https://doi.org/10.5392/IJoC.2019.15.4.008

Sketch Recognition Using LSTM with Attention Mechanism and Minimum Cost Flow Algorithm

Bac Nguyen-Xuan, Guee-Sang Lee
Dept. of Electronic and Computer Engineering
Chonnam National University, Kwangju, 500-070, Korea

ABSTRACT

*This paper presents a solution of the **'Quick, Draw! Doodle Recognition Challenge'** hosted by Google. Doodles are drawings comprised of concrete representational meaning or abstract lines creatively expressed by individuals. In this challenge, a doodle is presented as a sequence of sketches. From the view of at the sketch level, to learn the pattern of strokes representing a doodle, we propose a sequential model stacked with multiple convolution layers and Long Short-Term Memory (LSTM) cells following the attention mechanism [15]. From the view at the image level, we use multiple models pre-trained on ImageNet to recognize the doodle. Finally, an ensemble and a post-processing method using the minimum cost flow algorithm are introduced to combine multiple models in achieving better results. In this challenge, our solutions garnered 11th place among 1,316 teams. Our performance was 0.95037 MAP@3, only 0.4% lower than the winner. It demonstrates that our method is very competitive. The source code for this competition is published at: https://github.com/ngxbac/Kaggle-QuickDraw.*

***Key words**: Sketch Recognition, Deep Learning, Attention Mechanism, Long-Short Term Memory, Minimum Cost Flow.*

1. INTRODUCTION

The sketching was born as an essential from our ancestors to record their lives. Nowadays, sketch still holds a critical role as an effective communication tool. Recently, researchers have paid more attention to exploring sketch properties in various applications including sketch recognition [1]-[4], sketch-based image retrieval [5], [6], sketch segmentation [7]-[9] and sketch-based 3D model retrieval [10]. Studies related to sketching can be divided into two categories: sketch recognition and sketch generation. For sketch recognition, it is used in a wide range of applications. This will have an immediate impact on handwriting recognition and its robust applications in areas including OCR (Optical Character Recognition), ASR (Automatic Speech Recognition) and NLP (Natural Language Processing). However, it is an extremely challenging task to recognize sketches drawn by non-artists even for human being.

In comparison with image classification, sketch recognition being more challenging due to the following limitations. At first, a sketch has inherent ambiguities itself. Unlike traditional real images, sketches do not describe an object completely. Instead, sketches represent the abstract of an object in which many details are absent. Due to lack of rich texture details, different objects can be visualized in the same way. Secondly, there is a huge variation between real images and sketch images although they describe the same thing. Sketches is free-handwriting. Different users with different properties such as characteristics, habits, culturals, etc, may give a different sketch images which describes the same object. Thirdly, sketch only includes simple lines, curves and dots without rich texture information and color which makes sketch recognition be a difficult task.

The previous work on sketch recognition treats sketch recognition as an image classification task by considering sketch image as a binary image then extracting handcrafted features. Those features include the histogram of oriented gradient (HOG), scale-invariant feature transform (SIFT) and shape context. With the quantified feature descriptors, a typical classifier such as Support Vector Machine (SVM) is trained for object category prediction. However, sketch is constructed by a sequence of strokes. Therefore, representing sketch as the image totally lose the information of stroke orders. From the psychology point of view, they point out that this information plays an important role in recognition by human. This problem is also confirmed in computer vision. However, there is no research dealing with sequential characteristics in sketch recognition even that information is already available.

Recent years have witnessed the success of deep learning in image classification [11]. Similar neural network designs have also been used to address the recognition of problem of sketch images [3], [12]. Although these deep learning-based methods outperform the traditional ones, the unique properties of sketches, as discussed in the following, are often overlooked, leaving room for further improving the performance of sketch recognition A Sketch has two natural properties: visual pattern

* *Corresponding author, Email: gslee@jnu.ac.kr*
Manuscript received May. 2, 2019; revised Oct. 23, 2019; accepted Nov. 11, 2019

and sequential pattern which has high impact on sketch recognition. Firstly, the sketch image is described as series of strokes. These strokes have determined visual patterns to form main shapes and special details belonging to a specific object. For instance, a house has a door, zebra has a zebra crossing. Secondly, a sequential pattern describes how an object is drawn step by step such as left to right, top to bottom, clockwise, etc. For example, to draw a chicken, people often draw its head firstly, then body and legs. How to utilize these two useful patterns to improve the performance of sketch recognition is the main goal of this paper. In this paper, we firstly propose a recurrent neural network architecture for sketch object recognition which exploits the long-term sequential and structural regularities in stroke data. Second, we present the way to use pre-trained ImageNet models to utilize visual pattern of sketch image. Finally, an ensemble and post processing methods are introduced to combine two kinds of models which uses different features and patterns from sketch images.

2. RELATED WORK

In the prior works, researchers recognize sketches by drawing stroke by stroke to an image. To quantify a sketch image, the local features (Fisher Vectors with SIFT features [2], HOG features [13], bag-of-features [1]) are extracted. The typical methods such as SVM [1], [2] are then applied to recognize the unseen sketch. Besides, different studies such as [13] and [14] use active learning, multi-kernel feature learning to improve the performance slightly.

In recent year, deep neural network (DNN) gives a great success in many computer vision tasks, especially image classification and recognition. This framework shows an end-to-end learning strategy where hand-crafted features are not necessary. Related to sketch recognition, research efforts [3], [12], [15], [16] have been made to apply DNN and achieves better performances. Especially, [24] proposes a well-designed convolutional neural network (CNN) called Sketch-A-Net which has an architecture like AlextNet for sketch recognition. According to their experimental results, their method achieves 77.95% which surpasses the best result achieved by human 73.1% at the first time. [16] introduces a novel method to present features of sketches using DNN. They use real images as the reference to discover latent discriminative structures of sketch images and classify by using triplet. However, those methods still follow the existing learning procedures of image classification where the input image is a draw of sequential strokes. This approach does not take advantages of stroke orders.

To address this problem, in [17], the sequential information of sketches is exploited. They explicitly use sequential regularities in strokes with the help of recurrent neural work (RNN)

3. METHODOLOGY

In this section, we first introduce dataset and its representation. At the second part, a LSTM-based network is presented. Finally, image-based networks are discussed.

3.1 Dataset

In this challenge, we use Quick Draw Dataset which is a collection of millions of drawings across 340 categories, contributed by players of Quick, Draw! an online game where the players are asked to draw objects belonging to a particular object class in less than 20 seconds. The drawings were captured as timestamped vectors, tagged with metadata including what the player was asked to draw and in which country the player was located. Two versions of the data are given. The raw data is the exact input recorded from the user drawing, while the simplified version removes unnecessary points from the vector information. For example, a straight line may have been recorded with 8 points, but since you only need 2 points to uniquely identify a line, 6 points can be dropped. The simplified files are much smaller and provide effectively the same information.

In this dataset, samples are either recognized or unrecognized. During drawing stage, if players cannot draw the object within preset time of 20 seconds or the doodle is not recognized by the algorithm of Google behind the backend, this sample is considered as unrecognized. This information makes the challenge more difficult since we have to deal with uncertain labels.

Fig. 1. Quick Draw dataset

The dataset is not only represented as image, but also represented as a sketch which is a set of ordered strokes which is represented as following:

$$S = \{(x_i, y_i, s_i)\}_{i \in \{1 \ldots N\}}$$

where N is the number of points. x_i and y_i are the x-coordinate, y-coordinate of point $i-th$ correspondingly. s_i is the number indicates that the point belonging to which stroke. The same value of s_i shows that points are in the same stroke. The number of points N and number of strokes s_i can be different for each sample.

3.2 LSTM-based Network

In this part, we briefly summary LSTM network definition. Second, we discuss about an attention mechanism. Finally, an LSTM-based network using attention is proposed.

3.2.1 Long-Short Term Memory Network: In our work, we take LSTM as a basic network architecture to exploit the sequential characteristic of the sketches. A LSTM network is designed to avoid long-term dependence problem. It takes input sequence $X = (x_1, x_2, \dots, x_t)$ to output sequence $Y = (y_1, y_2, \dots, y_t)$ as following

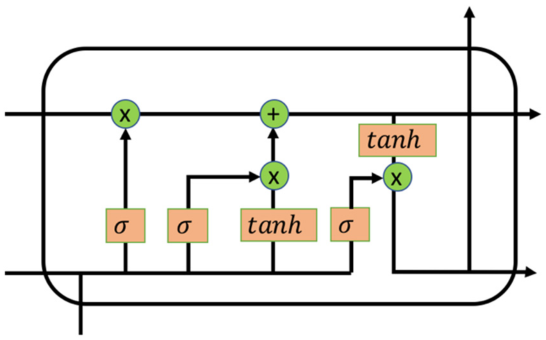


Fig. 2. LSTM cell

$$z_t = \sigma(W_z . [h_{t-t}, x_t])$$
$$r_t = \sigma(W_r . [h_{t-1}, x_t])$$
$$h_t^{\sim} = \tanh(W . [r_t * h_{t-1}, x_t])$$
$$h_t = (1 - z_t) * h_{t-1} * h_{t-1} + z * h_t^{\sim}$$

where x_t is $t-th$ input and h_t is the hidden state of LSTM. The operation $*$ denotes the element-wise vector product.

3.2.2 Feed-forward Attention: Attention mechanism is first introduced in NLP research field to capture temporal information and decencies of an entity word with others in a sentence when they are placed together. In Quick Draw dataset, an object is a sequence of strokes which is similar to a sentence where the stroke holds the role of the word. In fact, the order of strokes is very important for recognizing the object. For example, if we draw one circle, it can be an alarm, a football, a snow man or glasses. The model needs to see the next stroke to give the answer. If it is one circle again, the object can be a snow man or glasses because both of them requires at least two circles. To distinguish two objects, the model must consider the position of the second circle. The snow man is drawn by two circles aligning in horizontal axis where glasses are drawn by two circles aligning in vertical axis. This example shows the relationship and dependencies of strokes are essential for sketch recognition. To utilize this feature, we apply attention mechanism. Feed-forward attention is introduced in [18]. It simplifies attention mechanism by replacing the recurrent attention model with a feed forward one. This mechanism takes hidden state of LSTM cell and produces a single vector c from an entry sequence could be formulated as follows

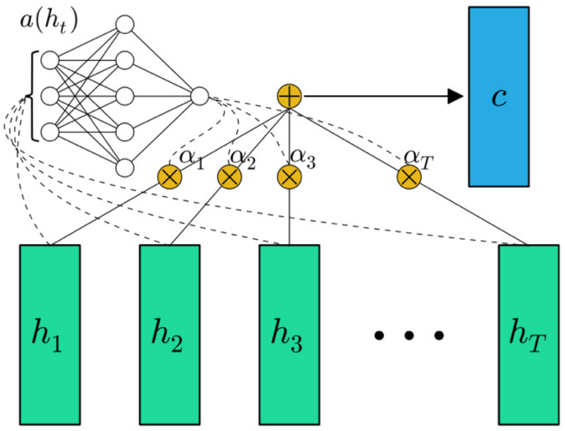


Fig. 3. Feed-forward attention [15]

$$e_t = a(h_t)$$
$$\alpha_t = \frac{\exp(e_t)}{\sum_{k=1}^{T} \exp(e_k)}$$
$$c = \sum_{t=1}^{T} \alpha_t h_t$$

where a is a learnable function. h_t and T are the $t-th$ and number of hidden states of LSTM respectively.

3.2.3 Network architecture: To learn the pattern of temporal strokes, we propose a LSTM-based network to classify doodles. The model uses a combination of 1-dim convolutional layers, LSTM cells, and a softmax output layer to classify the drawing. The recognition in Quick, Draw! is performed by a classifier that takes the user input, given as a sequence of strokes of points (x_i, y_i, s_i). Input data is fed into a series of 1-dimensional convolutions and batch normalization layers. Before going through a series of Bi-directional LSTM layers, the dimension of feature vector is reduced by pooling layers. However, pooling may lead to an information loss. Thus, we applied both max-pooling and average-pooling and kept them for final softmax classification layer. In this model, we use attention mechanisms [18] after LSTM layers to force model pays more attention to sequence of strokes. The concatenation of the outputs of all LSTM, attention and pooled information steps are fed into a softmax layer to make a classification decision among the classes of drawings that we know.

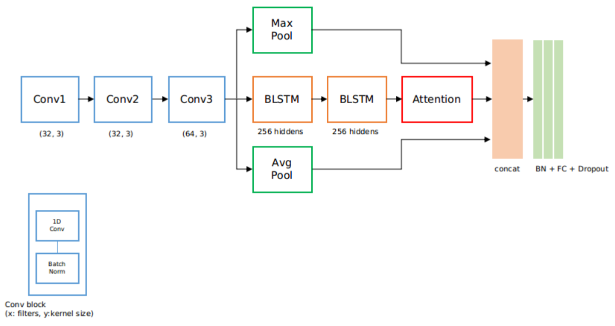


Fig. 4. Proposed LSTM-based network

3.3 Image based network

Different from LSTM-based network, the image-based network takes image data as the input while LSTM-based network uses sequence of strokes for the training. We use models pre-trained on ImageNet [19] to recognize sketches. The procedures follow the existing classification approach of deep learning. Due to hardware limitations, we only can draw the sketch into image which has size $64x64$ at first. Then, those images are fed into the model for training. Finally, we conclude to use 6 models: ResNet50 [20], Densenet169 [21], Xception [22], SEResNet50 [23], SEResNext101, and SEResnext50 [24].

4. EXPERIMENT

The training and testing processes are performed on NVIDIA GeForce GTX1080Ti 11G GPUs, using the deep learning framework Pytorch for image classifications models and Keras for LSTM models. In this section, we firstly introduce implementation details, including data augmentation, parameter setting and training strategy. Second, we present an ensemble method to combine the networks together and a post processing technique to balance prediction of classes in the test set.

4.1 Implementation

The LSTM network is trained by full data for 200 epochs. Each epoch is done by 100 steps. In each step, 1024 samples are randomly selected for training. Each sample is constructed by 196 sequences of strokes of points in (x, y, s), where s indicates whether the point is the first point in a new stroke. The network is trained by Adam [25] optimizer. The image networks aforementioned are used for fine-tuning. Each model is trained for 8 epochs. Each epoch is done by all data. At the first three epochs, we freeze all the top convolutional layers of network and only keep the last fully connected layer being updated parameters during the back-propagation process. The learning rate of this stage is 0.01. From rest of phase, we unfreeze all the layers with smaller learning 0.001 and allow them to be updated parameters. This training strategy brings a significant boosted result. During training, we apply Stochastic Weight Averaging (SWA) [26] algorithm and save 5 best states of the network. We do not use augmentation during training because we have a large scale of data, but we use two test-time augmentations (original and horizontal flipped image) for testing (TTA).

$$p_k = \frac{1}{M.T}\sum_{m\in M}\sum_{t\in T} p_k^{mt}$$

where p_k is the predicted probability of test sample $k-th$. M and T are the set of models and TTA correspondingly.

4.2 Evaluation metric

In the competition, all submissions are evaluated according to the Mean Average Precision @ 3 (MAP@3):

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

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 predictions per drawing

4.3 Probability calibration

In this challenge, 112,000 test samples are used for two evaluation phases. The first phase is called public evaluation with only 9% of data. The second is private evaluation with the rest of data. There is no overlap of data between two phases. The final ranking is made based on the private evaluation. The competitors do not know the distribution between two test sets. This point makes the competition to be more interesting and challenging.

We recognize that there are 112,000 test cases divided to 340 classes. We place the hypothesis that there are $\frac{112000}{340} =$ 330 test cases per class. In addition, the organizer sets a benchmark as 0.005 MAP@3 score in public evaluation phase only. In this benchmark, all the test cases have same predictions. Interestingly, from our hypothesis above, we apply MAP@3 formulation defined at Section 4.2 and measure that the public evaluation score. Our estimation result is also 0.005 which is exactly same as the benchmark. In conclusion, our hypothesis is proved and shows that the distribution of data is equal for all classes. With a large of data, it is more likely that number of samples per class is the same in private evaluation. Moreover, we analyze the prediction of the model and determine that the distribution of prediction between classes is unbalanced. Therefore, we apply Minimum Cost Flow Problem (MCFP) to solve this problem in order to balance predicted distribution between all classes. This method helps 0.1% boosting overall which is a big improvement.

In theory, the MCFP is defined as follows:

Let $G = (V, E)$ is a directed network with a set of vortexes V(also called as nodes) and set of edges S. For each edge $(i, j) \in E$ also has an associated cost c_{ij} that denotes the cost per unit flow on this edge and u_{ij} that denotes the capacity (maximum amount that can flow through the edge). Each vertex $i \in V$ has a value b_j. There are two kinds of vertex: supply and demand. Supply vertex has positive value while demand vertex has negative value. We call vertex is transshipment if $b_j = 0$. We call G is transportation network denotes as $G = (V, E, c, u, b)$ which is described as following.

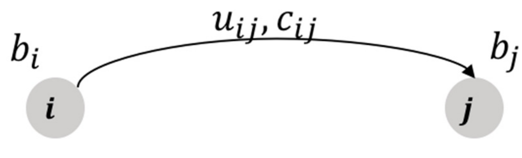


Fig. 5. A flow from supply node i to demand node j

We represent the flow on the edge $(i, j) \in E$ as x_{ij}, the model that minimum the cost flow is defined as:

$$Minimize\ z(x) = \sum_{(i,j)\in E} c_{ij}x_{ij}$$

Subject to:

$$\sum_{\{j:(i,j)\in E\}} x_{ij} - \sum_{\{j:(j,i)\in E\}} x_{ji} = b_j\ for\ all\ i \in V$$

$$0 \le x_{ij} \le u_{ij}\ for\ all\ (i,j) \in E$$

The detail of our application is represented as follows.

Let $N_{test} = 112000$ is the number of test sample. Let $N_{class} = 340$ is the number of class in our prediction. We defined a set of demand and supply nodes as follows:

$V_{demand} = \{V_i^d\}_{i=1,\dots N_{test}}$ with values: $b_i^d = -1$ for all i

$V_{supply} = \{V_j^s\}_{j=1,\dots N_{class}}$ with values: $b_j^s = \frac{N_{test}}{N_{class}}$ for all j

In this dataset, the number of sample per class is equal. Therefore, for each supply node, the value $b_j^s = \frac{N_{test}}{N_{class}}$ for all j. In the case that dataset is biased, the supply node value b_j^s is different for each class j and must satisfy the condition that:

$$\sum_{j=1}^{N_{class}} b_j^s = N_{test}$$

Let p_{ij} is the probability that our sample $i-th$ belongs to class $j-th$. We consider the edge from $V_i^d \to V_j^s$ has unit cost c_{ij}. In this case, $c_{ij} = p_{ij}$. However, the edge should go from supply nodes to demand nodes. Therefore, we define the edge from supply node V_j^s to demand node V_i^d as following

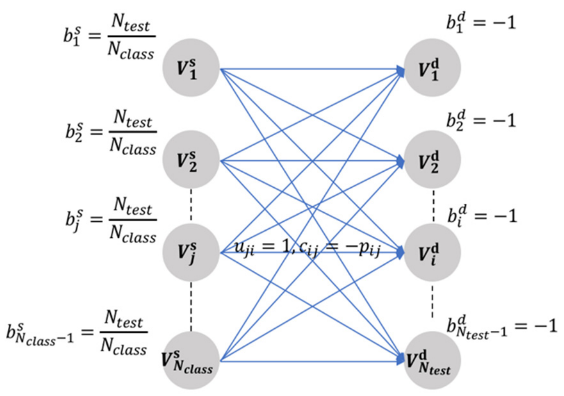
$$(j,i) \in E = V_{demand} \cup V_{supply}$$

has unit cost $c_{ji} = -c_{ij} = -p_{ij}$

The number of edge we define in this problem is:

$$N_{edge} = N_{class} * N_{test}$$

Each edge will have capacity: $u_{ij} = 1$.



$b_1^s = \frac{N_{test}}{N_{class}}$
V_1^s
$b_1^d = -1$
V_1^d
$b_2^s = \frac{N_{test}}{N_{class}}$
V_2^s
$b_2^d = -1$
V_2^d
$b_j^s = \frac{N_{test}}{N_{class}}$
V_j^s
$u_{ji} = 1, c_{ij} = -p_{ij}$
$b_i^d = -1$
V_i^d
$b_{N_{class}-1}^s = \frac{N_{test}}{N_{class}}$
$V_{N_{class}}^s$
$b_{N_{test}-1}^d = -1$
$V_{N_{test}}^d$

Fig. 6 Graph model to probability calibration using MCF

Now, the problem is changed to *Minimum cost maximum flow* since we want to find get as much flow as possible from the supply nodes to demand nodes with the shortest path (reaching from supply to demand nodes with minimum cost). To solve this problem, we use OR-Tool with pseudo code as follows

Algorithm 1 Algorithm for applying Min Cost Flow in order to balance predictions

```
Input: probability of testset after prediction
  $N_{test}$ is the number of testset.
  $N_{class}$ is the number of classes.
  Init Min Cost Flow Solver
  for j = 1 to $N_{class}$ do
      Set supply nodes with the value $N_{test}/N_{class}$
  end for
  for i = 1 to $N_{test}$ do
      Set demand nodes with the value $-1$
      for j = 1 to $N_{class}$ do
          Define the edge from $j \to i$
          $E_{j\to i} := u_{ji} = 1, c_{ji} = -p_{ij}$
      end for
  end for
  Solve Min Cost Flow problem
  for each edge betweens nodes do
      if flow in edge is greater than 0 then
          $i_{head}$ is the head's node number of edge
          $i_{tail}$ is the tail's node number of edge
          $Predict_{class}[i_{head}] = i_{tail}$
      end if
  end for
Output: balanced prediction of testset $Predict_{class}$
```

Fig. 7. Pseudo-code of algorithm for applying MCF

5. RESULTS

5.1 Submission results

This table below shows the performance of each single model on both evaluation phases

Table 1. Model performance on the public test set and private test set

Model	Public		Private	
	mAP@1	mAP@3	mAP@1	mAP@3
SEResNext50	**90.778**	**94.404**	**90.366**	**94.207**
SEResNet50	90.301	94.073	90.061	93.953
ResNet50	90.402	94.146	90.224	94.103
SEResNet101	90.230	93.985	89.994	93.887
Densenet169	90.007	93.833	89.649	93.695
Xception	89.854	93.711	89.440	93.524
LSTM	89.489	93.490	89.341	93.411
Ensemble	92.360	95.390	91.716	95.037
Winner (Ensemble)	-	95.750	-	95.480

Our best single model is SEResNext50 achieving 94.404 mAP@3 and 94.207 mAP@3 on public and private evaluation correspondingly. After ensemble, our score is 95.037 mAP@3 on private test set. The score is only 0.4% lower than the winner (95.480 mAP@3).

5.2 Performance comparison with these of SOTA methods

Table 2. Comparison of our performance with these of SOTA methods

Method	Accuracy
Human [1]	73.1%
HOG-SVM [1]	56.0%
MKL-SVM [14]	65.8%
Fisher-Vectors [2]	68.9%
Sketch-a-Net [3]	74.9%
Deep Visual-Sequential Fusion Model [27]	79.6%
SketchMate [28]	80.5%
Sketch-2RNN [29]	84.4%
LSTM (ours)	89.3%
SEResNext50 (ours)	90.4%

In this table above, [29] has LSTM based approach. This method is evaluated on QuickDraw dataset which is similar to our dataset. It is clear that Sketch-2RNN outperforms the previous researches which are evaluated on TU-Berlin sketch dataset. The reason is that QuickDraw dataset has million sketches compared to 20,000 sketches of TU-Berlin. Moreover, QuickDraw dataset has 350 objects while TU-Berlin has only 250 objects. With a large-scale dataset and a big number of objects, the results of QuickDraw dataset are more reliable. Our results of LSTM based approach shows that our method is much more efficient than Sketch-2RNN with a big gap of 5%.

5.3 Effect of probability calibration

We would like to analyze how good applying MCF to sketch recognition is. The figure below shows samples that model gives wrong predictions and the corrected answer after applying MCF algorithm.

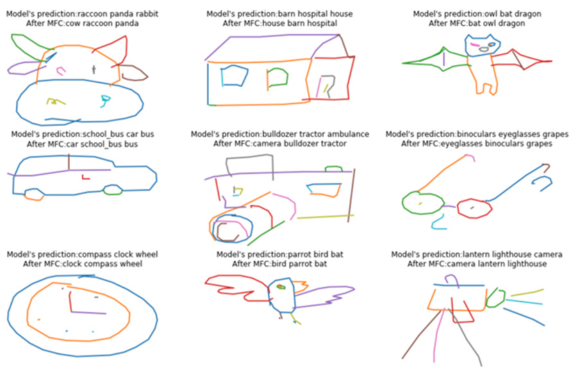

Model's prediction:raccoon panda rabbit
After MFC:cow raccoon panda
Model's prediction:barn hospital house
After MFC:house barn hospital
Model's prediction:owl bat dragon
After MFC:bat owl dragon
Model's prediction:school_bus car bus
After MFC:car school_bus bus
Model's prediction:bulldozer tractor ambulance
After MFC:camera bulldozer tractor
Model's prediction:binoculars eyeglasses grapes
After MFC:eyeglasses binoculars grapes
Model's prediction:compass clock wheel
After MFC:clock compass wheel
Model's prediction:parrot bird bat
After MFC:bird parrot bat
Model's prediction:lantern lighthouse camera
After MFC:camera lantern lighthouse

Fig. 8. Sample results of the model before and after applying MCF

At the first sample (top-left figure), it is clear that the *cow* is drawn. However, the top-3 predictions from the model, which are *raccoon*, *panda* and *rabbit*, are totally wrong. After applying MCF, the top-1 prediction is now *cow,* showing us an impressive because *cow* is not included in the top-3 predictions. It's remembered that, the evaluation metric is mean average precision. It means the answer must be not only correct, but also arranged by the confidence and priority. In this case, if we evaluate this sample by ground truth as *cow*, the score we will get if we trust the model is 0. But MCF algorithm helps us to achieve maximum score 1. Similarly, in the other samples, MFC not only helps model to correct the answer, but also help to rearrange the confident among answers.

6. CONCLUSION

In this paper, we have presented the solution to Quick Draw! Doodle Recognition Challenge. A novel sketch recognition model is proposed, which is based on LSTM and attention mechanism. The experiment shows that this model outperforms the previous methods. Last but not least, we introduce the method called probability calibration which uses MCF algorithm to balance the number of predicted classes. This technique can correct wrong predictions by the model and help to improve performance overall. Although the promising performance is achieved by our methods, there are still many issues which can be explored further such as stroke interval, drawing style, etc. In the next steps, we will look into this problem, refine the framework and design more solutions.

ACKNOWLEDGEMENT

This research was supported by the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (NRF-2018R1D1A3B05049058 & NRF-2017R1A4A1015559) and by Chonnam National University (Grant number: 2018-3293). The corresponding author is Guee Sang Lee.

REFERENCES

[1] Mathias Eitz, James Hays, and Marc Alexa, "How do humans sketch objects?," ACM Trans. Graph., vol. 31, no. 4, 2012, pp. 44:1-44:10. doi: 10.1145/2185520.2185540

[2] Rosa ́lia G. Schneider and Tinne Tuytelaars, "Sketch classification and classification-driven analysis using fisher vectors," ACM Trans. Graph., vol. 33, no. 6, 2014, pp. 174:1-174:9. doi: 10.1145/2661229.2661231

[3] Qian Yu, Yongxin Yang, Feng Liu, Yi-Zhe Song, Tao Xiang, and Timothy M. Hospedales, "Sketch-a- net: A deep neural network that beats humans," Inter- national Journal of Computer Vision, vol. 122, no. 3, 2017, pp. 411-425. doi: 10.1007/s11263-016-0932-3

[4] Jianhui Zhang, Yilan Chen, Lei Li, Hongbo Fu, and Chiew-Lan Tai, "Context-based sketch classification. In Brian Wyvill and Hongbo Fu, editors," Proceedings of the Joint Symposium on Computational Aesthetics and Sketch-Based Interfaces and Modeling and Non-

Photorealistic Animation and Rendering, Expressive 2018, Victoria, BC, Canada, August 17-19, 2018 ACM, pp. 3:1-3:10. doi: 10.1145/3229147.3229154

[5] Mathias Eitz, Kristian Hildebrand, TamyBoubekeur, and Marc Alexa, “Sketch-based image retrieval: Benchmark and bag-of-features descriptors,” IEEE Trans. Vis. Comput. Graph., vol. 17, no. 11, 2011. pp. 1624-1636. doi: 10.1109/TVCG.2010.266

[6] Rui Hu and John P. Collomosse, “A performance evaluation of gradient field HOG descriptor for sketch-based image retrieval,” Computer Vision and Image Understanding, vol. 117, no. 7, 2013, pp. 790-806. doi: 10.1016/j.cviu.2013.02.005

[7] Zhe Huang, Hongbo Fu, and Rynson W. H. Lau, “Data-driven segmentation and labeling of freehand sketches,” ACM Trans. Graph., vol. 33, no. 6, 2014, pp. 175:1-175:10. doi: 10.1145/2661229.2661280

[8] Lei Li, Hongbo Fu, and Chiew-Lan Tai, “Fast sketch segmentation and labeling with deep learning,” CoRR, abs/1807.11847, 2018. doi: 10.1109/MCG.2018.2884192

[9] Zhenbang Sun, Changhu Wang, Liqing Zhang, and Lei Zhang, “Free hand-drawn sketch segmentation,” In Andrew W. Fitzgibbon, Svetlana Lazebnik, Pietro Perona, Yoichi Sato, and Cordelia Schmid, editors, Computer Vision - ECCV 2012 - 12th European Conference on Computer Vision, Florence, Italy, October 7-13, 2012, Proceedings, Lecture Notes in Computer Science, Part I, vol. 7572, 2012, pp. 626-639. doi: 10.1007/978-3-642-33718-5_45

[10] Fang Wang, Le Kang, and Yi Li, “Sketch-based 3d shape retrieval using convolutional neural networks,” In IEEE Conference on Computer Vision and Pat- tern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015, IEEE Computer Society, 2015, pp. 1875-1883.

[11] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton, “ImageNet classification with deep convolutional neural networks,” In Peter L. Bartlett, Fernando C. N. Pereira, Christopher J. C. Burges, Le ́on Bottou, and Kilian Q. Weinberger, editors, Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems 2012, Proceedings of a meeting held December 3-6, 2012, Lake Tahoe, Nevada, United States., 2012, pp. 1106-1114. doi: 10.1145/3065386

[12] PatsornSangkloy, Nathan Burnell, Cusuh Ham, and James Hays, “The sketchy database: learning to retrieve badly drawn bunnies,” ACM Trans. Graph., vol. 35, no. 4, 2016, pp. 119:1-119:12. doi: 10.1145/2897824.2925954

[13] ErelcanYanik and TevfikMetinSezgin, “Active learning for sketch recognition,” Computers & Graphics, vol. 52, 2015, pp. 93-105. doi: 10.1016/j.cag.2015.07.023

[14] Yi Li, Timothy M. Hospedales, Yi-Zhe Song, and Shaogang Gong, “Free-hand sketch recognition by multi-kernel feature learning,” Computer Vision and Image Understanding, vol. 137, 2015, pp. 1-11. doi: 10.1016/j.cviu.2015.02.003

[15] Xiangxiang Wang, Xuejin Chen, and ZhengjunZha, “Sketchpointnet: A compact network for robust sketch recognition,” In 2018 IEEE International Conference on Image Processing, ICIP 2018, Athens, Greece, October 7-10, 2018, IEEE, pp. 2994-2998. doi: 10.1109/ICIP.2018.8451288

[16] H. Zhang, S. Liu, C. Zhang, W. Ren, R. Wang, and X. Cao, “Sketchnet: Sketch classification with web images,” In 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2016, pp. 1105-1113. doi: 10.1109/CVPR.2016.125

[17] Jun-YanHe, XiaoWu, Yu-GangJiang, BoZhao, and Qiang Peng, “Sketch recognition with deep visual- sequential fusion model,” In Qiong Liu, Rainer Lien- hart, Haohong Wang, Sheng-Wei “Kuan-Ta” Chen, Susanne Boll, Yi-Ping Phoebe Chen, Gerald Fried- land, Jia Li, and Shuicheng Yan, editors, Proceedings of the 2017 ACM on Multimedia Conference, MM 2017, Mountain View, CA, USA, October 23- 27, 2017, ACM, pp. 448-456. doi: 10.1145/3123266.3123321

[18] Colin Raffel and Daniel P. W. Ellis, “Feed-forward networks with attention can solve some long-term memory problems,” CoRR, abs/1512.08756, 2015.

[19] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei, “ImageNet Large Scale Visual Recognition Challenge,” International Journal of Computer Vision (IJCV), vol. 115, no. 3, 2015, pp. 211-252. doi: 10.1007/s11263-015-0816-y

[20] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, “Deep residual learning for image recognition,” CoRR, abs/1512.03385, 2015. doi: 10.1109/CVPR.2016.90

[21] Gao Huang, Zhuang Liu, and Kilian Q. Weinberger, “Densely connected convolutional networks,” CoRR, abs/1608.06993, 2016. doi: 10.1109/CVPR.2017.243

[22] Francois Chollet, “Xception: Deep learning with depth-wise separable convolutions,” CoRR, abs/1610.02357, 2016. doi: 10.1109/CVPR.2017.195

[23] Jie Hu, Li Shen, and Gang Sun, “Squeeze-and- excitation networks,” In 2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018, IEEE Computer Society, pp. 7132-7141. doi: 10.1109/CVPR.2018.00745

[24] SainingXie, Ross B. Girshick, Piotr Dolla ́r, ZhuowenTu, and Kaiming He, “Aggregated residual transformations for deep neural networks,” CoRR, abs/1611.05431, 2016. doi: 10.1109/CVPR.2017.634

[25] Diederik P. Kingma and Jimmy Ba, “Adam: A method for stochastic optimization,” CoRR, abs/1412.6980, 2014.

[26] Pavel Izmailov, DmitriiPodoprikhin, TimurGaripov, Dmitry P. Vetrov, and Andrew Gordon Wilson, “Averaging weights leads to wider optima and better generalization,” In Amir Globerson and Ricardo Silva, editors, Proceedings of the Thirty Fourth Conference on Uncertainty in Artificial Intelligence, UAI 2018, Monterey, California, USA, August 6-10, 2018, AUAI Press, pp. 876-885.

[27] Jun-Yan He, Xiao Wu, Yu-Gang Jiang, Bo Zhao, and Qiang Peng, “Sketch recognition with deep visual-sequential fusion model,” In Liu et al, 2017, pp. 448-456. doi: 10.1145/3123266.3123321

[28] Peng Xu, Yongye Huang, Tongtong Yuan, Kaiyue Pang, Yi-Zhe Song, Tao Xiang, Timothy M. Hospedales, Zhanyu Ma, and Jun Guo, “Sketch- mate: Deep hashing for million-scale human sketch retrieval,” In 2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018, pp. 8090-8098. doi: 10.1109/CVPR.2018.00844

[29] Lei Li, Changqing Zou, Youyi Zheng, Qingkun Su, Hongbo Fu, and Chiew-Lan Tai, “Sketch-r2cnn: An attentive network for vector sketch recognition,” CoRR, abs/1811.08170, 2018.

Bac Nguyen-Xuan
He received the B.S in Electronic and Telecommunication from University of Engineering and Technology, Vietnam National University, Vietnam in 2015. He was a Software Engineer in Japan for two years. Since then, he has been with the Electronics and Computer Engineering, Chonnam National University. His main research interests include emotion recognition, image processing and deep learning. Besides, he is an aggressive competitor at the Kaggle which is the largest community of data science, machine learning and deep learning. He achieved several competition medals and be horned to receive Master of Competition tier with the ranking as 65/124,000 (updated 2019/11/15) which belongs to the top 0.5% of best data scientists inside the Kaggle.

Guee-Sang Lee
He received the B.S. degree in Electrical Engineering and the M.S. degree in Computer Engineering from Seoul National University, Republic of Korea, in 1980 and 1982, respectively. He received the Ph.D. degree in Computer Science from Pennsylvania State University in 1991. He is currently a professor of the Department of Electronics and Computer Engineering at Chonnam National University, Republic of Korea. His primary research interests are image processing, computer vision, and video technology.