A Comparison of Keypoint Descriptors in the Context of Pedestrian Detection: FREAK vs. SURF vs. BRISK

Cameron Schaeffer
Stanford University CS Department
camerons@stanford.edu

Abstract

The subject of keypoint detectors has been, for the past 20 years, a popular topic of research in the computer vision community. In the last five years, though, the domain has shifted. The advent of "mobile devices" such as the smart phone and tablet computer has brought with it a demand for computer vision on new platforms with restricted computing cycles and limited memory. This shift to mobile devices has resulted in a growing need for faster and more memory efficient keypoint descriptors, for applications such as panorama stitching, tracking, and object recognition.

I present a comparison of two novel keypoint descriptors with the well-known SURF descriptor in the context of my pedestrian detector. I achieve over 90% accuracy using FREAK descriptors with a Radial Basis Function SVM classifier using a Bag of Words model. I give charts comparing the speed and accuracy of BRISK and FREAK with SURF in the context of a pedestrian detection, and give valuable statistics related to the training of such a detector, which in turn will shed some led on the descriptors themselves.

1. Introduction

The focus of the project is to perform interesting evaluations to help programmers decide which keypoint descriptor is most suitable for the pedestrian detector application, and to produce code for a programmer to test such descriptors on his own windows environment. To my knowledge, at the time of this writing, a Bag of Words pedestrian classifier for Windows for these binary feature descriptors has not been openly sourced (although several similar classifiers exist for the Linux environment). My classifier is designed for pedestrian detection on still images, but can be trivially extended to run in real time on video. The classifier can also label where in the scene pedestrians are located by taking images marked "positive", doing a bounding box sweep over the image, and re-running the classifier on each of these sub-images, although this feature is not analyzed in the report.

The pedestrian detector is written in C++ using MSVC on Windows. It relies on a standard Bag of Visual Words model, using SURF, BRISK, or FREAK features as the possible word types. The detector provides benchmarks on two state of the art binary descriptors, FREAK [1] and BRISK [2], comparing them to the widely known SURF keypoint descriptor [3]. The pedestrian detector is run with each descriptor in turn on the NICTA database [4, 5], and compares performance tradeoffs such as speed and accuracy in different situations.

1.1. Motivation

FREAK and BRISK are novel binary descriptors that far surpass the industry standards. It is very rare that a new keypoint descriptor performs so much better than the predecessors in this field; SIFT was left unchallenged for almost a decade before SURF was developed. Now that this new descriptor has been developed, analysis is required so it can be immediately used by android developers and security camera companies for applications such as panorama stitching, object detection, tracking, navigation, etc. Although the bag of words object detector has been around for 7 years [7], it is novel to use apply it using these new descriptors and gauge performance in detecting pedestrians. We can learn more about the accuracy of FREAK and BRISK by applying them to a pedestrian detection problem on a real dataset.

2. Background/Related Work

2.1. Overview of SURF Descriptor

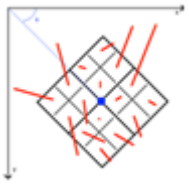


Figure 1. SURF HOG Descriptor

The feature vector of SURF is almost identical to that of SIFT. It creates a grid around the keypoint and divides each grid cell into sub-grids. At each sub-grid cell, the gradient is calculated and is binned by angle into a histogram whose counts are increased by the magnitude of the gradient, all weighted by a Gaussian. These grid histograms of gradients are concatenated into a 64-dimensional vector. The high dimensionality makes it difficult to use this in real time, so SURF can also use a

36-vector of principle components of the 64 vector (PCA analysis is performed on a large set of training images) for a speedup. SURF also improves on SIFT by using a box filter approximation to the convolution kernel of the Gaussian derivative operator. This convolution is sped up further using integral images to reduce the time spent on this step.

2.2. Overview of BRISK Descriptor

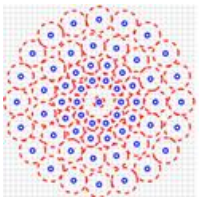


Figure 2. Brisk Sampling Pattern

BRISK is a 512 bit binary descriptor that computes the weighted Gaussian average over a select pattern of points near the keypoint (See Figure 2). It compares the values of specific pairs of Gaussian windows, leading to either a 1 or a 0, depending on which window in the pair was greater. The pairs to use are preselected in BRISK. This creates binary descriptors that work with hamming distance instead of Euclidean, and can be made to run extremely quickly using special SSSE hardware instructions for up to a 6x speedup [2].

2.3. Overview of FREAK Descriptor

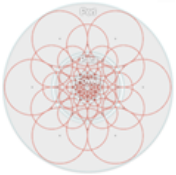


Figure 3. Freak Sampling Pattern

FREAK is also a binary descriptor that improves upon the sampling pattern and method of pair selection that BRISK uses. FREAK evaluates 43 weighted Gaussians at locations around the keypoint, but the pattern formed by these Gaussians is biologically inspired by the retinal pattern in the eye. The pixels being averaged overlap (see Figure 3), and are much more concentrated near the keypoint. This leads to a more accurate description of the keypoint as analysis will show.

The actual FREAK algorithm also uses a cascade for comparing these pairs, and puts the 64 most important bits in front to speed up the matching process, Unfortunately, this is not yet implemented in OpenCV so it does not affect our analysis, but we expect that when this is implemented, FREAK's matching step will speed up by an order of magnitude.

2.4. Related Work

There have been several benchmark studies similar to mine, evaluating keypoint descriptors in the past submitted to IEEE and others. [8] My work is novel though because FREAK, SURF, and BRISK are brand new descriptors that are not well analyzed yet. While some benchmarks actually compare all three of these descriptors [9], they have not given metrics a) in the context of pedestrian detection, b) on the windows platform, or c) using a Bag of Words framework. Much of the analysis already done concerns accuracy of correspondences, and while this is a valuable metric, it does not capture the behavior of an ensemble of keypoints as I do.

3. Approach

This section describes our algorithm step-by-step.

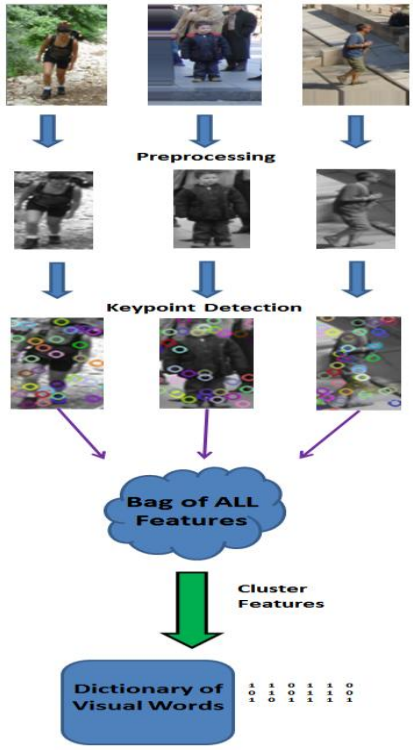


Figure 4. Building the Dictionary

3.1. Building the Dictionary

I begin by loading all positive images (with mirroring) and negative images (with sampling), and converting them to 8-bit grayscale for a total of 103172 images, half in 64x80 resolution, half in cropped from 1700x1200 resolution. I apply a DoG band pass filter to avoid aliasing, and then normalize image resolution to a fixed value to ensure that the features learned are uniform throughout training and testing. Next, I run the SURF keypoint detector to produce the keypoints. FAST [10] or multi-scale AGAST [11] keypoint detectors would also work fine instead if a speedup is desired. These keypoint vectors are fed into the descriptor (FREAK, BRISK, or SURF) to receive the keypoint vectors as described earlier. In negative images, if there are no keypoints, I just resample the image. I then take all the keypoints from every image ever seen (called the bag of features), and cluster them into a fixed number of clusters using K Means with Euclidean distance for SURF and hamming distance for the other two. The keypoint clusters from K-

Means become our codewords that we will use to describe images. We also store this vocabulary to a YAML file to save multiple hours on every subsequent test run.

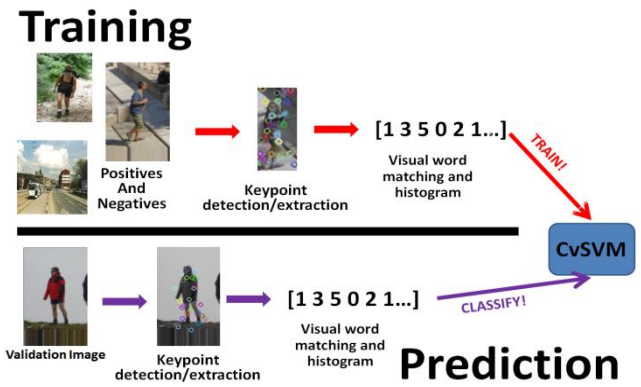


Figure 5. Training and Prediction Process

3.2. Training and Prediction

The next step is to train on our pedestrian dataset. We load the appropriate labeled positive and negative image files with bounding box information to begin training our pedestrian detector. From the positives, we crop to the region with the human, perform the same preprocessing as before, invoke the descriptor, and match each keypoint in the image to the closest feature in the dictionary. This is done with either a Brute Force Matcher OpenCV provides, or a FLANN based matcher. FLANN uses the Hierarchical K-means Tree for generic feature matching and this gives SURF an inherent advantage because binary features are not easily extended to hierarchical K Means. [12]

We make a histogram of the number of each visual word found in the image and this histogram (a vector of length Size of Dictionary) becomes our feature vector. For negative images, we do the same thing, but instead of cropping the human, we do 10 separate random crops. These feature vectors go into the built in CvSVM library (which takes in CvMats for training), and out comes a trained SVM. We save this SVM for later use using the OpenCV YAML save/load functions because the training step can take a very long time (hours on the full dataset). The prediction is performed in the same way: Take the validation image, detect and extract the keypoints. Match the visual words to dictionary words and histogram, then use the SVM to predict the class of this histogram feature vector.

4. Evaluation

Figure 6. Samples From NICTA Dataset

4.1. NICTA Dataset

I use the widely cited NICTA pedestrian dataset [5] to learn the dictionary/train/and evaluate on. The dataset has 25551 64x80 positive images and 5207 high resolution negative images. I mirror positives left/right for a total of 51102 positive training images. The negatives provided in NICTA are high resolution so I crop 10 sub images to get 52070 negatives (if there are no keypoints in the image, we reject the crop and try again). The dataset is explicitly divided into training and validation sets, with about 10% of the images being used in validation. All tests are performed on an Intel-2760QM @ 2.4 GHz with 8GB RAM. We do not multi thread for our benchmarks so only one core is used.

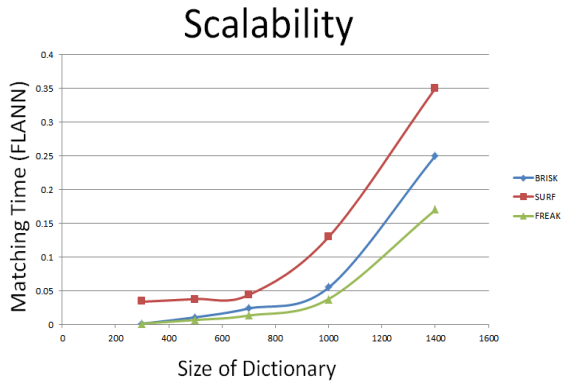


Figure 7. Matching Time vs. Size of Dictionary

4.2. Scalability of Keypoint Descriptors

We run tests to gauge the time it takes to match a keypoint to the entire dictionary of 1000 codewords. We do this for each descriptor on the same training set of all images, and we see from the graphs that SURF is roughly 2x slower than BRIEF, and BRIEF is about 1.5x slower than FREAK. There appears to be a roughly quadratic relationship between matching time and size of the dictionary for all three algorithms as we would expect for brute force matching. We see a similar quadratic effect with the FLANN based matcher as well, as seen in Figure 7.

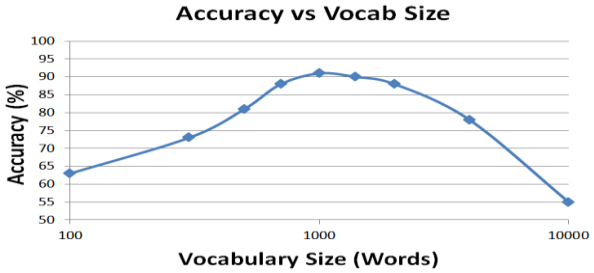
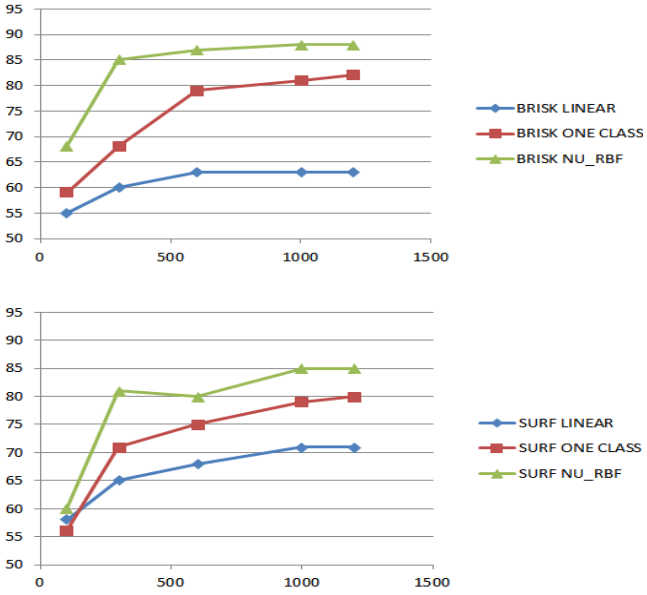


Figure 8. Choosing the Optimal Dictionary Size

4.3. Choosing Vocab Size to Avoid Over fitting

We determine the optimal dictionary size to achieve the maximum accuracy of the pedestrian detector. If we have too many words in the dictionary, we will overfit the training data and see quantization effects when we build our histogram (as seen by the rapid drop in accuracy as we get close to 10000 words). If we have too few words in the dictionary, we underfit the data and are not descriptive enough to distinguish between person and not person with only a few visual words. Notice that the number of vocabulary words needed to describe all the images can be partly dependent on the resolution of images (average number of keypoints). During testing, it was noticed that running on the INRIA dataset which has on average 5x as many keypoints as NICTA, that having roughly 3000 words in the vocabulary was optimal. This can be attributed to the fact that in higher resolution images, we get more keypoints per image so we should have a few more bins in the histogram without having to worry yet about quantization effects. The ratio of average number of keypoints to number of bins is important in the histogram process it would seem in order to avoid overfitting. The optimal number of bins for the NICTA dataset was about 1000 keywords for my rough sweep of vocabulary sizes using an exponential kernel SVM. For each vocabulary value in the plot, I did a log sweep of the SVM "C" value, and I display the optimal accuracy over all C values for each number of codewords in the plot to remove SVM calibration as a source of error. Because the accuracy is variable, I run the test on 4 sets of 1200 images that NICTA provides and plot average the accuracy over the 4 image test sets. Accuracy here is defined as percentage of correct classifications over the number of validation images.



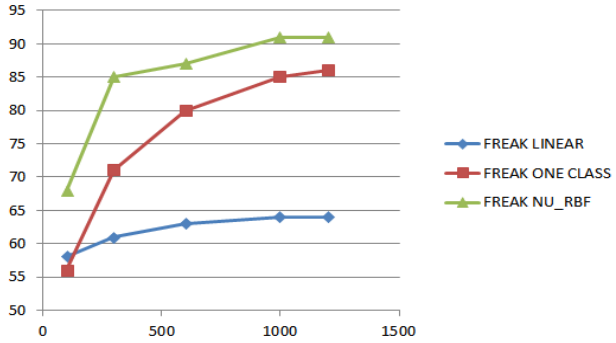


Figure 9. Accuracy (%) as a function of number of training images.

4.4. A Comparison of Accuracy vs. Training Size

We note that FREAK generally performs better than the other two algorithms (5% better on average than SURF) using a non-linear NU_RBF (nu-parameter tuned, radial basis function) SVM. We see that all the algorithms require about 200-300 training images to see a large spike in accuracy in the non-linear SVM and then accuracy begins to degrade after around 1100 training images. Also from this plot, we can see that the histogram of words for FREAK and BRISK are clearly not linearly separable because the linear SVM does not really improve much from random (achieving a peak of roughly 65% accuracy on the binary descriptors). The linear SVM seems to do well on SURF in comparison (around 6% better), but SURF in general is not as accurate as the other two with the SVM that performs best, the nonlinear exponential kernel SVM. The One Class SVM is an SVM built into OpenCV that takes in only positive training examples and attempts to wrap a decision boundary tightly around these examples. This SVM does not work as well as the RBF nonlinear one, but it still performs better than the linear version, supporting the theory that the data is not linearly separable. This could also occur because there are outliers in the positive data which affect the one class SVM much more because it has less training data to begin with. On the most trained non-linear SVM, FREAK performs at around 91% accuracy, while BRISK comes in second at 87% and SURF is close behind at 85% accuracy on the NICTA dataset with this bag of words pedestrian detector.

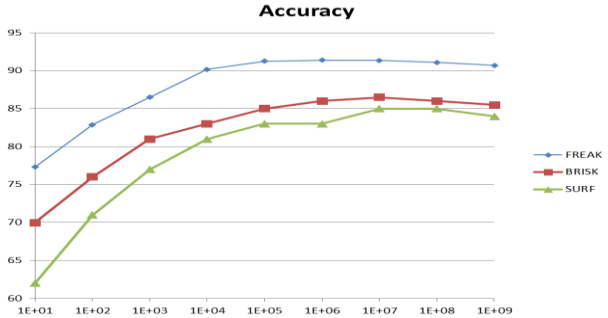


Figure 10: Accuracy (%) as a function of SVM Cost Parameter C (x axis).

4.5. Detector Weaknesses

SURF tends to underperform compared to FREAK and BRISK in accuracy. The binary FREAK provides a higher accuracy description of pedestrian related keypoints, but according to literature, this should be expected because it has a higher rate of matching correspondences (which I verified in testing) [12]. SURF scored higher recall and lower precision than BRISK and FREAK, implying that it is less conservative than the other two algorithms.

FREAK and BRISK generally did not perform well on pictures that had few keypoints and misclassified those. These tended to include people who were not facing the camera or who blended in well with the background, even though these cases were covered in training. However, the fault is more on the keypoint detector than the descriptor because it is the detectors job to provide enough keypoints for the descriptor in order to maintain accuracy. One problem with the descriptor though is that it tended to do rather poorly when the human pose was slightly off of upright (shoulders were not aligned). The training data seems to cover these cases so that is the fault of the keypoint descriptor to capture this pose accurately. I believe given more keypoints though, this could be made up for by stronger hits on other parts of the body than the shoulder. See Figure 11 examples.

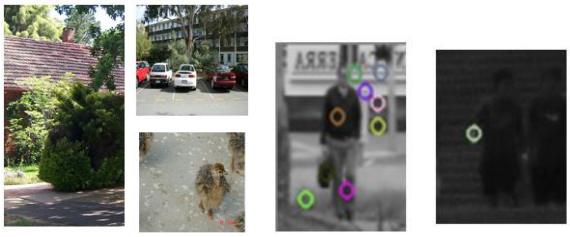


Figure 11 [Left]: Where's Waldo? SURF labeled these images as human. [Right]: Low keypoint images are often misclassified by FREAK and BRISK.

5. Conclusion

In conclusion, I have created a pedestrian detector using OpenCV for the windows platform. I determined that SURF is both a more robust and quick keypoint descriptor than BRISK and SURF, its forbearers. The detector works with 91% accuracy on the NICTA pedestrian dataset and most of the false positives can be attributed to lack of keypoints which are the fault of the SURF detector, or extremely difficult images. The framework I provide is a good start for someone trying to decide on a keypoint descriptor for their own applications. The work also shows that Bag of Words is a fine classifier to use with FREAK and gives the optimal dictionary sizes. Hopefully, over the next few years, we will witness a rise in computer vision applications on the mobile phone platform which is becoming more and more the most lucrative field in computer science. The invention of keypoint detectors such as FREAK and BRISK are the first step towards widespread computer vision applications for the mobile device industry, and before they become ubiquitous in this market, further performance analysis is required.

6. References

1. A. Alahi, R. Ortiz, and P. Vandergheynst. FREAK: Fast Retina Keypoint. In IEEE Conference on Computer Vision and Pattern Recognition, 2012.

2. Leutenegger, Chli, Siegwart. BRISK: Binary Robust Invariant Scalable Keypoints. ICCV 2011.

3. H. Bay, T. Tuytelaars, and L. Van Gool. Surf: Speeded up robust features. Computer Vision–ECCV 2006, pages 404–417, 2006.

4. K. Mikolajzyk and C. Schmid. "A Performance Evaluation of Local Descriptors", IEEE,Trans. Pattern Analysis and Machine Intelligence, vol.27, no.10, pp 1615-1630, October 2005.

5. G. Overett, L. Petersson, N. Brewer, L. Andersson and N. Pettersson, **A New Pedestrian Dataset for Supervised Learning**, *In IEEE Intelligent Vehicles Symposium*, 2008.

6. D. Lowe. "Distinctive Image Features from Scale-Invariant Keypoints", IJCV, 60(2):91–110, 2004.

7. L. Fei-Fei and P. Perona (2005). "A Bayesian Hierarchical Model for Learning Natural Scene Categories". *Proc. of IEEE Computer Vision and Pattern Recognition*. pp. 524–531.

8. Brehar, R. "A Comparative Study of Pedestrian Detection Methods using Classical Haar and HoG features versus BoW model Computed from Haar and HoG features" IEEE 2011, p299-306.

9. "A battle of three descriptors: SURF, FREAK, and BRISK" http://computer-vision-talks.com/2012/08/a-battle-of-three-descriptors-surf-freak-and-brisk/

10. Edward Rosten and Tom Drummond. "Fusing points and lines for high performance tracking", *ICCV Oct 2005.*

11. Elmar Mair, Gregory D. Hager, Darius Burschka, Michael Suppa, and Gerhard Hirzinger. "Adaptive and generic corner detection based on the accelerated segment test (AGAST)", In *Proceedings of the European Conference on Computer Vision (ECCV'10)*, September 2010.

12. Marius Muja, FLANN, Fast Library for Approximate Nearest Neighbors, 2011, http://mloss.org/software/view/143/.

IMPORTANT NOTE TO CS229 STAFF: This class was a joint project for both CS229 and CS331. I worked alone on it all quarter and hope to count the Machine learning aspect of it for this class and the computer vision aspect for CS331. The part for CS229 includes: the training process for the SVM such as C value tuning, procedure for tuning with respect to both C value and Vocab size as variables, save and loading SVM test harnesses, etc. The entire process of bag of words and general info is relevant to both classes. I have intentionally left out many sections related to computer vision work because they I feel they should not be double counted to keep this paper relevant to what should be said about machine learning, and I have done the same for the computer vision class with respect to the learning aspects.