Contours and Kernels: The Art of Sketching

Dan Guo, Paula Kusumaputri, Amani Peddada {DGUO1113,PAULAKSP,AMANIVP}@STANFORD.EDU

Abstract

Given sketches of everyday objects, we seek to accomplish two goals. The first is to classify the object depicted within the sketch. The second is to study different styles and varied interpretations in the depictions of the same object category. We employ supervised learning algorithms as the main method of classifying sketches into their respective classes, with our fine-tuned multiclass SVM attaining 57.7% accuracy. To understand varied styles and portrayals of objects – concepts that inherently are subjective and contain more vague metrics – we use unsupervised learning algorithms to discover common structure in the data. We find that by clustering, we group sketches with stylistically or structurally similar features. These groupings yield important insight into how we can obtain computational interpretation of more general, subjective qualities.

1. Introduction

Sketching has existed as one of the original ways humans have used to depict their narratives and journeys. Despite being such an ancient media of expression, this art form has been tremendously unexplored through computational methods, with only introductory work having been completed (Eitz et al., 2012). In our study, we aim to explore the domain of sketches in two ways.

First, we interpret sketches as carriers of information, depicting real life objects. In this part of the study, we want to see exactly how well sketches can be equated with their real life counterparts. Namely, we attempt to classify a sketch according to the object that is drawn.

Secondly, we consider sketches as an art form, where the individuality and creativity of the artist are emphasized. Given drawings of objects within a certain category, we note that there is a large degree of variability depending on the author of a sketch. We thus seek to understand what types of styles might be used in the creation of these drawings and what are different ways that artists might represent the same everyday object.

Stanford University

2. Dataset

Dataset. We base our analysis on an already existing crowd-sourced dataset, collected and released with the paper (Eitz et al., 2012). The dataset consists of 20,000 human-produced drawings, depicting 250 object categories. Each sketch is labeled with the correct object category, representing our ground truth values. The sketches are limited to only isolated objects, with no backgrounds, surrounding contexts, or colors, and are provided in png and svg (temporal order of strokes) format.

Features. Given the common use of bag-of-features feature extraction (Sivic et al., 2005) in image- and photo-based classification, we apply the same techniques to sketch-based classification. The features of sketches are derived using the Histogram of Gradients (HOG) technique (Dalal & Triggs, 2005). This featurizing algorithm divides the png images into a grid of cells; within each cell, the technique first builds a visual vocabulary of common line orientations. The method then counts frequencies of gradient orientation and builds a histogram from which 500 features can be extracted. The HOG descriptors are invariant to local scale geometric transformations, as the features are derived from a particular cell (Dalal & Triggs, 2005). We then use this feature representation, in the form of 500-dimensional real valued vectors, to represent each sketch.

3. Methods

We utilize various supervised learning algorithms for object classification in sketches. We test the algorithms (all except one-vs-one SVM) with 10 fold cross validation, reducing data loss. We also use unsupervised learning algorithms to understand the different styles and representations in a sketch. Because there does not necessarily exist a metric to test against, our style analysis is fairly qualitative and examines individual images to understand structure between sketches.

3.1. Supervised Learning

Note that KNN and GDA were algorithms used mostly for rapid prototyping. We implemented these to test the 10 fold cross validation wrapper algorithm and error analysis infrastructure.

3.1.1. K Nearest Neighbors

To serve as a guiding diagnostic, we implement a variant of the K Nearest Neighbor Classifier. The naive KNN classifier examines the training instance closest in Euclidean distance to a given test example by features, and then outputs that training example's category as the test example's prediction; this corresponds to the case when K = 1 (Guo et al., 2003).

We made several enhancements to this naive classifier. First it seemed less accurate for the label of the test example to be determined by only one training example. Therefore, we learned the optimal value of K by varying the values of K, finding that the best performance is achieved with K = 8 (see Figure 1).

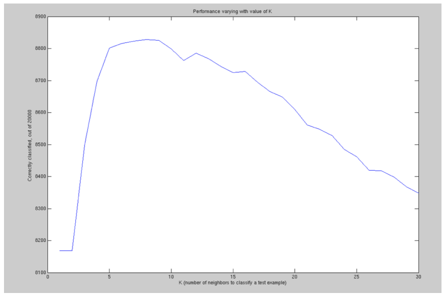

Figure 1. Plot of how classification accuracy varies with the value of K (number of neighbors). The accuracies are the average of 10 fold cross validation runs, with the modified voting scheme discussed.

We simultaneously modified how the K neighbors voted on the classification of the testing instance. A naive approach would be to tally up the classifications of the K neighbors and assign the test example the majority's classification (Guo et al., 2003). We instead enhanced the vote so closer neighbors have more weight in the classification of the test example. We categorize a test example X with N = $\{K$ nearest neighbors of X by Euclidean distance dist()$\}$ as the class i that maximizes $\sum_{n \in N} \frac{I\{class(n)=i\}}{dist(X,n)}$.

3.1.2. Gaussian Discriminant Analysis

The inspiration behind GDA is that our task is, fundamentally, predicting data on the similarity of input features, so it follows intuition to construct general models for our classes. Thus we describe the distribution of features for each of the 250 possible object categories. From this point, classifying a test example is equivalent to finding the class where the probability of observing the test example's features is maximized.

3.1.3. Multiclass SVM

With the high success of SVM's in classification-related problems, we implement two variants of the multiclass SVM as described in (Hsu & Lin, 2002).

One-vs-All SVM For each object category in our dataset, we train an L1 regularized, soft-margin SVM that classifies whether a given sketch belongs in that category or not. Thus, we learn a total of 250 SVM's. For a test example x, we loop through every classes's SVM, classifying x as i where i is

$$\arg\max_{i=1,\ldots,250} \sum_j \alpha_j^i K(s_j^i, x) + b$$

Of the different kernels experimented, the quadratic kernel $K(x,y) = (x^T y + c)^2$ performed optimally.

One-vs-One SVM For all pairs of categories X, Y, we generate an SVM ($SVM_{X,Y}$) that classifies whether a sketch is of X or Y. We classify test example Z as the class

$$C = \arg\max_C \sum_{X,Y:X<Y} I\{SVM_{X,Y} \text{ classifies Z as } C\}$$

Because this requires an SVM for all unique pairs of objects, the one-vs-one SVM algorithm requires $\binom{250}{2}$ SVM classifiers.

3.1.4. Neural Network

The superlative performance of neural networks within other computer vision tasks suggested they would be appropriate for this study. In specific, we build a single-layer Neural Network with 400 nodes, another with 600 nodes, a deep Neural Network with two layers of varying sizes, and 250 binary neural network classifiers for each category. We find that the network with two layers performed best, with 400 and 300 nodes, respectively, which follows a heuristic that the number of hidden nodes should be approximately the mean of the input and output layer sizes. Each Neural Network is a supervised feed-forward, back-propagation network, where the output layer is a softmax classifier, and the hidden layer contains a standard non-linear function (the $tanh$ function) applied at each level to our input features. During training, examples are fed through the network, and the results are used to calculate gradients that we then apply iteratively in Stochastic Gradient Descent to derive our matrices, biases, and other parameters, as is common with back-propagation (Rojas, 1996) .

Testing involves feeding new examples through the layers to obtain a vector of probabilities, which is then translated into the predicted category.

3.1.5. Hierarchical Classification with SVM

With the success of one-vs-all SVM, we decided to develop a more intuitive way of choosing which subsets of categories to train our SVMs on, to obtain more fine-tuned class recognition. We therefore chose to construct a *hierarchical tree of categories* (see Figure 2) from our data. We do so in the following manner: Begin with a set of 250 nodes (where each represents one object category). Iteratively create subtrees by setting the next two nodes (or other subtrees) that are most commonly misclassified for each other as children of a new internal node. This eventually generates a tree whose structure describes the level of misidentification between various sets of object categories, with entries sharing the same parent node being frequently misidentified. For every internal node in the tree, we then train a binary SVM to predict whether a test example belongs to the left or right child. Classifying a given example involves passing it through the tree, and outputting the first leaf node the sample reaches.

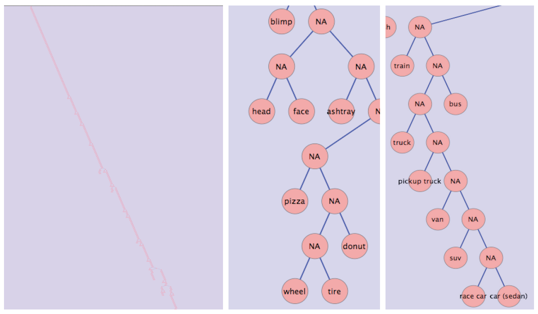

Figure 2. SVM Hierarchical Tree. The root of the tree is in the top left. Note that nodes that share a direct parent are more likely to be misclassified. We have two closeups showing the misclassifications in groups of vehicles and objects that look very similar.

3.2. Unsupervised Learning

3.2.1. K-Means Clustering

To understand what styles or representations of objects are commonly found, we strive to understand structure within an object category. For each object category, we run k-means clustering on its constituent points. However since a given object category may differ in the number of unique clusters, for each object category (e.g. mug, seagull, etc.) we learn the optimal number based on the data's silhouette values (Rousseeuw, 1987) of point i, $s(i) = \frac{b(i)-a(i)}{max\{a(i),b(i)\}}$ where $a(i)$ is the average similarity of i with all other data within the same cluster and $b(i)$ is the lowest average dissimilarity of i to any other cluster which i is not a member. We run k-means clustering using different numbers of starting centroids, each for multiple runs, and pick the number of cluster centers that routinely achieves the largest silhouette value over all data points.

4. Results

4.1. Supervised Learning Results

We test the data with various supervised learning algorithms, with the results below in table 1.

For sketch classification, we find that one-vs-all SVM produces the best results, with an accuracy of 57.7% that improves over the 56% performance provided by the paper (Eitz et al., 2012).

4.2. Unsupervised Learning Results

When we apply k-means to the object categories, we find optimal cluster numbers range from 1 to 10; we stop at this point, since further grouping begins to assign individual points to unique clusters, which does not yield additional information.

5. Discussion

KNN. KNN is a simple classifier examining the categories of the closest neighbors in order to make a classification. In a feature space where many different object categories exist in close proximity to one another, predicting simply by the nearness of neighbors is susceptible to noise. The error from KNN appears to be a variance problem that can likely be reduced with more data. A larger dataset from a representative distribution of artists can improve performance levels. We can understand this by recognizing that as the number of training sketches grows, a given test example will look more similar to a now larger set of training examples drawn from that same category. At the moment, we find that a test example is close in distance to a few sketches of the same class, but is also near drawings of other objects, which is the source of many misclassifications.

GDA. Observing the results in Table 1, it appears that GDA is susceptible to overfitting, as its training error is fairly low while its test error remains high, suggesting a variance problem with GDA. Specifically, GDA makes the assumption that the distribution of features given a class label is multivariate Gaussian. In many object categories, however, this assumption does not hold, since there are several common representations of the same object. Thus the variation of sketches may not follow a canonical Gaussian curve. In fact running a normality test confirms that most features with statistical significance are indeed not Gaussian given the class labeling.

Table 1. Training and Test Accuracy. All models (excluding one-vs-one SVM) were evaluated with 10-fold cross validation.

MODEL	TRAINING ACCURACY	TEST ACCURACY	DATA SET SIZE
GDA	99.76%	48.89%	18000 TRAINING, 2000 TESTING
KNN	100%	44.14%	18000 TRAINING, 2000 TESTING
ONE-VS-ALL SVM	100%	57.7%	18000 TRAINING, 2000 TESTING
ONE-VS-ONE SVM	100%	46.15%	18000 TRAINING, 2000 TESTING
NEURAL NETWORK	61.17%	43.45%	18000 TRAINING, 2000 TESTING
HIERARCHICAL CLASSIFICATION	100%	43.7%	18000 TRAINING, 2000 TESTING

SVM. The one-vs-one SVM is a cleaner algorithm to train as we build classifiers between only two distinct object categories, instead of the one-vs-all variant. The drawbacks, however, seem to outweigh this potential benefit. For instance, one-vs-one requires an SVM for every pair of distinct object categories, which is $\binom{250}{2}$ SVM's to train. This makes testing and handling the one-vs-one system extremely inefficient, such that 10 fold cross validation is infeasible. Additionally, it would not scale well to larger numbers of categories. Another drawback is our one-vs-one does not keep track of the score outputted by each SVM. Instead, the current implementation simply classifies an object by the class that the greatest number of pairwise SVM's decides on. This does not allow for more nuanced decisions, where the relative confidence from different SVM's is taken into account.

The one-vs-all SVM excels in the task of classification, outperforming the other algorithms by a considerable margin. This is perhaps because SVM's generally excel at binary classification tasks more so than other methods and are generally more robust, placing fewer restrictions on the structure of the data, as with GDA, for instance. Despite regularization, interestingly, the set of SVM's attains $\sim$100% accuracy on the training set. This implies that the data are linearly separable and do not have very large outliers, at least in the quadratic kernel space. It is also interesting to note that there are certain test examples where every one of the 250 SVM's achieves a negative score. This suggests that for each object X, the binary SVM classifies the test example as not of the object X. In many cases, this is an expected outcome, since the negative examples for each SVM encompass a larger range of features spanned by 249 categories, as opposed to those of the positive examples, which span only a single category. This could perhaps be addressed by picking random subsets of our negative examples for classifier training.

Looking at close ups in Figure 2 of our hierarchy tree, the common misclassifications of objects by the one-vs-all SVM demonstrate that object category labels play large roles in the accuracy, as there is significant overlap between some classes (either very similar to each other- barn and house- or one a subset of the other- chair and armchair).

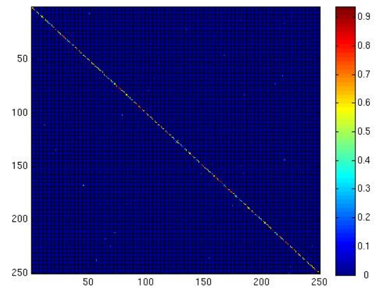

Figure 3. Confusion Matrix of Multiclass SVM (10-fold Cross Validation). Warmer values, or larger values correspond to more correct classifications. The left axis is truth classes and bottom axis is predicted classes.

This may suggest that we can actually build a set of categories that will allow us to classify more accurately, without losing additional information.

Hierarchical Classification. The idea behind implementing hierarchical classification is that with a clear hierarchy in object categories, SVM's can be used to decide between large groups of objects, and more specific SVM's can then progressively produce finer classifications. The central issue with this method is that the hierarchical tree of object categories that we generated is less balanced than what we had anticipated. Specifically, for the majority of the tree, an internal node has two children, one which is a leaf node and one which is the rest of the tree. Because of this structure, when we build an SVM classifier for each internal node, the classifier will often decide between a single object category (leaf node) and the rest of the objects (rest of tree). And if the SVM were to achieve a positive score on that object category, the algorithm simply classifies the test example as such. Since we do not consider the rest of the tree during our testing, there are many chances for a test example to incidentally be misclassified as the single object category. The hierarchical system thus did not lend itself well to classification because its structure is prone to premature classification decisions.

Neural Network. It is also interesting that the Neural

Network's performance was not as superlative as expected, given NN's common use in computer vision classification tasks. This might be simply due to the parameterization of the network – more finely tuned functions and weights may be needed – but most likely it is because of the large variability in the training and testing data, such that the network tries to capture too large a range of relationships between components within the sketches, thus producing less definitive results. Therefore, a standard network may not actually be suited to this task, and different variants of the network might need to be considered (Convolutional Neural Networks, for example).

K-Means Clustering. To understand styles and different interpretations in sketches, we cluster within particular object categories, using pairwise distance as a metric. These clusters capture quite well the variability with which the object is drawn. Quite often, the clusters within an object category mirror the diversity of drawing style and representations of the same object. To give a few examples, in Figure 4, we have samples from the two bear clusters. Cluster one represents more realistic bears with four paws. In cluster two, the bears are standing on its hind legs. These are fundamentally two different positions that a bear can assume and the clustering algorithm clearly separates these two cases. As a second example, the two clusters of vans are based on the level of detail in the sketch. The vans in cluster one are all very busy, with many lines sketched onto the side of the vehicle; the vans in the second cluster have an empty body and are more simplistic in flavor. The style of the sketch – in this case the rhetorical choice of the level of realism or abstraction to depict the van – is a way the algorithm has clustered the sketches. We likewise see in our clusters a similar trend with object orientation, where sketches within a category are grouped by alignment.

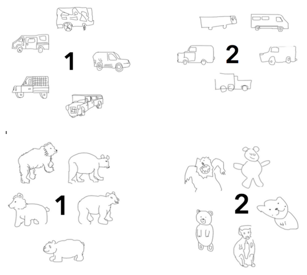

Figure 4. Vans (top) and Bears (bottom) Clustering. Clusters by k-means clustering.

Because our features are related to the distribution of line-orientations, we have confidence in the clustering results. Diverse orientations and representations of the same object produce varied orientations of lines, which the HOG features are able to capture and k-means is able to cluster between. With more abstract or simple sketching styles, the line orientation distributions will be more sparse and the features are able to distinguish between very busy and very cleanly drawn objects. Note however that there are object category clusters that do not have any apparent differences. For instance, we expected that clustering in the category of books would result in opened and closed books, but instead each cluster has both opened and closed books, and with varying viewpoints of the object.

6. Conclusion and Future Work

We have presented various algorithms to classify a dataset of sketches into their object categories. We have demonstrated that we are able to achieve reasonable classification of 57.7% using multiclass one-vs-all SVM. We also evaluated different classifiers with varying degrees of success, though none outperform the one-vs-all SVM.

We believe sketching is a vast domain to explore, and would like to continue improving performance on object recognition within sketches. We aim to further pursue the approach of a hierarchical cluster by building the hierarchical tree in a more sophisticated manner. This might produce a more balanced structure, thus improving classification.

Another possibility is to experiment with more elaborate features that include spatial and temporal information about the sketch, to better learn the more subjective components and drawing style. With a more complex feature extraction method, we could additionally cover up portions of the image, extract features from the image, and classify the image to understand which parts of the image are most correlated with its identity. This yields insight on what makes a certain category what it is and perhaps how we ourselves internally identify everyday objects.

References

Dalal, Navneet and Triggs, Bill. Histograms of oriented gradients for human detection. In *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 1 - Volume 01*, CVPR '05. IEEE Computer Society, 2005.

Eitz, Mathias, Hays, James, and Alexa, Marc. How do humans sketch objects? *ACM Trans. Graph. (Proc. SIGGRAPH)*, 31(4):44:1–44:10, 2012.

Guo, Gongde, Wang, Hui, Bell, David, Bi, Yaxin, and Greer, Kieran. Knn model-based approach in classification. In *On The Move to Meaningful Internet Systems 2003: CoopIS, DOA, and ODBASE*, pp. 986–996. Springer, 2003.

Hsu, Chih-Wei and Lin, Chih-Jen. A comparison of methods for multiclass support vector machines. *Neural Networks, IEEE Transactions on*, 13(2):415–425, Mar 2002. ISSN 1045-9227. doi: 10.1109/72.991427.

Rojas, Raúl. *Neural networks: a systematic introduction*. Springer, 1996.

Rousseeuw, Peter J. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, 20(0):53 – 65, 1987. ISSN 0377-0427.

Sivic, J., Russell, B.C., Efros, A.A., Zisserman, A., and Freeman, W.T. Discovering objects and their location in images. In *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, volume 1, pp. 370–377 Vol. 1, Oct 2005.