Classificação de imagens pela qualidade estética utilizando o WEKA

Sofia M. Moraes[1]

[1]Instituto de Informática – Universidade Federal de Goiás (UFG)
Goiânia – GO – Brazil

sofiamoraesh@gmail.com

Abstract. *Using the WEKA Data Mining software, a model was found that could predict whether an image has a low or high aesthetic quality. The model reached an accuracy rate of 67.75% using four filters from the imageFilters package ('ColorLayoutFilter', 'FCTHFilter', 'JpegCoefficientFilter' and 'PHOGFilter') and the 'RandomForest' Decision Tree Classifier.*

Resumo. *Por meio do software para Data Mining WEKA, foi encontrado um modelo capaz de predizer se uma imagem possui uma baixa (LOW) ou alta (HIGH) qualidade estética. O modelo atingiu uma taxa de acurácia de 67.75% utilizando quatro filtros do pacote imageFilters ('ColorLayoutFilter', 'FCTHFilter', 'JpegCoefficientFilter' e 'PHOGFilter') e o classificador baseado em árvore de decisão 'RandomForest'.*

1. Introdução

A classificação de imagens é um dos problemas mais conhecidos da ciência de dados e vem sendo explorado desde 1998 com o reconhecedor para dígitos manuscritos chamado LeNet [LeCun et al. 1998]. A arquitetura proposta foi evoluindo e foi consolidada em 2012 com a AlexNet [Romero et al. 2015], vencedora do desafio ImageNet [Deng et al. 2009]. Esse estilo de arquitetura é nomeado de Rede Neural Convolucional.

As Redes Neurais Convolucionais (ConvNets ou CNNs) são redes neurais artificiais profundas que podem ser usadas para classificar imagens, agrupá-las por similaridade (busca de fotos) e realizar reconhecimento de padrões e objetos [Data Science Academy 2018]. São similares às redes neurais tradicionais, com pesos e bias que precisam ser treinados, porém possuem singularidades em sua arquitetura que permitem o reconhecimento de padrões em imagens [Karpathy 2016].

Nesse contexto, se encontra a classificação de imagens pela sua qualidade estética. Com a alta variedade de instrumentos ópticos para captação de imagens, desde câmeras de smartphones até câmeras profissionais, a qualidade de uma imagem pode variar muito. Dessa forma, a classificação é vantajosa tanto para sites de avaliação de estabelecimentos comerciais, como o Yelp e o TripAdvisor, que recebem fotos fornecidas por seus usuários, quanto para os próprios estabelecimentos que as fotos mais atrativas sejam selecionadas. Outra aplicação se dá no desenvolvimento de recursos de softwares de fotografia com mais precisão, para fornecer ajustes automáticos de exposição, contraste e cores nas fotos, criar modos diferentes de fotografia ou até melhorar o pós-processamento das câmeras de smartphones.

A classificação de imagens pela sua qualidade estética possui suas peculiaridades. O atributo "beleza" é subjetivo, e pode variar de pessoa pra pessoa. É mais fácil dizer que um carro é um carro, do que dizer se um carro é bonito ou não. Propostas anteriores apresentam diferentes métricas para e avaliar se a qualidade estética da imagem é alta ou baixa.

Este estudo tem como objetivo classificar da melhor forma a qualidade estética de imagens utilizando dois datasets, um de fotos de animais e outro de prédios (paisagens de cidades). Para isso, será utilizado o software open source WEKA [Hall et al. 2009], desenvolvido na Universidade de Waikato, Nova Zelândia, com a utilização do pacote imageFilters [Mayo 2016]. Serão avaliados os dez filtros disponíveis a fim de encontrar a melhor alternativa para a solução do problema.

2. Trabalhos relacionados

2.1. Yelp Restaurant Photo Classification

O site Yelp para avaliação de estabelecimentos comerciais recebe uma média de cem mil fotos de usuários por dia para os diferentes restaurantes cadastrados no site. Dessa forma, era necessário uma forma de avaliar quais fotos deveriam ficar em destaque na página [Alex M. 2016]. Para isso, foi utilizada uma rede neural convolucional para classificar imagens por sua qualidade estética. O objetivo era selecionar as fotos com melhor qualidade para que elas fossem colocadas em destaque nas páginas dos restaurantes.

Foram utilizados como dados de treinamento metadados de fotos conhecidos como EXIF data. Exchangeable image file format (EXIF) é uma especificação seguida por fabricantes de câmeras digitais que que gravam informações sobre as condições técnicas de captura da imagem junto ao arquivo da imagem propriamente dita na forma de metadados etiquetados. Dados como orientação, compressão, resolução, tempo de exposição são anexados à imagem.

2.2. Google NIMA

Para capturar a subjetividade da avaliação estética de imagens, indo além da detecção de ruídos, borrões e distorções, foi proposta a alternativa NIMA: Neural Image Assessment [Esfandarani and Milanfar 2017], uma CNN treinada para predizer como um usuário avaliaria uma imagem de forma técnica e estética, dando para uma imagem uma nota de 1 a 10. Seu maior trunfo é a alta correlação entre as notas preditas e as reais dadas por humanos (ground truth).

Executando o algoritmo com a base de dados AVA (Aesthetic Visual Analysis) [Murray et al. 2012], onde cada foto foi avaliada por uma média de 200 pessoas, a classificação estética dessas fotos pelo NIMA corresponde de perto às pontuações médias dadas pelos avaliadores humanos. Além disso, as notas dadas pelo NIMA podem ser usadas para avaliar uma mesa imagem, sendo distorcida de várias maneiras.

2.3. Food Image Aesthetic Quality Measurement by Distribution Prediction

Para avaliar a qualidade estética de imagens de comida, pesquisadores da Universidade de Stanford propuseram um modelo [Lou and Yang] baseado no NIMA 2.2 com o objetivo de servir como uma alternativa melhor à proposta do Yelp 2.1. Fazendo testes com inputs de imagens gerais e específicos de fotos de comida, utilizando fotos de comida para

validação, Intenção de explorar se a especificidade das fotos interfere ou não no resultado. Imagens foram classificadas como “High” para alta qualidade ou “Low” para baixa qualidade, com uma distribuição estimada de quantos % das pessoas votariam de 1 a 10 se solicitadas para classificar a imagem. Os resultados foram satisfatórios, com o modelo atingindo 72% de acurácia quanto à classificação da foto (Figura 04). Entretanto, o treinamento utilizando fotos específicas de comida não apresentaram resultados tão bom quanto esperado, concluindo então que a qualidade estética das fotos é um atributo mais generalizado.

3. Experimento e Resultados

3.1. Datasets

Neste estudo, foram utilizados dois datasets, um contendo fotos de animais e outro contendo fotos de prédios/paisagens urbanas. O objetivo é analisar grupos de imagens com características diferentes, como por exemplo: foco em um ponto específico ou em uma paisagem, ambientes naturais ou ambientes urbanos etc. Através disso, faz-se uma observação, analisando se a especificidade das imagens influencia na sua classificação pela estética.

Os datasets utilizados foram disponibilizados gratuitamente pela Figure Eight Inc. [Figure Eight Inc. 2014]. São eles: “How beautiful is this image? (Part 3: Animals)” e “How beautiful is this image? (Part 2: Buildings and Architecture)”. Em ambos os datasets, colaboradoes avaliaram as imagens fornecidas uma escala de cinco pontos, de “inaceitável” (fotos borradas, distorcidas etc.) a “excepcional” (fotos de alta resolução que poderiam ser usadas profissionalmente). O conjunto de dados inclui uma URL para cada imagem, uma pontuação média (de 1 a 5) para a qualidade da imagem.

As imagens selecionadas para cada etapa do estudo foram separadas em uma planilha, sendo classificadas em duas categorias: baixa e alta qualidade. Para isso, as imagens receberam a tag "LOW", caso sua nota média seja menor que 3, sendo portanto considerada de baixa qualidade, ou a tag "HIGH", caso sua nota média exceda 3, sendo considerada então como de alta qualidade. A classificação pode ser melhor entendida abaixo, na Tabela 1.

Tabela 1. Classificação das imagens conforme sua nota

	"LOW"(baixa qualidade)	"HIGH"(Alta qualidade)
Nota(N)	N <3	N >= 3
Obs.: Os valores das notas (N) variam entre 1 e 5		

As imagens selecionadas então, foram obtidas através de suas URLs, por meio de um script em python, utilizando as bibliotecas ’csv’ e ’urllib.request’ e separadas em um diretórios. Após o download, no diretório onde se encontram as imagens, foi utilizado o comando ’convert’no terminal do linux para redimensionar as imagens para uma largura (width) máxima de 256 pixels. A padronização tem o objetivo de otimizar o processo posterior de análise pelo WEKA. Exemplos das imagens que serão utilizadas de input podem ser encontrados na Figura 1.

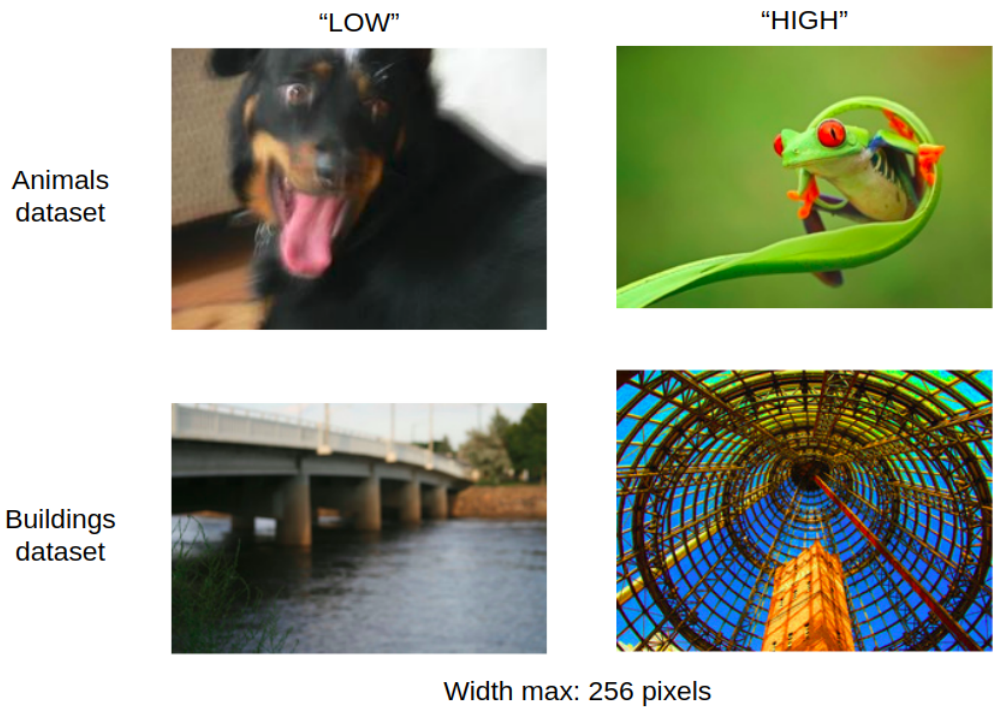

Figura 1. Exemplares dos datasets

4. Experimentos e Resultados

Para realizar as três etapas do experimento, será usado o pacote *imageFilters*, que pode ser obtido no próprio WEKA, através do Package Manager. Este pacote possui 10 filtros e precisa de um ARFF file (Attribute-Relation File Format - padrão do WEKA) com as instâncias e atributos a serem trabalhados. Neste caso, o primeiro atributo será o nome do arquivo de imagem (filename string) e o segundo, a classe em que se encaixa (class LOW,HIGH). Os filtros que podem ser encontrados no pacote estão listados abaixo, com suas descrições originais em inglês e um breve resumo em portugês.

AutoColorCorrelogramFilter is a batch filter for calculating a color correlogram from an image. A color correlogram encodes the spatial correlation of colors in an image, and is an effective feature that is robust to changes in viewing position and camera zoom.
Resumo: Calcula o correlograma das cores de uma imagem.

BinaryPatternsPyramidFilter is a batch filter for extracting a pyramid of rotation-invariant local binary pattern histograms from images. Each local binary pattern represents an intensity pattern (e.g. an edge or a corner) around a point. A histogram of local binary patterns therefore encodes the larger scale patterns that occur across regions of images. Local binary patterns are useful for texture and face recognition.
Resumo: Extrai histogramas para Local Binary Patterns de uma imagem. É uma forma eficiente e simples de descrever textura.

ColorLayoutFilter is a batch filter for extracting MPEG7 color layout features from images. This filter divides an image into 64 blocks and computes the average color for each block, and then features are calculated from the averages.

Resumo: Extrai o layout espacial das cores através do padrão MPEG-7.

EdgeHistogramFilter is a batch filter for extracting MPEG7 edge histogram features from images. Edges are the lines or discontinuities in an image. An edge histrogram is therefore a summary of the directions that the edges are going in across an image.

Resumo: Extrai histogramas de bordas para imagens através do padrão MPEG-7. Representa a distribuição espacial das bordas.

FCTHFilter is a batch filter for extracting FCTH color features from images. FCTH stands for 'Fuzzy Color and Texture Histogram', and as the name suggests, these features encode both color and texture information in one histogram. One bonus of this feature is that it is very small – limited to 72 bytes per image – and therefore suitable for large image datasets.

Resumo: Extrai o histograma FCTH, que codifica informações de cor e textura de uma imagem. Bastante eficiente para grandes datasets.

FuzzyOpponentHistogramFilter is a batch filter for extracting a fuzzy 64-bin histogram, based on the Opponent color space, from images.

Resumo: Extrai um histograma das imagens baseados em "Opponent color space"

GaborFilter is a batch filter that uses a Gabor wavelet to extract texture features from images. Gabor filters are very common in computer vision, and the features should be invariant to rotation.

Resumo: Extrai informações de textura das imagens baseado em "Garbor wavelet"

JpegCoefficientFilter is a batch filter for extracting JPEG coefficients from images. Converting an image to the JPEG file format discards information that it imperceptible to humans and in the process produces a sequences of quantized coefficients that are part of the compressed representation of the image. These coefficients are the features computed by this filter.

Resumo: Extrai coeficientes JPEG das imagens.

PHOGFilter is a batch filter for extracting PHOG features from images.PHOG stands for 'Pyramid Histogram of Oriented Gradients', and as the name suggests, it encodes information about the orientation of intensity gradients across an image.

Resumo: Extrai PHOG histogramas das imagens. Codifica informações sobre a orientação dos gradientes de intensidade em uma imagem..

SimpleColorHistogramFilter is a batch filter for extracting color histogram feature from images. This is the most basic color feature that can be computed; essentially, it three histograms (one for red, one for green, one for blue) each of which has 32 bins. Each bin contains a count of the pixels in the image that fall into that bin.

Resumo: Extrai um histograma sobre as cores de uma imagem. É a forma mais simples das informações sobre cores serem computadas.

4.1. Primeira etapa

Escolhendo de forma arbitrária um filtro para ser a base do experimento, avaliou-se pares de filtros (o escolhido mais outro da lista) para avaliar se os resultados de classificação melhoraram ou piorarm. Dessa forma, o experimento possui 10 testes, um onde a classificação foi feita somente com o filtro base, e outros 9 com as duplas de filtros. Para a

classificação, será escolhido também de forma arbitrária um classificador baseado em árvore de decisão, já que classificadores em árvore são muito mais rápidos em treinamento e testes do que classificadores tradicionais (como um SVM) [Bosch et al. 2007]. O filtro base escolhido foi o 'ColorLayoutFilter', o classificador escolhido foi o 'J48'. Para realizar a classificação, o atributo "filename"deve ser removido após a aplicação do(s) filtro(s).

Para realizar a classificação, foram organizados 3 diferentes inputs de dados: um contendo 64 imagens do dataset de animais, outro contendo 64 imagens só do dataset de prédios e um com 128 imagens correspondente à união dos dois anteriores. Todos os inputs são equilibrados, sendo 50% imagens LOW quality e 50% HIGH quality. As proporções para classificação são: 70% para treinamento, e 30% para validação. Os resultados obtidos podem ser encontrados nas Tabelas 2 3 e 4 respectivamente. As porcentagens apresentadas estarão em vermelho se o filtro piorou a classificação, em verde se melhorou ou preto se não mudaram se comparado à baseline.

Tabela 2. Classificação do dataset com 64 imagens de animais

Ordem do teste	**FIltro(s)**	**% de classificações corretas**
1º	**ColorLayoutFilter (CLF)**	**42.1% (baseline)**
2º	CLF + EdgeHistogramfilter	36.8%
3º	CLF + FCTHFilter	47.3%
4º	CLF + FuzzyOpponentHistogramFilter	31.5%
5º	CLF + GaborFilter	42.1%
6º	CLF + JpegCoefficientFilter	63.1%
7º	CLF + PHOGFilter	26.3%
8º	CLF + SimpleColorHistogramFilter	36.8%
9º	CLF + AutoColorCorrelogramFilter	42.1%
10º	CLF + BinaryPatternsPyramidFilter	68.4%

Tabela 3. Classificação do dataset com 64 imagens de prédios

Ordem do teste	**FIltro(s)**	**% de classificações corretas**
1º	**ColorLayoutFilter (CLF)**	**57.8% (baseline)**
2º	CLF + EdgeHistogramfilter	52.6%
3º	CLF + FCTHFilter	73.6%
4º	CLF + FuzzyOpponentHistogramFilter	57.8%
5º	CLF + GaborFilter	57.8%
6º	CLF + JpegCoefficientFilter	73.6%
7º	CLF + PHOGFilter	36.8%
8º	CLF + SimpleColorHistogramFilter	63.1%
9º	CLF + AutoColorCorrelogramFilter	57.8%
10º	CLF + BinaryPatternsPyramidFilter	73.6%

Tabela 4. Classificação do dataset com 128 imagens, união dos dois inputs anteriores

Ordem do teste	FIltro(s)	% de classificações corretas
1º	**ColorLayoutFilter (CLF)**	**50% (baseline)**
2º	CLF + EdgeHistogramfilter	57.8%
3º	CLF + FCTHFilter	63.1%
4º	CLF + FuzzyOpponentHistogramFilter	63.1%
5º	CLF + GaborFilter	50%
6º	CLF + JpegCoefficientFilter	76.3%
7º	CLF + PHOGFilter	68.4%
8º	CLF + SimpleColorHistogramFilter	52.6%
9º	CLF + AutoColorCorrelogramFilter	63.1%
10º	CLF + BinaryPatternsPyramidFilter	84.2%

4.1.1. Resultados parciais (primeira etapa)

A medida em que o input de dados aumenta, os filtros parecem apresentar maiores resultados positivos, com exceção do 'GaborFilter' que não apresentou nenhuma influência e será descartado para a segunda etapa do experimento. Os resultados parecem congruentes entre si, com exceção ao 'PHOGFilter', que apresentou uma piora com os testes individuais, mas melhorou o resultado obtido quando os dois datasets foram unificados, ou seja, o filtro só parece ser eficiente para datasets maiores, portanto será mantido junto com os outros para a próxima etapa. As altas taxas de acurácia obtidas na terceira classificação com 128 imagens são bem singulares, havendo probabilidades ínfimas de repetição em inputs maiores. A especificidade dos datasets também não pareceu influenciar de forma significativa a classificação das imagens. Uma visão geral dos resultados obtidos pode ser encontrada nas Tabelas 5 e 6 a seguir:

Tabela 5. Resultados gerais com inputs individuais de animais e prédios

	Teste 1 (animais) [64 imagens]	Teste 2 (prédios) [64 imagens]
Melhora	ColorLayoutFilter FCTHFilter JpegCoefficientFilter BinaryPatternsPyramidFilter	ColorLayoutFilter FCTHFilter JpegCoefficientFilter SimpleColorHistogramFilter BinaryPatternsPyramidFilter
Indiferente	GaborFilter AutoColorCorrelogramFilter	FuzzyOpponentHistogramFilter GaborFilter AutoColorCorrelogramFilter
Piora	FuzzyOpponentHistogramFilter EdgeHistogramfilter PHOGFilter SimpleColorHistogramFilter	EdgeHistogramfilter PHOGFilter

Tabela 6. Resultados gerais com input misto (animais + prédios)

	Teste 3 (animais + prédios) [128 imagens]
Melhora	ColorLayoutFilter EdgeHistogramfilter FCTHFilter FuzzyOpponentHistogramFilter JpegCoefficientFilter PHOGFilter SimpleColorHistogramFilter AutoColorCorrelogramFilter BinaryPatternsPyramidFilter
Indiferente	GaborFilter
Piora	-

4.2. Segunda etapa

Para a segunda etapa do experimento, serão mantidos todos os filtros que passaram na primeira parte do experimento. Entretanto, abordagem para a avaliação dos filtros é diferente. O princípio é adicionar os filtros um a um e efetuar a classificação. Se a partir do segundo input o filtro não apresentar uma melhora na classificação, ele será retirado. Caso o filtro aumente a porcentagem de classificações corretas, ele será mantido. Será seguida a mesma sequência de aplicação de filtros da primeira etapa do experimento.

Para realizar a classificação, foi organizado um dataset equilibrado com 1024 imagens, sendo 512 do dataset de animais, sendo metade destes a tag "LOW"e a outra metade com a tag "HIGH"e 512 do dataset de prédios, seguindo a mesma proporção de imagens com baixa e alta qualidade. O primeiro filtro a ser testado foi o 'ColorLayoutFilter'. As proporções para classificação são: 70% para treinamento, e 30% para validação. O classificador escolhido foi o 'J48'. Os resultados obtidos podem ser encontrados na Tabela 7. As porcentagens apresentadas estarão em vermelho se o filtro piorou a classificação ou em verde se jouve melhora em comparação à baseline anterior. Novas baselines são geradas a medida que a combinação de filtros atinge resultados melhores.

Tabela 7. Classificação do dataset com 1024 imagens. Input equilibrado com fotos dos datasets de animais e prédios

Ordem do teste	Filtro(s)	Classificações corretas (%)	Avaliação
1º	ColorLayoutFilter	**57.6% (baseline 0)**	-
2º	ColorLayoutFilter, EdgeHistogramFilter	54.3%	Filtro será descartado
3º	ColorLayoutFilter, FCTHFilter	58.3%	Filtro será mantido
4º	ColorLayoutFilter, FCTHFilter, FuzzyOpponentHistogramFilter	56.0%	Filtro será descartado
5º	ColorLayoutFilter, FCTHFilter, JpegCoefficientFilter	**60.2% (baseline 1)**	Filtro será mantido
6º	ColorLayoutFilter, FCTHFilter, JpegCoefficientFilter, PHOGFilter	**61.5% (baseline 2)**	Filtro será mantido
7º	ColorLayoutFilter, FCTHFilter, JpegCoefficientFilter, PHOGFilter, SimpleColorHistogramFilter	57.9%	Filtro será descartado
8º	ColorLayoutFilter, FCTHFilter, JpegCoefficientFilter, PHOGFilter, AutoColorCorrelogramFilter	57.9%	Filtro será descartado
9º	ColorLayoutFilter, FCTHFilter, JpegCoefficientFilter, PHOGFilter, BinaryPatternsPyramidFilter	57.3%	Filtro será descartado

4.2.1. Resultados parciais (segunda etapa)

Através da segunda etapa do experimento, o conjunto de filtros que melhor serviu ao problema e apresentou a melhor porcentagem de classificações corretas foram: 'ColorLayoutFilter', 'FCTHFilter', 'JpegCoefficientFilter' e 'PHOGFilter' com 61.56% de classificações corretas utilizando o classificador 'J48'. Esses são os filtros que passam para a terceira etapa do estudo.

4.3. Terceira etapa

Os filtros que passaram na segunda etapa serão utilizados na terceira etapa do estudo. O objetivo é encontrar o melhor classificador em árvore para o problema. Serão testados 7 classificadores que aparecem na aba weka/classifiers/trees. Os resultados obtidos podem ser observados na Tabela 8 abaixo.

Tabela 8. Classificação do dataset com 1024 imagens. Classificadores e seus resultados

Ordem do teste	Classificador	Classificações corretas (%)
1º	J48	65.56%
2º	LMT	64.49%
3º	**RandomForest**	**67.75% (melhor resultado)**
4º	RandomTree	56.2%
5º	REPTree	55.37%
6º	DecisionStump	59.60%
7º	HoeffdingTree	63.84%

4.4. Resultados

Concluído o experimento, o melhor modelo encontrado para classificar imagens pela sua qualidade estética utilizando o WEKA e o pacote imageFilters foram:

Filtros: 'ColorLayoutFilter', 'FCTHFilter', 'JpegCoefficientFilter' e 'PHOGFilter'.
Classificador: 'RandomForest'.

Com esta configuração foi obtido um total de 67.75% de classificações corretas no dataset misto de 1024 imagens, utilizando 70% dos dados para treinamento e 30% para validação. A matriz de confusão (Tabela 9) para esta classificação apresentou uma pequena quantidade a mais de falsos positivos para "HIGH", mostrando um maior acertos na identificação de imagens de baixa qualidade. Para avaliar a performace do modelo, também foi feita a validação cruzada (Cross Validation) com 10 Folds, a porcentagem de classificações corretas obtida foi de 66.99%.

Tabela 9. Matriz de confusão para a classificação

LOW	HIGH	<– classified as
105	42	LOW
57	103	HIGH

Todos os buffers de resultados gerados pelo WEKA, assim como os datasets e inputs utilizados e os resultados obtidos podem ser encontrados no seguinte repositório: https://github.com/Moraesofia/topicos-2018.

5. Conclusão

O estudo apresentou resultados satisfatórios, porém não ideais. Classificação binária para um atributo tão subjetivo como a qualidade estética de uma imagem não é a melhor alternativa a ser utilizada. Algo como a abordagem NIMA 2.2, que gera para uma imagem uma nota em uma escala de 1 a 10 pode trazer mais precisão quanto à qualidade estética da imagem. Entretanto, conclui-se que o WEKA (e seu pacote imageFilters) é capaz de resolver problemas de classificação de imagens. Trabalhos futuros podem incluir o teste do modelo encontrado em datasets de imagens com características diferentes e em maiores quantidades. Outro aspecto que pode ser estudado é a análise dos filtros com uma abordagem mais técnica, considerando suas funções e não a partir somente da percepção. Dessa forma seria possível investigar, por exemplo, o comportamento do filtro 'PHOGFilter', que só se mostrou eficaz a partir de inputs maiores.

Referências

Alex M. (2016). Finding beautiful yelp photos using deep learning. https://engineeringblog.yelp.com/2016/11/finding-beautiful-yelp-photos-using-deep-learning.html. Accessed: 2018-12-05.

Bosch, A., Zisserman, A., and Munoz, X. (2007). Image classification using random forests and ferns. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pages 1–8. IEEE.

Data Science Academy (2018). Capítulo 10 – as 10 principais arquiteturas de redes neurais. http://deeplearningbook.com.br/as-10-principais-arquiteturas-de-redes-neurais/. Accessed: 2018-12-05.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 248–255. Ieee.

Esfandarani, H. T. and Milanfar, P. (2017). NIMA: neural image assessment. *CoRR*, abs/1709.05424.

Figure Eight Inc. (2014). Data for everyone. https://www.figure-eight.com/data-for-everyone/. Accessed: 2018-12-05.

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., and Witten, I. H. (2009). The weka data mining software: An update. *SIGKDD Explor. Newsl.*, 11(1):10–18.

Karpathy, A. (2016). Cs231n convolutional neural networks for visual recognition. *Neural networks*, 1.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.

Lou, J. and Yang, H. Food image aesthetic quality measurement by distribution prediction.

Mayo, M. (2016). Imagefilter - weka filter that uses lire to extract image features. Github repository `https://github.com/mmayo888/ImageFilter`. Accessed: 2018-12-05.

Murray, N., Marchesotti, L., and Perronnin, F. (2012). Ava: A large-scale database for aesthetic visual analysis. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 2408–2415. IEEE.

Romero, A., Ballas, N., Kahou, S., Chassang, A., Gatta, C., and Bengio, Y. (2015). Imagenet classification with deep convolutional neural networks. In *International Conference on Learning Representations*.