DeformSketchNet: Deformable Convolutional Networks for Sketch Classification

Natasha Goenawan
natagoh@stanford.edu

Mandy Lu
mlu355@stanford.edu

Abstract

Compared to natural images or photographs, human sketches are less detailed with high variation in how different people sketch the same objects, causing effective image classification techniques for natural images such as CNNs to underperform. Previous models for sketch classification approach this problem with extensive data preprocessing, handcrafted features, and scale-invariant algorithms such as SIFT and BoW, which can be complex and time-consuming. We introduce the use of deformable convolutions, which augment spatial sampling locations in convolutions to learn robustness to geometric transformations. We have found that deformable convolutional networks are an easily trained end-to-end approach to sketch classification which improves classification performance.

1 Introduction

Humans have used sketches to express and record their ideas since the dawn of civilization. However, sketch classification is a relatively unexplored problem that can yield insights into how humans perceive, categorize, and represent objects. Sketch classification differs from natural image classification in that sketches are less visually complex than photographs. This can pose several problems. First, whereas photographs are rich in visual information, the sparsity and abstract nature of sketches can make feature extraction and classification difficult, especially with a large amount of object categories. In addition, sketches can vary widely by quality and artistic interpretation, making sketch classification particularly challenging. While people generally agree on what an object looks like in a basic sense, how they ultimately visualize it can vary. Thus, a key challenge in sketch recognition is accommodating geometric variations and transformations in object scale, pose, viewpoint, and part deformation.

Human perception of objects and subsequent transcription may primarily capture high level features and relative location of parts, omitting a lot of information present in natural images. Initially, we sought to use a deformable parts model to learn these relative offsets, but DPMs require extensive processing power for multiclass classification. However, the intuition behind applying deformable convolutions to sketch recognition may be similar. In this paper, we explore the effectiveness of deformable convolutions on multiclass sketch classification.

2 Related Work

2.1 Background

The seminal Eitz et al. [2] paper "How Do Humans Sketch Objects?" utilized a bag-of-features model for feature extraction and multi-class support vector machines to classify sketches. Recent papers on sketch classification primarily draw from Eitz' work and either slightly modify or provide verification of the reproducability of the techniques first described by Eitz et al. [Zhu, B., Quigley, E. 2015]. More recently, Google's Quick, Draw! (`https://quickdraw.withgoogle.com/`) is an online "game" that can identify your sketches in 20 seconds or less.

Eitz et al. [2] was able to demonstrate classification rates can be achieved for computational sketch recognition by using local feature vectors, bag of features sketch representation and SVMs to classify sketches. Schneider et al. [6] then modified the benchmark proposed by Eitz et al [2] by making it more focused on how the image should like, rather than the original drawing intention, and they also used SIFT, GMM based on Fisher vector encoding, and SVMs to achieve sketch recognition. Previous work on sketch recognition generally extracts hand crafted features from the sketch followed by feeding them to a classifier. Yu et al. [11] proposed Sketch-a-Net, a different type of CNN that is customizable towards sketches while Sarvadevabhatla et al. [5] used two popular CNN architectures (ImageNet and a modified LeNet) to fine-tuned their parameters on the TU-Berlin sketch dataset in order to extract deep features from CNNs to recognize hand-drawn sketches. The current state-of-the-art is DeepSketch2 [8], where they propose a ConvNet for classification but they also include feature extraction and similarity search in their model.

2.2 Deformable Convolutional Layers

In general, there are two ways to handle geometric transformations and variance in input images. The first is to add the desired variations to the training dataset. The second is to use transformation-invariant features and algorithms such as SIFT. However, both of these solutions use prior knowledge and this prevents generalization by the model.

Furthermore, the sparsity of features in human sketches hampers the effectiveness of convolutions. For example, the fine details that CNNs use to classify natural images may not be at all relevant in sketches due to their sparse nature. Also, because people have varied interpretations on what an object is, they don't always draw similar objects and this can pose additional problems for CNN classification.

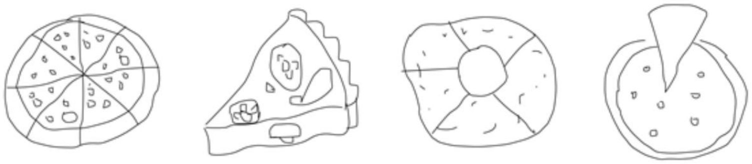


Figure 1: Different interpretations of a pizza

In contrast, deformable convolutions are a recent development that adds 2D offsets to the regular grid sampling locations in standard convolution and enables free form deformation of the sampling grid. The offsets are learned without supervision from the preceding feature maps, via additional convolutional layers. These deformations allow for generalization of various transformations for scale, aspect ratio, and rotation and hence may better capture the primary conceptual features that humans remember about objects.

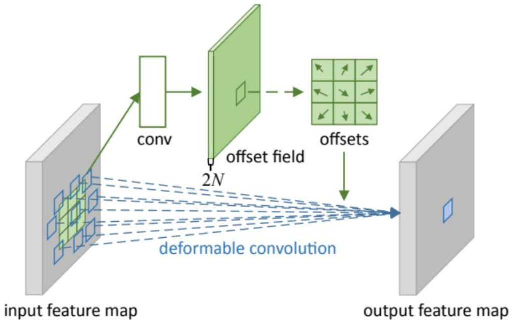


Figure 2: Illustration of 3×3 deformable convolution

As proposed by Dai et al. [1], the deformable convolution consists of 2 steps: (1) sampling using a regular grid $\mathcal{R}$ over the input feature map x; (2) summation of sampled values weighted by w. In standard convolution, each output on the feature map y is given by

$$y(p_0) = \sum_{p_n \in \mathcal{R}} w(p_n) \cdot x(p_0 + p_n) \tag{1}$$

By contrast, in deformable convolution, the regular grid $\mathcal{R}$ is augmented with offsets $\{\Delta p_n | n = 1, \ldots, N\}$ where $N = |\mathcal{R}|$. Then, each location p_0 on the output feature map y is given by

$$y(p_0) = \sum_{p_n \in \mathcal{R}} w(p_n) \cdot x(p_0 + p_n + \Delta p_n) \tag{2}$$

The offsets are obtained by applying a convolutional layer over the same input feature map. During training, both the convolutional kernels for generating the output features and the offsets are learned simultaneously. To learn the offsets, the gradients are backpropagated through a bilinear interpolation of Eq. (2).

3 Methodology

3.1 Dataset

For this project, we will be utilizing the TU-Berlin sketch database collected from 1,350 participants from the crowdsourcing platform Amazon Mechanical Turk. This database consists of 20,000 sketches divided evenly across common 250 object categories. Images are provided as 1111×1111px PNG files. To make the dataset more manageable, we first resized every image to 128×128px using bilinear interpolation. For training we used a 80/20/20 train/dev/test split, which is a common distribution for our medium size dataset of 20,000 images. Several prior models have found metrics on this model, listed in results.

To preprocess the data, we split the images in the TU-Berlin sketch dataset into 16000 training, 2000 dev and 2000 test classes. We then resize the images into 128x128 pixel files and apply a random horizontal flip to the training images to augment the data. We chose to implement minimal data augmentation in order to better study the effectiveness of standalone deformable convolution layers at learning geometric invariance.

- `https://github.com/mlu355/DeepSketch/`

3.2 Baseline Model

Our baseline is a 5-layer convolutional neural network with 3 convolutional layers and 2 fully connected layers. We apply a softmax function to the output in order to classify our dataset into one of 250 classes.

The final baseline accuracy is 48.7%, which is significantly less than human classification at 73% accuracy and the current best model at 77% accuracy (DeepSketch) [7]. This means there is both a 28% gap between the baseline and the current best machine model which is around human performance.

3.3 Proposed Model: DeformSketchNet

Our proposed model, DeformSketchNet, is an 8-layer convolutional neural network with 5 convolutional layers, 1 deformable convolutional layer, and 2 fully connected layers. We apply a softmax function to the output in order to classify our dataset into one of 250 classes. Basic regularization techniques such as dropout, batch normalization, and learning rate decay have been added.

4 Results

Our proposed model, DeformSketchNet, achieved a test accuracy of 62.6%. In comparison, this is better than both our standard CNN baseline and the original classifier proposed by Eitz et al. which utilized an SVM classifier trained on extracted SIFT features [2]. Current state-of-the-art models achieve much higher accuracies, with DeepSketch and DeepSketch 2 respectively reaching 75.4% and 77.7% cross-validation accuracies. However, these models utilize pretrained models and

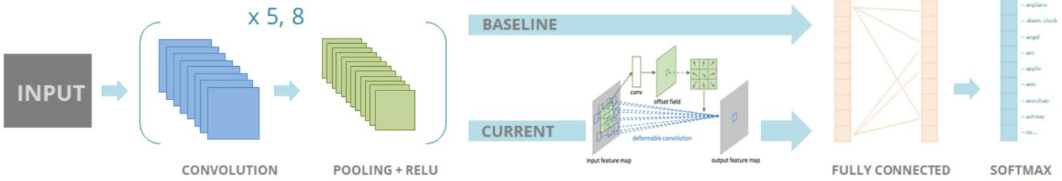


Figure 3: Diagram of our baseline and deformable convolution layer model. Note the addition of a deformable convolutional layer in our proposed model.

Table 1: Deformable convolutional network architecture.

Layer	Kernel	Filters	Stride	Padding	Output Size
Conv	3×3	32	1	1	$1 \times 128 \times 128$
ReLU	–	–	–	–	
MaxPool	2×2	–	–	–	
Conv	3×3	64	1	1	$64 \times 64 \times 64$
ReLU	–	–	–	–	
MaxPool	2×2	–	–	–	
Conv	3×3	64	1	1	$64 \times 32 \times 32$
ReLU	–	–	–	–	
MaxPool	2×2	–	–	–	
Conv	3×3	128	2	1	$128 \times 17 \times 17$
ReLU	–	–	–	–	
MaxPool	2×2	–	–	–	
Conv	3×3	256	2	1	$256 \times 4 \times 4$
ReLU	–	–	–	–	
MaxPool	2×2	–	–	–	
Deform Conv	3×3	128	1	1	$128 \times 4 \times 4$
ReLU	–	–	–	–	
MaxPool	2×2	–	–	–	
2 Fully Connected					250

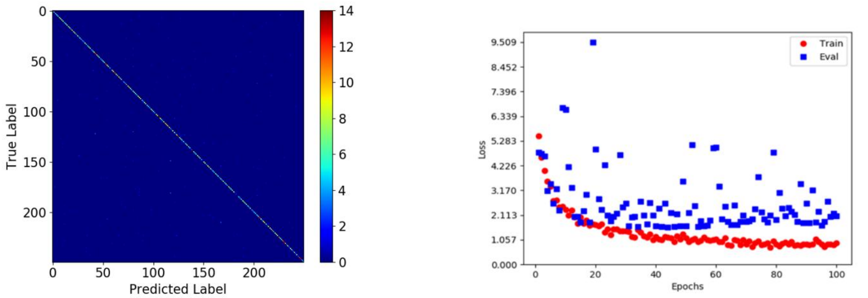
complex, hand-crafted features; in addition, DeepSketch2 considers stroke order. By contrast, we have demonstrated that deformable convolutional networks are an easily trained end-to-end approach to sketch classification which improves classification performance. In addition, our model performs similarly to the ResNet + Dropout model by Lu and Tran with 65.6% accuracy which uses more data augmentation, ResNet blocks and a much deeper network (15 layers) compared to our model.

Table 2: Hyperparameters

Hyperparameter	Baseline	DeformSketchNet
learning rate	1×10^{-3}	1×10^{-3}
batch size	32	64
epochs	50	60
dropout rate	0.8	0.1
weight decay	n/a	0.001
confusion factor	n/a	0.2

Table 3: Test accuracy comparison of our model versus other methods.

Model	Accuracy
Human [2]	73 %
SIFT-variant+BoF+ SVM [2]	56 %
IDM+SVM [5]	71.30 %
DeepSketch ConvNet [7]	75.42 %
DeepSketch2 ConvNet [8]	77.69 %
ResNet + Dropout [4]	65.6%
Baseline - ConvNet	49 %
Current - DeformConvNet	62.6 %



(a) Confusion matrix

(b) DeformSketchNet train and eval loss

Figure 4: We found that using accuracy as a primary metric leads to the imbalanced class problem, common in multi-class classification problems. To combat this, we used accuracy and a confusion matrix to evaluate performance and incorporated confusion matrix into our loss function. Visualization of the confusion matrix provides a more intuitive understanding of the performance of multiclass classification.

5 Conclusion and Future Work

We have found that deformable convolutional networks present an exciting new prospect for end-to-end sketch classification. In the future, we hope to explore what the best methods for incorporating deformable convolutions are, such as how many deformable convolution layers to add and with what parameters. In addition, we would like to tune our hyperparameters more or incorporate any pre-trained CNN models as some of our inspiration papers did. Additionally, further research could use a larger data set such as Google's Quick Draw Dataset which consists of 50 million drawings across 345 categories. This project is easily extendable and poses an interesting and relatively unexplored problem that can yield insights into how humans perceive, categorize, and represent objects.

References

[1] Dai Jifeng, Qi Haozhi, Xiong Yuwen, Li Yi, Zhang Guodong, Hu Han, and Wei Yichen. Deformable convolutional networks. arXiv:1703.06211, 2017.

[2] Eitz, M., Hays, J., & Alexa, M. (2012). How do humans sketch objects? ACM Transactions on Graphics,31(4), 1-10. doi:10.1145/2185520.2335395

[3] Felzenszwalb, P., Girshick, R., McAllester, D., & Ramanan, D. (2010). Object detection with discriminatively trained part based models. IEEE Transactions on Pattern Analysis and Machine Intelligence, 32(9), 1627-1645.

[4] Lu, W., Tran, E.. (2017). Free-hand Sketch Recognition Classification. CS 231N Project Report.

[5] Sarvadevabhatla, R. and Babu., R. (2015) Freehand sketch recognition using deep features.

[6] Schneider, R., and Tuytelaars., T. (2014) Sketch classification and classification-driven analysis using fisher vectors. ACM Transactions on Graphics (TOG), 33(6):174.

[7] Seddati, O., Dupont, S., & Mahmoudi, S. (2015). DeepSketch: Deep convolutional neural networks for sketch rec- ognition and similarity search. 2015 13th International Workshop on Content-Based Multimedia Indexing (CBMI). doi:10.1109/cbmi.2015.7153606

[8] Seddati, O., Dupont, S., & Mahmoudi, S. (2016). DeepSketch2: Deep convolutional neural networks for partial sketch recognition. 2016 14th International Workshop on Content-Based Multimedia Indexing (CBMI). doi:10.1109/ cbmi.2015.7153606

[9] Yesilbek, K., Sen, C., Cakmak, S., & Sezgin., T. (2015). Svmbased sketch recognition: which hyperparameter interval to try? Proceedings of the workshop on Sketch-Based Interfaces and Modeling, 117–121.

[10] Yin, R., Monson, E., Honig, E., Daubechies, I., & Maggioni, M. (2016). Object recognition in art drawings: Transfer of a neural network. 2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). doi:10.1109/icassp.2016.7472087

[11] Yu, Q., Yang, Y., Liu, F., Song, Y., Xiang, T., and Hospedales., T. (2016). Sketch-a-net: A deep neural network that beats humans. International Journal of Computer Vision, 1–15.

[12] Zhu, B., Quigley, E. (2015). Sketch-based Object Recognition. CS 231A Project Report.

[13] `https://github.com/ChunhuanLin/deform_conv_pytorch`

[14] `http://cybertron.cg.tu-berlin.de/eitz/projects/classifysketch/`