GROUP 9: DRAWING A CONCLUSION – PREDICTING DOODLES USING NEURAL NETWORK

Yifu Hu, Zihuan Qiao, Weixi Li, Calvin Guan
{yfhu,zhqiao,liweixi,cxguan}@bu.edu

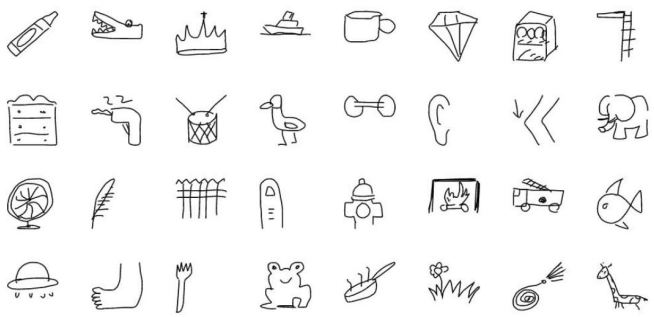

Figure 1. Quick, Draw! doodles

1. Project Task

We are participating in the Quick, Draw! Doodle Recognition Challenge competition on Kaggle. The goal of our project is to predict the labels for doodles drawn in the Google Quick Draw game. Quick Draw game prompts a label, the player attempts to sketch the label, and the Google AI guesses the label from the drawing. The input is a doodle image, some examples are shown in figure 1. Our output is the predicted probability that the drawing is of each of the 340 categories, the 3 categories with the highest probabilities will be submitted as the prediction. The challenges of this project lie in two aspects: very large and noisy dataset. Doodles in the training set can be incomplete or not sufficiently relevant to the labels, as the organizer stated, our task is "to build a recognizer that can effectively learn from this noisy data and perform well on a manually-labeled test set from a different distribution".

2. Related Work

Convolutional neural networks (CNN) have been successfully applied to solving the large-scale image classification problem. Krizhevsky et al. [1] proposed the standard structure of CNN which consists of stacked convolutional layers, followed by local response normalization, ReLU nonlinearity, and some fully-connected layers. Based on this structure, variants of CNN models were proposed, aiming to learn a more powerful model at a lower computational cost while seeking to reduce overfitting. The VGG-16 algorithm is proposed by Karen Simonyan and Andrew Zisserman [2]. This CNN algorithm grows the very deep net with 16 layers and a large number of weights in each layer. Salient parts in the image can have an extremely large variation in size. To solve this problem in image classification, GoogLeNet [3] is designed to be "wider" and more computationally efficient. Deep residual networks (ResNet) is proposed by Kaiming He, etc. and is designed to handle vanishing gradients encountered in deep CNN's. It uses a residual learning framework to ease the training of networks that is substantially deep. The Squeeze-and-Excitation Networks (SE-Net) won the first place for the ILSVRC 2017 classification competition. Unlike prior research which sought to strengthen the representational power of a CNN by enhancing the quality of spatial encodings throughout its feature hierarchy, the SE-Net [5] focused on the channel relationship and proposed a novel architectural unit, which they term the "Squeeze-and-Excitation" (SE) block, that adaptively recalibrates channel-wise feature responses by explicitly modeling interdependencies between channels.

3. Dataset and Evaluation Metric

The dataset that we will be using comes from the open sourced Quick Draw Dataset. Where the training data consist of over 50 million drawings across 340 categories of drawings. The test data has 112,163 drawings without labels. There are two forms of data – raw and simplified. The raw dataset contains the pixel coordinates of drawings and the time since the first point of when that pixel was placed, for each stroke made in the drawing. The simplified dataset removes

the timing information, standardizes the coordinates, resamples strokes with 1-pixel spacing, and simplifies strokes. For simplicity, we only work with the simplified dataset in out project.

In the data preprocessing step, we converted the original data which is presented in the form of pixel coordinates of drawings to images. In the preliminary experimental phase, we randomly sampled 10% of the full dataset, visualized around 13,000 images (256*256) from each of 340 categories to form our preliminary dataset and randomly divide into sets as shown in Table 1.

Table 1. Small dataset (10% of the full dataset) summary: number of images in each category, 340 categories in total.

Training	Validation	Test	Total
10,400 (80%)	1,300 (10%)	1,300(10%)	13,000

Metric: Our evaluation metric is the Mean Average Precision @3 (MAP@3). For each drawing, we submit three ordered predicted labels. The score is 1 if the first label is correct, .5 if the second is correct, .33 if the third is correct, and 0 if none is correct. The average of the scores is taken over every drawing in the test data. Formally, MAP@3 is defined as

$$MAP@3 = \frac{1}{U}\sum_{u=1}^{U}\sum_{k=1}^{min(n,3)} P(k),$$

where U is the number of scored drawings in the test data, $P(k)$ is the precision at cutoff k, and n is the number of predictions per drawing.

4. Approach

The input for one sample is a doodle image. Our output is an array of length 340, corresponding to the probabilities the doodle should belong to each of the 340 categories. Eventually, the top three most possible categories will be predicted. As a multi-class classification problem, the loss function we are using is the cross-entropy loss, which is defined as

$$L(\theta) = -\frac{1}{N}\sum_{i=1}^{N}\sum_{k=1}^{K} y_{i,k} log(h_k(x_i,\theta)) \ ,$$

where N denotes the number of total images, K denotes the number of classes (in our problem $K = 340$), $y_{i,k}$ is the indicator of image *i* belonging to class *k*, and $h_k(x_i,\theta)$ is the hypothesis for the corresponding model.

Given such a large dataset with over 0.13 million images per class, it's very time-consuming (more than 60h per epoch) to train a deep neural network on it, especially when SCC is the only GPU source we have access to. Thus, we need a way to get a subset of the full dataset and train deep neural networks on the smaller dataset. An ideal choice of the small data set would lead to a lower computational cost at the expense of reasonable accuracy loss. In addition to the randomly sampled small dataset introduced in the last section, we applied denoising techniques and proposed a way to get denoised sub-datasets.

The denoising technique works in the following steps: first train a model with SE-ResNet on the random 10% subset of the full dataset. We choose SE-ResNet because it's the most state-of-the-art deep neural network model for image recognition in our candidate models. Then we evaluate and obtain the feature vector of each image in the whole dataset using the trained model. For each class, rank the softmax of feature vectors with ascending Euclidean distance to the one-hot vector corresponding to that class. By using the Euclidean distance, we assume the feature vectors are Gaussian distributed with the identity covariance matrix. The Euclidean distance between two vectors x and y is defined as $d(x, y) = \sqrt{x^T \cdot y}$. Then the estimated center of each category is acquired by taking the average of the top 5% ranked images' feature vectors. Finally, pick the top 1% (2%, 5%, and 10% respectively) ranked images and form version1 (version2, version5, version10, respectively) denoised subset of the full dataset.

Finally, in the model selection step, we are going to compare four classical variants of the CNN model described in the related work section, namely VGG16, InceptionV4, ResNet50 and SE-ResNet50 in terms of their validation accuracy and cross-entropy loss. Since the test results on the 10% random dataset outperformed the results corresponding to the denoised dataset, which will be shown in the result section, the comparison experiment will be using the 10% random dataset.

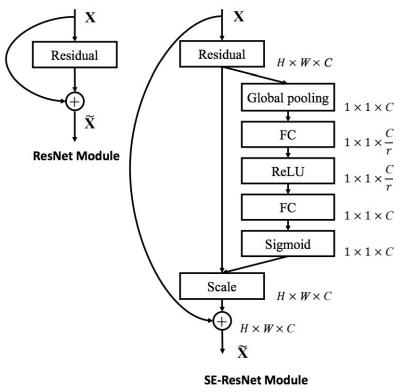

Figure 2. Structure of SE-ResNet vs ResNet model

Although our approach is based on the existing classical models, we retrain all the models mentioned using the Google Quick Draw dataset from scratch. In order to tackle with the big size and low quality of the quick draw dataset, a lot of work also needs to be done on data preprocessing and data denoising. To better visualize the effect of our data denoising method, t-SNE[6] visualization is also implemented accordingly.

5. Training and Denoising

After the initial training of SE-ResNet, we obtained millions of 340-dimensional feature vectors of quickdraws. To get more information about the training data, we randomly picked feature vectors of 128 classes out of 340 classes then use t-Distributed Stochastic Neighbor Embedding (t-SNE) technique [6] to reduce the dimension of 340-d feature vectors to 2-d then plot the scaled features using the 2-d vectors. As shown in Figure 3, the training data is very noisy. So we use our denoising scheme to make the data cleaner.

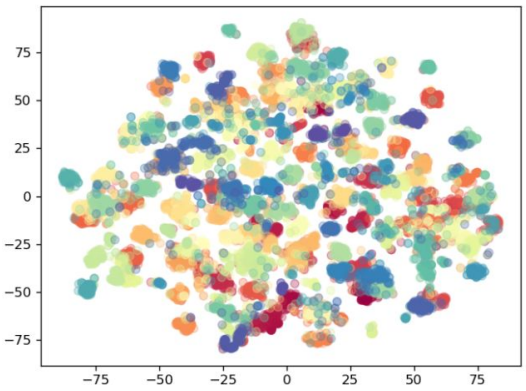

Figure 3. 128 classes t-SNE visualization with noise

After removing most of the noisy feature vectors, we plot Figure 4 by using the same metric as plotting Figure 3. As shown in Figure 4, the data is much cleaner. Thus, we apply our denoising metric to the full data and get the training datasets as described in section 3 then retrain our model.

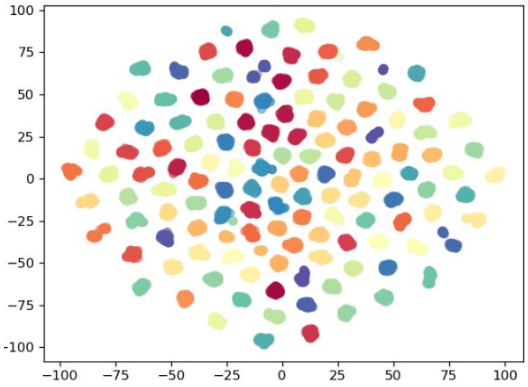

Figure 4. 128 classes t-SNE visualization after denoising

Table 2 shows the datasets we use and the results of training. From Table 2, we could infer that our denoising scheme reduced the variance of the data, resulting in over-fitting of the re-trained models.

Table 2. SE-ResNet50 training results

Datasets	Training Accuracy	Kaggle Score
Denoised 1%	99.28%	63.87%
Denoised 2%	99.63%	69.76%
Denoised 5%	99.63%	71.74%
Denoised 10%	99.43%	74.39%
Random 10%	77.66%	84.42%

6. Results

Epoch accuracy and top-3 accuracy for Inception-V4, Resnet50, and VGG16 models for 3 epochs are provided below. Table 3 shows that the SE-ResNet50 model maintained the highest accuracy and the lowest loss for each epoch. Thus we will use the Se-ResNet50 model moving forward.

Table 3.(Accuracy, **Top-3 Accuracy**, Loss) comparison of different networks

Epoch	SE-ResNet 50	ResNet50	Inception-V4	VGG-16
1	75.21%, **82.87%**, 0.85	75.25%, **82.78%**, 0.86	74.67%, **82.38%**, 0.87	74.27%, **81.80%**, 0.92
2	77.13%, **84.35%**, 0.78	76.97%, **84.13%**, 0.79	76.13%, **83.51%**, 0.82	73.56%, **81.29%**, 0.95
3	77.61%, **84.70%**, 0.76	76.86%, **N/A**, 0.80	75.28%, **82.93%**, 0.85	N/A, **N/A,** N/A

Learn Rate Comparison

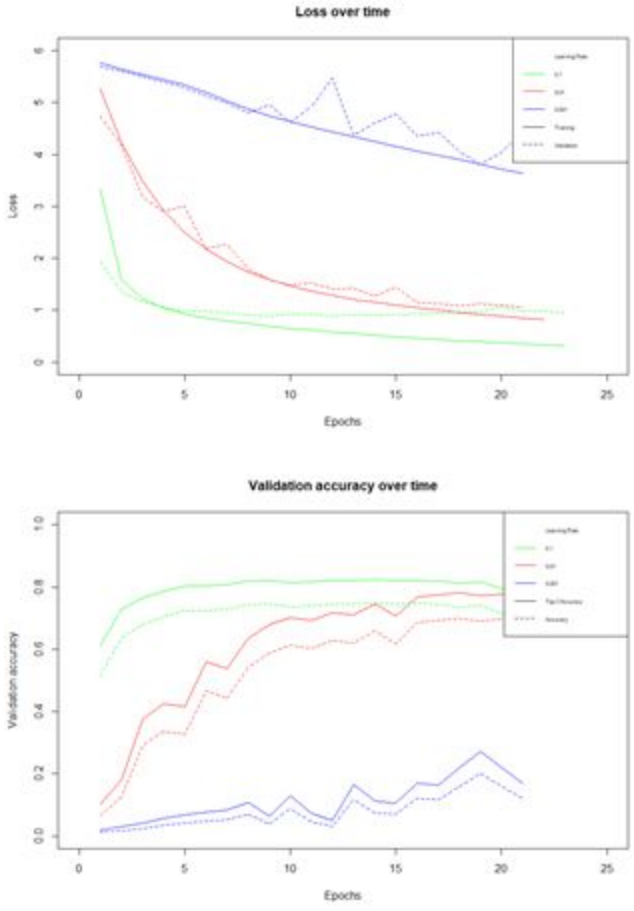

Figure 4. Loss and Accuracy of Learn Rates

We compared 3 different learning rates to gauge its effect on our Se-ResNet50 model. The learning rates are scaled by the batch size. We sampled 1% of the drawings for each label in order to run more epochs. Figure 4 shows that the smallest learning rate reduces loss very slowly, thus training is very slow. On the other hand, the highest training rate decreases training loss quickly, but the validation loss suggests overfitting occurs after a few epochs. Our results suggest that with multiple epochs, choosing a moderate learning rate is optimal.

Table 4. Kaggle Leaderboard Example

#	change	Team	Score	Entries
1	-	[ods.ai] Pablos	0.95480	79
2	+2	Guanshuo Xu	0.95330	76
3	-1	mgchbot	0.95327	79
...	-			
928	-	Team CS542	0.84563	2

7. Future Work

Our future work will focus on two aspects: first, come up with a better data denoising scheme that will both serve the purpose of denoising and preventing overfitting. One promising way is to divide the full dataset into several subsets according to their Euclidean distance to the estimated center, for example, top 0-40%, 40%-80%, 80%-90%, and 90%-100%. Then assign a different percentage of data sampled from each subset. The subsets closer to the estimated center will have priority over the others. In this way, we can keep the variation of training dataset and get rid of the noisy data at the same time. Another future work direction is to enhance the computational capability to handle training on the full dataset.

8. Timeline and Roles

Task	**Deadline**	**Lead**
Data conversion	11/05/18	Weixi
Run basic models and compare them	11/08/18	all
Implement and run the baseline model using default data	11/19/18	Weixi, Yifu
Denoise data	11/19/18	Yifu, Zihuan
Finalize model	12/06/18	Yifu, Calvin
Experiment with the final model	12/09/18	Zihuan, Weixi
Prepare report and presentation	12/10/18	all

References

1) Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). ImageNet Classification with Deep Convolutional Neural Networks. Advances In Neural Information Processing Systems, 1–9.
2) Simonyan, K., & Zisserman, A. (2014). Very Deep Convolutional Networks for Large-Scale Image Recognition.
3) Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., … Rabinovich, A. (2015). Going deeper with convolutions. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition(Vol. 07-12-June-2015, pp. 1–9). IEEE Computer Society.
4) He, K., Zhang, X., Ren, S., Sun, J.(2015). Deep residual learning for image recognition. IEEE Conference on Computer Vision and Pattern Recognition (CVPR).
5) Hu, J., Shen, L., Albanie, S., Sun, G. & Wu, E.(2018). Squeeze-and-Excitation Networks. IEEE Conference on Computer Vision and Pattern Recognition.
6) L.J.P. van der Maaten and G.E. Hinton. Visualizing High-Dimensional Data Using t-SNE. Journal of Machine Learning Research 9(Nov):2579-2605, 2008.