DRAWING A CONCLUSION PREDICTING DOODLES USING NEURAL NETWORK

{ YIFU HU, ZIHUAN QIAO, WEIXI LI, CALVIN GUAN } BOSTON UNIVERSITY

INTRODUCTION

Quick, Draw! Doodle Recognition Challenge is a competition on Kaggle:

- Input: doodle images.
- Output: probabilities that the drawing is of each of the 340 categories, the 3 categories with the highest probabilities will be submitted as the prediction.
- Loss function: cross entropy loss ($K = 340$)

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

- Difficulties: i) size of dataset is very large (over 110K per category); ii) noisy data; iii) high computational cost and the only available GPU source is SCC at BU

Figure 1: Quick, Draw!

DATASET

- Training data: over 50 million drawings across 340 categories.
- Test data: 112,163 drawings.
- Small dataset: randomly sample around 13,000 images from 340 categories, 10% of the full dataset.
- Denoised datasets: 1%, 2%, 5% and 10% of the most relevant doodles that are closer to the estimated center.
- Random dataset: randomly sample 10% images of full dataset

REFERENCES

[1] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.

[2] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. *CoRR*, abs/1709.01507, 2017.

METHODS

Given such a large dataset with over 0.13 million images per class, it's very time-consuming (more than 60h per epoch) for training a deep neural network on it, especially when using the SCC.
Thus, we decided to form a subset of the full dataset by extracting a small percentage (e.g., 10%) of images from each class. The simplest way is to pick 10% of the images randomly. However, after taking the noisy nature of the dataset into consideration, we first come up with a denoising scheme to screen out the most recognizable portion of the full dataset. It works as follow:

1. Train a model with SE-net on a random 10% subset of the full dataset;
2. Evaluate and obtain the feature vector of each image in the whole dataset using the trained model;
3. Compute and rank the Euclidean distance between each softmax-ed feature vector and its ground-truth one-hot vector; Suppose D^i is a data in the full dataset, C is the estimated center, then the distance between them is: $d(D^i, C) = \sqrt{D^{iT} \cdot C}$.
4. Pick the top 1% (2%, 5%, 10%, respectively) images with the smallest distances and form version1 (version2, version5, version10, respectively) of the full dataset.

We then train our network on these four subsets individually and compare them with the random 10% subset. The result shown in section "Result 1" suggests that this denoising schedule failed to denoise the original dataset. The reason may be that this denoising scheme removes the variations of images in the same class, thus causing overfitting during training. We then directly use this random 10% subset for the rest of our experiments.
We then compared the performance of four models - VGG16, ResNet50, InceptionV4, and SE-ResNet50 on the randome 10% subset of the dataset to determine the main model to use. Our comparison showed that SE-ResNet50 came out on top in both efficiency and accuracy.
In Result 2, we illustrate the underlying framework behind the SE-ResNet50 model that we used as our main model. In addition, we compare the effect of using different learning rates on the training performance of the model.

FUTURE RESEARCH

The experiments show that the amount of high quality data is the key. In the denoising experiments, the over-fitting implies that the variance in the denoised data is very low. However, the t-SNE visualization suggests this method indeed reduced the amount of ambiguity between similar classes. A better scheme may be needed for denoising.

RESULTS 1

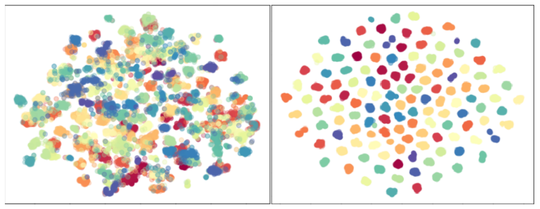

Figure 2: 128 classes t-SNE visualization with noise and after denoising

After the initial training of SE-ResNet, we obtained millions of 340-dimensional feature vectors of drawings. We randomly picked feature vectors from these classes and then used t-Distributed Stochastic Neighbor Embedding(t-SNE) technique to reduce the dimension of feature vectors to 2, thus allowing plotting of the figures on a two-dimensional graph. Figure 2 shows that the training data is very noisy. We removed the noisy feature vectors and re-plotted the features using t-SNE. As shown in Figure 2, the data is much cleaner.
Thus, we apply our denoising metric to the full data and get the training datasets then retrain our model. Table shows the datasets we use and the results of training. From the Table, we could infer that our denoising scheme reduced the variance of the data, resulting in over-fitting of the retrained models.

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%

Table 1: SE-ResNet50 training results

RESULT 2

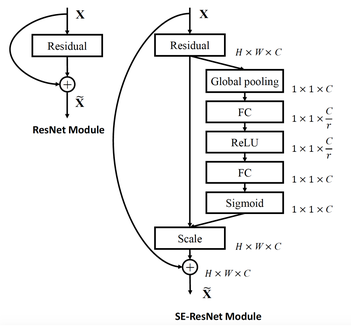

Figure 3: The schema of the original Residual module (left) and the SE-ResNet module (right)

- Training for learning rate comparison was done on 10% of the small data in order to simulate more epochs.
- Learning rates are scaled by the batch size.
- Higher learning rates resulted in faster decrease in loss and faster increase in accuracy.
- Using the highest learning rate, the validation loss stopped decreasing after a few epochs, and even started to increase, signifying over-fitting.

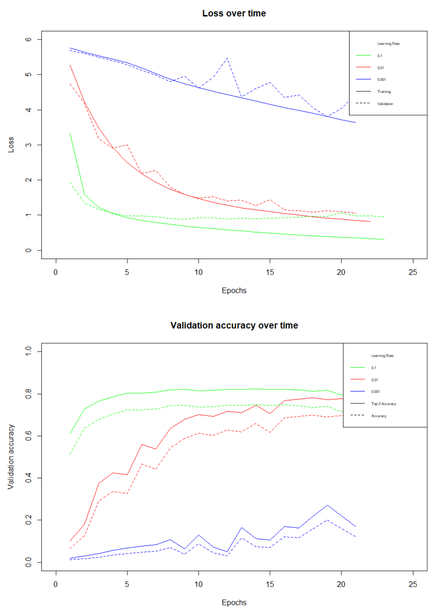

Figure 4: The effect of different learning rate for SE-ResNet50