

# Sketch Classification using Deep Convolutional Neural Network for Interactive Drawing Interface

---

Byeongchan Lee  
[lee@sas.upenn.edu](mailto:lee@sas.upenn.edu)

Pedro Rizo  
[prizo@seas.upenn.edu](mailto:prizo@seas.upenn.edu)

Nikhil Jamdade  
[jnikhil@seas.upenn.edu](mailto:jnikhil@seas.upenn.edu)

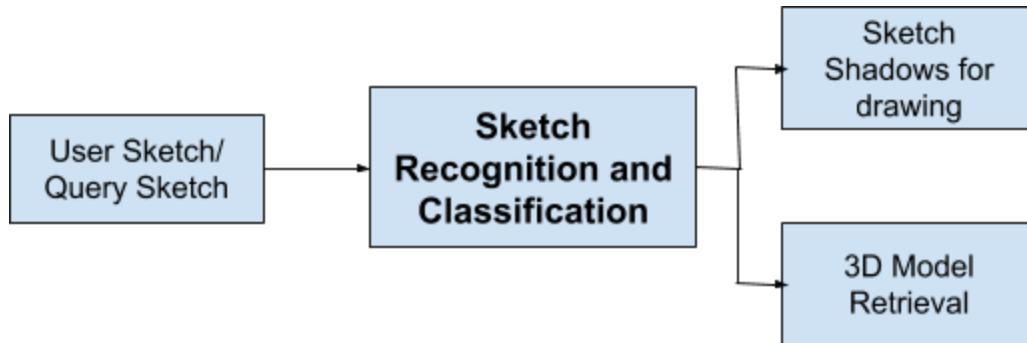
## Abstract

*An interactive interface is designed to improve drawing skills and retrieve 3D shapes based on sketch classification and sketch matching. The system involves input, output and computational steps to recognise and classify free-hand sketch using deep learning neural network approach. This paper explores the classifier built using a convolutional neural network (CNN) based on the Sketch-a-Net's CNN architecture. We not only replicated the architecture but further experimented with the model by applying it to unseen data ("real world data") to understand zero-shot learning. Moreover, we have experimented with the architecture with multi-channel generalisation that encodes sequential ordering in the sketching process, and a multi-scale network ensemble with joint Bayesian fusion that accounts for the different levels of abstraction exhibited in free-hand sketches. The primary objective of this project is to develop and experiment with set of algorithms and classifiers in order to accurately classify drawing / human drawn sketches into correct categories.*

## 1. Introduction

Sketches are very intuitive to humans and have long been used as an effective communicative tool. With the proliferation of touchscreens, sketching has become a much easier undertaking for many – we can sketch on phones, tablets and even watches. The recognition of free hand drawn sketches/drawing is extremely challenging task unlike image recognition. This is particularly because of sketches are highly iconic and abstract with highly varied different level of abstractions/details. Sketches lack visual cues as they consist of black and white lines instead of colors. Initially sketch and image recognition generally followed conventional methods such as feature extraction using HOG or SIFT etc and feeding them to the classifier like SVM. However these methods does not count the unique abstract and sparse nature of sketch, moreover they ignore sequential stroke ordering in sketch. The project follows deep neural network (DNN) approach from Sketch-a-Net. Recently DNN especially deep convolutional neural network have achieved tremendous success in replacing representation hand crafting with representation learning for variety of vision and Machine Learning problems. An

important advantage of DNNs, particularly CNNs compared with conventional classifiers such as SVM's lies with closely coupled nature of presentation learning and classification i.e from raw pixels to class labels in a single network which makes the learned feature representation maximally discriminative.



<Figure 1: Flow Chart for Basic Project Overview>

## 2. Background

### 2.1 Data Augmentation

Data Augmentation is commonly used with CNNs to reduce overfitting. We performed data augmentation by replicating the sketches with a number of transformations. For each input sketch, we did horizontal and vertical shifts.

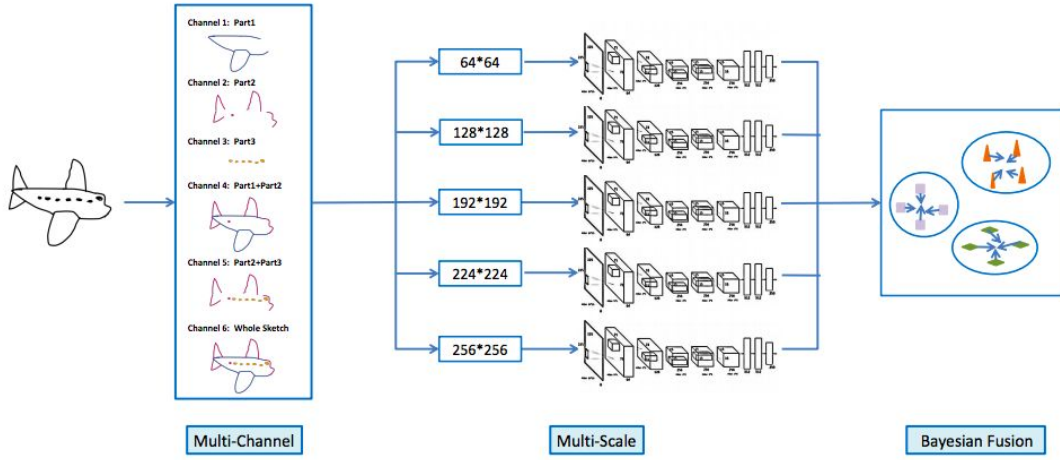
### 2.2 Zero Shot Learning

Zero-shot learning (ZSL) is a variant of multi-class classification problem where no training data is available for some of the classes. Most ZSL algorithms use some connection between the available information and the unseen classes. Zero-shot learning is being able to solve a task despite not having received any training examples of that task. It entails the classification of images where there are no labeled training data.

## 3. Data

The data set contains a total of 20,000 sketches that were obtained by crowdsourcing and were made by non-experts/(non artists), which provides value because we want to capture the “rough” or often “inaccurate” drawings that people use to represent high level abstractions of daily things like a bird or a toothbrush. Furthermore, the 20,000 sketches are divided into a total of 250 categories that try to provide a representative sample of many of the daily words used in human interactions. Each of the categories contains a total of 80 sketches.

## 4. Methodology



<Figure 2: Overall Framework>

### 4.1 CNN for Sketch Recognition

It uses five convolution layers each with rectifier (ReLU) units, while the first, second and fifth layers are followed by max pooling. The filter size of the sixth convolution layer is  $7 \times 7$  which is the same as the output from the previous pooling layer which works like fully connected layer. The softmax loss is placed after the final layer. The unique aspects of the architecture for sketch modelling compare to other network are larger first layer filter, no local response normalisation, larger polling size, higher dropout etc.

Index	Layer	Type	Filter Size	Filter Num	Stride	Pad	Output Size
0		Input	-	-	-	-	$225 \times 225$
1	L1	Conv	$15 \times 15$	64	3	0	$71 \times 71$
2		ReLU	-	-	-	-	$71 \times 71$
3		Maxpool	$3 \times 3$	-	2	0	$35 \times 35$
4	L2	Conv	$5 \times 5$	128	1	0	$31 \times 31$
5		ReLU	-	-	-	-	$31 \times 31$
6		Maxpool	$3 \times 3$	-	2	0	$15 \times 15$
7	L3	Conv	$3 \times 3$	256	1	1	$15 \times 15$
8		ReLU	-	-	-	-	$15 \times 15$
9	L4	Conv	$3 \times 3$	256	1	1	$15 \times 15$
10		ReLU	-	-	-	-	$15 \times 15$
11	L5	Conv	$3 \times 3$	256	1	1	$15 \times 15$
12		ReLU	-	-	-	-	$15 \times 15$
13		Maxpool	$3 \times 3$	-	2	0	$7 \times 7$
14	L6	Conv(=FC)	$7 \times 7$	512	1	0	$1 \times 1$
15		ReLU	-	-	-	-	$1 \times 1$
16		Dropout (0.50)	-	-	-	-	$1 \times 1$
17	L7	Conv(=FC)	$1 \times 1$	512	1	0	$1 \times 1$
18		ReLU	-	-	-	-	$1 \times 1$
19		Dropout (0.50)	-	-	-	-	$1 \times 1$
20	L8	Conv(=FC)	$1 \times 1$	250	1	0	$1 \times 1$

<Table 1: The Architecture of Sketch-a-Net >

## 4.2 Modelling Sketch Stroke Order with Multiple Channel

A multi channel architecture is designed to model the sequential ordering of strokes in each sketch. The sequential ordering is strong cue in human sketch recognition. The method discretize the strokes into three sequential groups and treating these parts as different channels in first layer. These three parts are further used to generate six images containing combinations of the strokes parts. First three images contain the three parts alone, next contain pairwise combinations of strokes and third is original sketch of all parts.

## 4.3 A Multi-scale Network Ensemble with Bayesian Fusion

A multi-scale network ensemble to address the variability in abstraction and sparsity, followed by a joint Bayesian fusion scheme to exploit the complementarity of different scales. In this method each network in the ensemble is learned a model of varying coarseness by blurring training data to different degrees. The 5 network ensemble are created by blurring -downloadsampling and then upsampling by to the original 256 x 256 pixel image size. The downsample sizes are 256, 224, 192,128,64. Each network in the ensemble is independently trained by backdrop using one of these blur levels.

## 5. Results

First, we explored the unseen data performance of our own implementation of classifier based on CNN and the classifier based on Sketch-a-Net methodology.

Trained over 50 classes	Trained over 250 classes
	
Train Accuracy : 0.943	Train Accuracy : 0.6716
Test Accuracy : 0.42	Test Accuracy : 0.224

<Table 2: CNN-based Classifier Unseen Data Performance>

The results shown above, can be seen initially as a low accuracy, but in both cases, the accuracy obtained on unseen data is considerably higher than the pure random choice result. In the case of the 50 classes, the random guess is 0.02 and the obtained result on the test is 0.42, which corresponds to a 21x increase compared to the random guess, which makes the result more acceptable. The same happens in the case of the 250 classes in which the random guess is 0.004 and the accuracy obtained on unseen data is 0.224. In this case we obtained a 56x increase to the random guess. Hence, the results for the full 250 categories might seem low at a first glance but are actually better considering the complexity of the 250 class model.

It's worth mentioning that the results provided above do not include the stroke order feature extraction of the original dataset and we are just working with the pure sketch images. Furthermore, the results presented were obtained by training our CNN in only 35 examples per class, since the model proved too complex for our machines, so we had to scale down the training sample of data to be able to correctly train the network. Finally, because of the same constraint of computational resources available, we were unable to perform the desired data augmentation techniques described in the original paper. Despite not being able to implement them in the final results, we experimented with expanding the training set of sketches by flipping each sketch horizontally, vertically, in both axes, and rotating 90 degrees and -90 degrees. Hence, for each image we were able to expand the original training set of images by 5x. This presented an increase especially in test performance for the experiments performed.


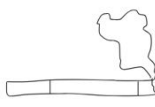


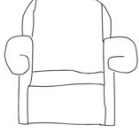

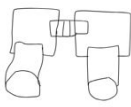



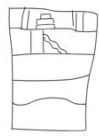

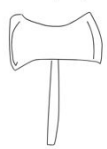



Having mentioned the limitations of our implementation, we can compare with the original Sketch-a-Net based classifier's results (0.6397 for the first epoch, and 0.6411 test accuracy for the second epoch). With this comparison, we can observe the great improvement that the classifier can obtain by performing both the data augmentation techniques on the training data and by taking into account the stroke order feature set. Hence, these results show promise for future work, and highlighting the role of data augmentation and correct feature representation.

Secondly, we explored the possibility of zero shot learning by not providing training data for select few classes - chair, cigarette, panda, and pumpkin - and then having the classifier recognize some instances of those classes. The idea behind this experiment was to show that the classifier is actually able to recognize the different shapes and parts that come together to create the sketches. This also is meant to show promise of how the model can learn what a sketch means, by being able to associate it with other similar items or classes. The results are shown in Table 2 and the visual interpretation is shown in Figure 3.

	<b>Top1</b>	<b>Top2</b>	<b>Top3</b>
<b>Chair</b>	Armchair (102/ 200 = 51%)	Bear (14 / 200 = 7%)	Axe (8 / 200 = 4%)
<b>Cigarette</b>	Arm (60 / 200 = 30%)	Banana (48 / 200 = 24%)	Baseball Bat (47 / 200 = 23.5%)

<b>Panda</b>	Binoculars (25 / 200 = 12.5%)	Bookshelf (18 / 200 = 9%)	Bear (18 / 200 = 9%)
<b>Pumpkin</b>	Canoe (67 / 200 = 33.5%)	Apple (35 / 200 = 17.5%)	Butterfly (15 / 200 = 7.5%)

<Table 2:Results: Zero Shot Learning Experiment>

Chair 	Cigarette 	Panda 	Pumpkin 
Armchair (51%) 	Arm (30%) 	Binoculars (12.5%) 	Canoe (33.5%) 
Bear (7%) 	Banana (24%) 	Bookshelf (9%) 	Apple (17.5%) 
Axe (4%) 	Baseball Bat (23.5%) 	Bear (9%) 	Butterfly (7.5%) 

<Figure 3: CNN-based Classifier: Zero Shot Learning Experiment>

To expand a bit more on the results, the classifier wasn't provided with training examples of "chair", "cigarette", "panda" or "pumpkin". The results show some interesting insights into how the classifier works and learns the classes. For example, as expected, the class "chair" was properly associated with "Armchair" on 51% of the times, given the resemblance on the form of their sketches. On the case of the "Cigarette", it was almost evenly recognized as "Arm", "Banana" and "Baseball Bat" with 30%, 24% and 23.5% respectively. In this second example we can see that all the categories hold some resemblance of having similar and mostly linear shapes with not a lot of filling and mostly very simple patterns. In the case of the "Panda", the results were a bit surprising since the top result was "Binoculars" with 12.5%, which after taking a closer look makes sense given the shape of the distinctive eyes of the "Panda" models given in the Sketch data set and it's resemblance with the shape of the front part of the "Binoculars". Not as surprising, we see the class "Bear" coming in third with 9%. Finally, the "Pumpkin" class was classified as "Canoe" 33.5% of the times. This is due to the similar shape that the typical inner pumpkin lines with the overall canoe sketch. In second place is the "Apple" class with 17.5%.

These results of the Zero-Shot experiment show promise on the potential of learning certain classes just by the resemblance of the sketches to some other similar classes. The ideal result behind this is that eventually even without having seen a single example of “Chair”, but having seen similar classes such as “Armchair”, the CNN would be able to correctly understand the “Chair” as a class on its own. This is something that could be explored as a future work.

## **6. Discussion**

### **6.1 Unseen Data Performance**

For testing unseen data with our implementation of only CNN-based classifier, same number of images from each 50 classes are used. If the model did not learn after training and guessed arbitrarily on test data, the result would have been near 0.02. Given this base line, we can conclude that our implementation of convolutional neural network for sketch recognition can effectively recognize and classify unseen data. However, the result is relatively low compared to the experiment data for the sketch-a-net implementation that incorporates stroke ordering and multiscale network ensemble with Bayesian fusion, which produces an unseen data accuracy of 0.64 after just two epochs. The result confirms that unique features found in sketches such as stroke orders can be the key to performance improvement for the sketch recognition system.

### **6.2 Possibility of Zero Shot Learning**

The predictions for the labels, for which no instances is provided to the model based on our CNN implementation for training, exhibited an interesting result. Top 3 predictions for those labels with no training data - “chair”, “cigarette”, “panda”, and “pumpkin” - shared some important characteristics with the actual label. They tend to have similar orientation, shape, or pattern. We saw this result as a promising sign for the possibility of zero-shot learning since this may indicate that the model / network is actually learning semantic information about the sketch, which, in turn, can be used to inform a sketch recognition system so that it can predict labels without any training data but with provided semantic information.

## **Reference(s)**

- [1] M. Eitz, J. Hays, and M. Alexa. *How do humans sketch objects?* In SIGGRAPH, 2012
- [2] Q. Yu, Y. Yang, F. Liu. *Sketch-a-Net: a Deep Neural Network that Beats Humans.* In BMVC 2015