Sketch Recognition: Template Matching

Trung-Hieu Hoang
FIT - HCMUS
1512159@student.hcmus.edu.vn

Thanh-Lam Do
FIT - HCMUS
1512275@student.hcmus.edu.vn

Tuan-Anh Le-Duong
FIT - HCMUS
1512002@student.hcmus.edu.vn

Duy-Tam Nguyen
FIT - HCMUS
1512479@student.hcmus.edu.vn

Minh-Triet Tran
FIT - HCMUS
tmtriet@fit.hcmus.edu.vn

Abstract

Sketch images is an effective way to convey information that is hard to describe using text. Furthermore, with the popularity of touch devices, sketch understanding is an interesting area for computer vision scientists. The authors implement "template matching algorithm" to recognize sketch image and experiment them on a part of the TU Berlin sketch dataset. We adjust parameters in available method and analysis the result. In achievement, our highest accuracy up to 32.47%. For the future work, we want to improve this technique for getting higher accuracy and apply in building application that using sketches to search for products on-line.

Keyword: *free-hand sketch, sketch recognition, template matching, support vector machine.*

1. Introduction

Free-hand human sketches are usually incredible intuitive to humans. In some way, a sketch speaks for a hundred words and it is easily to recognized across cultures and languages. One of the most important tasks of computer vision is developing Sketch Understanding System.

The exiting image search-engines‘ have already done a good job in catalog-level image [1]. In some way, it does not have so much meaning because people can easily describe using text. Imagine that you go to your friends house and there is a beautiful table that you really want to buy it. If you search the keyword "table" on the Internet, you will found thousands kind of table. It is not easy to find the table that you really want to buy. In contrast, you takes out your smart-phone and draws sketch the table using fingers; the Sketch Recognition System will found the most familiar table with the users sketch.

Sketch classification is a challenging task even for humans because of its diversity and abstract structure[2]. The first basic Sketch Classification System was developed in 2012 by Mathias Eitz, James Hays and Marc Alexa [3]. They designed a robust visual feature descriptor for sketches. This feature permits not only unsupervised analysis of the dataset, but also the computational recognition of sketches. In the following year, SYMetric-aware Flip Invariant Sketch Histogram (SYM-FISH) structure was used in sketch classification [4]. In 2015, Free-hand sketch recognition by multi-kernel feature learning was developed and achieved the accuracy 65.81% [5].

Deep Learning was established in 2015 by Y LeCun, Y Bengio, G Hinton. Since there, Deep Learning is used in a lot of fields and one of them is object recognition. In image sketch-understanding field, Deep Convolutional Neural Network a powerful tool for researcher to build partial sketch understanding systems [2, 6].The Sketch understanding system using this CNN can reach the accuracy up to 77.69% [6]

In this paper, we re-implement "Template matching algorithm" and adjust its parameters. We reach the highest accuracy up to 32.47%. Although this is not the best available methods in 2017 but we think it will help us easier to implement and understand the algorithm with low configured computer.

We mention **Related Works** in section [2]. Section [3] will explain about the data-set and issues related. Sketch

recognition by Template Matching method is presented in section [4]. Content in section [5] is our experiment's result. We will also outline the experimental process and suggest a capable application in the future.

2. Related Works

In 2017, Machine Learning and Deep Learning occupy an important role in the Computer Vision field. They are used to solves the difficulties of predicting including classification, recognition, detection, etc. However, to implement these techniques require a large memory and time performance.

There are Sketch Image Recognition experiments using Machine Learning or Deep Learning. "Deep Sketch: Freehan Sketch recognition using deep features"[7] uses two popular CNNs. Another project is "Deep Sketch: Deep Convolutional Neural Networks for Sketch Recognition and Similarity Search" [8] reach accuracy 75.42% on TU-Berlin dataset and "Sketch-a-Net: a Deep Neural Network that Beats Humans"[9] reach accuracy up to 77.06% with . They use Deep Convolutional Neural Networks. And any other methods[10][11][12]

Our experiments result is not good as experiments above. However, we implement and experiment using Image Gradient - this method seemly does not receive much attention. Our experiments find out many problems to exploit. From that, it provides directional approaches to reduce performance of algorithm and optimizing on preparation step when using Machine Learning.

3. The Sketch Database

We use the publicly freehand sketch database from sketch dataset of TU-Berlin benchmark. Dataset contains of 20,000 unique sketches distributed over 250 object categories. Each sketch has a size of 1111px x 1111px. Type of all pictures in dataset are PNG (Portable Network Graphics). We pick randomly 40% dataset to train and predict.

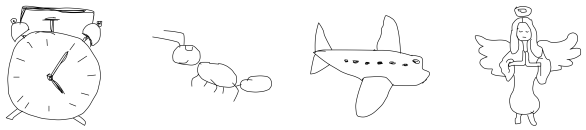

Figure 1 4 sketch images from TU-Berlin Benmark

4. Method

4.1. Overview

The Template matching algorithm considers the Image Gradient of each sketch - image. The main idea to recognize is each different drawing object will have different histograms of gradient direction. For the pre-processing step, we will calculate the percentage of gradient-direction in each image. For prediction step, we will apply SVM for linear regression.

Particularly, first we apply Sobel operator[13] to find the direction and magnitude of gradients. After that, we divide the gradients into k bins base on its direction. We calculate and choose features of objects and train it by using Support Vector Machine model. Finally we recognize a new data by calculate the errors between it and objects and get the label with the smallest error.

From our survey on TU Berlin dataset, we use this method because of two conclusions following:

More than 80% cases, sketches of same objects produce histograms with same widths that have features is defined as high density bins are smilar.

More than 67% cases, sketches of different objects produe histograms with same widths that have features are different.

4.2. Gradient Vector: Sobel

In our first pre-processing step, we scale all images to size of 255px x 255px. Then we try Sobel operator[13].

Sobel is an operator or filter to find the 2-D image Gradients and emphasizes regions of high spatial frequency that correspond to edges. Sobel works with grayscale image. Of course, sketch images are always presented with two colors. Even, in many cases, the image have more than two colors, we can union any colors. After applying Sobel, we receive two parameters: magnitude and the direction.

We apply Sobel Kernel 3x3 in small image such as images with 100px x 100px or images with thin brushstroke, we see the gradient's directions clearly. But with large image, kernel 3x3 does not show the directions well and exactly. In this case, we apply larger kernel, for example, 5x5 or more. You can apply another operators which are same as Sobel, for example, Prewitt, Scharr, etc.[14] They return similar results.

4.3. Divide into k bins

After processing with Sobel operator, we plot the histogram[15] of gradient's direction.

We have a set of Image Gradients. Since the gradient, whose is equal 0, does not have direction, we consider the gradient of which the magnitude is not equal 0. In this case, it is called meaning gradient and the pixels which have meaning gradient is called meaning pixel.

We plot the histogram of meaning gradient collection. We divide it into k bins such as number of bins is 10, 15, 20, 22, 25, 27, 29, 50, etc in sequence. However, we see the best range of k's choice is $[23..30]$. We choose two alarm clock images and two air plane images randomly and plot

their histograms, results as:

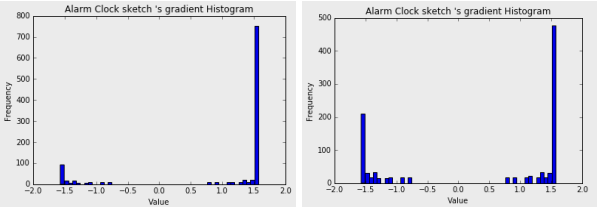

Figure 2.1 Alarm clock sketch's gradient histograms

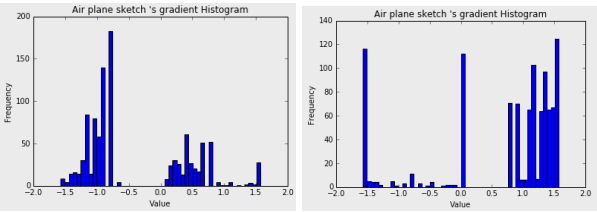

Figure 2.2 Air plane sketch's gradient histograms

In figure 2.1 we can see, histograms of two alarm clock images have one feature clearly. In Figure 2.2 we can see, histograms of two air plane images are not similar alarm clock's. Histograms of them is not clearly as Figure 2.1, they produce similar densities in histograms but two similar bins in two histograms have a certainly distance. We present method to solve this problem in 4.4.

We divide 180° into k equal bins. Each bin contains the number of direction's values calculated as follows:

$$\begin{aligned} range[i] &= \big[L[i], R[i]\big) \quad i = 0, 1, 2...k-1 \\ L[i] &= \frac{\pi}{k} * i + \frac{-\pi}{2} \\ R[i] &= \frac{\pi}{k} * (i+1) + \frac{-\pi}{2} \end{aligned} \tag{1}$$

The interval of gradient's direction is $[\frac{-\pi}{2}, \frac{\pi}{2}]$ following above formula, $\frac{\pi}{2}$ is in no bins. That does not make sense. However, about meaning, $\frac{-\pi}{2}$ and $\frac{\pi}{2}$ are present one in the direction.

As we mention above, the best range of k's choice is [23..30]. We can not demonstrate this choice but we notice that after adjusting parameters k from 5 to 60.

That can be explained as follows: In case with small k, in other words, there is a little bins so the presentation of features is not clear. With large k, the intervals of each bins are too small, so with the gradients that have close directions by many reasons of drawing, they are divided into different bins. It is too difficult to identify object's features.

After that, in bin i^{th}, we count the number of meaning pixels which have the direction belong to $range[i]$, in other words, the direction's pixels is in $[L[i], R[i])$.

$count[i]$ is the number of meaning pixels in $range[i]$.

Because of the difference among the number of meaning pixels in images, we calculate the ratio of number of meaning pixels in each bins to number of all meaning pixels in the image:

$$P[i] = \frac{count[i]}{\sum_{i=0}^{k-1} count[i]} \tag{2}$$

4.4. Nomalize Data

An object in sketch images or normal images are plentiful in the shape, size and tilt. We solve the problem about size by scaling at first. Now we normalize all sketch images of an object.

We fix on one to three images. Then we rotate all images left that their tilt become more synchronous. Tile's problem is showed below:

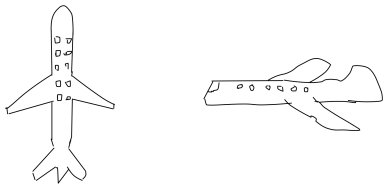

Figure 3 Air plane sketch from dataset

In Figure 3, two air plane's image are similar, however one object is drawn horizontallyone, the other drawn vertically.

We base on the sets of k bins we make on previous step. For an image, we have a vector of percentage of meaning pixels in all bins as $[P[0], P[1], ...P[i]...P[k-1]]$

We look at images, choose only one to fix and get the vector of this image. After that, with all images left, we shift right all element in the vector k times. Each times, we check the errors between the processing image with the fixed image. We define:

$P[i]$: percentage of meaning pixels in bin i^{th} of the fixed image.

$M[i]$: percentage of meaning pixels in bin i^{th} of the processing image.

$N[i][j]$: percentage of meaning pixels in bin i^{th} of the processing image in j^{th} turn.

$error[i][j]$: error in bin i^{th} between the fixed image and the processing image in j^{th} turn.

$E[j]$: average error of the processing image in turn j^{th}

$$\begin{aligned} x &= \mid P[i] - N[i][j] \mid \\ error[i][j] &= \frac{1}{1+e^x} - 0.5 \\ E[j] &= \sum_{i=0}^{k-1} error[i][j] \end{aligned} \tag{3}$$

Result vector certainly is a vector made from available vector and have a $\min(E)$ after shift right t times. We save result vector as vector T.

4.5. Support Vector Machine (SVM)[16]

A Support Vector Machine (SVM) is a supervised learning[17] (we given labeled training data and predict label of new data). SVM is a discriminative classifier formally defined by a separating hyperplane.

In this experiment, a training data point is viewed as a k-dimensional vector (vector T). SVM help us find linear regressions to classify m classes (m objects). For m classes, this algorithm return $\frac{m(m-1)}{2}$ linear regressions.

Now we present about the linear regression between two classes. In SVM algorithm, it give a hyperplane which has the maximum margin, it means distance between the nearest observation and the hyperplane should be maximized.

We use SVM on hyperplane k dimensions. We present a data point as:

$$X = \begin{bmatrix} p[0] & p[1] & p[2] & \dots & p[k-1] \end{bmatrix}$$

As simple machine learning, we train data to classify two objects, first step in SVM is finding parameters for this funtion:

$$f(x) = \beta_0 + \beta^T x, \tag{4}$$

In (4) β is known as the weight vector and β_0 as the bias.

The solution of function[1] is infinity that SVM fix as below:

$$|\beta_0 + \beta^T x| = 1 \tag{5}$$

VM use geometry to find the distance between a point x and a hyperplane (β, β_0):

$$\text{distance}_{\text{ support vectors}} = \frac{|\beta_0 + \beta^T x|}{||\beta||} = \frac{1}{||\beta||}. \tag{6}$$

so double the distance called M is the closest distane between two data points that one point belong to first object, another point belong to second object.

$$M = \frac{2}{||\beta||} \tag{7}$$

The target of SVM is finding the maximum of M mean finding the minimum of $||\beta||$. SVM support the constraints model the requirement for the hyperplane to classify correctly all the training examples x_i. Formally,

$$\min_{\beta,\beta_0} L(\beta) = \frac{1}{2}||\beta||^2 \text{ subject to } y_i(\beta^T x_i + \beta_0) \geq 1 \; \forall i, \tag{8}$$

where y_i represents each of the labels of the training examples.

SVM using Lagrange multipliers to obtain the weight vector β and the bias β_0 of the optimal hyperplane.

Then we present the multiclass classification.

4.6. Prediction

A new data point X is presented by a k -dimensional vector. To predict, with each linear regression, SVM algorithm find the distance from point X to 2 margins and choose the class with smaller distance. Class label is the class with highest count.

5. Conclusion

In this paper, we have presented template matching algorithm to recognize sketch image. Our right prediction archive average 32.47%. Although it is not good as many available methods, we want to show an idea about using gradient to recognize sketch image because twos reason: it is easier to understand the algorithm; it can be implemented with low configured computer.

This method is not a best way, it have problems in many cases we present below:

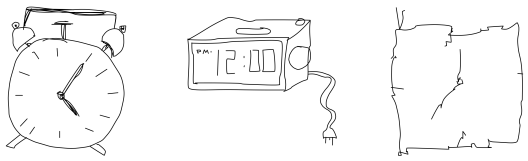

Figure 4 Alarm clock sketches from TU-Berlin dataset

With each objects, there are many sketches with different expressions (see Figure 4). The weakness of the algorithm is passive in training, with different expressions, it is fail on presenting data on hyperplane k dimensions and find a function f(x).

Another reason is many objects have histograms whose bins are equal high. A It is difficult to get the features of the object base on gradient.

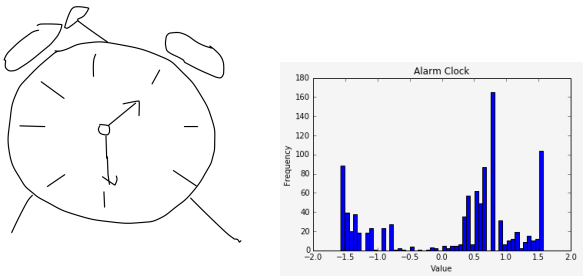

Figure 5.1

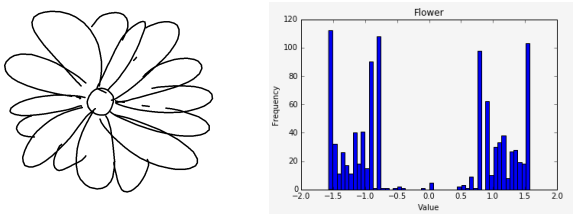

Figure 5.2

Figure 5.1 is an alarm clock sketch from dataset and its histogram. Figure 5.2 is a flower sketch from dataset and its histogram. They are pretty similar on the shape of histogram.

References

[1] Y.-Z. S. T. X. T. M. H. C. C. L. Quian Yu, Feng Liu, "Sketch me that shoe," *CVPR*, pp. 800–807, 2016.

[2] E. Boyaci and M. Sert, "Feature-level fusion of deep convolutional neural networks for sketch recognition on smart phones," *IEEE International Conference on Consumer Electronics (ICCE)*, 2017.

[3] M. A. Mathias Eitz, James Hays, "How do humans sketch objects?," *ACM Trans. Graph*, vol. 31, no. 4, pp. 1–10, 2012.

[4] S. L. X. G. L. L. Xiaochun Cao, Hua Zhang, "Sym-fish: A symmetry-aware flip invariant sketch histogram shape descriptor," *ICCV*, pp. 313–330, 2013.

[5] S. M. Omar Seddati, Stephane Dupont, "Free-hand sketch recognition by multi-kernel feature learning," *Computer Vision and Image Understanding*, 2015.

[6] S. M. Omar Seddati, Stephane Dupont, "Deep sketch 2: Deep convolutional neural networks for partial sketch recognition," *IEEE*, 2016.

[7] B. R. V. Sarvadevanbhatla R. K, "Freehand sketch recognition using deep features," *Computer Vision and Pattern Recognition*, 2015.

[8] M. S. Seddati O, Dupont S, "Deep sketch: Deep convolutional neural networks for sketch recognition and similarity search," *Content-Based Multimedia Indexing 13th International Workshop*, 2015.

[9] S. Y. X. T. H. T. Yu Q, Yang Y, "Sketch-a-net: a deep neural network that beats humans," *Computer Vision and Pattern Recognition; BMVC*, 2015.

[10] L. J. H. T. Field M, Valentine S, "Sketch recognition algorithms for comparing complex and unpredictable shapes," *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence*.

[11] P. B. J. J. P. J. W. A. T. P. Hammond T, Logsdon D, "A sketch recognition system for recognizing free-hand course of action diagrams," *Proceedings of the Twenty-Second Innovative Applications of Artificial Intelligence Conference*.

[12] S. Y. L. X. S. L. Liu L, Shen F, "Deep sketch hashing: Fast free-hand sketch-based image retrieval," *Computer Vision and Pattern Recognition*, 2017.

[13] S. I, "An isotropic 3x3 image gradient operator," 2014.

[14] L. G, "Prewitt, sobel and scharr gradient 5x5 convolution matrices," 2012.

[15] T. B. Dalal N, "Histograms of oriented gradients for human detection," *Computer Vision and Pattern Recognition*, 2005.

[16] S.-T. J. Christianini N, "An introduction to support vector machines and other kernel-based learning methods," *Cambridge University Press*, 2000.

[17] V. V. N. Chapelle O, Haffner P, "Support vector machines for histogram-based image classification," *IEEE Transactions on Neural Networks*, 1999.