Deep Neural Networks for Sketch Recognition

Article · January 2015

Source: arXiv

2 authors, including:

Timothy Hospedales
The University of Edinburgh
128 PUBLICATIONS **2,539** CITATIONS

SEE PROFILE

DEEP NEURAL NETWORKS FOR SKETCH RECOGNITION

Yongxin Yang & Timothy M. Hospedales

Electronic Engineering and Computer Science
Queen Mary, University of London
{yongxin.yang, t.hospedales}@qmul.ac.uk

ABSTRACT

Deep Neural Networks (DNNs) have recently outperformed traditional object recognition algorithms on multiple large-scale datasets, such as ImageNet. However, the model trained on ImageNet fails on recognising the sketches, because the data source is dominated by photos and all kinds of sketches are roughly labelled as 'cartoon' rather than their own categorises (e.g., 'cat'). Most of sketch recognition methods still heavily rely on the hand-craft feature extraction techniques, thus it is interesting to know whether DNNs can eliminate the needs of specific feature engineering in this area, and how the architecture is designed for sketch recognition purpose. To the best of our knowledge, it is the first deep neural network model for sketch classification, and it has outperformed state-of-the-art results in the TU-Berlin sketch benchmark. Based on that, we outline a sketch image retrieval system in a unified neural network framework.

Index Terms— Convolutional Neural Networks, Deep Learning, Sketch Recognition

1. INTRODUCTION

Sketch recognition targets on classifying human drawn sketches into categories. As a sub-task of object recognition, it remains a degree of uniqueness because the sketches usually lack colour and texture information compared to photos, which makes classification a non-trivial problem. On the other hand, DNNs based models significantly improved the performance of object recognition in recent years, however little research has been conducted specifically for recognising sketch objects.

We start from evaluating the sketches classification performance of a DNN model trained on ImageNet [1]. By uploading a cat sketch to Clarifai[1] [2], we can see that the possible tags are cartoon, character, funny etc. (Fig. 1(a)). Some of them are semantically correct, but we actually expect 'cat' as the result. Further, if we ask for similar images based on the prediction, we may just get some random cartoons, which is undesirable (Fig. 1(b)). This can be easily interpreted: ImageNet dataset is dominated by photos, and most of the sketches are labelled as 'cartoon' rather than their own categories. Given the large domain shift between photos and sketches, fine-tuning is unlikely to achieve the desirable performance. Therefore we present a deep neural network (sketch-DNN) designed for sketch recognition from scratch, and its precision on TU-Berlin sketch dataset [3] is 72.2%, which is comparable with human performance 73%.

The contributions of our work are:

- To the best of our knowledge, it is the first deep neural network based algorithm for sketch recognition. It surpasses the state-of-the-art results in the TU-Berlin benchmark.
- We reveal the keynotes for building sketch-DNN in contrast to the existing DNNs that are designed for photos.
- DNN makes it possible to construct a sketch image retrieval system within a unified neural network framework.

The rest sections are organised as follows. In section 2, we review the existing work on general image recognition and specific studies for sketches. In section 3, we introduce and explain the architecture of sketch-DNN in detail. Section 4 includes the experiments and the comparison with the existing methods. Section 5 concludes the paper and outlines a new sketch image retrieval system involving with sketch-DNN as future work.

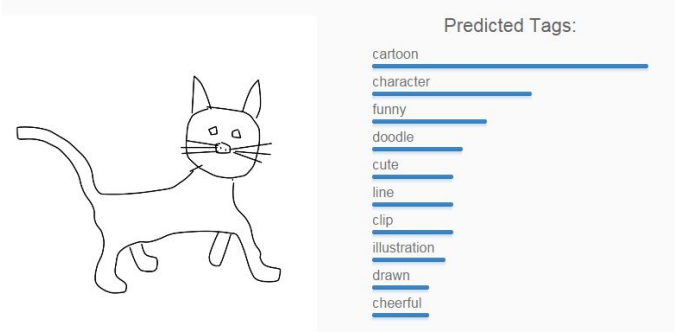
2. RELATED WORK

In this section, we will briefly review the work in general image classification as well as the specific approaches for sketches.

2.1. Image Classification

Image classification is a classic supervised learning problem. Given the images from training set and the corresponding labels, a learning algorithm is expected to produce a model

[1] http://www.clarifai.com/



(a) Cat Sketch Recognised by a DNN Model

(b) Similar Images based on the Predicted Tags

Fig. 1: A demo that shows the limited capability of a DNN model trained on ImageNet for sketch recognition: (a) is the results of predicting a cat sketch, the top-1 tag is 'cartoon", which is semantically correct but not the category name. (b) based on the prediction of (a), the similar images are arbitrary cartoons rather than other cat sketches as expected.

(classifier) that assigns correct labels to the images in testing set. There are two separate pipelines.

The first one employs some signal processing techniques such as SIFT [4] and HOG [5] for feature extraction, which usually decompose an image into a set of descriptors. Then some unsupervised algorithms, mainly dictionary learning, build a dictionary (bases for the descriptors) such that an image can be eventually coded as a feature vector (usually the weights of bases). The typical examples are bag-of-words [6] using K-means, and fisher vector [7] using Gaussian Mixture Models or other types of generative models. Finally, a supervised learning algorithm trains a classifier on these vectorised image representations and labels.

The second is the neural network approach. It usually starts from the raw pixels of an image directly, layer-by-layer training enables the neural network to distinguish different objects. The representative researches in this area are LeNet [8] for handwritten digit recognition and AlexNet [9] for general image classification.

The latter approach is potentially better in the light of its joint learning process in contrast to the first one that builds on separate blocks of both unsupervised and supervised algorithms. A drawback of neural networks is the difficulty of training, esp. for the ones with multiple layers (termed as deep neural network). Thanks to the modern GPU, the training time is now acceptable with a single machine even for millions of images.

2.2. Sketch Recognition

Sketch recognition is a relatively new field, and most of studies in this area focus on the first pipeline mentioned in previous section. For example, [3] uses a SIFT-variant and bag of words model. [10] assembles multiple low-level features and form a star-graph for each sketch. [11] uses SIFT and GMM-based fisher vector. So far, we have not found public work on a designing DNN model for sketch recognition. Another interesting topic is sketch image retrieval [12, 13, 14, 15], and it could gain benefits from better sketch recognition solution.

3. MODEL

It remains an open question how to design the architecture of deep neural networks, but a typical structure starts from multiple convolutional layers, with one or more fully-connected layers following. We adopt this pipeline: first we place five convolutional layers, and each of them is followed by a rectifier layer (ReLU) [16], while the first, second and fifth layers are followed by a maximum pooling layer (Maxpool) respectively. The filter size of the sixth convolutional layer (index: 14) is 6×6, which is the same as the output from previous pooling layer, thus it is precisely a fully-connected layer. Then two more fully connected layers are appended. Dropout [17] is applied on the first two fully connected layers. The last layer (top layer) has 250 output units corresponding to 250 categories, upon which we place a softmax loss function to produce the loss for back-propagation. Bias terms are present and initialised 0.1. The initial weights are drawn from a Gaussian distribution with $\mu = 0$ and $\sigma = 0.01$.

The details of the designed sketch-DNN is listed in Table 1. For tidy representation, we do not distinguish convolutional and fully-connected layers explicitly.

3.1. Design Keynotes

Most of the existing DNNs are proposed without explaining why the parameters, such as filter size, stride, filter number, pooling size, are set to be so.

It is nearly impossible to exhaustively verify the effect of every free parameter, nevertheless we summarise some findings that are considerably different from the DNNs for photographic images such as AlexNet [9] and DeCAF [18].

Ind	Type	Filter Size	Filter Num	Stride	Pad
1	Conv	16×16	128	3	0
2	ReLU	-	-	-	-
3	Maxpool	3×3	-	2	0
4	Conv	5×5	256	1	2
5	ReLU	-	-	-	-
6	Maxpool	3×3	-	2	1
7	Conv	3×3	512	1	1
8	ReLU	-	-	-	-
9	Conv	3×3	512	1	1
10	ReLU	-	-	-	-
11	Conv	3×3	256	1	0
12	ReLU	-	-	-	-
13	Maxpool	3×3	-	2	1
14	Conv	6×6	4096	1	0
15	ReLU	-	-	-	-
16	Dropout	Rate: 0.55	-	-	-
17	Conv	1×1	4096	1	0
18	ReLU	-	-	-	-
19	Dropout	Rate: 0.55	-	-	-
20	Conv	1×1	250	1	0

Table 1: The architecture of sketch-DNN

3.1.1. Larger First Layer Filter

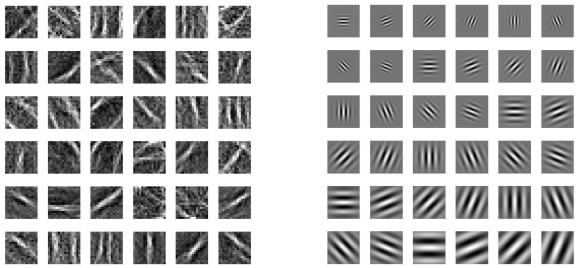
The size of filters in the first convolutional layer might be the most sensitive parameter, on which all the following works are built.

We find that it is necessary to use much larger filters (esp. in the first layer) for sketch classification. The ordinary choice of this parameter for photo recognition is 7×7 or 11×11 when the image size is around 200×200. However, we find that 16×16 filters work better than the smaller ones for sketches of the similar size. The reason might be that the sketch lacks texture information, e.g., a small round-shaped patch can be recognised as eye or fastener in a photo (texture-rich) but it is generally infeasible for sketches. These larger filters help to capture more structured rather than textured information.

As illustrated in Fig. 2, the filters in the first layer tend to produce biologically plausible feature detectors like stroke. These filters in Fig.2(a) are visually similar to Gabor filters [19], and this phenomenon has been noticed by other researchers such as [20].

3.1.2. More Filters

A reasonable deduction could be that sketch-DNN uses fewer filters than photo-DNN, because photos are more informative in terms of their colour and texture. Unexpectedly, we still find that more filters generally lead to better performance. A possible explanation is that sketches lack consistency on the same object: a large cat and a small one may share furry information in photos but they could be totally different in sketches. The intra-class variability of sketches is significantly larger than real world images due to user variability in sketching style and abstraction, thus more filters are needed to ensure that enough variants for the same object can be captured.



(a) A subset of filters in the first layer (b) A subset of real parts of Gabor filters

Fig. 2: Illustration of the filters in the first convolutional layer: (a) randomly selected filters from the first layer in our model, and (b) the real parts of some Gabor filters. we can find some visual similarities between.

3.1.3. Remove Local Response Normalization

Local Response Normalization (LRN) [9] implements a form of lateral inhibition, which is found in real neurons. In their work, LRN could be more correctly termed as "brightness normalisation". However, we find that LRN does not help in our case because brightness is not present in sketches. Removing LRN layers makes the learning faster without sacrifice of performance.

3.1.4. Overlapped Pooling

For a long time, 2×2 maximum pooling with stride 2 is the most popular choice of building convolutional neural networks. It efficiently reduces the size of the layer by 75% while bringing translations/distortions invariance to certain degree. In our model, we use it with a small modification: 3×3 pooling size overlapping with stride 2. We find this small change brings around 1% improvement without too much extra time consumed. The same strategy has been found in [9].

3.1.5. Deeper but Higher Dropout Rate

Deeper neural network brings better performance. This applies to the sketch-DNN as well. However, a drawback is that more complex network structure leads to the over-fitting problem. We adopt a classic method – dropout [17] – to prevent over-fitting. For the sketch classification task, we do not have so many data as ImageNet, so we use a much higher dropout rate (the probability that randomly deactivates a unit) – 55%.

Method	Feature	Classifer	mAP
[3]	SIFT-variant; BoG	SVM	56%
[10]	HOG, LBP etc.; Stargraph	KNN	61.5%
[11]	SIFT; Fisher Vec. (GMM)	SVM	68.9%
Ours	Deep Neural Network	MLP	**72.2%**
Human	-	-	**73%**

Table 2: Results Comparison

4. EXPERIMENTS

We evaluate the designed network on the TU-Berlin sketch dataset [3]. It contains 250 unique object categories, and each of them has 80 images sketches equally, which forms 20000 single-channel images.

The original size of each image is $[1111, 1111, 1]$ (Width, Height, Colour Channel). As the first step of preprocessing, we rescale the image to 220×220. Then we subtract the means of all pixels.

Note that the dataset is evenly distributed, i.e., each category has the same number of instances, the mAP (mean average precision) is equal to precision.

Our model is implemented with MatConvNet [21]. The optimisation method is mini-batch gradient descent, and the number of instances in one batch is 50. The learning rate is 0.001 for weights with decay multiplier $= 1$ and 0.002 for biases without decay. We do three-fold cross-validation with the same setting as suggested by [3]: the experiments are carried out three times, and 33% of data are chosen to test in turn with the rest 67% training each time. Finally, sketch-DNN achieves 72.2% mAP that outperforms the alternatives, and it has been very close to human performance in this task.

To compare with previous methods, we briefly summarise their results and the key techniques in Table 2. Though [10] uses 4-fold cross-validation, which brings a slightly larger training dataset (75% v.s. 67%), we still include their result as a representative example of sketch-specific engineered feature (star-graph).

The training is carried out on a single NVIDIA K40c card, with the GPU mode on, the memory consumption (video card only) is about 11G, and it takes around 6 minutes for a full pass of all training instances. The forward-feed (testing) process handles around 80 images per second.

5. CONCLUSION AND FUTURE WORK

We first proposed a deep neural network architecture for sketch recognition (sketch-DNN), and discussed the differences on design compared to the DNNs for photographic images. The model achieves highest accuracy among many existing methods, but there is still room to improve by data augmentation: the dataset can be enriched 10 times easily through (i) cropping five patches (four corners and centre: $1 \rightarrow 5$) and (ii) flipping them horizontally (mirror: $5 \rightarrow 10$). These transformations are label preserving and generally helpful [22]. Despite of being commonly used tricks in training DNNs, they are intentionally skipped to ensure the fair comparison with other studies that do not use them.

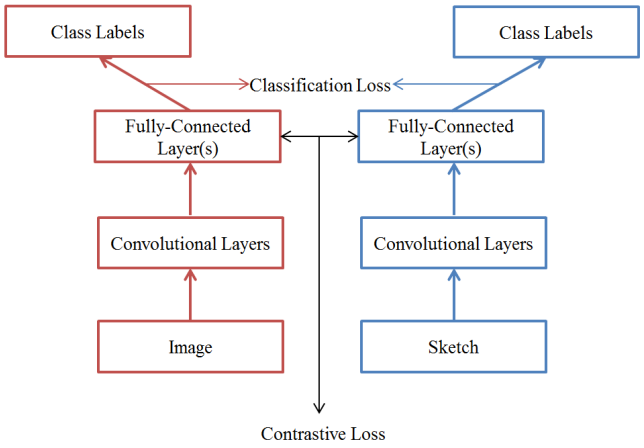


Fig. 3: A blueprint of sketch image retrieval system

In the future work, we will involve this sketch-DNN with a larger sketch image retrieval system illustrated in Fig. 3.

This two-branch neural network structure is inspired by [23]. The key differences are (i) we do not share the parameters for convolutional layers because photos and sketches are not supposed to have the same architecture, which we have discussed in section 3.1 and (ii) the functionality for classification is still kept for better discriminative purpose. On top level, the output units of one or more fully connected layers, on which the contrastive loss function [23] is placed, can serve as a representation (vectorised coding) for the sketch, then the image retrieval is achieved by measuring distance between sketch and image coding. Contrastive loss function, intuitively, can gather close the coding of intra-class objects and push away the inter-class objects. Classification loss function again benefits for inter-class discriminative purpose. Due to the new designed sketch-DNN, the whole system can be jointly trained by standard back-propagation, which potentially leads to better performance compared to the alternatives built by separately trained blocks.

Acknowledgements We gratefully acknowledge the support of NVIDIA Corporation for the donation of the GPUs used for this research.

6. REFERENCES

[1] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A Large-Scale Hierarchical Image Database," in *CVPR09*, 2009.

[2] Matthew D. Zeiler and Rob Fergus, "Visualizing and understanding convolutional networks," in *Computer Vi-*

sion - ECCV 2014 - 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part I, 2014, pp. 818–833.

[3] Mathias Eitz, James Hays, and Marc Alexa, "How do humans sketch objects?," *ACM Trans. Graph. (Proc. SIGGRAPH)*, vol. 31, no. 4, pp. 44:1–44:10, 2012.

[4] David G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vision*, vol. 60, no. 2, pp. 91–110, Nov. 2004.

[5] Navneet Dalal and Bill Triggs, "Histograms of oriented gradients for human detection," in *In CVPR*, 2005, pp. 886–893.

[6] J. Sivic and A. Zisserman, "Video Google: A text retrieval approach to object matching in videos," in *Proceedings of the International Conference on Computer Vision*, Oct. 2003, vol. 2, pp. 1470–1477.

[7] Florent Perronnin, Jorge Snchez, and Thomas Mensink, "Improving the fisher kernel for large-scale image classification," in *IN: ECCV*, 2010.

[8] Le Cun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Handwritten digit recognition with a back-propagation network," in *Advances in Neural Information Processing Systems*. 1990, pp. 396–404, Morgan Kaufmann.

[9] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems 25*, F. Pereira, C.J.C. Burges, L. Bottou, and K.Q. Weinberger, Eds., pp. 1097–1105. Curran Associates, Inc., 2012.

[10] Yi Li, Yi-Zhe Song, and Shaogang Gong, "Sketch recognition by ensemble matching of structured features," in *In British Machine Vision Conference (BMVC)*, 2013.

[11] Rosália G. Schneider and Tinne Tuytelaars, "Sketch classification and classification-driven analysis using fisher vectors," *ACM Trans. Graph.*, vol. 33, no. 6, pp. 174:1–174:9, Nov. 2014.

[12] M. Eitz, K. Hildebrand, T. Boubekeur, and M. Alexa, "Sketch-based image retrieval: Benchmark and bag-of-features descriptors," *Visualization and Computer Graphics, IEEE Transactions on*, vol. 17, no. 11, pp. 1624–1636, Nov 2011.

[13] Mathias Eitz, Kristian Hildebrande, Tamy Boubekeur, and Marc Alexa, "An evaluation of descriptors for large-scale image retrieval from sketched feature lines," *Computer & Graphics*, 2010.

[14] Rui Hu, Mark Barnard, and John P. Collomosse, "Gradient field descriptor for sketch based retrieval and localization.," in *ICIP*. 2010, pp. 1025–1028, IEEE.

[15] Rui Hu and John Collomosse, "A performance evaluation of gradient field hog descriptor for sketch based image retrieval," *Comput. Vis. Image Underst.*, vol. 117, no. 7, pp. 790–806, July 2013.

[16] Y. LeCun, L. Bottou, G. Orr, and K. Muller, "Efficient backprop," in *Neural Networks: Tricks of the trade*. 1998, Springer.

[17] Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," *arXiv preprint arXiv:1207.0580*, 2012.

[18] Jeff Donahue, Yangqing Jia, Oriol Vinyals, Judy Hoffman, Ning Zhang, Eric Tzeng, and Trevor Darrell, "Decaf: A deep convolutional activation feature for generic visual recognition," in *ICML*, 2015.

[19] D. Gabor, *Theory of communication*, Institution of Electrical Engineering, 1946.

[20] Marijn Stollenga, Jonathan Masci, Faustino J. Gomez, and Jürgen Schmidhuber, "Deep networks with internal selective attention through feedback connections," *CoRR*, vol. abs/1407.3068, 2014.

[21] A. Vedaldi and K. Lenc, "Matconvnet – convolutional neural networks for matlab," *CoRR*, vol. abs/1412.4564, 2014.

[22] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman, "Return of the devil in the details: Delving deep into convolutional nets," in *British Machine Vision Conference*, 2014.

[23] Sumit Chopra, Raia Hadsell, and Yann LeCun, "Learning a similarity metric discriminatively, with application to face verification," in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*. IEEE, 2005, vol. 1, pp. 539–546.