Sketch-Based Wireframing

Savinay Narendra[1] and Krishna G[2]

Abstract— **This report discusses an approach to improve the sketch based Wireframing system using machine learning. With User interfaces being developing away from the classical WIMP (windows, icons, mouse and pointing) paradigm, existing wireframing tools are more often a hindrance than a benefit to the designers in the initial stages. To allow the designers to have the freedom of sketch, FREE-STYLE was developed that detects shapes using geometric features and map them to corresponding functionality. However, this interactive sketch system lacks a little in recognition. The aim of this report is to discuss an approach to improve the accuracy of the system using Machine Learning, so that it can handle variations in the intended sketches.**

I. Introduction

When professional designers first start thinking about a visual interface, they often sketch rough pictures of the screen layouts. Their initial goal is to work on the overall layout and structure of the components, rather than to refine the detailed look-and-feel. Designers use these sketches and other ”low-fidelity techniques” to quickly consider design ideas, later shifting to interface construction tools or handing off the design to a programmer. Unfortunately, this transition forces the designer to specify too many details.

Much of the design literature recommends drawing rough sketches of design ideas, yet most interface construction tools, and even prototyping tools, require the designer to specify much more of the design than a rough sketch allows. These tools force designers to bridge the gap between how they think about a design and the detailed specification they must create to allow the tool to reflect a specialization of that design.

Another key lesson from the design literature is the value of iterative design. It is important to iterate quickly in the early stages of design because that is when radically different ideas can and should be examined. The need to turn out new designs quickly is hampered by tools that require detailed designs. This over-specification can be tedious and may also lead to a loss of spontaneity. Thus, the designer may be forced to abandon computerized tools until later in the design process or forced to change design techniques in a way that is not conducive to early creative design.

Additionally, research indicates that the use of current interactive tools in the early stages of development places too much focus on design details like color and alignment rather than on the major interface design issues, such as structure and behavior. Wong found that colleagues give more useful feedback when evaluating interfaces with a sketchy look. The designers reported that current user interface construction tools are a hindrance during the early stages of interface design. What designers need are computerized tools that allow them to quickly sketch rough design ideas.

The performance of a developed interactive web-based wireframing tool is aimed to improve in this project. It allows website designers to quickly sketch a website having certain html elements and in response, our tool develops a website containing all the sketches that the user drew converted into the corresponding html element having custom values. Designers can test the interface at any point, not just when they finish the design. When they are satisfied with their early prototypes, they can have FREE-STYLE transform the sketch into an operational interface using real widgets, according to a particular look and feel.

Deep learning models can achieve state-of-the-art accuracy, sometimes exceeding human-level performance. Models are trained by using a large set of labeled data and neural network architectures that contain many layers. They have a state of the art performance on image classification. Convolutional Neural Networks are very similar to ordinary Neural Networks. They are made up of neurons that have learnable weights and biases. Each neuron receives some inputs, performs a dot product and optionally follows it with a non-linearity. The whole network expresses a single differentiable score function: from the raw image pixels on one end to class scores at the other. And they have a loss function (e.g. SVM/Softmax) on the last (fully-connected) layer. ConvNet architectures make the explicit assumption that the inputs are images, which allows us to encode certain properties into the architecture. These then make the forward function more efficient to im- plement and vastly reduce the amount of parameters in the network. CNNs when trained with proper regularization can achieve superior performance on visual object recognition tasks.

We propose to use Convolutional Neural Networks to identify the sketches drawn. Our approach is to convert

*This work was not supported by any organization

[1]Savinay Narendra is with Department of Computer Science and Engineering, Texas A&M University, College Station, Texas 77843 savinay.90@gmail.com

[2]Leela Krishna C. G. is with Department of Computer Science and Engineering, Texas A&M University, College Station, Texas 77843 glk.c.93@tamu.edu

the sketches drawn to images and then feed it into a CNN model to classify it.

II. Related Work

Even though sketching interfaces are not new, with work by [1] being one among the first, the invention of mouse and the ergonomic problems with the light pen relegated pen-based interactions to limited applications.

Our work draws inspiration from [2] SILK (Sketching Interfaces Like Krazy), an informal sketching tool that combines many of the benefits of paper-based sketching with the merits of current electronic tools. With SILK, designers can quickly sketch an interface using an electronic pad and stylus, and SILK recognizes widgets and other interface elements as the designer draws them. Unlike paper-based sketching, however, designers can exercise these elements in their sketchy state. SILK also supports the creation of storyboards, that is, the arrangement of sketches to show how design elements behave such as how a dialog box appears when the user activates a button. Storyboards are important because they give designers a way to show colleagues, customers, or end users early on how an interface will behave. After the designer tests the interface and iterates the design as needed, SILK transforms the rough design to a more finished looking implementation. But, our system resembles a more real world scenario which automatically generates the website in real- time while sketching the HTML elements on the canvas. This provides real- time feedback to the website designers, thereby improving their efficiency and quality of work. It has a drawback of being heavy for web applications and takes long time to give feedback. A fast, simple and compact approach to recognize scribbles drawn with a stylus is presented by [3].

Our current work is an extension of the work done by Narendra et. al [4] which uses geometric features of shapes to recognize the html elements.

A. Problem

A system is developed by recognizing the elements using the geometric properties [4], but it has restrictive rules placed for recognition. To remove the restrictive rules, improve the accuracy of the system and build a more robust system, a Machine Learning based approach is analyzed and discussed.

III. Approach

A. Proposed Solution

The problem of recognizing sketch based designs is formulated as a Image based recognition problem. The sketches drawn by the users is processed and saved as an image, and is recognized using CNNs.

During the data collection phase, users were shown few symbols and were asked to draw. The symbols, as shown in

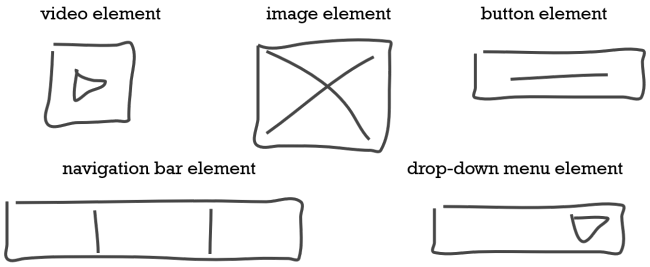

Fig. 1. Images of Sketches to be learned

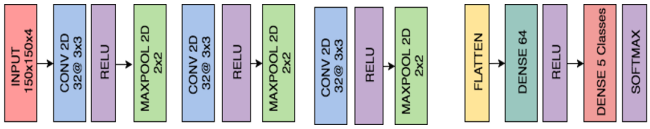

Fig. 2. Architecture of the CNN used

Fig. 1, contains images of sketches for users to be drawn. The sketches are converted to images using inbuilt package functions, resized to have the shape (150x150) and saved to be used during CNN training.

The data collected is divided and fed in to the CNN to be learned and the model parameters are saved. This model is then integrated to the system for recognition.

The images are used to train a CNN with the architecture shown in Fig. 2. The input is fed into 3 Convolution Layers with RELU activation unit and a Maxpooling layer. The resultant is then flattened and is passed through a fully connected layer with RELU activation and dropout rate of 0.5 to prevent over-fitting. It is then finally fed to a fully connected layer of 5 classes using soft-max classifier.

Among the few architectures that were experimented using 5-fold classification, the proposed architecture achieved highest accuracy with 72.06%.

B. Data

We've obtained 143 images in total during the collection phase. This is divided into Training and testing data to be fed into CNN that is programmed using Keras[5]. The data is collected using an online system shown in Fig. 3. Also the images were augmented using data-augmentation techniques in Keras. The collected data is then used to train CNN over 100 epochs and the model is saved.

The saved model is then integrated to our system and currently is being used to display the probability of the user sketch belonging to each of the classes. It can be checked at [6]. We weren't able to do user study.

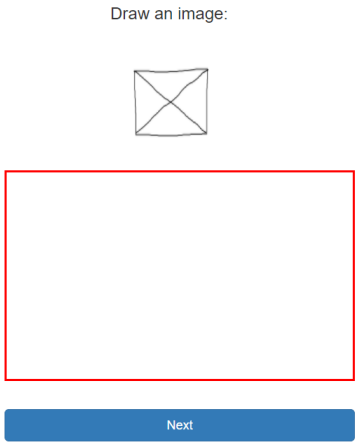

Fig. 3. Screen-shot of system used for Data Collection

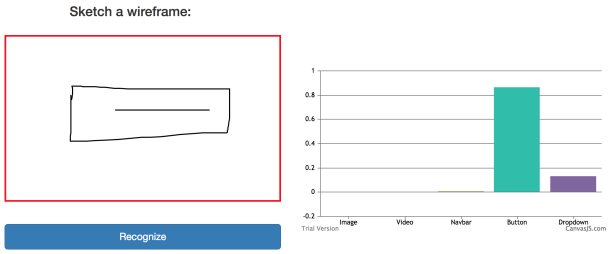

Fig. 4. Positive Case of the System

IV. Results

After integrating the trained model to the system, some users were asked to check the performance of the system. A positive case (an image is recognized as an image) can be seen in Fig.4, where the sketch is classified correctly with high probability. A mediocre case (a button is recognized as a button but also predicts it to be a drop-down with a small probability) is shown in Fig.5 where the sketch is classified into 3 classes with middling probability. A negative case (a drop-down is recognized as navbar and drop-down with almost equal probabilities) is shown in Fig.6 where the sketch is classified incorrectly. The system always classified images correctly while it was confused between images and videos. Also the system was not able to recognize drop-downs with a high confidence confusing it sometimes with buttons or navbars.

A. Discussion

One of the reasons the system is classifying incorrectly is because of less data and less variation in the data. The system was unable to learn the classes from the data provided to it. Also, currently the sketched lines are too thin and the trained model does not find too many variations in the pixel color of each image generated from the sketch. The sketched lines drawn should be thicker to capture so

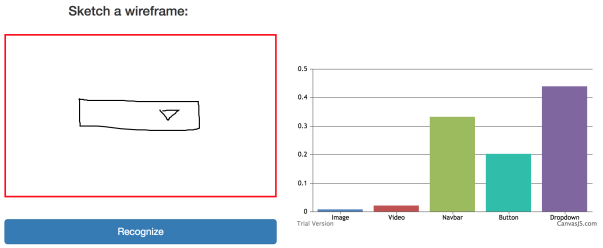

Fig. 5. Mediocre result of the System

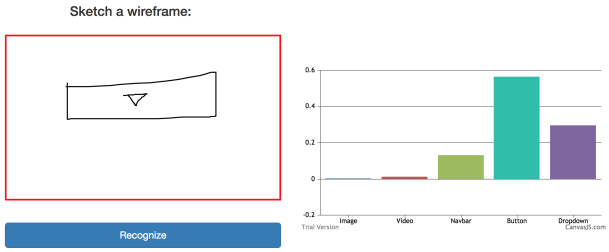

Fig. 6. Negative Case of the System

the model is able to differentiate and recognize different images

B. Future Work

One way to improvise the system's accuracy is to gather more amount of data and use it to train the CNN. Another direction is to integrate more types of sketches and finally use them to convert into html elements. We can develop a system to develop websites right from the sketches. The height and width of each element can be captured while being sketched and a html elements can be made on the webpage using the extreme coordinates of the drawn sketches. We see this work as a tool for developing websites from sketches.

V. Conclusion

This report analyses an approach to improve the accuracy of sketch-based wireframing system using machine learning. A convolution neural network is trained using the images of the sketches and the learned model is integrated to the system. This system enables non-techies to design and build their personal websites effectively. It is examined that the CNN was unable to learn properly with the amount of data provided. Next task is to gather more data, with variation and create a more accurate system.

References

[1] I. E. Sutherland, "Sketchpad a man-machine graphical communication system," *Transactions of the Society for Computer Simulation*, vol. 2, no. 5, pp. R–3, 1964.

[2] J. A. Landay, "Silk: sketching interfaces like krazy," in *Conference companion on Human factors in computing systems*. ACM, 1996, pp. 398–399.
[3] M. J. Fonseca, C. Pimentel, and J. A. Jorge, "Cali: An online scribble recognizer for calligraphic interfaces," in *AAAI spring symposium on sketch understanding*, 2002, pp. 51–58.
[4] S. Narendra, S. Dey, J. Coad, S. Polsley, and T. Hammond, "Freestyle: A sketch-based wireframing tool."
[5] F. Chollet *et al.*, "Keras," 2015.
[6] K. Narendra, http://a.co/5JQ8nRf, 2017.