DSP Feature Analysis with CIFAR-10

Sergio Pallas Enguita
Department of Electrical and Computer Engineering
Old Dominion University
Norfolk, USA
Spall009@odu.edu

***Abstract*—This article explores the application of digital signal processing (DSP) techniques for extracting meaningful features from digital images, focusing on color analysis, shape analysis, and edge detection. Utilizing MATLAB and Google Colab environments for practical implementation, this study demonstrates how DSP methods can significantly enhance the processing and analysis of digital images for machine learning (ml) applications.**

***Index Terms*—Machine Learning, VGG16, InceptionV3, weights, bias, Digital Sinal Processing, Feature Selection, Signals, Filtering, Gradient, transform, CIFAR-10, Computer Vision.**

I. Introduction

Digital Signal Processing (DSP) stands as a powerful tool in the realm of machine learning (ML), offering a sophisticated means to analyze, modify, and synthesize signals. The application of DSP in feature selection for ML algorithms is pivotal, transcending the traditional boundaries of image processing to influence a wide array of fields such as audio analysis, telecommunications, healthcare, and financial systems. This integration plays a critical role in enhancing the efficiency and accuracy of ML models by ensuring that only the most relevant, informative features are used for training.

The essence of DSP in feature selection lies in its ability to transform raw data into a format that is more suitable for ML algorithms. Through techniques such as filtering, Fourier transforms, and wavelet transforms, DSP can highlight important characteristics of the data while discarding noise and redundant information. This preprocessing step is crucial, as it directly impacts the learning algorithm's ability to generalize from the training data to unseen data, thereby affecting the overall model performance. [1]

In audio signal processing, for instance, DSP techniques can extract features that capture the essence of sound, such as pitch, tempo, and timbre, which are invaluable for applications like speech recognition, music classification, and emotion detection. In telecommunications, DSP enables the extraction of features from signal modulations, aiding in signal classification and network optimization. The healthcare sector benefits from DSP through the enhanced analysis of biomedical signals (e.g., ECG, EEG signals), facilitating accurate disease diagnosis and patient monitoring. Similarly, in financial systems, DSP techniques can help identify patterns in time-series data, leading to more informed decision-making in algorithmic trading and risk management. [2] [3]

Moreover, the push towards real-time processing and the need for models to operate under resource constraints (e.g., in IoT devices) is shaping the way DSP and feature selection are approached. Edge computing and the development of lightweight DSP algorithms are becoming increasingly important, enabling the deployment of powerful ML models in environments where computational resources are limited. [4]

This project intends to demonstrate the functioning of some of these DSP methods and how they can be implemented using platforms both like Matlab and GoogleColab.

II. Background and Dataset

A. CIFAR-10 DataSet

The CIFAR-10 dataset consists of 60,000 32x32 color images divided into 10 classses, with 6,000 images per class. This variety provides a dataset that is complex enough to challenge ML models without being as overwhelming as larger datasets like ImageNet. CIFAR-10 has also been widely recognized and used throughout the DSP and ML communities which allows us to compare the results and methods found in this project. The images in this dataset are also small enough for it not be computationally demanding but big enough for filters and features to be separable from the base image. [5]

Overall CIFAR-10 is a very versatile dataset that has been widely tested throughout the community allowing us to have a solid base on which to base our project.

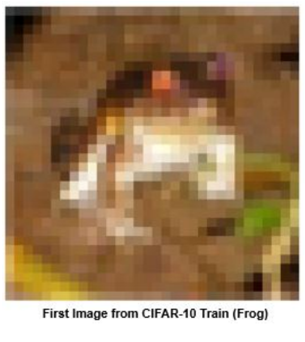


Fig. 1. First Image of CIFAR-10 training batch [6]

B. Feature Selection Techniques

Feature selection serves as a bridge between raw data and ml models. It involves identifying and selecting the most useful features (i.e., input variables) that contribute to the predictive accuracy of the model, while removing the irrelevant data. This process not only enhances the performance of ml models but also improves interpretability and efficiency. Understanding how feature selection works and its impact on ml models is fundamental to understand the Machine Learning and Data Science fields.

1) Color Analysis: Color Analysis involves the examination and manipulation of color components in digital images to extract meaningful information. Images are typically represented in color spaces such as RGB (Red, Green, Blue), HSV (Hue, Saturation, Value), or others, where each pixel's color is encoded as a combination of basic components. Some common color analysis techniques are color space transformation, histogram equalization, and filtering enhance specific colors or to separate objects based on color. In this case we will be using color histograms to perform color analysis. This consists on taking a histogram representation of the color distribution in an image. It is a graphical representation of the tonal values of an image, in this case there are three separate histograms (one for each color value RGB). [7]

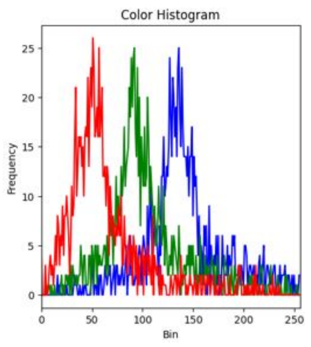


Fig. 2. Color Histogram for Frog [6]

2) Edge Detection Analysis: Edge Analysis focuses on identifying points in an image where the brightness changes sharply or, more formally, has discontinuities. These points typically correspond to object boundaries, texture changes or other significant transitions. DSP-based edge detection filter, such as Sobel, Prewitt and Canny detectors, operate by convolving the image with an operator that is designed to highlight spatial frequency components that correspond to these edges. [8] In this case we will use the Canny edge detection algorithm, which is a robust and widely-used method for identifying edges in digital images. The algorithm uses a Gaussian filter to smooth the image and computes the gradient in magnitude for each pixel. It then uses different thresholds to separate the pixels into different groups that allow it to separate the sharp edges from the rest of the images, this edges are then used to train the model. [9]

3) Shape Analysis: Shape Analysis involves extracting shape features from objects to classify, compare or track these objects. Some common techniques are contour detection, morphological operations (such as dilation, erosion, opening and closing) and the extraction of shape descriptors (such as Fourier descriptors, moment invariants and curvature scale space features). These DSP techniques helps in transforming the visual representation of shapes into a form that can be

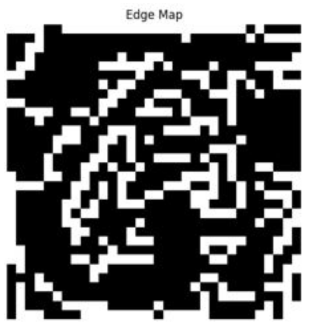


Fig. 3. Canny Edge Analysis for Frog [6]

efficiently analyzed and processed by ml algorithms. To implement this method we will convert the images from color space RGB to a gray-scale space and then apply a findContours() method upon that image. This will find curves or figures that share a similar shade or intensity in the gray-space. These is usually most useful in eliminating the background of an image. This method is very popular when the objective is to detect caligraphy or specific figures. It uses the already done Canny Edge detection and groups the edges into shapes or contours that are later passed on to the model. It is easy to think about shape analysis as a step further from edge analysis. [10]

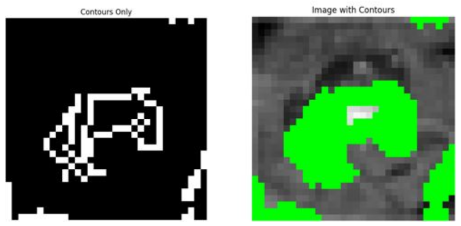


Fig. 4. Contour/Shape Analysis for Frog [6]

C. Machine Learning Algorithm

When choosing a ml model there were several factors into consideration. The model has to be simple enough to not have a high computation cost but complex enough to learn with the given dataset. It would also need to be implemented in Google Colab and/or Matlab, and it would need to be able to take the created feature selections as input. [11]

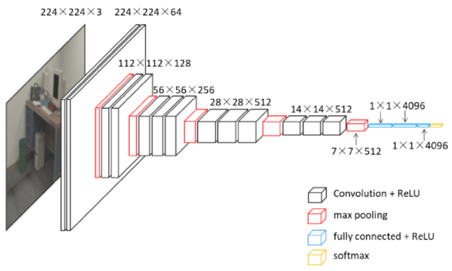


Fig. 5. VGG16 Model Architecture [11]

As a result a pre-trained version of VGG16 was used, this is also easy to implement because the ml algorithm has been widely uesd and is already prebuilt in google colab. For the base run the pretrained version of this algorithm was used without any modification done to it.

Afterwards the same model was used with a modification to the last layers, where the layers were combined with the new feature selections that have been made as an input. This layer is then the one that connects to the end of the model that predicts the class of each image.

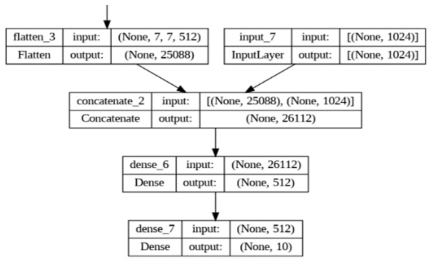

flatten_3 | input: | (None, 7, 7, 512)
Flatten | output: | (None, 25088)
input_7 | input: | [(None, 1024)]
InputLayer | output: | [(None, 1024)]
concatenate_2 | input: | [(None, 25088), (None, 1024)]
Concatenate | output: | (None, 26112)
dense_6 | input: | (None, 26112)
Dense | output: | (None, 512)
dense_7 | input: | (None, 512)
Dense | output: | (None, 10)

Fig. 6. VGG16 and Feature Selection Architecture [6]

This architecture allows us to use the pre-trained model with the addition of the results from the feature selection. The combination is used at the end because we are using a model that has already learned previously. In summary the feature selections are being used as the input for an already trained model, the results could vary if the model was trained from scratch.

Additionally InceptionV3 was implemented through GoogleColab again with pre-trained learning. The objective of this was to compare the results with VGG16 and give the results more credibility. The model was implemented using google colab (for computational cost reasons) but could be implemented in Matlab as well using the same code provided for VGG16.

InceptionV3 is similar to VGG16 because they are both Convolutional Neural Networks but differs in the architecture as instead of performing a compete convolution it instead performs several smaller convolution and then uses a concatenation of these to learn from the images.

Due to the fact that both of these models are based on Convolution it is expected that they will do better with Edge and Shape features but it is unclear how they will hande a color histogram input.

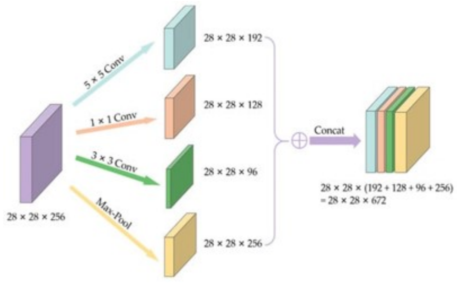
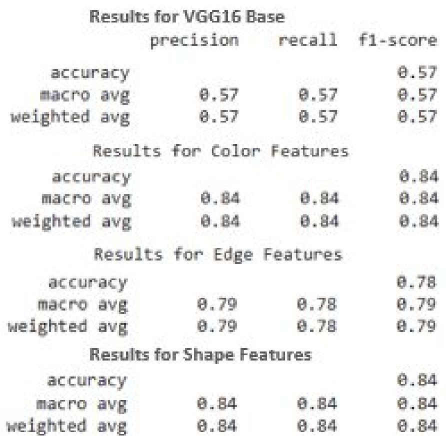


Fig. 7. InceptionV3 Architecture [12]

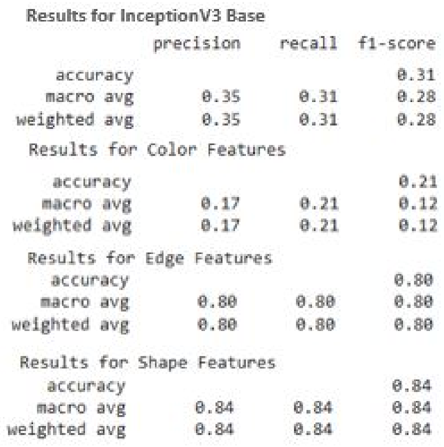


```
Results for VGG16 Base
              precision    recall  f1-score

    accuracy                           0.57
   macro avg       0.57      0.57      0.57
weighted avg       0.57      0.57      0.57

          Results for Color Features
    accuracy                           0.84
   macro avg       0.84      0.84      0.84
weighted avg       0.84      0.84      0.84
          Results for Edge Features
    accuracy                           0.78
   macro avg       0.79      0.78      0.79
weighted avg       0.79      0.78      0.79
          Results for Shape Features
    accuracy                           0.84
   macro avg       0.84      0.84      0.84
weighted avg       0.84      0.84      0.84
```

Fig. 8. Results for VGG16 Model [6]



```
Results for InceptionV3 Base
              precision    recall  f1-score

    accuracy                           0.31
   macro avg       0.35      0.31      0.28
weighted avg       0.35      0.31      0.28
 Results for Color Features
    accuracy                           0.21
   macro avg       0.17      0.21      0.12
weighted avg       0.17      0.21      0.12

 Results for Edge Features
    accuracy                           0.80
   macro avg       0.80      0.80      0.80
weighted avg       0.80      0.80      0.80

 Results for Shape Features
    accuracy                           0.84
   macro avg       0.84      0.84      0.84
weighted avg       0.84      0.84      0.84
```

Fig. 9. Results for Inception Model [6]

Results

To evaluate each model three scores will be used; Precision, Recall and F-1 Score. This variables are widely used to evaluate the efficiency at which different ml models predict/classify.

Analysis and Conclusion

Initial results would aim towards DSP Filtering or Feature Selection having a positive impact on ml algorithms (at least VGG16). Shape Feature Results are surprising for VGG16 as they were expected to decrease accuracy due to both models being convolutional neural networks. The other two feature analysis were expected to increase accuracy due to the fact that they focus in spatial features and have their output shown as figures.

Feature Analysis if chosen correctly is due to increase the efficiency of the model. The training time should decrease as well but with such a small batch it is difficult to see any significant differences between runs. Bigger datasets would be needed to make any certain claims about this.

The entire project was ran in Matlab first and then moved into Google Colab to make a run with a bigger batch size using Google Colabs cloud GPU's.

Future Works and Remarks

-There are several more feature extraction methods for Image Analysis as well as other methods for other types of data. This is just an introductory project to the topic.

-Implement this same project with other machine learning algorithms that don't work through convolution to see how they manage with other features that could be presented to the model in other methods aside from images or spatial maps.

-A run with a bigger batch would reaffirm some of the results and give a better insight on how effective these methods are.

-There is a lot of parameter tuning that was not adressed in this article as this is a DSP problem and not a machine learning project. Fine tuning this parameters could also lead to better/different results.

References

[1] AO Yan-Li. Introduction to digital imag e pre-processing and segmentation. *2015 Seventh International Conference on Measuring Technology and Mechatronics Automation*, 2015.

[2] Biomedical Signal and Image Processing MIT Lecture Notes, Accessed March 2024 at: https://ocw.mit.edu/courses/hst-582j-biomedical-signal-and-image-processing-spring-2007/pages/lecture-notes/.

[3] Pranab Dhar, Hee-Sung Jun, and Jongmyon Kim. Design and implementation of digital filters for audio signal processing. pages 332–335, 07 2008.

[4] Frederic Defaux. Grand challenges in image processing. *Frontiers in Signal Processing*, 2021.

[5] CIFAR-10 is a public dataset from the Computer Science Department at University of Toronto, available at: https://www.cs.toronto.edu/ kriz/cifar.html.

[6] Code Implementations found in Author's GitHUB: https://github.com/Spchiq/DSP-Feature-Analysis.

[7] Nishchol Mishra and Sanjeev Sharma. Color texture based image retrieval system. *International Journal of Computer Applications*, 2011.

[8] John Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1986.

[9] Bing Wang and ShaoSheng Fan. An improved canny edge detection algorithm. *Second International Workshop on Computer Science and Engineering*, 2009.

[10] 2020, Raqueeb Shaikh, OpenCV(findContours) Detailed Guide, Medium, Accesed 3/23/2024 at: https://medium.com/analytics-vidhya/opencv-findcontours-detailed-guide-692ee19eeb18.

[11] Yujin Chen, Ruizhi Chen, Mengyun Liu, Aoran Xiao, Dewen Wu, and Shuheng Zhao. Indoor visual positioning aided by cnn-based image retrieval: Training-free, 3d modeling-free. *Sensors*, 18:2692, 08 2018.

[12] Xinyang Wang. A recognition method of ancient architectures based on the improved inception v3 model. *MDPI: Computer Vision, Pattern Recognition, Machine Learning, and Symmetry*, 2022.