OBJECT RECOGNITION WITH LIMITED DATA SETS AND COMPLEX ENVIRONMENTS

Jordan Leiker

Georgia Institute of Technology

ABSTRACT

Modern machine learning-based object recognition and image classification requires training sets in the order of hundreds-of-thousands to millions of images to achieve favorable accuracies. Prediction accuracy drops rapidly in the absence of large data sets and is exacerbated if the training set is not representative of the actual testing environment.

This paper presents a solution to nullify the effects of training a convolutional neural network (CNN) on images in non-representative environments, thereby enabling the CNN to be trained on an entirely independent set of images than what will be used for testing and without having a detrimental effect on the prediction accuracy.

Index Terms— Image Classification, Neural Networks, Template Matching, Image Processing

1. INTRODUCTION

The CURE-OR data set will be used for the training and testing of this papers proposed algorithm; it contains 9900 training images and 6600 test images with 10 object categories and 18 artificially imposed distortions. The CURE-OR training data has greatly simplified '2D' backgrounds, while the test data is a mix of 2D and 3D backgrounds. In the context of this paper, 2D backgrounds are 'simple' backgrounds either consisting of a white screen or a significantly blurred background; 3D backgrounds are unblurred backgrounds with significant detail, such as a book case containing many books with text on the bindings.

The addition of 3D backgrounds in the test data, combined with the relatively small number of images and added distortions, is perfect for testing out the proposed algorithms ability to separate the object identification task from the 3D background not present in the training data.

2. APPROACH

The algorithm uses two convolutional neural networks (CNNs), one for the image classification and another for distortion identification. The high-level CNN software library, Keras, was used in conjunction with the machine learning library TensorFlow to build and train the CNNs.

Built-in Keras functions for image augmentation and model training were used to artificially increase the size of the CURE-OR training set as well as perform the actual CNN training. Using the image augmentation functions, Keras added varying levels of shear, zoomed in and out, and horizontally flipped the images in an attempt to build a more robust model during training. To use the Keras training functions, the images needed to be sorted in to a 'train' and a 'validation' folder, with subfolders corresponding to the classes desired for each model to train on, e.g. the object classification 'train' and 'validation' directories had 10 subfolders each, with those folders containing images of only one of the object types. From there, Keras handled the randomization of the images as well as working both forwards and backwards through the training sets each epoch.

The two CNN's had identical layers with the exception of the final dense layer reflecting the total number of categories for the given neural network. The object classifying network had 10 object types, so the final dense layer had a width of 10; the distortion classifying CNN had 7 layers and a final dense layer width of 7. The reason the distortion CNN only had a width of 7 when there were 18 total distortions in the CURE-OR data set is because the RGB and grayscale images with similar distortions were grouped, dropping the total from 18 to 9, and then the 'No Challenge' and 'Re-Size' categories were dropped. The justification for dropping these categories is that we did not want to allow the neural network to train on images that had 'nothing present to learn'.

Finally, to augment the CNN's predictions, a number of image processing steps were taken with the goal of presenting an image to the CNN for prediction that had as much non-object related information removed, i.e. distortions and backgrounds. Dippykit, OpenCV, and custom libraries were used for the image processing.

3. ALGORITHM

The algorithm can be broken down in to four sections: (1) Pre-Processing (2) Distortion detection and removal (3) Object location and background smoothing (4) Image class prediction. These sections will each be described in detail. Figure 1 shows the complete image processing pipeline.

3.1. Pre-Processing

The pre-processing section converts all of the input images to a standard shape. This is accomplished by converting a grayscale input image to RGB by replicating the pixel intensities three times, as well as resampling the image to as close to the max image size of 968x648 as possible. Any images that go slightly over or under the max image shape during the resample process are either cropped down or padded (using reflection).

The CNN's require that all input images be the same shape; resampling was picked versus simply padding the images because it has the added benefit of automatically correcting 'Re-Size Distortion'. By resampling prior to any categorization, any misclassifications due to the 'Re-Size Distortion' in the CURE-OR dataset are virtually eliminated.

3.2. Distortion Detection and Removal

The output of the distortion classification CNN is used to select from a set of image processing functions depending on which distortion is detected. For instance, if salt and pepper noise is detected the image is run through a bank of median filters; this bank of filters is composed of two 3x3, two 5x5, and three 7x7 median filters. The bank of median filters was selected after a set of subjective tests that yielded the 'best looking image'; this is potentially one area that could be tuned to yield better performance.

If Gaussian blur is detected, the image is run through a Laplace sharpening block that uses the 3x3 extended Laplace kernel. The blur is so intense for the higher levels of Gaussian distortion that a larger Laplace kernel would be more useful for sharpening, but this would negatively impact the lower levels of distortion correction.

At this time, only the median filter bank and Laplace sharpening blocks are implemented; salt and pepper noise and Gaussian blur are the only statistically significant mis-identifications present in the validation set. The algorithm lends itself well to extending capabilities of this distortion removal section; it would be simple to add more features, such as histogram-equalization for under/oversaturation and contrast correction, or a custom algorithm for other noise corrections.

3.3. Object Location and Background Smoothing

Given that the 3D backgrounds are the biggest hurdle to overcome in this image classification set, it is critical to remove as much influence from the background as possible. To do this, the algorithm locates the object and then blurs the background

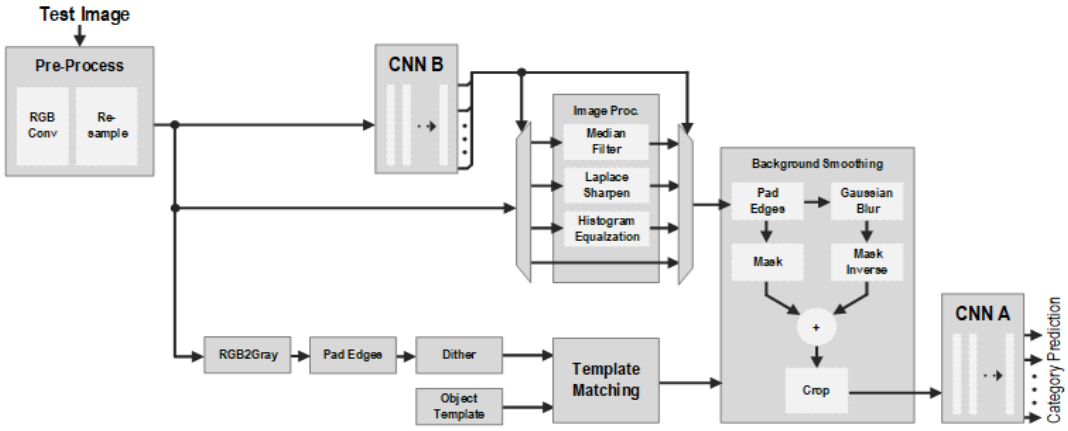

Figure 1. Image Processing Pipeline

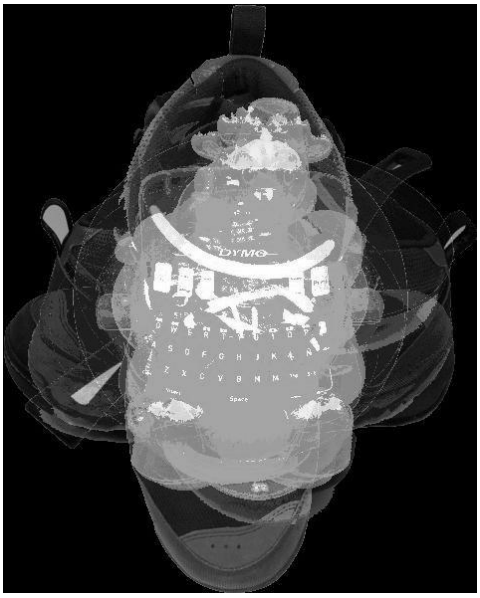

Figure 2. Morph Template

surrounding the object. By blurring, instead of removing entirely, there is less of a chance that the 'edge' between the unmodified and modified region will look like an object feature. Additionally, the blur more closely resembles the backgrounds of the 2D training set.

A template matching based approach was used to locate the object in the image [1]. The template used is a 'morph' of all of the objects, specifically the 'morph' was created by performing the following for all of the objects in all orientations, and then logically OR-ing the results together, the 'morph' template is shown in Figure 2:

1) Cropping all of the objects from the simplest-background training images
2) Thresholding and blob-clustering to get an outline of the object
3) Locating the maximum contour and using it to create a mask
4) Masking the original image with the 'blob'-mask
5) Find a bounding rectangle around the now-masked object
6) Crop to the bounding rectangle, then re-size so that the object is centered and can be OR-ed with the others

Prior to using the 'morph' template for the template matching, the original image is converted to grayscale [2], padded such that the morph-template can be slid over the entire original image area, and then dithered (add salt and pepper noise). The dithering was an important step to reduce the morph-templates correlation score with details in the background. For example, a dark corner of a book case may be confused with the dark coloring on the shoe that is port of the morph-template.

The template matching process itself used the OpenCV 'matchTemplate' function with Normalized Correlation Coefficients as the comparison algorithm; when compared to the Squared Error and Cross-Correlation variants, the Correlation Coefficients consistently outperformed the rest. The results from this are then analyzed to determine the best-match pixel location, which is forwarded to the background smoothing blocks.

The background smoothing blocks use the 'cleaned image' (the output of the distortion prediction and image processing blocks) and creates a Gaussian-blurred copy. Then, using an elliptical mask the size of the morph-template, masks off the area around the detected object in the un-blurred image and with an inverse mask captures the area around the ellipse in the blurred version of the image. Both the mask and inverse masks are added together and used to replace the region-of-interest in the full-sized blurred image. The result of this is a blurred version of the test image, with only an elliptical region surrounding the detected object left un-blurred.

3.4. Image Class Prediction

The last step in the algorithm is sending the cleaned and partially blurred image to the object classification CNN. The prediction class in Keras can either print out a single class label corresponding to the guess, or it can print out a list of prediction percentages for each of the object categories. The algorithm described in this paper only uses the label output, because the final 'softmax' layer in the object classification CNN tends to greatly skew the prediction percentages in favor of whichever object is detected.

4. RESULTS

Prediction accuracy on the test set was 24% when using the full image processing pipeline; compared to the neural network alone at 22%. The algorithm does not consistently remove enough of the 3D backgrounds influence to have a high prediction accuracy. In some cases when the object is offset from

the details of the 3D background, for example a series of pictures that have the object placed near several magazines on a table top, the algorithm works perfectly because the magazines are far enough from the object that the they fall outside of the mask and are blurred.

Despite the fact that the background smoothing does not work exceedingly well, the distortion prediction correctly identifies the salt and pepper noise and gaussian blur, and the corrections from the pre-processing blocks and image cleanup blocks result in an approximately 10% increase in prediction accuracy on the validation set. This increase in prediction accuracy is due to eliminating misidentifications due to 'Re-Size Distortion' as a result of the resampling pre-processing step and cutting the misses due to salt and pepper noise in half thanks to the distortion prediction and median filtering. See Table 1 for a complete comparison of misidentifications with and without the pre-processing steps.

Unfortunately, the increase in accuracy on the validation data does not translate to the test set, as the 3D background affects swamp out the improvements gained by the pre-processing. Additionally, the processing time to complete the entire 6600 image test set grew from 1 hour to approximately 24 hours.

Table 1. Comparison of Mis-Identifications With and Without Pre-Processing

Distortion	Resize	Satur.	Gauss	LensDirt	S&P
W/o Proc.	54	10	76	3	32
W/ Proc.	2	11	76	7	18

5. CONCLUSION

The distortions can be overcome provided the information in the image hasn't eroded too much, as was the case with the high-level Gaussian blurs, but if the CNN models are not trained with images representative of the test set the prediction accuracies will be low and inconsistent. In this case, the models were trained with images that did not have backgrounds as detailed as the ones found in the test set, and as a result, the features in the 3D backgrounds were consistently mistaken as the objects that the model was initially trained on.

6. FUTURE WORK

This algorithm failed to make a major improvement on the prediction accuracy because it could not nullify enough of the 3D background influence. To continue on from the work presented in this paper, the primary focus should be improving the object location and background smoothing method. For instance, to improve the object location it may yield better results to use an edge-based template matching algorithm instead of using true template matching, which often failed due to similar intensities between the templates and the pixels in the 3D backgrounds. Instead of a generic ellipse mask for the background smoothing steps, using some kind of contour tracing approach starting at the located object and working outwork may result in a tighter unblurred region surrounding the object.

Even though the improvements outlined above may improve the quality of the algorithm discussed in this paper, it may yield better results to take on a different approach altogether. Time may be better spent training the object prediction CNN with a more robust training set, such as an augmented training set that reproduces the objects across a number of complicated 3D backgrounds; or perhaps introducing dithering in the training set. It is hypothesized that training the model to identify objects through added dithering would yield a model that could identity objects in a dithered test set; which could be another method to break up 3D background influence.

12. REFERENCES

[1] K. Ahuja, P. Tuli, "Object Recognition by Template Matching Using Correlations and Phase Angle Method," *International Journal of Advanced Research in Computer and Communication Engineering*, Vol 2, Issue 3, March 2013

[2] P. Swaroop, N. Sharma, "An Overview of Various Template Matching Methodologies in Image Processing," *International Journal of Computer Applications*, pp. 8-14, November 2016.