On the benefit of synthetic data for company logo detection

Christian Eggert, Anton Winschel, Rainer Lienhart

On the Benefit of Synthetic Data for Company Logo Detection

Christian Eggert   Anton Winschel   Rainer Lienhart

Multimedia Computing and Computer Vision Lab
University of Augsburg
Universitätsstr. 6a, 86153 Augsburg, Germany
{ christian.eggert, anton.winschel, rainer.lienhart } @ informatik.uni-augsburg.de

ABSTRACT

In this paper we explore the benefits of synthetically generated data for the task of company logo detection with deep-learned features in the absence of a large training set. We use pre-trained deep convolutional neural networks for feature extraction and use a set of support vector machines for classifying those features. In order to generate sufficient training examples we synthesize artificial training images. Using a bootstrapping process, we iteratively add new synthesized examples from an unlabeled dataset to the training set. Using this setup we are able to obtain a performance which is close to the performance of the full training set.

Categories and Subject Descriptors

I.5 [**Pattern Recognition**]: Miscellaneous; I.4.3 [**Image Processing and Computer Vision**]: Enhancement

General Terms

Algorithms, Experimentation, Performance

Keywords

Logo detection, Logo classification, Synthetic training data, Dataset augmentation

1. INTRODUCTION

It is in the nature of machine learning algorithms to generalize previously seen training examples to unknown instances. For a robust prediction it is vital for the classifier to have access to a large variety of training examples. However, building a dataset can be a slow and tedious task and training examples are hard to come by for some machine learning problems.

Usually there are two approaches to tackle this problem: One way is to augment the training set by generating syn- thetic variations of the collected examples. Another ap- proach is to use the already collected examples to search for

Figure 1: **A suggestion (red) for user feedback generated by our bootstrapping process.**

related examples on a large unstructured dataset, browsing through the suggestions and deciding which ones to add to the training set. The problem with the latter approach is located in the second step, since related examples usually mean small variety.

For the task of company logo detection it is compelling to use both approaches to reinforce each other. Company logos are a special case because typically, only the context in which a logo appears changes strongly while the actual logos themselves usually have relatively uniform appearances. Context is comparatively easy to synthesize as we will describe in section 2. Due to the uniformity of the company logos, similarity search has a good chance of success - especially when used in conjunction with high-level feature representations like features extracted from deep convolutional neural networks (DCNNs).

DCNNs have revolutionized many areas of computer vision. Typically, the convolutional layers of such a network learn a feature representation while the last layers - usually fully connected - are responsible for classification. Being a data driven approach, DCNNs typically require a large number of labeled instances to train. However, it has been found [3] that the features extracted by the convolutional layers generalize well to other computer vision tasks on different datasets. In order to repurpose a DCNN for a different task, the last layers tend to be discarded and the output of the previous layers are used as input for a different classifier, such as support vector machines (SVMs) or random forests.

In this paper we apply the principles mentioned above to the problem of company logo detection and recognition in the absence of a large dataset. For our scenario we will as-
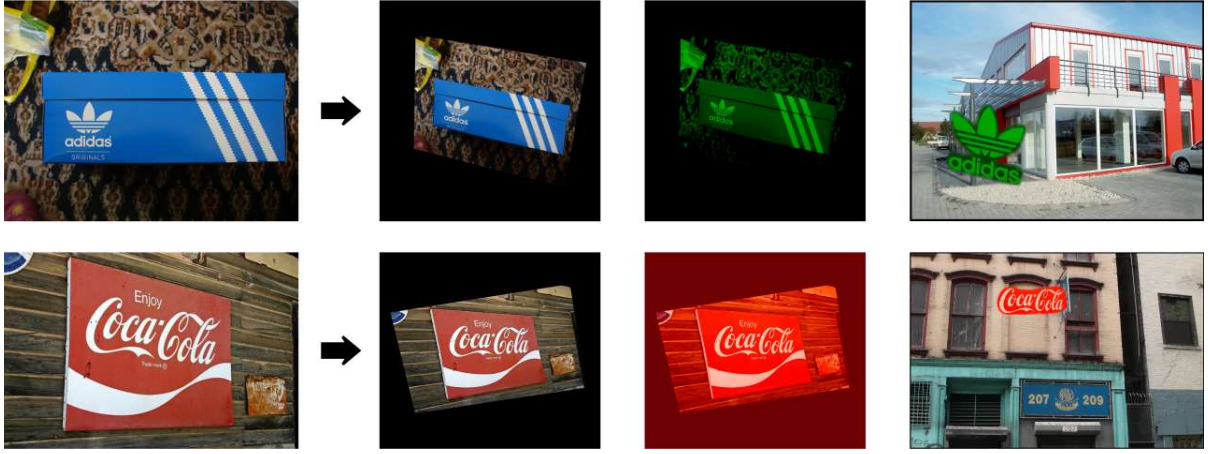
Figure 2: Two examples of synthetically generated images. Starting from a base image, a random perspective transform is applied, followed by gaussian blur (not shown), color augmentation (exaggerated for viewing purposes) and background replacement.

sume a limited number of labeled training examples with pixel-level annotations and an unlabeled number of images, some of which may contain additional instances of the logo. This could for example be achieved by downloading a set of images from an online photo-sharing website using the logo names as keywords. Furthermore, we require a set of images that is most likely free of the logos that we are attempting to detect. Again, this could be achieved by searching an online photo-sharing site with random keywords such as "landscape".

Our contributions are as follows:

1. We show that by training a classifier from synthetic training examples alone we are able to obtain a performance that is comparable to a classifier trained from real images.

2. We show that in a single-shot scenario synthetic training examples can be beneficial to boost the effectiveness of bootstrapping.

2. SYNTHETIC DATA GENERATION

Since we are using DCNNs solely as feature extractors and assume them to come pre-trained, we are only left to train the SVMs used for classifying the DCNN features. In the absence of a large database, we turn to synthetically generated training images. The generation process is depicted in figure 2.

For every class we start off with a small number b of base images from which the synthetic images are to be derived. For every base image we synthesize artificial training images in the following way:

1. We select a random perspective transform and warp both image and mask accordingly. The transformation is given by

$$M = PTR_xR_yR_z \qquad (1)$$

where the image is regarded to be on a 3d-plane translated from the origin by T while R_x, R_y and R_z represent the rotations of this plane around the coordinate axis associated with the angles Θ_x, Θ_y and Θ_z. P defines a perspective projection. For reasonably large, yet not extreme variations, we limit the allowed angles to the range $-50° \leq \Theta_x \leq 50°$,

$-50° \leq \Theta_y \leq 50°$ and pick $\Theta_z \in \{0°, 90°, 180°, 270°\}$ from a uniform distribution.

2. In order to generate a larger visual variety, we modify the color of the warped image by using the approach of [2]. We calculate the eigenvalues $\alpha_i, i = 1, 2, 3$ and the corresponding eigenvectors p_i on the covariance matrix of RGB values in the original base image. For each image we draw three random numbers γ_i distributed according to $\mathcal{N}(0, 0.5)$ and add to each pixel $[I_{x,y}^R, I_{x,y}^G, I_{x,y}^B]^T$ a linear combination of the eigenvectors, given by

$$\sum_{i=1...3} \gamma_i \alpha_i p_i \qquad (2)$$

3. The logo is randomly blurred by convolution with a gaussian kernel $G(x, y, \sigma)$ to simulate different scales. This is done by picking $\sigma \in \{0, 0.5, 1.0, 1.5\}$ from a uniform distribution.

4. We remove the background of the logo using the pixel-level annotations of the training set and copy the warped and color-modified logo into a new image which contains no logos. This is done to prevent the classifier from adapting to features extracted from the background since the DCNN only processes rectangular bounding boxes.

On the synthesized images, we apply selective search [7] in order to obtain candidate regions. The candidate regions are split into positive and negative examples as described in section 3. Finally, we extract the DCNN features for the image patches defined by the selective search windows.

3. CLASSIFICATION PIPELINE

For our experiments we use the following classification pipeline: We are using selective search [7] to generate up to 2000 bounding boxes with object proposals per image. Images are limited to a maximum side length of 1024 pixels and are rescaled if necessary. Each bounding box defines an image patch which is resized to match the input dimensions of the neural network and is propagated through the DCNN. For our experiments, we use the 16-layer very deep architecture by [5].

1284

We use the DCNN output after the first fully connected layer[1] without applying the ReLU [2] or employing the dropout. We have found the 4096-dimensional features extracted by this configuration to yield the best results. One exception is the output of the last pooling layer which delivers slightly better results but are impractical to use due to their high dimensionality.

The extracted features are L_2-normalized and used as input for a series of linear SVMs. We train a separate SVM for every class. Each candidate region is classified according to the SVM which yields the highest score. We require a confidence of at least 0.5 for a positive classification. If no SVM yields a score greater than 0.5, the candidate region is classified as negative. A positive classification for the image is given by the predicted class of the highest-scoring candidate region.

For the initial training, we select candidate regions C whose overlap − calculated using intersection over union (IoU) − with the groundtruth annotation G is $\mathrm{IoU}(C, G) \geq 0.7$ as positive examples for the SVM. As negative examples we choose candidate boxes with an overlap between $0.1 \leq \mathrm{IoU}(C, G) \leq 0.3$. Since the number of negative examples typically is a lot higher than the number of positive examples, we only use a random subset of the negative examples for training.

4. MINING NEW TRAINING EXAMPLES

It is possible to train a decent classifier on purely synthetic data (see section 5). In this section we investigate a bootstrapping process which uses a rudimentary classifier - trained only from a few real images - to mine new training examples from a dataset containing unlabeled instances of company logos. The newly found real training examples can then be used as a basis to synthesize more training images.

Starting with a set of pre-trained SVMs - e.g. from a previous training run on synthetic images - we use our classifier to predict the class labels of object proposals on a previously unlabeled set of images as described in section 3. Typically, the number l of instances a user is willing to label is limited. In order to extract the number of training examples that are likely to be maximally beneficial to the performance of the classifier, we follow the idea of version space bisection by [6]. We select $l = 5$ training examples which are located closest to the decision surface of each SVM for the user to classify. Intuitively, this approach provides the classifier with labels to instances it is most uncertain about. In manually labeling these examples, the classifier is therefore provided with the currently most informative labels.

If classified as a positive example by the user, we generate synthetic versions of this example as described in section 2. We extract DCNN features from both the original example as well as from the synthetic variations. If classified as a logo by the user, the features are added as positive examples to the corresponding SVM responsible for this class. To all other SVMs the example is added as a negative example.

Additionally we make use of the set of logo-free images, which is used to replace the background of synthesized logo images (see section 2), for mining hard negatives. For this purpose, we select at most one candidate box per image that is classified as positive with the highest confidence on

_____

[1]This layer is called FC6 in the CAFFE [1]-Model provided by [5]
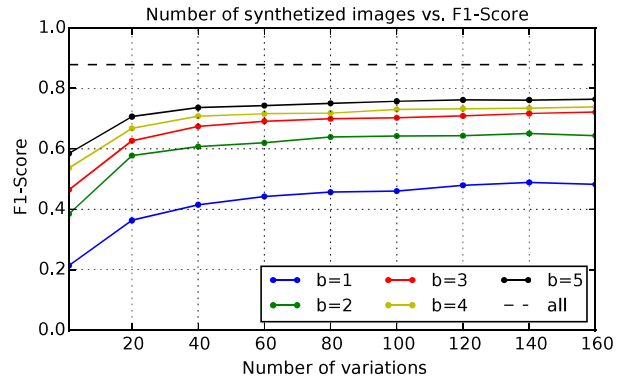

Figure 3: Influence of the number of variations on the recognition performance for different numbers of base images. The dotted line indicates the performance when trained with all training examples (no synthetic images)

the logo-free dataset and add them as negative examples to the SVM which classified this image as positive.

5. EXPERIMENTS

We conduct our experiments on the FlickrLogos-32 dataset [4] which consists of 4280 training and validation images and 3960 test images containing both images of 32 logo classes and images containing no logos. The training set contains 10 examples per class while both the validation and the test set contain 30 positive examples each. Additionally, the validation set contains 3000 non-logo images.

First we establish the baseline for our logo recognition system by combining the positive examples from both training and validation set and use them for training. Using the training process described in section 3 and after two rounds of hard negative mining on the non-logo part of the validation set, we obtain a precision of 0.996, a recall of 0.786 and a F1-Score of 0.879.

In our first experiment we evaluate the performance of the classifier when trained on a purely synthetic training set using a fixed amount of base images. This will help us to determine a reasonable number of synthetic images to generate per real example and will serve as a baseline to evaluate the impact of the bootstrapping procedure. We start with a single base image per class $b = 1$ and progressively generate more synthetic images from each base image. Figure 3 shows the F1-Score for the logo recognition system as a function of the number of variations v. Remarkably, only a few base images are required to achieve a performance which is close to the performance of the recognition system trained with the full training set. Unsurprisingly the influence of synthetic training examples diminishes as the number of base images b increases, since the base images can already provide enough variation for the SVMs.

In our second experiment we use the classifiers trained in the first experiment as a starting point for the bootstrapping process described in section 4 (synth). For comparison we use the same bootstrapping process without dynamically synthesizing new training examples (w/o synth).
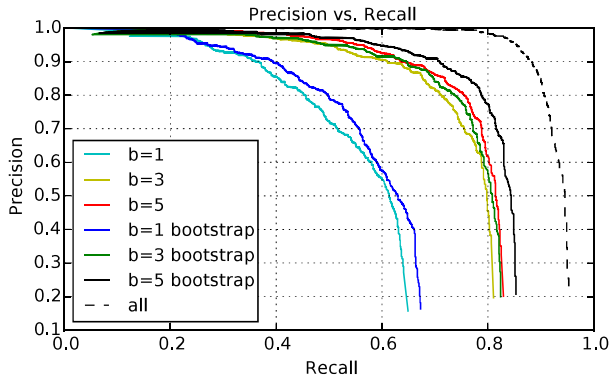
Figure 4: Precision-Recall curve for synthesized training sets with $b = 1$, $b = 3$ and $b = 5$. The dotted line indicates the performance when training with all examples (no synthetic images). In the case of synthetic training data, the number of variations is $v = 40$. Three learning rounds were performed for bootstrapping.

Learning rounds	b	1	2	3	4
synth	1	0.483	0.632	0.646	0.681
synth	3	0.730	0.791	0.795	0.807
synth	5	0.770	0.803	0.815	0.819
w/o synth	1	0.515	0.517	0.514	0.517
w/o synth	3	0.730	0.748	0.751	0.759
w/o synth	5	0.770	0.791	0.792	0.790

Table 1: Influence of the number of learning rounds ($l = 5$) on the performance of the classifier for different number of base images given as F1-Score. The initial classifiers were trained with $v = 40$.

In order to simulate a user providing input, we use the annotations provided from the FlickrLogos validation set. In each training round we select $l = 5$ examples from each SVM which are classified according to the overlap with the groundtruth.

While this approach for simulating user input yields reproducible results, it does have the weakness that it relies on the assumption that all logo instances in the dataset have been labeled. Clearly this is not always the case. In the process, we noticed some limitations in the annotations of the FlickrLogos dataset: One such example is shown in figure 1. The bounding box around the groundtruth annotations is shown in green which is a typical annotation for this particular class. The bounding box marked in red shows a suggestion for a new training example as generated by our bootstrapping process. Despite clearly being an instance of the company logo used in a different context, this example will be classified as a negative example during the bootstrapping process since instances like this are not annotated in the dataset, possibly confusing the classifier.

Table 1 shows the results of this experiment in terms of the F1-Score. While in every scenario synthetic training images are beneficial to the bootstrapping process, the effects are most strongly pronounced when the number of initial training images is small. This is especially obvious in the case of a single training image. As the initial training set grows larger the effects of bootstrapping diminishes as well as the effect of synthetic images. However, using this bootstrapping strategy we are still able to achieve 91% of the performance of the original dataset while using only 8% of the original training images in the case of $b = 3$.

Figure 4 shows the effect of mining new training examples in terms of precision and recall. We show the performance of classifiers trained from synthetic data and compare it with the performance of the classifier that has been subjected to three rounds of bootstrapping with synthesizing from newly mined training examples.

6. CONCLUSION

We have shown that training sets for company logo classification can be successfully enhanced with synthetic images when little training data is available. This is especially true when used in conjunction with a bootstrapping strategy. In this work we have exclusively used pre-trained DCNNs for feature extraction. We expect strong performance gains for DCNNs that have been specifically adapted to the task of logo detection. We imagine that synthetic images could also be beneficially employed in this regard which we will investigate in future works.

7. ACKNOWLEDGMENTS

8. REFERENCES

[1] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. *CoRR*, "abs/1409.1556", 2014.

[2] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Neural Information Processing Systems (NIPS)*, pages 1097–1105. 2012.

[3] A. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson. Cnn features off-the-shelf: An astounding baseline for recognition. In *IEEE Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 512–519, June 2014.

[4] S. Romberg, L. Pueyo, R. Lienhart, and R. van Zwol. Scalable logo recognition in real-world images. In *ACM International Conference on Multimedia Retrieval (ICMR)*, pages 25:1–25:8, 2011.

[5] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*, 2015.

[6] S. Tong and D. Koller. Support vector machine active learning with applications to text classification. In *International Conference on Machine Learning (ICML)*, pages 999–1006, Stanford, California, June 2000.

[7] J. Uijlings, K. van de Sande, T. Gevers, and A. Smeulders. Selective search for object recognition. *International Journal of Computer Vision*, 104(2):154–171, 2013.