Sketchai: using FCNS to Extract Line ART Drawings

Raghav Jadia, Siddharth Sampat, Ritika Pandey, Manohar R, Supriya P.

Abstract: ***The digital revolution has improved every field of human lives. And field of ART is no exception. The rapid development of modern technology and techniques has made an impact on the work of painters, sketch artists and even comic book writers. One would be hard pressed to find artists in this day and age who haven't heard of modern tools and applications such as Adobe Photoshop, GIMP, etc. But though these applications have undoubtedly made the lives of artists better, the use of AI for art is still at a nascent stage. The current work aims at developing a web-based application called SketchAI which uses Artificial Intelligence to simplify rough sketches or art work and extract the simplified line drawing. Fully Convolution Networks or FCNs are used to automate the task of sketch simplification. Dataset of rough sketches and corresponding line art drawings are used to train the system to extract line art. Different parameters such as number of epochs, loss functions etc. are considered for experimentation along with different subsets and augmentations of the data. Finally, comparisons of different methods are done to integrate deep learning models with a web application.***

Keywords: Line Art Extraction, Sketch Simplification, Sketch Shading, Fcn

I. INTRODUCTION

SketchAI is a web-based tool that aims to use AI to help make the life of artists easier by automating certain tasks. SketchAI uses Deep Learning to automate the task of line art extraction and sketch simplification. Every artist first draws a rough sketch to express his ideas and design rather than focusing on fine details. Using the rough sketch, artists physically trace the jagged draft to create a smooth drawing. But, it's a wearisome and prolonged method. Thus, the system uses a Fully Convolutional Networks to try and automate the task of Line Art Extraction.

Sketch AI system performs the following tasks:

1. Sketch Simplification: Rough raster sketches such as pencil sketches are taken and fully clean drawing can be obtained by using Fully Convolutional Network which is a completely automated process.
2. Comic Colorization: Colorized comic books are often more visually appealing than non-colored ones, and with the advent of digital distribution methods, color printing costs aren't an issue. But actual colorization requires skilled artist. A tool for automatic comic colorization will therefore be highly helpful for comic writers. The system automate a subset of this task, specifically, colorization of individual characters, using a type of neural network known as Generative Adversarial Networks or GAN.
3. Sketch Shading: The task of shading a picture after drawing it is a hard and cumbersome one. The system uses GAN for the task of auto shading of cartoon characters.
4. Landscape Generation: Creating beautiful paintings of landscapes or scenery such as beaches, forests, mountains etc. is something which artists have been doing since time immemorial. The SketchAI tool designed uses GANs to help you automatically generate landscape images of mountains based on doodles.

Revised Manuscript Received on November 15, 2019

Raghav Jadia, Students, Sir M Visvesvaraya Institute of Technology, Bangalore

Siddharth Sampat, Students, Sir M Visvesvaraya Institute of Technology, Bangalore

Ritika Pandey, Students, Sir M Visvesvaraya Institute of Technology, Bangalore

Manohar R, Assistant Professor, Dept of ISE, Sir M Visvesvaraya Institute of Technology, Bangalore

Supriya P, Assistant Professor, Dept of CSE, Nitte Meenakshi Institute of Technology, Bangalore.

manohar_is@sirmvit.edu, supriya.p@nmit.ac.in

II. RELATED WORK

Edge detection is one of the research areas where most researchers are contributing on. Many researchers have proposed Edge detectors such as Canny Edge Detector [5] which uses multi stage algorithm that can detect edges in images, and Laplacian of Gaussians (LoG) which is used to detect edges as well as any noise in the image. But, these detectors rely heavily on the gradients and thus some of the high contrast black and white images create confusions during edge detection. The authors of [4] and [3] have made an attempt to detect boundaries in natural images. However, these detectors are designed to detect the complete structure of objects and hence do not perform well in extracting fine structural lines.

For extraction of structural lines, [1] proposes a method using manga which is based on FCNs with skip connections. Adoption of this technique is limited since the method uses CNN structure and training data is composed of manga image which are mostly black & white images with high screen tones. However, on the other hand most of the rough sketches have very low screen tones. Edgar Simo Serra et.al, [2] have proposed CNN-based sketch simplification. Accordingly, this involves creation of a network structure to train the dataset which is intended to simplify the sketch. The system designed in this paper is built upon FCN network structure. The image size is fixed to 565x565, reducing the model size and padding. However to simplify sketches, completely new dataset is been fabricated and then augmented, which is required for training the network from scratch.

III. CONVOLUTIONAL NEURAL NETWORKS

Convolutional Neural Networks extends Artificial Neural Networks (ANNs) concepts where the weights are shared across the layers.

ANNs helps to derive solution for an unknown complex function. The complex function involved in the proposed system is converting a rough sketch into the corresponding line art representation. The network is made up of several layers of units or neurons. Each layer is an image of multiple channels of size h x w , where h signifies height and w signifies width. Considering C as channel, the image will be a vector in the $R^{C \cdot h \cdot w}$. Input layer will be the first layer and the size corresponds to the size (H × W) of the input grayscale image, i.e., h corresponds to H, w corresponds to W and C = 1. Likewise, the output layer will be the last layer having the same size as the image. Consecutive successive layers are allied intricately with the help of bias mapped operation:

Convadd : $R^{C \cdot h \cdot w} \rightarrow R^{C' \cdot h' \cdot w'}$ (i)

where, C, reflects the number of channels of the layer L, h reflects the height of the layer L, w, reflects the width of the layer L, C', reflects the number of channels of the layer L', h' reflects the height of the layer L' and w' reflects the width of the layer L'

For every channel of C' in layer L', the map can be defined as convolution with kernel of size $C \times k_h \times k^w$ that is followed by adding up of a constant "bias" image.

Consider, $W^{c'}_{c,i,j}$ as component of kernel and b'_c as constant bias of channel C' in the layer L'. Followed by, value $y_{c',u,v}$ of particular pixel on (u, v) in the channel C' of layer L' could be specified by:

$$y_{c',u,v} = b_{c'} + \sum_{i=-k'_h}^{k'_h} \sum_{j=-k'_w}^{k'_w} \sum_{c=1}^{C} W^{c'}_{c,i+k'_h,j+k'_w} \; x_{c,u+i,v+j}$$

(ii)

where, $(x_{c,s,t})$ is multichannel image of the layer L , $k'_h = (k_h - 1)/2$, and $k'_w = (k_w - 1)/2$.

The layers are interconnected with the shared weight W independent of the spatial location. [6] have erudite on kernel and bias by back-propagation. Hence, the kernel and bias together confer to $C \cdot C' \cdot k_h \cdot k_w + C'$ learnable weights of every pair of successive layer. Kernel size and the number of channel in the layer amounts the number of weights.

IV. MODEL

The model architecture used in the system is based on the Fully Convolutional Network (FCN) [2]. The architecture is been modified to take only fixed size input images: 565x565 and with all padding removed. This helps reduce model size and increase performance.

FCNs are a modification of traditional CNNs where instead of a fully connected dense layer, the model is built only from layers such as down-convolution, flat-convolution, pooling and up sampling (also known as up-convolution or transposed convolution) as shown in Figure 1.

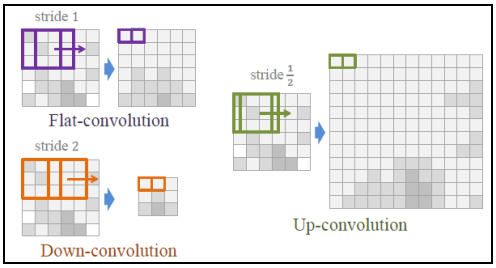

Figure 1: FCN

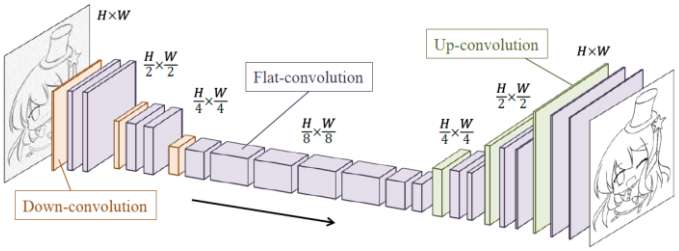

Figure 2: Sketch Simplification Model Architecture

Figure 2 represents the model which considers down convolution, flat convolution and up convolution. Down convolution considers 2 stalk for halving the image size, flat convolution considers 1 stalk for maintaining the size of the image and up convolution with stalk of 1/2 doubling the size of the image.

Convolutional layers with increased strides replace the pooling layers which sink the resolution from previous layer. The first layer of the model acts as encoder that helps in compressing the image dimensionally, the second layer process and extract the necessary outline of the sketch image and the third layer along with last layer act as decoder converting the representation to a grayscale image with equivalent resolution.

After each convolution, a non-linear activation function ReLU is used :

ReLU(x) = max (0, x)
(iii)

In the last layer, the activation function used is Sigmoid to squeeze the output to the range of (0,1):

Sigmoid(x) = 1/(1 + e^ − x),
(iv) .

The weight is learned using back propagation and the loss function used is based on the mean square error criterion:

$$l(Y, Y^*, M) = ||M \odot (Y - Y^*)||^2_{FRO}$$

(v)

V. DATASET

To train the model to extract line art, a dataset is fabricated consisting of 66 anime sketches with their corresponding line art version. Since the dataset considered has very less number if images for training, training data is augmented using tools like Photoshop, Xnconvert etc. and three filters-tone, slur and noise. Random crops of the training data are also designed. The augmented image is as shown in Figure 3.

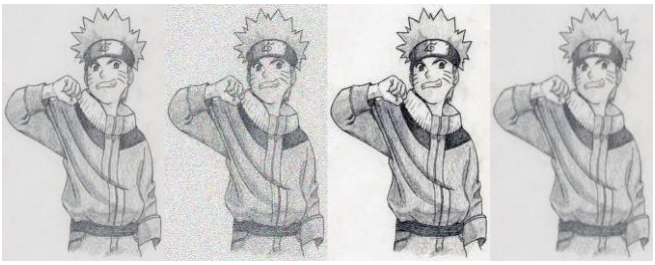

Figure 3: Image Augmentation: (a): original image,
(b):noise, (c): tone, (d): slur

V. WEB APPLICATION

The proposed methodology application is developed as a web based application. The web application is built using HTML, HTML5, JavaScript and JQuery for the frontend and django for the backend. The user can upload an input image or draw on the HTML5 canvas depending on the application. After uploading the image, the user has to press the convert button. This calls the AI model in the backend with the uploaded image as the input to the model. In case the tool uses the HTML canvas and drawing as the mode of input, then the AI model is called whenever the user lifts the cursor from the canvas.

The current work also test the various ways one can integrate deep learning models with the web application. The three methods implemented and compared are discussed below:

A. TensorflowJS

TensorflowJS is a library which allows one to run deep learning models directly using JavaScript. First the model is converted to .h5 format using the tensorflowjs python package. Then the model from JS code is invoked by linking the CDN for tensorflowjs to the web page. During the process, it was observed that as the model size becomes larger, the loading time for the model increases. In fact, tensorflowjs takes the longest time to load compared to other methods of integrating deep learning models with the website.

B. Flask API

In this method, the Python based micro framework is used to build the API which serves the trained Keras model. The client then calls the API by sending a POST request with the input image data in the message body. The API then returns the output image in the response. The advantage of this method is that it is platform independent in terms of the client. For example, the same API can be called using a POST request from JS code as well as Python. So using this method, one can potentially deploy and leverage the power of AI of AI on different platforms such as Android, iOS etc.

The disadvantage of this method is the issue of latency that arises. This means that this method is not suitable for real time use.

C. Integrated model

In this method, the deep learning model is loaded in the Django backend itself. The model is loaded using settings.py which thus allows the models to be loaded before the server starts. The advantage of this method is that during the development stage, one can easily develop and test the models with the frontend. Also, this method works well even when the model size is large. Thus, this the method is chosen for integrating the model with the frontend in the current work.

VI.RESULTS

The system is trained for 450,000 epochs with a batch size of 3 using Paperspace Cloud GPU. The same model is used for inference and testing. It is been observed that the system is able to extract meaningful line art drawings from rough sketch images. An example of line art extraction is as shown in figure 4.

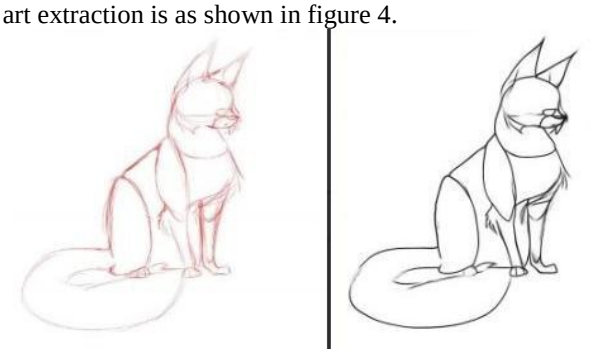

Figure 4: Line art extraction from rough sketches.

The system is also designed to extract line art from old sketches, manuscripts etc. The snapshot of the result is shown in figure 5. The system is tested on pictures of pages from Leonardo Da Vinci's notes. It was found that the performance of the model wasn't as good as on normal rough sketches. This is due to the fact that system was trained for only a few images of anime sketches and the image size was fixed 565x565.

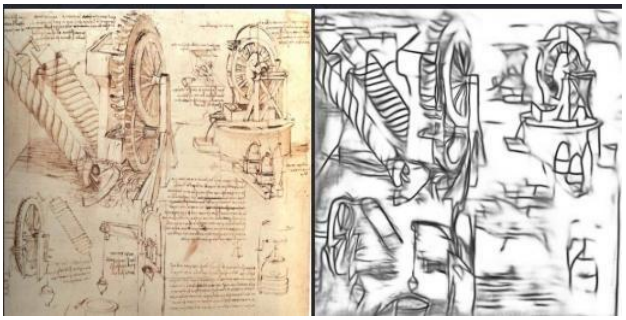

Figure 5: Line art extraction from old sketches.

VII. CONCLUSIONS

The model proposed presents a novel automated system which accepts rough sketches and gives high quality simplified images as output. This model is based on Fully Convolutional Network which optimizes the process of simplification of images. The current results show that approach adopted gives good performance for rough sketches of fixed image size. Despite the achievement, there is still scope for improving the performance by training the model on a larger number of images with more variation.

The model can also be further trained to extract text and art from inscriptions, manuscripts.

REFERENCES

1. Chengze Li, Xueting Liu, Tien-Tsin Wong, 2017. Deep extraction of manga structural lines, ACM Transactions on Graphics, Vol. 36, July 2017.
2. Edgar Simo-Serra, Satoshi Iizuka, Kazuma Sasaki, Hiroshi Ishikawa, 2016. Learning to Simplify: Fully Convolutional Networks for Rough Sketch Cleanup, SIGGRAPH 2016.
3. Iasonas Kokkinos, 2016. Pushing the boundaries of Boundary Detection using Deep Learning, ICLR 2016.
4. Phillip Isola, Daniel Zoran, Dilip Krishnan, Edward H. Adelson, 2014. Crisp Boundary Detection Using Pointwise Mutual Information, European Conference on Computer Vision (ECCV), 2014.
5. John Canny, 1986. A Computational Approach to Edge Detection, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. PAMI- 8, No. 6, November 1986.
6. Rumelhart, David E., Geoffrey E. Hinton, and Ronald J. Williams. "Learning representations by back-propagating errors." Cognitive modeling 5.3 (1988): 1.

AUTHORS PROFILE

Raghav Jadia completed his Bachelor's in Information Science & Engineering from Sir M Visvesvaraya Institute of Technology, Bangalore. Their areas of interest include Machine Learning and Artificial intelligence.

Siddharth Sampath completed his Bachelor's in Information Science & Engineering from Sir M Visvesvaraya Institute of Technology, Bangalore. Their areas of interest include Machine Learning and Artificial intelligence.

Ritika Pandey completed her Bachelor's in Information Science & Engineering from Sir M Visvesvaraya Institute of Technology, Bangalore. Their areas of interest include Machine Learning and Artificial intelligence.

Mr. Manohar R is working as Assistant Professor, Department of Information Science and Engineering, Sir M Visvesvaraya Institute of Technology, Bangalore. His area of interest includes Machine Learning, NLP

Mrs. Supriya P, Assistant Professor, Department of Computer Science & Engineering, Nitte Meenakshi Institute of Technology, Bangalore. Her area of interests includes User Interface Technologies, Mobile Application Development