-arte, reduz ainda mais o tempo de detecção do objeto, enquanto a acurácia é ligeiramente melhorada em relação à Fast R-CNN.

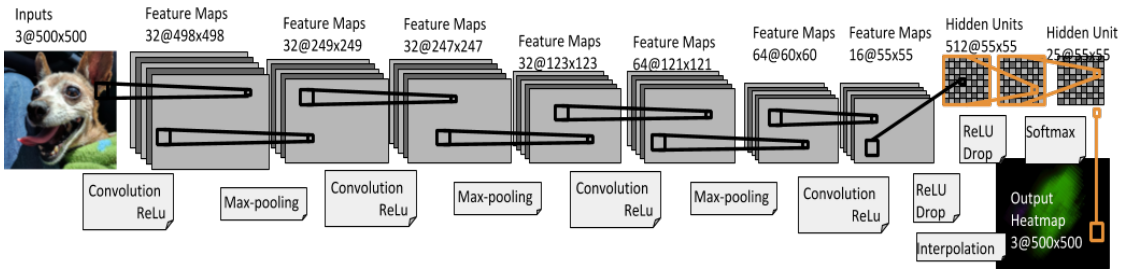

Figura 1. Arquitetura da FLODNet.

3. *Fast and Lightweight Object Detection Network* (FLODNet)

Para alcançar alta velocidade e assim superar em tempo de processamento redes do estado-da-arte, a *Fast and Lightweight Object Detection Network* (FLODNet) [de Oliveira et al. 2018] sacrifica a exatidão no posicionamento das detecções usando um modelo de CNN com apenas 10 camadas, como apresentado na Fig. 1. Além disso, todos os passos de detecção e reconhecimento são feitos analisando a imagem por completo apenas uma vez. Em vez de usar janelas deslizantes de proporções diferentes para a detecção, como faz a Faster R-CNN [Ren et al. 2015], a arquitetura proposta utiliza apenas uma convolução com máscara de tamanho fixo sobre o *feature map*, para realizar uma detecção mais rápida. Essa simplificação pode afetar algumas tarefas, mas aquelas que não dependem do posicionamento exato das detecções podem se beneficiar do baixo custo computacional da FLODNet.

3.1. Obtenção e Preparação da Base de Dados

Uma coleção de 200 imagens de pratos contendo uma ou mais classes de alimentos categorizados em 12 classes foi obtida em uma cafeteria nos Estados Unidos, especificamente para este projeto. Todas as imagens foram obtidas em 1936x1296 *pixels*. Posteriormente, elas tiveram seu conteúdo segmentado e classificado manualmente com o auxílio do software LabelMe [Russell et al. 2008]. O conjunto de imagens foi então dividido em dois grupos, treino e teste, de modo que as imagens de teste não sejam utilizadas durante o treino. Para aumentar o número de imagens, técnicas de sintetização de imagens foram utilizadas e são apresentadas posteriormente. Utilizando essas técnicas foi possível obter 20.000 novas imagens, totalizando 20.200 para treino e teste.

3.2. Modelo

Uma vez que cada *pixel* é um neurônio na primeira camada da CNN, ao reduzir o tamanho da imagem o número de neurônios e convoluções também é reduzido. Como o objetivo do projeto é reduzir o custo computacional, as imagens foram reduzidas para 500x500 *pixels*, como uma forma de acelerar o processo de treinamento e reconhecimento. Durante a execução a imagem subamostrada é processada pela FLODNet, que classifica cada região. Como a imagem foi redimensionada, as regiões classificadas são então expandidas para coincidir com o tamanho da imagem original e, utilizando essas regiões, as caixas de delimitação são geradas.

A arquitetura da FLODNet, explicada com mais detalhes em [de Oliveira et al. 2018], pode ser vista na Fig. 1. Existem três camadas convolucionais de 3x3, cada uma delas usando *rectified linear unit* (ReLU) como funções de

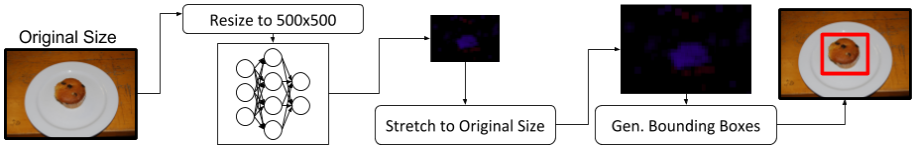

Figura 2. Diagrama mostrando o pipeline de execução do modelo.

ativação e, em seguida, por uma camada de *maxpool*. Uma camada convolucional 6x6 sobre o *feature map* divide regiões que são enviadas para as camadas totalmente conectadas (Fig. 2), que são responsáveis por classificar essas regiões como uma das classes treinadas, criando uma representação semelhante a um mapa de calor. Uma vez que a proporção e o tamanho do *feature map* são diferentes da imagem original, as coordenadas das regiões classificadas não podem ser usadas diretamente. A saída contém um grande número de pequenas regiões detectadas e é necessário fazer o cálculo de proporção para obter o valor final.

Para reduzir o *overfit* no treino, cada camada totalmente conectada implementa uma técnica de *dropout*. Essa técnica remove temporariamente um neurônio escolhido aleatoriamente, bem como todas as suas conexões, durante uma execução. Com isso, são reduzidas co-adaptações nos dados de treinamento [Baldi and Sadowski 2013]. O *dropout* melhora o desempenho das redes neurais em uma grande variedade de aplicações, sendo sua única desvantagem o aumento do tempo de treinamento [Srivastava et al. 2014].

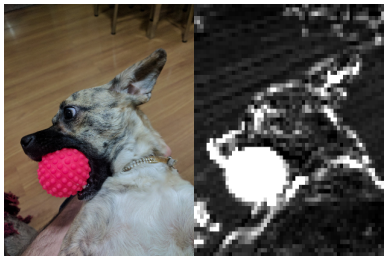

Figura 3. Representação criada utilizando a união de todos os *feature maps*.

Após completar a detecção e o reconhecimento, as regiões detectadas como pertencentes a uma das classes são analisadas para remover detecções de pouca confiança (com probabilidade inferior ao limiar de confiança estabelecido). As regiões de alta confiança remanescentes são combinadas com base nas classes e na área de sobreposição, para criar caixas de delimitação maiores. Após a execução, uma nova imagem é exportada com as caixas de delimitação, destacando os objetos detectados e reconhecidos.

3.3. Treinamento

As imagens são pré-processadas por um *script* que prepara o conjunto de dados para treinamento. Um arquivo *Comma-Separated Values* (CSV) é criado mapeando os objetos que podem ser encontrados em cada região das imagens, para que a rede possa ser treinada para classificar essas regiões. O modelo foi implementado no TensorFlow [Abadi et al. 2015] usando o Python visando fácil utilização em diferentes dispositivos e arquiteturas, permitindo que o projeto seja treinado e executado em CPU ou GPU.

Para treinar completamente uma CNN, é necessária uma grande quantidade de

dados com alta variabilidade. Isso aumenta a probabilidade da rede aprender a detectar características de baixo nível corretamente, aumentando a acurácia em novos dados e reduzindo o *overfit*. As características de baixo nível representam os atributos mais básicos da imagem, como por exemplo: bordas, círculos, linhas e cores. Uma vez que as camadas intermediárias de uma CNN são geralmente extratores de características que podem ser generalizadas para conjuntos de dados diferentes, uma solução comum para o treinamento é usar um conjunto de dados maior para pré-treinar a rede e, em seguida, retreinar as camadas completamente conectadas para reconhecer as classes de alimentos [Oquab et al. 2014]. O conjunto de dados *Dogs vs. Cats* do Kaggle [Kaggle 2004], que contém 25.000 imagens, foi usado para pré-treinar a FLODNet. A quantidade de dados, bem como a variabilidade de cores e formas neste conjunto de dados, ajudam a generalizar características de baixo nível nos filtros convolucionais.

Um modelo de CNN como AlexNet [Krizhevsky et al. 2012] pode consumir até 11 GB de memória da GPU durante o treino, tornando impossível o seu treino completo, na maioria das GPUs modernas. Uma vez que é esperado que a FLODNet seja usada em ambientes com recursos computacionais limitados, o treinamento foi dividido em fases, como apresentado na Fig. 1. Cada fase treina uma parte da rede utilizando uma camada extra completamente conectada para classificação e são treinadas no conjunto de dados do Kaggle. A primeira fase treina as quatro primeiras camadas. Para a segunda fase, os pesos treinados na primeira fase são mantidos inalterados, enquanto o resto da parte convolucional é adicionado à rede como camadas treináveis. Após o treinamento das camadas convolucionais, as camadas completamente conectadas são adicionadas e a rede está pronta para ser treinada para detecção e reconhecimento usando o conjunto de dados alvo. Para alcançar uma convergência mais rápida e evitar o *overfit* devido à ordem das amostras durante o treino, os dados de treinamento são reordenados em cada *epoch* [Ioffe and Szegedy 2015]. Aliado ao *batch* de 32 imagens, foi possível treinar a rede usando menos de 3 GB de memória de vídeo.

3.4. Métricas

Para medir a exatidão das detecções é utilizada a razão entre a interseção e a união das regiões delimitadas pelas caixas (IoU), como descrito na Equação 1.

$$\text{IoU(A,B)} = \frac{|A \cap B|}{|A \cup B|} \text{ sendo } 0 \leq \text{IoU(A,B)} \leq 1 \tag{1}$$

Como mostrado em [Krähenbühl and Koltun 2014], o IoU representa com exatidão a similaridade entre as regiões de pixels. Os objetos detectados são considerados corretos se coincidirem com a classe esperada e se suas caixas de delimitação se sobrepuserem em pelo menos 50% em relação à detecção original, chamada de *ground truth*.

Para medir a qualidade das detecções é utilizada a acurácia, como visto na Eq 2:

$$\text{Acurácia} = \frac{\text{Verdadeiros Positivos + Verdadeiros Negativos}}{\text{Verdadeiros + Negativos}} \tag{2}$$

Para se obter o IoU geral, denominado de IoU_g, a média dos IoUs obtidos em cada detecção de um mesmo objeto, em toda a base de dados, é calculada como apresentado na Equação 3. Nesse caso, a variável $bbox$ representa uma única detecção, que é comparada

ao *ground truth*. Essa métrica é útil para medir a exatidão de detecção das caixas de delimitação propostas.

$$\mathrm{IoU_g} = \frac{\sum_{bbox=0}^{\text{bboxes-1}} \text{IoU(bbox,ground truth)}}{\text{número de bboxes}} \tag{3}$$

Contudo, como uma vantagem da FLODNet é o tempo reduzido de detecção e reconhecimento, propõe-se aplicar uma métrica que leve o tempo de reconhecimento em consideração. A métrica IoU por tempo, $\mathrm{IoU_t}$, mostrada na Equação 4, permite avaliar o desempenho da rede por tempo de reconhecimento.

$$\mathrm{IoU_t} = \frac{\text{IoU x Acurácia}}{\text{Tempo de Reconhecimento}} \tag{4}$$

4. Resultados

Embora diversas adaptações têm sido feitas em estabelecimentos comerciais, escolas e locais públicos para incluir socialmente pessoas com deficiência visual, alguns lugares ainda não suportam esse tipo de infraestrutura [de Freitas Alves et al. 2009]. As pessoas com deficiência visual, parcial ou total, geralmente são capazes de identificar objetos através do toque. No entanto, alguns fatores tornam essa tarefa difícil, ou mesmo impossível, de executar, tais como: falta de conhecimento prévio do objeto, semelhança com outros objetos ou impossibilidade de tocar. Além disso, restrições alimentícias podem tornar o simples ato de comer arriscado. Algumas vezes, as refeições podem ser difíceis de reconhecer, mesmo por pessoas sem deficiência visual. Alguns fatores podem aumentar a importância do reconhecimento, por exemplo, se a pessoa é intolerante a um determinado alimento. A detecção e o reconhecimento de alimentos são consideradas tarefas difíceis, pois existem vários tipos diferentes de alimentos e combinações possíveis [Kagaya et al. 2014].

Os resultados obtidos foram inconsistentes quando não foi feito o passo de pré-treinamento com a base Dogs vs. Cats [Kaggle 2004], apresentando baixa acurácia durante o treino. Isso ocorreu porque a quantidade e a variabilidade de amostras da base de imagens de alimentos não foram suficientes para treinar a CNN, o que mostra que o pré-treinamento é um passo necessário. A Fig. 4 apresenta os primeiros resultados obtidos com a utilização do pré-treinamento, onde as caixas de delimitação propostas pela FLODNet são comparadas com aquelas do *ground truth*, mostrando que a FLODNet foi capaz de detectar e gerar as caixas de delimitação para o conjunto de dados usado na fase de teste. Devido ao modelo pré-treinado, foi possível retreinar a rede para o novo conjunto de dados em cerca de 4 horas e alcançar acurácia de 100%.

Como visto na Tabela 1, a Faster R-CNN supera as anteriores em tempo de reconhecimento e, portanto, é a melhor alternativa para comparar com a FLODNet. Sendo assim, a Faster R-CNN [Ren et al. 2015], estado-da-arte, foi escolhida como base de comparação e o seus valores de IoU também foram calculados usando a Equação 3, como apresentado no gráfico da Fig. 5.

No geral, os alimentos com textura mais detalhada e granulada obtiveram resultados de IoU melhores nas duas redes, enquanto que os com textura e contorno mais suaves,

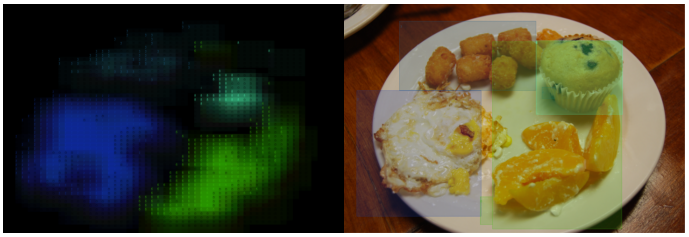

Figura 4. Mapa de calor e caixas de delimitação geradas.

Tabela 1. Comparação entre CNNs

	Arquivo	Tempo Ret.	Tempo Rec.	Acurácia
R-CNN	Variável	-	47302ms	-
Fast R-CNN	477MB	-	2185ms	-
Faster R-CNN	523MB	362 minutos	385ms	100%
FLODNet	95MB	248 minutos	113ms	100%

como por exemplo o bolo, obtiveram resultados piores nos testes.

A classe de alimentos com menor diferença de IoU entre as redes foi o ovo frito, tendo em vista que o contorno irregular e as diferenças de textura entre amostras dificultaram o treinamento na rede mais complexa. Por conta da multiplicidade de instâncias devido à grande variedade de posições, a batata foi a classe de alimentos com maior diferença entre as redes, colocando a Faster R-CNN em vantagem em relação à FLODNet [Ren et al. 2015].

Como apresentado na Tabela 1, em comparação com a Faster R-CNN [Ren et al. 2015], a FLODNet foi suficiente para obter a mesma acurácia (100%), no mesmo banco de dados. A simplicidade da FLODNet se reflete no arquivo de treino que, por conta do menor número de conexões e camadas, fica 5 vezes menor, o que reduz o espaço requrido para execução. Além disso, o tempo necessário para realizar o retreino e o reconhecimento também são menores. Como o reconhecimento é realizado muitas vezes, uma redução de 3 vezes no tempo em relação ao estado-da-arte representa uma grande vantagem para a FLODNet. Esse tempo reduzido também se traduz em menor consumo de energia, aumentando a autonomia quando usado em dispositivos móveis. Portanto, uma métrica que leva em consideração o tempo é mais indicada para a comparação entre as redes. Para essa comparação usamos a Equação 4, gerando o gráfico apresentado na Fig. 5.

Uma vez que a FLODNet utiliza convoluções sempre de mesmo tamanho sobre o *feature map*, ao contrário da Faster R-CNN [Ren et al. 2015], que utiliza janelas de detecção com tamanho e aspecto diferentes, o IoU foi impactado negativamente por conta do tamanho e posicionamento das caixas de delimitação encontradas pela FLODNet. Como mostrado na Fig. 4, as dimensões e o aspecto das caixas de delimitação são uma aproximação do formato ideal. No geral, o índice de IoU entre as caixas de delimitação encontradas pela FLODNet e as verdadeiras são ligeiramente piores do que os alcançados pelo Faster R-CNN [Ren et al. 2015]. Em contrapartida, essas características fazem com que a FLODNet, também com a acurácia em 100%, necessite de um tempo de execução menor, compensando pela perda de exatidão em relação ao posicionamento das caixas de delimitação. O empate na acurácia mostra o ganho de eficiência em relação à CNN mais

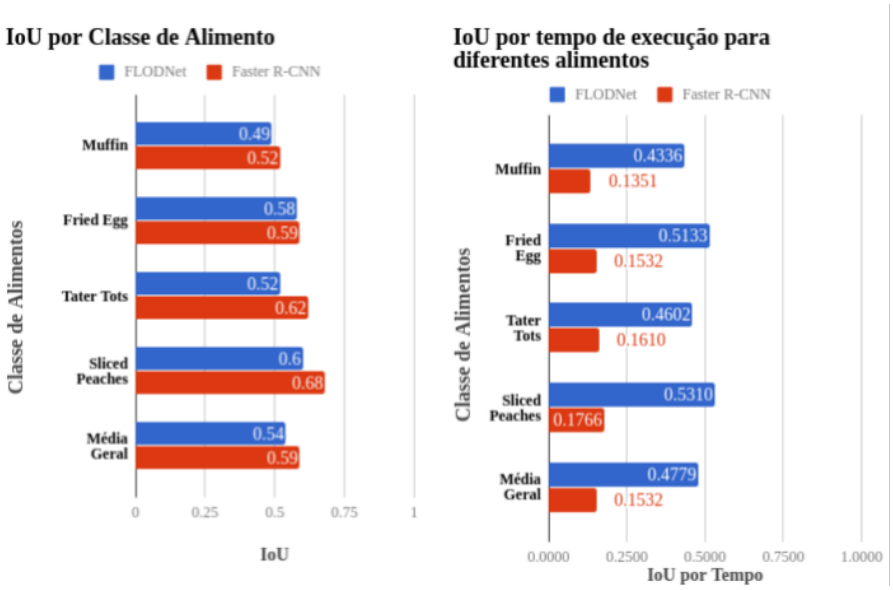

Figura 5. Gráfico apresentando os resultados obtidos nos testes.

profunda.

5. Conclusão

Como apresentado nos resultados, a FLODNet é capaz de identificar os alimentos presentes na imagem, três vezes mais rápido que o estado-da-arte, ambos com 100% de acurácia. Com isso, a FLODNet tem menor custo computacional da avaliação das imagens, reduzindo os requisitos de hardware e o tempo geral de execução, mantendo a qualidade final da classificação. Em contrapartida, a FLODNet pode não ser indicada para situações onde o posicionamento, o aspecto e o tamanho das caixas delimitadoras são críticos. Para aplicações médicas, como análise de tecido tumoral, a necessidade de exatidão no tamanho e posicionamento requer o uso de abordagens mais confiáveis nesse quesito.

Neste artigo foi proposta a métrica IoU_t, que leva em consideração o IoU, a acurácia e o tempo de reconhecimento para medir quantitativamente o desempenho das redes. A FLODNet foi 3,11 vezes melhor que o estado-da-arte considerando essa métrica, mostrando que a diferença de tempo supera a perda de 0,05 na média geral de IoU.

A principal contribuição deste trabalho em relação às técnicas consideradas estado-da-arte no contexto de detecção e reconhecimento de alimentos são o ganho em velocidade de execução, uso reduzido de recursos e espaço requerido, mantendo-se a acurácia de classificação em muitas aplicações, tais como a apresentada neste artigo. Muitas vezes, alimentos em um mesmo prato já possuem uma área de sobreposição, o que anularia a vantagem de uma rede mais complexa. Com essa tarefa, é importante obter a detecção em tempo real e economia de energia em dispositivos de baixa especificação. Alguns desses dispositivos podem depender de baterias ou painéis solares, o que destaca a importância do baixo consumo de energia que surge como consequência do uso reduzido de recursos.

A FLODNet cumpre o objetivo de reduzir o espaço requerido e acelerar o tempo de detecção, reconhecimento e treinamento. Assim como o caso mostrado neste artigo, di-

versas outras aplicações também podem se beneficiar da FLODNet sem serem fortemente impactadas pelo posicionamento das caixas de delimitação. Entre outros, a detecção de placas e a detecção de sinais rodoviários são exemplos de tarefas que não rigorosas com a exatidão no posicionamento das caixas de delimitação, mas se beneficiariam da detecção rápida.

Agradecimentos

Os autores gostariam de agradecer os revisores, que contribuíram para melhorar a versão final do artigo. Além disso, o primeiro autor gostaria de agradecer aos demais autores, seus orientadores, por acreditar na ideia da FLODNet como trabalho de conclusão de curso e orientar no desenvolvimento da proposta, realização dos testes e escrita do artigo. Finalmente, como a ideia do projeto surgiu durante um estágio de verão na Pennsylvania State University, o primeiro autor gostaria também de agradecer à CAPES pela oportunidade de participar do programa de intercâmbio Ciência Sem Fronteiras.

Referências

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., and Zheng, X. (2015). TensorFlow: Large-scale machine learning on heterogeneous systems. Software available from tensorflow.org.

Albers, S. (2010). Energy-efficient algorithms. *Communications of the ACM*, 53(5):86–96.

Alexe, B., Deselaers, T., and Ferrari, V. (2010). What is an object? In *Computer Vision and Pattern Recognition*, pages 73–80.

Alexe, B., Deselaers, T., and Ferrari, V. (2012). Measuring the objectness of image windows. *IEEE transactions on pattern analysis and machine intelligence*, 34(11):2189–2202.

Baldi, P. and Sadowski, P. J. (2013). Understanding dropout. In *Advances in neural information processing systems*, pages 2814–2822.

de Freitas Alves, C. C., Monteiro, G. B. M., Rabello, S., Gasparetto, M. E. R. F., and de Carvalho, K. M. (2009). Assistive technology applied to education of students with visual impairment. *Revista Panamericana de Salud Pública*, 26(2):148–152.

de Oliveira, B. A. G., Ferreira, F. M. F., and d. S. Martins, C. A. P. (2018). Fast and lightweight object detection network: Detection and recognition on resource constrained devices. *IEEE Access*, 6:8714–8724.

de Sande, K. E. V., Uijlings, J. R., Gevers, T., and Smeulders, A. W. (2011). Segmentation as selective search for object recognition. In *IEEE International Conference on Computer Vision (ICCV)*, pages 1879–1886.

Endres, I. and Hoiem, D. (2010). Category independent object proposals. In *European Conference on Computer Vision*, pages 575–588. Springer.

Felzenszwalb, P. F., Girshick, R. B., and McAllester, D. (2010). Cascade object detection with deformable part models. In *Computer vision and pattern recognition*, pages 2241–2248.

Girshick, R. (2015). Fast r-cnn. In *IEEE International Conference on Computer Vision*, pages 1440–1448.

Girshick, R., Donahue, J., Darrell, T., and Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. In *IEEE conference on computer vision and pattern recognition*, pages 580–587.

Hosang, J., Benenson, R., and Schiele, B. (2014). How good are detection proposals, really? In *25th British Machine Vision Conference*, pages 1–12.

Ioffe, S. and Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning*, pages 448–456.

Kagaya, H., Aizawa, K., and Ogawa, M. (2014). Food detection and recognition using convolutional neural network. In *22nd ACM international conference on Multimedia*, pages 1085–1088.

Kaggle (2004). Dogs vs. cats competition. `https://www.kaggle.com/c/dogs-vs-cats/data`. (Acessado em 06/02/2017).

Krähenbühl, P. and Koltun, V. (2014). Geodesic object proposals. In *European Conference on Computer Vision*, pages 725–739.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105.

Oquab, M., Bottou, L., Laptev, I., and Sivic, J. (2014). Learning and transferring mid-level image representations using convolutional neural networks. In *IEEE conference on computer vision and pattern recognition*, pages 1717–1724.

Ren, S., He, K., Girshick, R., and Sun, J. (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99.

Russell, B. C., Torralba, A., Murphy, K. P., and Freeman, W. T. (2008). Labelme: a database and web-based tool for image annotation. *International journal of computer vision*, 77(1-3):157–173.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958.

Zhang, Z., Warrell, J., and Torr, P. H. (2011). Proposal generation for object detection using cascaded ranking svms. In *Computer Vision and Pattern Recognition*, pages 1497–1504.