EAI Endorsed Transactions
on Creative Technologies
Research Article EAI.EU

LW-Sketch-Net on Mobile Phones

Ni Kong[1,*], Hong Hou[2] and Zhe Bai[3]

[1]School of Information Science and Technology, Northwest University, Xi'an, China

Abstract

With the popularity of smart terminals, people tend to draw simple sketches to express emotions and ideas in communication, which means the era of reading pictures is coming. Therefore, in the field of sketch recognition, the application of deep network models in mobile devices is an irreversible trend. However, using deep learning method, most existing works with good performance has a large number of parameters. In order to further improve recognition speed and ensure the accuracy, we propose a lightweight neural network architecture to recognize sketch object. Specifically, we apply depthwise separable convolution into the network to reduce parameters and adjust the network effectively for the sparsity of sketch. Outperforming the state-of-the-art approaches, we achieves 85.8% and 84.1% on TU-Berlin and QuickDraw benchmarks respectively. Furthermore, the number of parameters is reduced to a large extent, which are 5% and 20% of the amount of Sketch-A-Net and MobileNets. We also develop a sketch recognition application on mobile phones to demonstrate the proposed scheme.

Keywords: Sketch Recognition, Lightweight Networks, Depthwise Separable Convolution.

Received on 15 March 2019, accepted on 21 April 2019, published on 26 April 2019

doi: 10.4108/_____

1. Introduction

In recent years, the popularity of touch-based smart phones terminals has largely changed the way people communicate with each other. Due to the intuitiveness and simplicity of sketch, people tend to draw sketch at any time and send interesting stick figures to express emotions and ideas [1]. Thus, an effective means of social contact is provided by sketch for people with different culture and language. For better human-computer interactions, research on sketch so far includes sketch recognition [2], sketch-based image retrieval [3] and sketch synthesis [4].

In 2012, a study collected 20,000 freehand sketches for 250 categories and analysed that recognition accuracy of human was only 73.1% [2]. Sketch recognition is even a challenging task for computer. The main reasons are as follows. (1) Sketch is different from image, which lacks rich texture details. (2) Sketch is highly abstract and ambiguous that represents an object or depicts a story in a few strokes. (3) Sketch has large shape variations because different people would draw different sketches for the same object.

The goal of sketch recognition is to identify the object category of an input sketch. Traditional works mainly extracted hand-crafted features which were coupled with Bag-of-Words to yield a final feature representations for sketch. But the accuracy is lower than human recognition because these features constructed by natural image are not suitable for sketch. Recent year researchers built networks aimed at hand-drawn sketch using deep learning methods, and the accuracy was improved obviously compared to traditional methods. However, these existing approaches have more parameters and recognition speed is slow. With the promotion of smart phone, the networks model should be refined to improve recognition speed while ensuring accuracy. In order to identify sketch using deep network models on mobile phones, it is meaningful to study lightweight networks.

In this work, we propose a lightweight neural network for sketch which achieves good accuracy on the QuickDraw and TU-Berlin benchmarks. Furthermore, the amount of network parameters is greatly reduced and recognition speed is

*Corresponding author. Email: kongni929@163.com

improved. Finally, we apply the lightweight recognition framework in the sketch recognition application on mobile smartphones.

The contributions of our work are as follows.

(1) We apply the depthwise separable convolutions firstly for sketch recognition in order to reduce network parameters. The parameters of our network is only 1M which is 20% of the amount of MobileNets.

(2) In response to the sparsity of sketch stored in computer, we improve the lightweight sketch network effectively making it more suitable for recognition task. Our method achieves 85.8% and 84.1% accuracy on both QuickDraw datasets and TU-Berlin benchmarks, which outperforms the state-of-the-art approaches.

(3) We develop a sketch recognition application for smart devices using the lightweight neural network.

2. Related work

2.1. Sketch datasets

The collection of sketches is a long and complicated process. In order to ensure the availability and diversity of datasets, a large number of people need to draw different sketch objects. Thus, the research of sketch lacks large-scale datasets compared with image datasets such as ImageNet [5]. But some medium-scale sketch datasets are as follows. The TU-Berlin datasets [2] with 20,000 sketches used crowd-sourcing plat-form to send tasks and required participants to draw a sketch in 30 minutes. The Sketchy datasets [6] was the first collection of sketch-image pairs including 75,000 sketches of 125 categories. Yu [7] contributed two sketch-image datasets for shoes and chairs. However, these datasets are too small to suit for training deep neural net-works. In 2017, the problem was alleviated by Ha and Eck [8] who contributed a large-scale datasets, QuickDraw, containing 50 millions of sketches for 345 categories. Additionally, the datasets was very diverse because participants only had 20 seconds to draw and the sketches were not pre-processed. In this paper, we constructed a sub-datasets containing 10,200,000 sketches called QuickDraw-10M to perform experiments.

2.2. Sketch Recognition Method

Most prior works used traditional image classification methods and extracted shallow hand-crafted feature for sketch recognition. Eitz [2] used the histogram of oriented gridients (HOG) as sketch local features and built a visual dictionary with 500 visual words, then turned each sketch into a histogram vectors with fixed dimensions. Finally, they used KNN and SVM to identify sketches with accuracy of 45% and 56% respectively. A new shape context descriptor was designed by Cao [9], namely symmetric-aware flip invariant sketch histogram to refine the sketch shape context function. And they obtained 53% recognition performance on the TU-Berlin sub-dataset with only 24 classes using SVM. Li [10] constructed a multi-kernel learning framework by combining several local features included HOG, SSIM, Daisy and Attributes to represent sketch completely. They obtained the weighted sum of the kernel to make the best use of the distinguishing ability of each feature and representation, and finally reached recognition accuracy of 65.81%. However the recognition accuracy was far inferior to human performance.

The main reasons for the low accuracy are as follows. (1) These feature descriptors of the sketch are grafted from the descriptors of natural images. In other words, all feature descriptors are not designed directly for sketches and sketches cannot be properly represented. (2) Researchers ignored the characteristics of sketches such as diversity, sparseness and stroke order which are import section when spending a lot of effort on constructing descriptors.

Advancement in deep learning has significantly influenced image recognition since 2012. Sarvadevabhatla [11] first tested AlexNet and LeNet on sketch, and AlexNet performed better than LeNet and every methods using hand-crafted features. The ground-breaking work of Yu [12], for the first time, beat human on sketch recognition task by constructing a convolutional neural network (CNN) namely Sketch-a-Net until 2015. The same team tried to improve the recognition performance by designing data augmentation techniques in [13]. Zhao combined the deep convolutional neural network [14] and recurrent neural network (RNN) [15] to perform the task. Leveraging the inherent sequential nature of sketch, Sarvadevabhatla [16] addressed the problem as sequence learning task. These studies improve the performance to some extent. Yet these models are relatively large due to using deep neural network. Moreover, the recognition task consumes more time and the speed is slow.

The above sketch recognition methods based deep learning mainly include two aspect, a method only uses CNN to extract sketch space features and the other method that combines CNN and RNN simultaneously extract space and stroke order features. These methods can better adapt to the sparseness and abstractness of sketches to improve the ability of computers to recognize sketches. However, the amount of parameters of these methods is relatively many resulting in longer recognition times and slower speeds, which have few practical application significance. Our focus is to address the problem by designing a lightweight deep neural network which takes recognition accuracy and speed into consideration.

2.3. Sketch recognition applications

With the popularization of touch screen devices, people tend to interact directly with smart machines to gain more fun and knowledge recently. In addition, the sketch has become a new way of communication in network because of its convenient and interesting features. Therefore, some sketch recognition algorithms for smart devices have begun to develop.

In 2015, Xiao [17] introduced the PPTLens system to convert sketch images captured by smart phones to digital flowcharts in PowerPoint and they proposed an effective stroke extraction strategy. This research can be seen as an early attempt at hand-drawn sketch recognition. In 2017,

Google released an AI experiment application called Quick Draw [8], which uses recurrent neural network to guess the sketches that users draw in 20 seconds based on strokes. And the game stopped when it recognized the category of the drawing sketch which is the same as the category of the prompt by the system. In 2018, Sert [18] designed and implemented an application to identify sketches in smart devices. The application contained two modules: a sketch recognition program and a sketch application on mobile phones. The sketch recognition program was a web service program that performs sketch processing and recognition tasks. The sketch application on mobile phones allowed users to draw sketches and sent the sketch to the sketch recognition program. Then the sketch recognition program used a pre-trained SVM model to predict sketch categories, and sent the recognition results back to the sketch application on mobile phones.

In the above mentioned sketch recognition systems, the recognition model is deployed on the server to complete the recognition task. In this way, the recognition time is delayed due to the network and model and the results cannot be fed back in time. Thus, there is no recognition system on the mobile phones which directly embeds models into the mobile devices currently because the recognition models contain many parameters and take up too much memory.

In our work, the sketch recognition application we built on mobile phones is a new attempt. We use the proposed lightweight neural network for sketch recognition to speed up recognition while ensuring accuracy.

3. Methodology

3.1. LW-Sketch-Net

Our lightweight neural network (LW-Sketch-Net for short) ensure recognition accuracy by applying depthwise separable convolutions and adjusting the network for the sparsity of the sketch. Meanwhile, the recognition speed is improved to a large extent. Fig. 1 illustrates our overall framework.

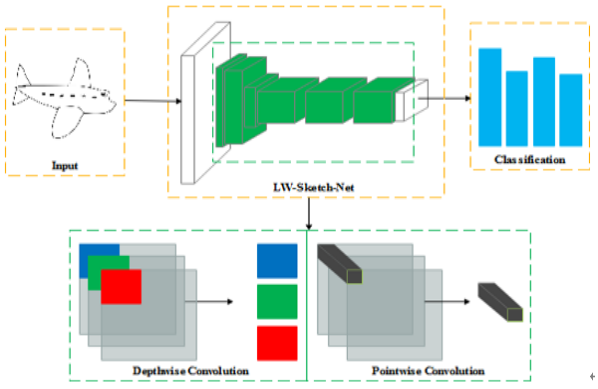

Figure 1. LW-Sketch-Net architecture.

Depthwise separable convolutions

LW-Sketch-Net applies the depthwise separable convolutions proposed by Google [19]. In a standard convolution, both filters and channels input into a new set of out-puts in one step. But the depthwise separable convolutions split the operation into two layers, a separate layer for filtering and another layer for combining channels. Specifically, the depthwise convolution learns the spatial feature on each channel of the input sketch and then the pointwise convolution combines channels features. The decomposition can simplify training models and reduce network parameters. Fig. 2 shows the detailed steps of two convolutions.

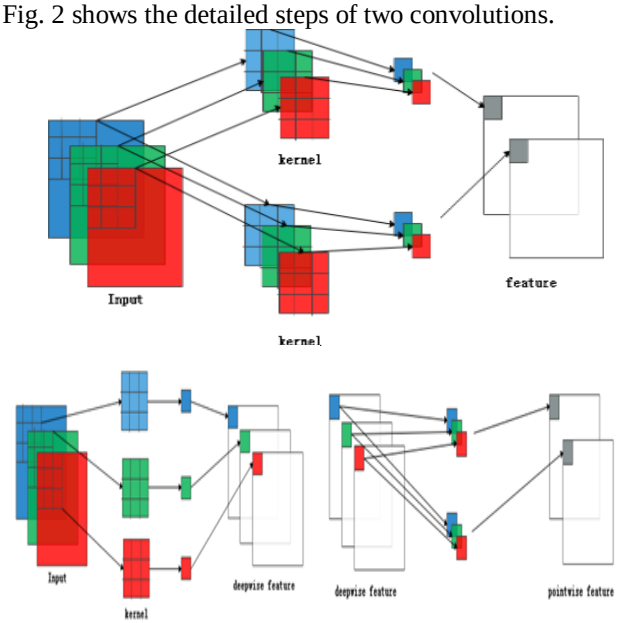

Figure 2. Detailed steps for standard and depthwise separable convolutions

A convolutional layer takes as input a W×H×IC feature map and produces W×H×OC feature maps, where W and H is the spatial width and height of input feature map, IC and OC are the number of input and output channels. The convolution kernel K of size K×K, standard convolutions have the computational combining features and channels. The computational cost of convolution is listed as follows.

$$W \times H \times IC \times K \times K + W \times H \times IC \times OC \times 1 \times 1 \quad (1)$$

$$W \times H \times IC \times OC \times K \times K \quad (2)$$

$$1/OC + 1/K^2 \quad (3)$$

Equation (1) is the total computational cost of depthwise separable convolutions and Equation (2) for standard convolutions. The ratio of two above equations is the reduction computation by using the new convolution form as listed in (3).

LW-Sketch-Net structure

The LW-Sketch-Net structure is built on depthwise separable convolutions as mentioned in the previous section. The specific network architecture is shown in Table 1.

There are some commonalities between LW-Sketch-Net and other sketch-oriented convolutional network architectures. As the common forms in neural networks, depthwise convolutional layers and pointwise convolutional

Table 1. LW-Sketch-Net Architecture

Layer	Type	Filter Size	Filter Num	Output Size
	Input			64×64×3
L1	Conv	9×9	64	64×64×64
L2	Conv_1	9×9	64	64×64×64
	dw	9×9	64	64×64×64
	Conv_1 pw			
L3	Conv_2	9×9	128	32×32×64
	dw	9×9	128	32×32×128
	Conv_2 pw			
L4	Conv_3	9×9	128	32×32×128
	dw	9×9	128	32×32×128
	Conv_3 pw			
L5	Conv_4	9×9	256	16×16×128
	dw	9×9	256	16×16×256
	Conv_4 pw			
L6	Conv_5	9×9	256	16×16×256
	dw	9×9	256	16×16×256
	Conv_5 pw			
L7	Conv_6	9×9	512	8×8×256
	dw	9×9	512	8×8×512
	Conv_6 pw			
L8	Conv_7	9×9	512	8×8×512
	dw	9×9	512	8×8×512
	Conv_7 pw			
L9	Global Average Pooling			1×1×512
L10	Soft max			1×1×340/250

layers are followed by a batchnorm and ReLU nonlinearity with the exception of the final fully connected layer which feeds into a softmax layer for classification However, combined with the unique characteristics of the sketch, more unique aspects in LW-Sketch-Net architecture are as follows.

Larger First Convolutional Layer Filters. The size of the filters in the first convolutional layer is a relatively sensitive parameter since all subsequent processing depend on the output of this layer. Generally, the size in most neural networks is 3×3. But we find that larger filters are more appropriate for sketch through experiments because more contextual information can be captured. The sketches stored in the computer are sparse, and the usual convolution filters size is too small for each convolution operation to learn valid information. Thus we set the size of the first convolution filters to 9x9 to capture more information.

Larger Depthwise Convolutional Layer Filters. As mentioned in previous section, the depthwise convolutions aim to learn spatial feature on each channel of the input sketch. When the size of the filters is small, it is difficult to obtain effective information because sketch is too sparse in the computer compared to the natural image. Thus, a natural intuition would be: is it possible if we can enlarge the size of filters in each depthwise convolutional layer. So avoiding the computational complexity during training and obtaining more information between sketch categories, we use 9×9 filters size in each depthwise convolutional layers.

Fewer Convolution Filters per Layer. In the training of natural image, the first convolutional layer filter of the deep convolutional network mainly extracts information such as image edges and chromatic aberrations. In order to obtain more complete information, the number of filters is generally 64 and the last layer is up to 1024. The sketch lacks colour and texture information, so the number of filters used for natural image recognition is too redundant for sketch recognition task. Moreover, the edge information learned by some filters is repeated, so reducing the number of filters in the sketch recognition network has no significant impact on the acquisition of information. Based on the above reasons, we reduce the number of filters in each layer, respectively 64, 128, 256 and 512, to reduce the amount of network parameters.

Attention Mechanism

Aiming at the diversity of sketches, we should learn the global features of sketches in addition. The attention mechanism is introduced to make the network not only pay attention to the spatial features of sketches, but also consider the distribution of channel weights in feature maps.

As we all known, there are limitations in global information capture because the traditional convolutional neural network loses a large amount of image information during sampling and layer-by-layer transferring. In recent years, many studies on the attention mechanism have emerged in image classification, image segmentation, and image classification.

In 2018, a non-local neural network with a certain generality was proposed by Wang [20] who was inspired by the application of non-local means in image. He aimed to overcome the problem of the traditional convolutional neural network and thought that all feature points should be considered for weighted calculation. And the network is too concerned about the shortcomings of local features. Hu [21] proposed the Squeeze-and-Excitation network in 2019 to explicitly model the interdependence between feature channels. Specifically, the importance of each feature channel is automatically obtained through learning, and then useful features are improved according to the importance to suppress features that are not useful for the current task.

We embed non-local network blocks and Squeeze-and-Excitation blocks into depthwise convolutions respectively, so that the recognition network can learn the global space features and channel features of sketches accurately.

3.2. Sketch recognition application for smartphone

As mentioned in 2.3, some researchers design and implement sketch recognition application to provide users with a tool. As shown in the Fig. 3, they conduct a mobile sketch application

and a web sketch service respectively, which communicate and transfer data to each other. Specifically, the mobile sketch application mainly provides the user with the function of drawing a sketch, then saves and uploads the image to the web server. The web sketch service identifies the received sketches through recognition model and then returns the identified results to the mobile sketch application. Finally, the mobile sketch application presents the recognition result to the user through the interface.

However, these approaches make the recognition time-limited and users cannot get results quickly after submitting sketches. Therefore, the most important thing is to deploy a lightweight recognition model into the recognition system, so that the model can be directly embedded in the user's mobile phones. In this way, we can improve the recognition speed when the user performs sketch recognition and shorten the recognition time. We directly embeds the recognition model into the mobile phones in the sketch recognition application, so that users can quickly obtain the recognition results.

Figure 3. Illustration of traditional sketch recognition system

Our sketch recognition system on mobile phones is an application scenario of our LW-Sketch-Net. The system provides users with functions such as drawing sketches, uploading sketches, and identify sketches. The illustration is shown in the Fig.4. On the one hand, users can use models which have been trained and embedded in the system for sketch recognition. On the other hand, the sketches uploaded by users make the sketch dataset larger. Thus, diversified and noisy sketch dataset is conducive to training accurate and lightweight recognition models. In this way, it can not only improve the accuracy of lightweight sketch recognition models, but also improve the user satisfaction when recognizing sketches. Specifically, a sketch recognition model is trained using a sketch library, and then the model is embedded in our sketch recognition application on mobile phones. The application provides users with the ability to draw sketches, and uses the embedded model to identify user sketches. Users can also upload sketches to the sketch library to achieve the purpose of automatically expanding the sketch dataset. A larger and more diverse sketch dataset can provide a better data foundation for model training.

Specifically, we apply the Tensor Flow Lite framework to embed and load the model on the mobile phone. The framework is a lightweight solution proposed by Google for mobile and embedded devices in 2018. Firstly, we train the recognition models on the sketch dataset and convert the format of models to a format that Tensor Flow Lite can recognize. And then we put the models into the resource folder of android studio along with the category label file, so that these can be embedded into the mobile phone memory. Then we create the interpreter in Tensor Flow Lite and run the models. Finally we get the information such as the identified sketch category, time spent and display the information on the user interface.

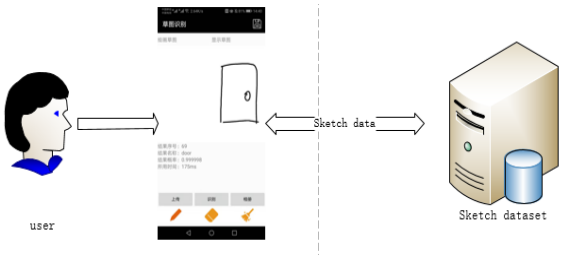

Figure 4. Illustration of sketch recognition application on mobile phones

It is worth noting that our application embeds the LW-Sketch-Net directly into the phone's memory, rather than placing the recognition models on the server. The application receives the user-entered sketches directly on the phone and then performs predictive calculations on the sketches. Then the recognition results are displayed on the user interface. The sketch recognition application is a scenario for the framework we propose. Due to the small amount of parameters in LW-Sketch-Net, the application can get recognition results faster, which further improves the recognition speed.

4. Experiments

4.1. Datasets and settings

We evaluate our model on two benchmarks, the QuickDraw datasets and the TU-Berlin datasets. The first datasets collected by Google in 2017 consists of 50 million hand-drawn sketches with 345 categories which is the first large-scale sketch datasets. Experience has shown that 20,000 sketches are enough to represent a sketch category. So we randomly select 30,000 sketches in each category, where 20,000, 5000 and 5000 are used for training, verification and testing respectively. A total of 10,200,000 sketches are used in our experiments, which we call Quickdraw-10M.

TU-Berlin [2] datasets presented by the Technical University of Berlin in 2012 which is the commonest benchmark in existing researches. It is divided into 250 categories which includes 80 sketches and it has 20,000 sketches totally. Due to the small scale, data augmentation is commonly with deep convolutional neural networks (CNNs) to avoid over-fitting. We replicate sketches with a number of transformations. Specifically, for each input sketch, we do rotation in the range [50, 130, -50, -130] degrees and the total of training instances is 100,000. We rescale all sketches to 64×64 pixels.

4.2. Competitors

We have compared with popular CNNs baselines because our model uses convolutional neural networks. Specifically, we have compared: (1) AlexNet [22], a deep network with five

convolutional and three fully-connected layers, (2) VGGNet-16 [23] has 13 convolutional layers, 3 fully connected layers, and 5 pooling layers. This network was already a deeper network in 2014, (3) ResNet-50 [24] is a representative of the classic residual network, with 50 weight layers, and (4) MobileNets [19], a light weight deep neural networks, based on a streamlined architecture that uses depthwise separable convolutions.

To prove the effectiveness of our architecture, we have studied several other alter-natives on the TU-Berlin benchmark. Here we briefly describe these methods. Sketch-A-Net [12] is the first CNNs framework for sketch recognition. Deep-Sketch [14] and deep-CRNN-sketch [15] are based on CNNs and recurrent neural networks (RNNs). AlexNet-FC [16] proposes a combined architecture with AlexNet and RNNs for sketch recognition task. DVSF [25] uses ensemble of networks to learn the visual and temporal properties of the sketch.

4.3. Results and discussions

Effect of unique aspects of LW-Sketch-Net

To evaluate the effect of the size of the first layer filters, we conduct six experiments by changing the size from 3 to 15. The results are listed in Table 2. The accuracy on two datasets is 83.3% and 83.5% respectively when size is 9×9, which outperform the accuracy when size is 3×3. It is worth noting that the accuracy is not greatly improved when the size is 15×15. The convolution filter is larger, the receptive field will be larger, so that the more picture information is seen and the better the features obtained. However, the larger convolution filters lead to the more the parameters. As shown in Table 2, the increase of the parameter amount is greater than the improvement of the accuracy rate, so we select the convolution filter size is 9×9. Thus, it is necessary to balance accuracy and parameter quantity when selecting the optimal size.

Table 2. Effect of larger first convolutional layer filters

Datasets	Filters size	Accuracy (%)	Parameters (MB)
QuickDraw	3×3	82.9	2.47
	9×9	83.3	2.49
	15×15	83.4	2.52
TU-Berlin	3×3	79.8	2.38
	9×9	83.5	2.40
	15×15	83.8	2.42

We have performed four experiments on two benchmarks to validate the effect of depthwise convolutional layer filters size when the first layer filters size is 9×9. The results are shown in Table 3. More sketch information have be learned by the larger size of depthwise convolutional filters. It achieves 85.1% accuracy on the Quick-Draw-10M datasets and 86.2% accuracy on the TU-Berlin benchmarks, more than 1.8% and 2.7% improvement compared to the situation when the size is 3×3.

Table 3. Effect of larger depthwise convolutional layer filters

Datasets	Filters size	Accuracy (%)	Parameters (MB)
QuickDraw	3×3	83.3	2.49
	9×9	85.1	2.69
TU-Berlin	3×3	83.5	2.40
	9×9	86.2	2.54

Additionally, we have compared the normal number of filters per layer with the special situation that half number of filters every layer. It is noted that the size of first layer and depthwise convolutional layers are 9×9. The detailed results are listed in Table 4. The recognition accuracy is 83.7% and 85.3% respectively on two datasets. Furthermore, the ratio of filter reduction is significantly larger than the ratio of the accuracy reduction. In terms of the amount of parameters, the number of filters in the entire network is reduced by half, and the parameter amount is 30% of the total when the number of filters is not reduced.

Table 4. Effect of fewer convolutional filters per layer

Data sets	Filters num	Accuracy (%)	Parameters (MB)
Quick Draw	64-128-256-512-1024	85.1	2.69
	64-64-128-256-512	83.7	0.85
TU-Berlin	64-128-256-512-1024	86.2	2.54
	64-64-128-256-512	85.3	0.79

Effect of Attention Mechanism

We have tried to increase the attention mechanism on the lightweight network, and performed experimental comparison on two data sets respectively. The results are shown in Table 5. We can find that the recognition ability of the original lightweight network is improved after applying attention to the field of sketch recognition. Specifically, it achieves 84.1% accuracy on the QuickDraw-10M datasets and 85.8% accuracy on the TU-Berlin benchmarks, more than 0.4% and 0.5% improvement compared to the situation when the network do not apply attention mechanism. Ac-cording to Table 5, we can think that the lightweight network is not yet able to effectively extract the features of the sketches. Under the influence of the non-local module, the lightweight network can identify more sketch space features. The Squeeze-and-Excitation blocks is stimulated to reweight the importance of the channel and the network make further adjustments.

Table 5. Effect of Attention Mechanism

Datasets	Attention Mechanism	Accuracy (%)	Parameters (MB)
QuickDraw	No apply	83.7	0.85
	apply	84.1	0.93
TU-Berlin	No apply	85.3	0.79
	apply	85.8	0.87

Comparison against Competitors.

We report the sketch recognition results of our LW-Sketch-Net on QuickDraw datasets, compared to state-of-the-art works in Table 6. The following observations can be made: (1) LW-Sketch-Net outperforms the state-of-the-art photo-oriented CNNs model on the large-scale datasets. AlexNet obtains only 64.5% which has the worst performance among all deep models due to its rough and simple structure. ResNet-50 and VGGNet-16 outperform AlexNet because there are more layers involved in net-works. The appropriate number of network layers helps to capture more sketch features, (2) our model obtains the accuracy by a significant margin, with 3.6% improvement over the performance of RNN&CNN+CEL+SCL [26], only a method de-signed for sketch on QuickDraw datasets. Furthermore, the accuracy of the Sketch-A-Net is 11.9 % lower than the LW-Sketch-Net. The results show that our network has good recognition ability on large-scale datasets, and (3) LW-Sketch-Net is superior to MobileNets while there is much smaller 20% of the total number of parameters. The parameters amount of our network are the least among all networks while maintaining accuracy. It is clear that our network is required for sketch recognition task.

Table 6. Recognition accuracy on QuickDraw datasets

Model	Accuracy (%)	Parameters (MB)
AlexNet[22]	64.5	60
ResNet-50[24]	66.8	25
VGGNet-16[23]	68.3	138
Sketch-A-Net[12]	72.2	17
MobileNets[19]	77.3	4.2
RNN&CNN + CEL +SCL[26]	80.5	123
LW-Sketch-Net	84.1	0.93

Most of the previous research on sketch recognition was based on TU-Berlin benchmark. Our work achieves 85.8% recognition accuracy and clearly outperforms other methods as Shown in Table 7. Specifically, Deep-Sketch and Sketch-A-Net only apply CNNs to recognize sketch, which achieves 69.2% and 74.9% accuracy. Compared to CNN-based methods, the sequential features of the strokes can be captured by RNNs. The recognition accuracy of deep-CRNN-sketch obtains 71.8% which combined the CNNs and RNNs based on Deep-Sketch. DVSF models the visual and sequential patterns of the strokes and achieves 79.6% accuracy. AlexNet-FC which poses the sketch recognition task as a sequence model using gated recurrent unit achieves 85.1%.

Table 7. Recognition accuracy on TU-Berlin datasets

Model	Accuracy (%)
Deep-Sketch[14]	69.2
deep-CRNN-sketch[15]	71.8
Human[2]	73.1
Sketch-A-Net[12]	74.9
Sketch-A-Net[13]	77.9
DVSF[25]	79.6
AlexNet-FC[16]	85.1
LW-Sketch-Net	85.8

Generally, our LW-Sketch-Net outperforms over the performance of competitors, not only the CNN-based methods but also combined CNN and RNN frameworks. It should also be noted that our method has fewer parameters by a significant margin. Our network parameters are only 0.93M, which is much less than the parameters of the CNN-only method, let alone the method of combining CNNs and RNNs.

5. Conclusion

In this work, we propose a lightweight neural network architecture by using depthwise separable convolutions for sketch recognition and develop a sketch recognition application for mobile devices. Leveraging on a large-scale QuickDraw-10M and TU-Berlin benchmarks, we explore the LW-Sketch-Net with fewer parameters, which is understudied in prior works. The experiments demonstrate that our model outperforms other methods. In the field of sketch recognition, our research provides the effective directing for the combination of deep neural networks and mobile terminals.

References

[1] H.Y. Li, H. Hou, X. Guo. (2017) The exploration on global village language: Harmony. In *ICSCA '17 Proceedings of the 6th International Conference on Software and Computer Applications*, 171-175.

[2] M. Eitz, J. Hays, M. Alexa. (2012) How do humans sketch objects? *ACM Transactions on Graphics* (TOG). 4(31).

[3] J. Song, Y.Z. Song, T. Xiang, T.M. Hospedales. (2017) Fine-grained image retrieval: the text/sketch input dilemma. In *BMVC*.

[4] Y. Li, Y.Z. Song, T.M. Hospedales. (2017) Free-hand sketch synthesis with deformable stroke models. *International Journal of Computer Vision.* 169-190.

[5] J. Deng, W, Dong, R, Socher, et al. (2009) ImageNet: a large-scale hierarchical image database. In *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2009)*, Miami, Florida, USA. IEEE.

[6] P. Sangkloy, N. Burnell, C. Ham, and J. Hays. (2016) The sketchy database: learning to retrieve badly drawn bunnies. *ACM Transactions on Graphics*. 122, 3, 411-425.
[7] Q. Yu, F. Liu, Y.Z. Song, T. Xiang, T. M. Hospedales, and C.C. Loy. (2016) Sketch me that shoe. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR).*
[8] D. Ha and D. Eck. (2017) A neural representation of sketch drawings.
[9] X.C. Cao, H. Zhang, S. Liu, X.J. Guo, and L.C. Liang. (2013) SYM-FISH: A symmetry-aware flip invariant sketch histogram shape descriptor. In *ICCV '13 Proceedings of the 2013 IEEE International Conference on Computer Vision*, 313-320.
[10] Y. Li, T. M. Hospedales, Y.Z. Song, and S. Gong.: Free-hand sketch recognition by multi-kernel feature learning. Computer Vision and Image Understanding. 2015, 137, pp. 1-11.
[11] R.K. Sarvadevabhatla, R.V. Babu. (2015) Freehand sketch recognition using deep features. Computer Science.
[12] Q. Yu, Y. Yang, Y. Z. Song, T. Xiang, T.M. Hospedales. (2015) Sketch-a-net that beats humans. In *BMVC 2015*.
[13] Q. Yu, Y. Yang, F. Liu, Y.Z. Song, T. Xiang, T.M. Hospedales. (2017) Sketch-a-net: A deep neural network that beats humans. In *IJCV 2017*, 3(122), 411-425.
[14] P. Zhao, F. Wang, H.T. Liu, Sh. Yao. (2016) Sketch recognition using deep learning. *Journal of Sichuan University*.
[15] P. Zhao, F. Wang, H.T. Liu, Sh. Yao. (2018) A sketch recognition method based on deep convolutional-recurrent neural network.
[16] R.K. Sarvadevabhatla, J. Kundu, R.V. Babu. (2016) Enabling my robot to play pictionary: recurrent neural networks for sketch recognition. *ACMMM*.
[17] C. Xiao, C. Wang, L. Zhang. (2015) PPTLens: Create digital objects with sketch images.
[18] Sert. M, Boyacı. Emel. (2019) Sketch recognition using transfer learning. *Multimedia Tools and Applications*.
[19] A.G. Howard, M. Zhu, B. Chen. (2017) MobileNets: Efficient convolutional neural networks for mobile vision applications.
[20] X.L. Wang, R. Girshick, A. Gupta and K.M. He. (2018) Non-local neural networks.
[21] J. Hu, S. Li, G. Sun. (2017) Squeeze-and-Excitation networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. (99):1-1.
[22] A. Krizhevsky, I. Sutskever, G.E. Hinton. (2012) Imagenet classification with deep convolutional neural networks. In *NIPS'12*, 1097-1105.
[23] K. Simonyan, A. Zisserman. (2015) Very deep convolutional networks for large-scale image recognition.
[24] K. He, X. Zhang, S. Ren. (2016) Deep Residual Learning for Image Recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
[25] J. He, X. Wu, Y. Jiang, B. Zhao, Q. Peng. (2017) Sketch recognition with deep visual-sequential fusion model. In *Proceedings of the 25th ACM international conference on Multimedia*, 448-456.
[26] P. Xu, Y. Huang, T. Yuan. (2018) SketchMate: Deep Hashing for Million- Scale Human Sketch Retrieval.