Brand logo classification

Jia-Han Chiam
Stanford University
450 Serra Mall
jiahan@stanford.edu

Abstract

This paper studies the use of convolutional neural networks and bag-of-words models constructed from SIFT features for finding brand logos in photographs uploaded to social media websites. We find that the CNNs produce much better results than the bag-of-words models, with the former achieving a classification accuracy of up to 86.98% on our dataset compared to the 11.04% accuracy achieved by the latter. As training neural networks involve learning a large number of parameters, and our dataset contains relatively few training examples (40 per class), we use data augmentation to increase the effective size of our training set by creating taking each training image and creating multiple random crops around the bounding box of the logo. Apart from attempting to classify entire images, we also use image segmentation to generate region proposals where the logo might reside, which we can classify separately before assigning to each image the label of the region whose predicted classification has the highest confidence. Both of these methods result in dramatic improvements in classification accuracy. not

1. Introduction

1.1. Motivation

Social media websites contain large amounts of unlabelled image data, from which many useful insights can be extracted. Many companies closely monitor their web presence by keeping track of, for example, any mentions of their brand name on Twitter or Facebook. However, a significant fraction of social media content consists of user generated images. Instead of tweeting the word 'Starbucks', a user may instead post an Instagram photo containing a Starbucks cup. The company in question, or third party market researchers, may find it useful to be able to detect such events, even if the accompanying text does not contain a mention of their brand name, stumping their usual brand tracking tools. The ability to effectively detect such 'visual mentions' clearly has many commercial applications.

1.2. Approach

We approach the problem as both a classification problem (classifying images that contain one or more instances of a single logo) as well as a detection problem (determining whether or not an image contains a logo).

We attack the problem by using to finetuning a CNN model that was pretrained on the ImageNet Large Scale Visual Recognition Challenge. As training a neural networks is a very data-hungry process, we also analyze the use of data augmentation to boost the performance of our CNN, creating additional training images by generating random crops of the images in our train set.

We also generate SIFT features for the logos in our training set and perform k-means to convert the SIFT descriptors into a histogram of cluster membership counts. We then train a support vector machine to predict the classification of each 'bag of words'.

As the brand logos may be only an incidental part of each photograph, occupying only a small region of the image, we also study the use of a sliding window method and an image-segmentation based approach (proposed by van de Sande, Uijlings, Gevers and Smeulders [5]) for generating region proposals where the logo might be.

2. Experiment

2.1. Data

For our research, we use the FlickrLogos-32 dataset [4], which provides images taken from Flickr that contain occurrences of 32 different brand logos. There are 70 labelled examples for each class, and the authors of the dataset divide the data into train, validation and test sets containing 10, 30 and 30 images respectively, where the training examples are handpicked to be close-up, unobscured images where the logo covers a large portion of the image (most of these logo appearances in the other images are incidental; the logos are not the central focus of the image and may only occupy a small region in the image, complicating the

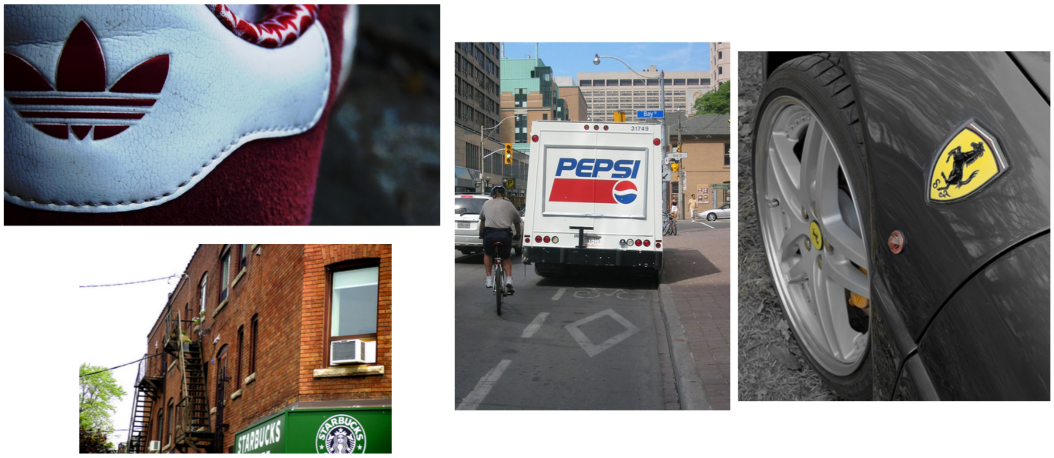

Figure 1. Photographs from the FlickrLogos-32 dataset

detection problem). However, due to the small size of the dataset, we opt to forgo the validation set and combine the train and validation sets into a training set of 40 images.

Each image may contain one or more instances of a brand logo, but not instances of different logos. The dataset also contains 6000 images with no logo in them, split evenly into a validation and test set. These can be used to calibrate a precision-recall curve, as the bulk of images in the wild will indeed not contain a logo and should not be falsely classified as having one. Due to space and time constraints, we only used the validation set (containing 3000 images) for testing.

The dataset also contains masks that indicate the locations of the logos in the images, as well as the coordinates for the bounding boxes of the logos.

2.2. Evaluation

Setting aside the logo-less images, we first treat our problem as a classification problem over 32 categories. Then each example in our test set can be assigned exactly one label, and we can measure the accuracy of our classifiers in predicting the label of each test image.

We can also include the logo-less images in our test procedure and treat our problem as a detection one. We can configure our models to label as 'logo-less' those images whose predictions are made below a certain level of certainty, and then see how our classification rate and false positive rate change as we vary this confidence threshold parameter.

3. Training

3.1. Bag of Words with SIFT features

We use OpenCV [1] to compute the SIFT descriptors inside each masked region in each training image. We then ran k-means to convert each set of SIFT descriptors into a histogram of cluster membership counts. For each test image, we then computed a k-dimensional histogram vector (normalized to sum to 1), and use the Scikit-Learn [3] python package to train a SVM classifier with an RBF kernel to solve the 32-class classification problem.

3.2. Convolutional Neural Networks

3.2.1 Introduction

We use the caffe [2] neural network implementation to finetune the BLVC CaffeNet Model, which was pretrained on the ImageNet Large Scale Visual Recognition Challenge. We are required to do this for two reasons. Firstly, we do not have enough data to train a full neural net with thousands of parameters from scratch. Secondly, we do not have the computing resources to perform the full training procedure. We replace the last fully connected affine layer of the CaffeNet reference model with one that has 32 outputs, corresponding to our 32 classes, and train the model to predict the class labels for our training images. Our transfer learning approach is very similar to the one adopted in Caffe's Finetune-Flickr-Style [2] example, where we set a low global learning rate of 0.001 but allow our newly introduced final affine layer to learn 10 times as quickly.

Caffe also allows five different crops (center and the four corners) to be extracted automatically from the image during the training phase, as a form of primitive data augmentation, but we explore a more robust and expansive method of data augmentation using the information we possess about the bounding boxes of the logos. It also has an option for mirroring the train images randomly during the testing phase, which is useful for the ImageNet classification, where membership in most categories is invariant under mirroring (e.g. a teapot whose spout faces left and

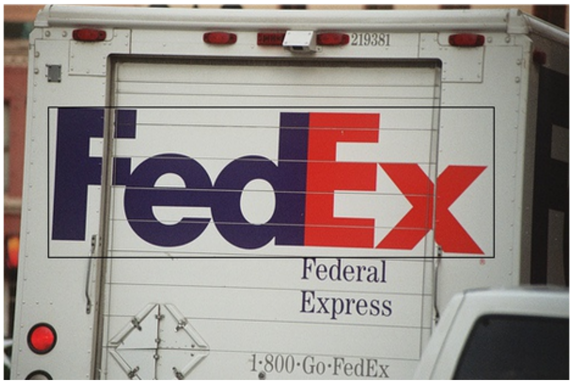

Figure 2. v2 model: Crop images according the bounding boxes of the logos

one whose spout faces right are nevertheless both teapots), whereas we expect a Coca Cola logo to almost always read left to right, so training our model to learn both orientations is not a useful generalization.

We train three separate models which differ on how they preprocess the training data. For all the models, we use a batch size of 50 for stochastic gradient descent in the training phase. We also resize the images to 227×227 (after performing any cropping required by the model), regardless of the initial aspect ratio.

3.2.2 v1

For our first model, we train our neural net on the full, uncropped train images. We do so for 2500 iterations, which correspond to approximately 50 thousand epochs (there are $32 \times 40 = 1280$ images in our train set). We decrease the learning rate by a factor of 10 every 500 iterations (10k epochs).

3.2.3 v2

For our second model, we train our neural net on the cropped bounding box of each image (see Figure 2), (if there is more than one logo, we only crop the first bounding box). The other parameters are set identically to the v1 model.

3.2.4 v3

For our third model, we perform a customized form of data augmentation. Since we have only 40 training images per class, we can dramatically inflate the size of our training set by making 100 random crops from each image in our training set (see Figure 3). The rectangular crops are created by drawing the coordinates of the opposite corners of the cropped region from a uniform distribution on the interval between the coordinates of the corners image and the corners of the bounding box of the logo. If there is more than one logo, then we choose the one whose bounding box was listed first in the bounding box file provided with the dataset.

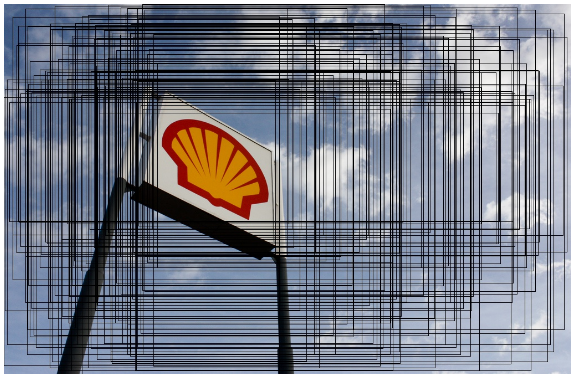

Figure 3. v3 model: Perform 100 random crops around the bounding box

Our data augmentation approach guarantees that the crops contain at least one full logo, and that we do not accidentally crop and irrelevant logo-less area of the image and attempt to train on that. We found that augmenting our training set in this way resulted in significant improvements (almost 10%) in our classification accuracy.

Since we have a much larger training set, this time, we run SGD for 50,000 iterations, and decrease the learning rate by a factor of 10 every 10,000 iterations.

3.3. Tuning the Hyperparameters

Since we do not have a cross validation set, we did not want to overfit to the training set by excessively optimizing the parameters. Nevertheless, we did attempt to vary a couple of important model parameters. For our bag-of-words model, we experimented with the number of clusters k we would divide the SIFT descriptors into during our k-means clustering step.

For our CNN model, we also tried increasing the regularization strength to minimize overfitting on our small dataset

4. Testing

4.1. Generating region proposals

When classifying the test images, we have to deal with the problem of correctly making predictions for images where the logo in question only occupies a small region of the image. First, we analyze the performance of our models when we feed each test image in its entirety to the classifier. Apart from that, we also test various methods for generating region proposals, which are bounding boxes which we suspect may approximately mark the boundary of the logos in question. Since both our bag-of-words-based SVM clas-

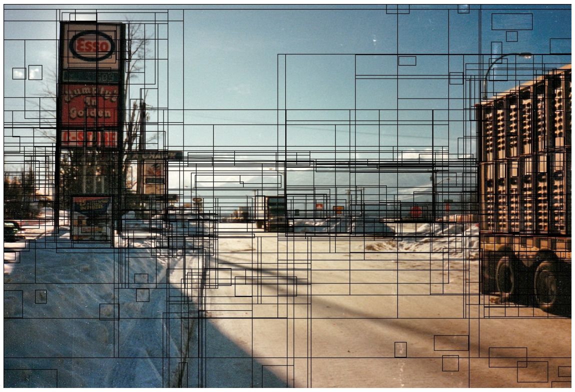

Figure 4. Regions proposed by selective search

sifier and our CNN classifier can be configured to generate probabilities for each class, we can classify each region proposal independently and select the label of the region that gets classified with the highest confidence.

4.1.1 Sliding Window

For the bag of words model, we can divide the image into an $n \times n$ square grid, and attempt to classify all $\binom{n}{2}^2$ possible subgrids of the image, picking the region classification with the highest SVM score as our final label. This is very computationally intensive, with a time complexity of $O(n^4)$, so we were only able to test values of n up to 5.

4.1.2 Selective Search

Van de Sande, Uijlings, Gevers and Smeulders [5] proposed a method for segmenting an image and using the segments to generate region proposals where objects are hypothesized to reside (see Figure 4). There were an average of 675 region proposals generated per image in our test set. While the sliding window technique is content-insensitive, the selective search method uses the image features to generate smarter region proposals, allowing us to generate more fine-grained region proposals without having the number of regions to test blow up too quickly.

4.2. Pruning region proposals

We found that false positives often involved very small region proposals, or proposals with very skewed aspect ratios. We would expect a Google doodle to be much longer than it is tall, and for the bounding box Pepsi logo to be approximately squarish, for example. Hence, we can prune region proposals whose aspect ratio is significantly different from the average aspect ratio of the original logo in the train image (the average width divided by the average height). We can also try discarding very small region proposals, as our v3 CNN model was trained on images that often contained logos that only occupied a small region of the image, so we can expect reasonably good performance even if our region proposals are larger than ideal and contain significant dead space around the relevant logo.

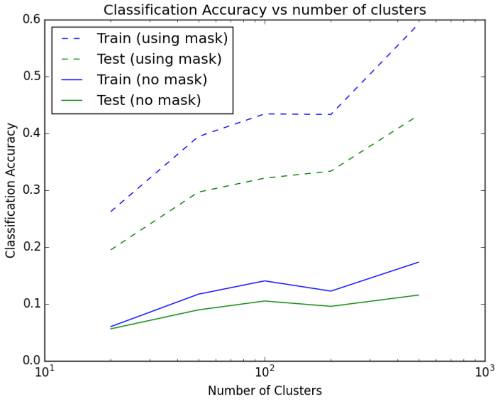

Figure 5. Classification Accuracy of bag of words model increases as number of clusters increase

5. Results

5.1. Bag of Words Model with SIFT features

We found the increasing the number of clusters produced in the k-means step resulted in an increase in accuracy for the bag of words model (see Figure 5). However, due to space and computational constraints, we were unable to test beyond $k = 500$, so it is possible that the optimal number of clusters is higher than that. At $k = 500$, we get a test accuracy of 11.56% when testing entire images. We used this value of k for the rest of our test procedure.

We also found that using a sliding window grid actually resulted in a decrease in test accuracy, as a larger number of candidate windows resulted in a larger number of potential false positives. Even if the closest-fit region proposal was correctly classified, an irrelevant candidate proposal was often classified wrongly with an even higher confidence (see Figure 6). Similarly, selective search resulted in a decrease in our classification accuracy.

For reference purposes, we also attempted classifying the test images based only on the masked test image region, and using this metadata, otherwise prohibited under our experimental conditions, we obtained a much higher accuracy of 43.2%, indicating that the main issue lay with correctly identifying the candidate region containing the logo.

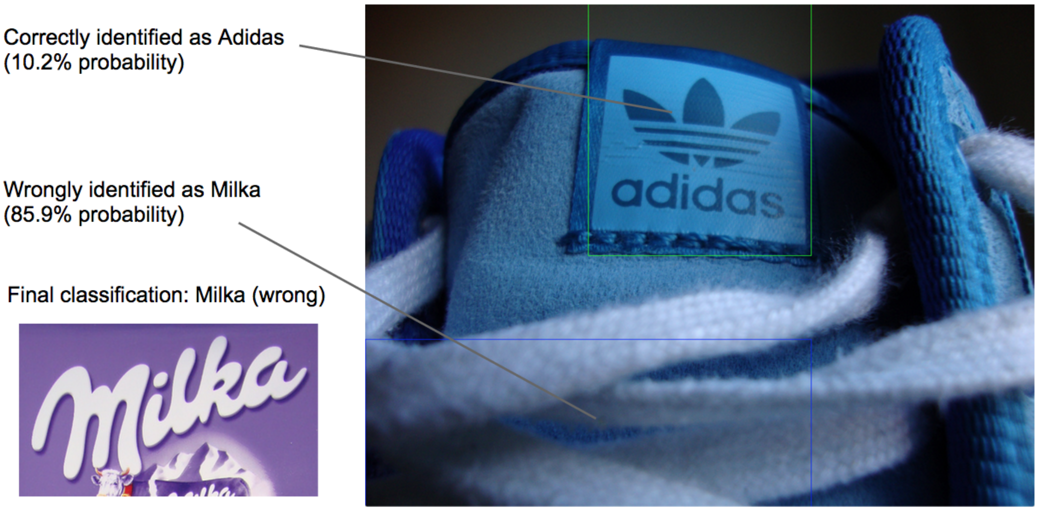
Correctly identified as Adidas
(10.2% probability)
Wrongly identified as Milka
(85.9% probability)
Final classification: Milka (wrong)

Figure 6. Adidas image with actual window correctly identified, alternate window misidentified with higher confidence by BOW classifier

5.2. Convolutional Neural networks

5.2.1 Testing on Entire Images

We found that the v2 model did very poorly when training entire images with a classification accuracy of 22.40%, since we only trained it on the cropped logos. In comparison, the v1 model had an accuracy of 60.42%, and the v3 model, with the data augmentation preprocessing step, had a much higher accuracy of 69.79%. In other words, the data augmentation procedure allowed us to increase the classification accuracy of the model by almost 10%.

5.2.2 Selective Search Region Proposals

When incorporating the selective search algorithm into our testing procedure to generate region proposals, we managed to obtain another significant increase in the classification accuracy of our models. Now that we were able to identify candidate locations for the logos in the images, we managed to obtain a classification accuracy of 85.0% for the v2 model and 85.9% for the v3 model.

5.2.3 Pruning Region Proposals

We define the skew ratio of a candidate bounding box for a logo to be the ratio between aspect ratio of the box and the aspect ratio of the average logo in the training set (where we take the inverse of the ratio if appropriate to get a value larger than 1). Then we can discard all region proposals with skew ratios larger than γ, for some threshold γ, and feed the remaining proposals into our v3 model. We experimented with various values of γ and determined that a threshold value of approximately 2.5 produces the best results (see Figure 7).

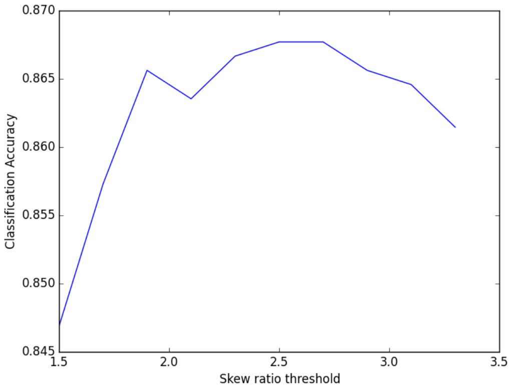

Figure 7. Classification accuracy as we vary the skew ratio threshold γ

We can also discard all region proposals which are less than $1/m$ the area of the full image, to cut down on false positives. We found that value of $m = 50$ produces the highest classification accuracy (see Figure 8).

We find that we improve our accuracy over the base v3 by approximately 1%, from 85.9% to 86.98%, by combining these two pruning techniques.

5.2.4 Parameter Regularization

We found that varying the regularization parameter did not result in a significant improvement our classification accuracy, contrary to our expectations (see Figure 9). We per-

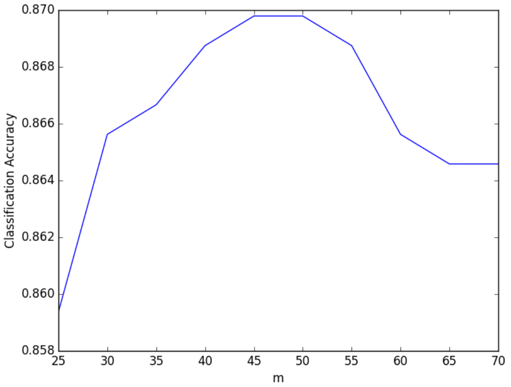

Figure 8. Classification Accuracy when we discard all region proposals smaller than $1/m$ the pixels in the full image, as m varies

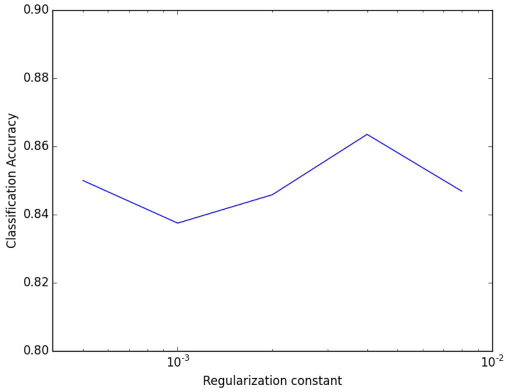

Figure 9. The classification accuracy fluctuated but did not seem to show a consistent trend in either direction when the regularization constant was varied.

Model	Accuracy
v1	60.42%
v2	22.40%
v3	69.79 %
v2 (w/ SS)	85.0 %
v3 (w/ SS)	85.9 %
v3 (w/ SS and pruning)	86.98 %

Table 1. Summary of results

formed this experiment using our v2 model with selective search and no region pruning, as it was much faster to train than the v3 model.

5.2.5 Detection vs Classification

All our analysis so far has focused on the problem of classifying images that are known to contain at least one label.

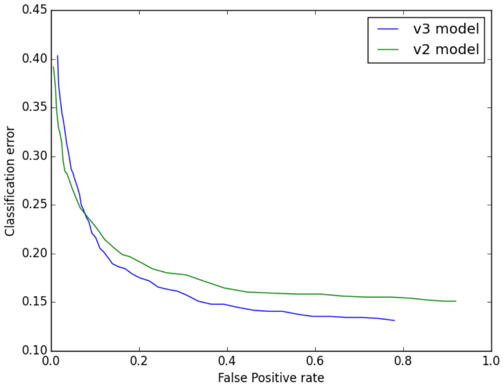

Figure 10. Classification Accuracy vs False positive Rate

Confidence Threshold	Classification Error	FP Rate
0.99	0.151	0.920
0.999	0.160	0.446
0.9999	0.215	0.122
0.99999	0.284	0.0317
0.999999	0.392	0.00533

Table 2. v2 model: Classification error and false positive rate tradeoff as confidence threshold varies

However, images take from the wild will mostly contain no relevant logos in them, and it is importantly to be able to determine which images have logos and which do not. We can do this by setting a minimum confidence threshold, and labelling as 'logo-less' any of the images whose highest prediction probability over all the region proposals is less than the threshold.

We can then plot the curve of classification error (number of images containing logos that were labelled as 'logo-less', or as containing a logo of a different brand) against false positive rate (number of logo-less images labelled as containing a logo) (see Figure 10). We do this for both the v3 model (with selective search and region pruning) and the v2 model (with selective search and no region pruning).

We see that v2 model has a lower classification error when the threshold is set higher, while the v3 model has a lower classification error when when the threshold is lower. This is because we train the v3 model on crops that contain large amounts of non-logo context, and it may be inadvertently learning irrelevant features that trigger on logo-less images. If we care solely about the classification problem, or are tolerant of false positives, the v3 model is superior, while the v2 model is superior if we want to minimize false positives.

Confidence Threshold	Classification Error	FP Rate
0.99	0.131	0.780
0.999	0.145	0.423
0.9999	0.179	0.184
0.99999	0.238	0.008
0.999999	0.3125	0.0257

Table 3. v3 model: Classification error and false positive rate tradeoff as confidence threshold varies

6. Conclusions

We found that our CNN models produces much better results than the bag-of-words based models. With our CNNs, we managed to achieve a final accuracy of 86.98%, compared to 11.04% for our bag-of-words model. This is likely due to the fact that the CNN models are much more expressive and encode much more information (e.g. about the location of various features). While CNNs normally require a large amount of data to train, we managed to achieve a high level of accuracy with a small training set through the use of transfer learning. The CNNs also only used the bounding boxes for the image logos in the preprocessing phase, which are much cheaper to create by hand than the image masks.

We also found that our models performed well at the detection problem, with very low false positive rates possible for a fairly reasonably tradeoff in classification accuracy.

7. Future work

Applying data augmentation to the CNN model results in higher classification accuracy at the cost of more false positives. To minimize the number of false positives, we can be stricter about cropping our images, for example, ensuring that the logo takes up at least a certain fraction of the cropped image so that we do not train on images where the logo only occupies a tiny region of it, leading the classifier to possibly learn irrelevant features.

Besides specifically trying to recognize the logo, we can also try to recognize other relevant objects in the image that may provide useful hints to our classifier. For example, a beer bottle is likely to contain the logo of a beer company on it, and coffee cup is more likely to have a Starbucks logo than a Shell logo on it. A general object recognition toolkit can detect these common objects, and we can use these detections as additional features for our classifier. We can also either manually construct these semantic relations between brand and objects, or attempt to learn them by seeding each brand with a small number of canonical objects (e.g. Milka → chocolate bar, Starbucks → coffee) and using existing semantic databases like WordNet to find similar nouns with which we can expand our list of related objects. Alternately, we can learn them directly from training imagery, although that will require significantly more training data than we have access to currently.

References

[1] G. Bradski. Opencv. *Dr. Dobb's Journal of Software Tools*, 2000.

[2] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*, 2014.

[3] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[4] S. Romberg, L. G. Pueyo, R. Lienhart, and R. van Zwol. Scalable logo recognition in real-world images. In *Proceedings of the 1st ACM International Conference on Multimedia Retrieval*, ICMR '11, pages 25:1–25:8, New York, NY, USA, 2011. ACM.

[5] K. E. A. van de Sande, J. R. R. Uijlings, T. Gevers, and A. W. M. Smeulders. Segmentation as selective search for object recognition. In *IEEE International Conference on Computer Vision*, 2011.