Doodle Recognition and Generation through Neural Networks

Lydia Xu (leediaxu@stanford.edu), Vera Xu (veraxrl@stanford.edu), Ying Chen (yingchen107@stanford.edu)

Video link: https://youtu.be/bsYS2xgKWgQ

What we are solving?

Can we recognize the "bee" in the below human-drawed doodles?

- Classification for sketch drawings or doodles has been a popular and challenging task in Computer Vision
- Applied the state-of-the-art neural models to the Google **Quick, Draw!** Dataset with image-based models for the doodle recognition task.
- Employed data augmentation and MobileNet comparison
- Explored RNN-based generative model for generation

Motivation

- Build an educational tool for kids to learn how to draw
- Create a classification tool to understand graphic symbols or logographic characters (such as Chinese characters)

Data

- Google **Quick, Draw!** Dataset
  - 50 millions real-user drawing collected
  - 340 label categories (e.g. bee, apple, river, etc.)
  - Data format: Nx3 stroke vector
- Data example

A stroke vector where each stroke consists of two lists of points.
- Word: class label "bee"
- List of points in x axis
- List of points in y axis


```
{
 "word":"bee",
 "drawing":[
  [
   [130,113,99, ...],
   [72,40,27, ...]
  ],
  [
   [76,28,7],
   [136,128,128]
  ],
  ...
 ]
}
```


- Stroke-based vs image-based
  - We transformed origin dataset into image-based model since sequential strokes doesn't provide much additional insights above the completed drawing
- Stream process for loading large size data
  - We split the data into 100 shards, each shard contains 340 categories as whole information.
  - 90 shards for training, 5 shards for validation and 5 shards for testing

Approach

- CNN-based architecture with categorical cross-entropy loss and ReLU activation layers (Figure 1 and Table 1)
- Data Augmentation: flip horizontally and random zoom (0.8-1.2). Selectively augment only 50% of the training data.

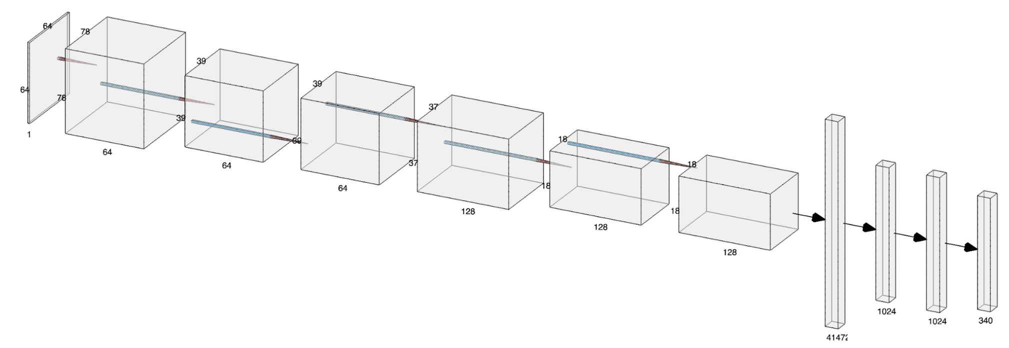


Figure 1. Convoluted Neural Network Architecture Diagram

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 78, 78, 64)	640
max_pooling2d (MaxPooling2D)	(None, 39, 39, 64)	0
dropout (Dropout)	(None, 39, 39, 64)	0
conv2d_1 (Conv2D)	(None, 37, 37, 128)	73856
max_pooling2d_1 (MaxPooling2D)	(None, 18, 18, 128)	0
dropout_1 (Dropout)	(None, 18, 18, 128)	0
flatten (Flatten)	(None, 41472)	0
dense (Dense)	(None, 1024)	42468352
dropout_2 (Dropout)	(None, 1024)	0
dense_1 (Dense)	(None, 340)	348500

Table 1. Convoluted Neural Network Architecture Table

- Keras MobileNet: a model using depth-wise separable convolutions to reduce computation and enhance efficiency. Using deeper and more complicated neural networks.
- Stretch goal: RNN-based generative model Magenta sketch-rnn to generate drawings based on pre-trained models.

Figure 2. Depth-wise Separable Convolution

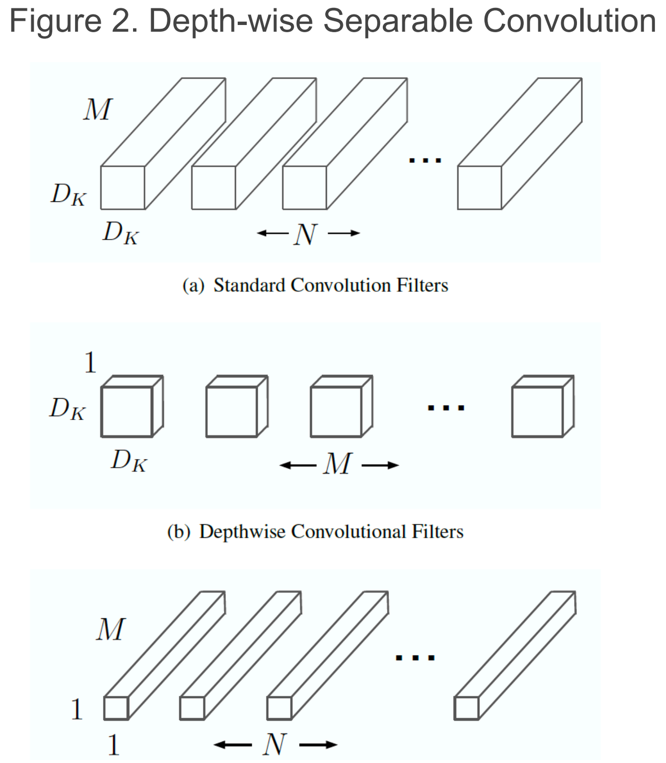


Table 2. MobileNet Model Architecture Table

Type / Stride	Filter Shape	Input Size
Conv / s2	$3 \times 3 \times 3 \times 32$	$224 \times 224 \times 3$
Conv dw / s1	$3 \times 3 \times 32$ dw	$112 \times 112 \times 32$
Conv / s1	$1 \times 1 \times 32 \times 64$	$112 \times 112 \times 32$
Conv dw / s2	$3 \times 3 \times 64$ dw	$112 \times 112 \times 64$
Conv / s1	$1 \times 1 \times 64 \times 128$	$56 \times 56 \times 64$
Conv dw / s1	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 128$	$56 \times 56 \times 128$
Conv dw / s2	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 256$	$28 \times 28 \times 128$
Conv dw / s1	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 256$	$28 \times 28 \times 256$
Conv dw / s2	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 512$	$14 \times 14 \times 256$
$5\times$ Conv dw / s1	$3 \times 3 \times 512$ dw	$14 \times 14 \times 512$
$5\times$ Conv / s1	$1 \times 1 \times 512 \times 512$	$14 \times 14 \times 512$
Conv dw / s2	$3 \times 3 \times 512$ dw	$14 \times 14 \times 512$
Conv / s1	$1 \times 1 \times 512 \times 1024$	$7 \times 7 \times 512$
Conv dw / s2	$3 \times 3 \times 1024$ dw	$7 \times 7 \times 1024$
Conv / s1	$1 \times 1 \times 1024 \times 1024$	$7 \times 7 \times 1024$
Avg Pool / s1	Pool 7×7	$7 \times 7 \times 1024$
FC / s1	1024×1000	$1 \times 1 \times 1024$
Softmax / s1	Classifier	$1 \times 1 \times 1000$

Howard, A.G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., & Adam, H. (2017). MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. ArXiv, abs/1704.04861.

Result and Analysis

Table 3. Accuracy for Different CNN-models

	Test Loss	Test Accuracy	Top_3_Accuracy
Baseline	1.888	55%	N/A
CNN	1.899	56.94%	75.54%
MobileNet	0.6995	81.59%	93.17%

- Vanilla CNN model beats the baseline by 1.7% after fine tuning with an accuracy of 56.77%.
- MobileNet reaches a high accuracy of 81.59%, beating the baseline by 26.59%.
- Introducing "Top 3 Accuracy" as a metrics because many categories look alike or they are hard to learn. For example, it is hard to distinguish between duck and swan:

Figure 3. Images with Duck label

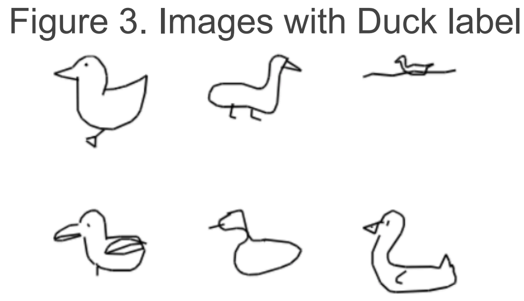
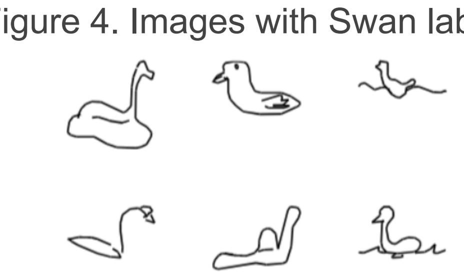
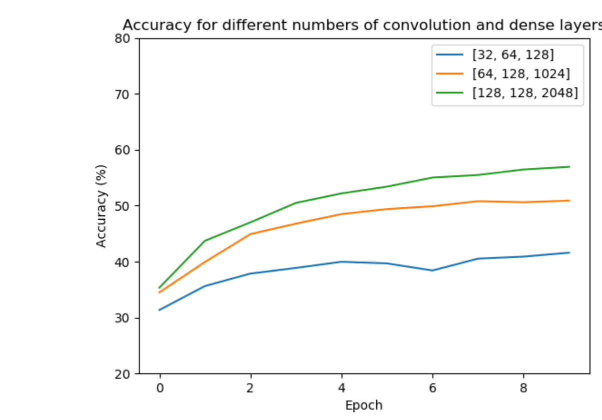
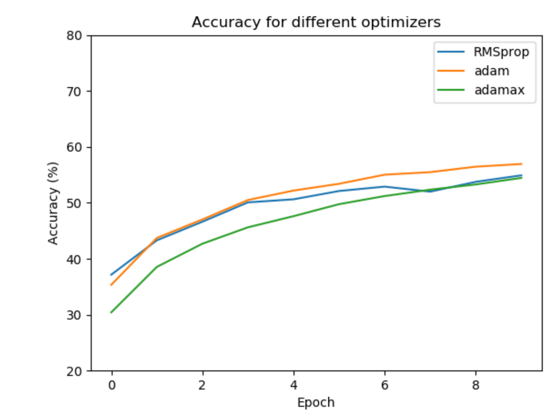


Figure 4. Images with Swan label



- Stroke-based model doesn't out-perform image-based model.
- Data Augmentation improved accuracy slightly by less than 1%.
- Different optimizers, image sizes and number of CNN layers have different effect on the results.





Conclusion

- Complicated CNN models after fine tuning can be very good at doodle classification (MobileNet for example).
- It is hard to achieve perfect accuracy scores as many human-generated drawings are strongly subjective. Even other human cannot distinguish between different categories.
- Image-based CNN models perform comparatively to baseline's stroke-based RNN models
- Future Work: doodle generation based on trained models