COMP 537 Intelligent User Interfaces Term Project
Signature Generator

Volkan Cirik
Koc University

Artificial Intelligence Lab
vcirik@ku.edu.tr

Abstract

This document briefly explains the work done for term project of COMP 537 Intelligent User Interface. The Signature Generator is a system that learns the characteristics of one's signature. The system is able to discriminate the several signatures and also mimics the users signature. It can be used as a digital signature for electronic documents or online banking systems. All the work done in this project can be found on Github[1]

1 Introduction

After the 2000s, the digital world is essential part of our lives. The most of the communications is done through digital mediums. Authentication is a part of processes of these mediums. It has even become a legal requirement (The Information Technology Act, 2000). There are several solutions to solve authentication problem in online transactions. These solutions include smart devices(either as a handset or as a software) producing unique keys. Here we suggest to use users own signature to generate signatures that can be used for both signing documents and as a unique key for any kinds of transactions.

The following sections of this document are as follows, ...

2 Signature Generator

Signature Generator make use of the machine learning techniques to learn the characteristics of one's signature. The following sections explain the general flow of the system and the mathematical foundation of the learning system.

2.1 Pipeline

Figure 1, shows the general flow of the algorithm.

In step (a), the signatures are collected from the user. The user provides 16 copies of his/her signature and provides a PNG image file. In step (b), the image file is binarized and reduced to 600x400 size. In step (c) individual signature images of size 120x100 are extracted. In step (d), each signature image is serialized. If a pixel value is greater than some treshold it will be assigned as on or of as a binary input. After this step we have 16 binary vectors of size 12000 as input for learning system.

In step (e), the learning step is done. The following subsection explains the learning step. After learning step, in step (f) either discrimination is done; to decide whether the given image is an instance of the user's signature or not or an instance of the user's signature is generated.

2.2 Deep Belief Networks with Restricted Boltzmann Machines

Restricted Boltzmann Machines(RBMs) are are probabilistic generative models that can be applied to both labeled and unlabeled data. Application domains of RBMs include image processing, speech recognition.

RBMs can be used to learn the parameters of an unknown probability distribution. An RBM is a bipartite graph with visible and hidden units connected with weight edges. In general both visible and hidden units take binary values. Visible units correspond to an observation and hidden units represent the dependencies between the components of the observation. Since the marginal distribution represented with the RBM is not tractable, Monte Carlo computation methods, namely Gibbs sampling, is used.

Deep learning, or Deep Belief Network(DBN), is the collection of algorithms in machine learning that try to learn layered models of inputs, commonly neural networks. The layers in models correspond to distinct levels of concepts,or features, where higher-level concepts are defined

[1] github.com/wolet/signature-generation

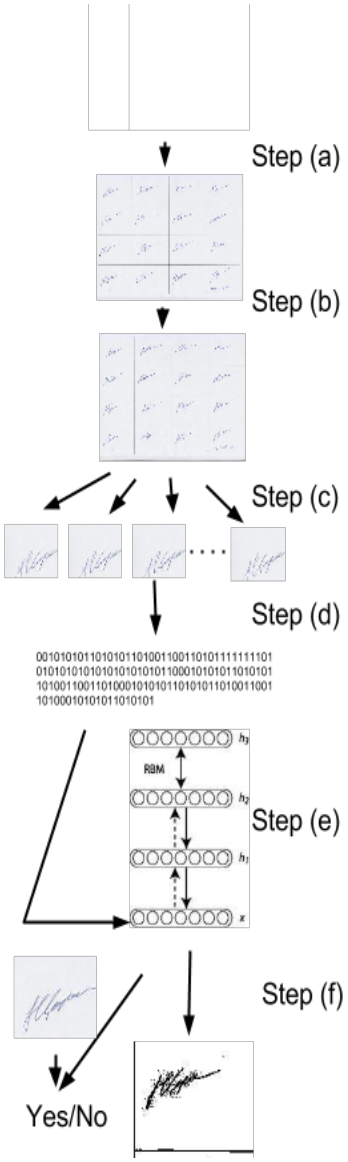

Figure 1: The flow of the system

and learned from lower-levels concepts (Y Bengio, 2009).

In the implementation step, as a bootstrapping step (Edwin Chen,, 2011) is forked on Github, a clean and pure implementation of RBMs. For the implementation of DBNs, (Bengio,Yoshua et al., 2007) and (G Hinton, 2010) is heavily used.

The first layer of the DBN is the visible layer, the input of this layer is provided by the step (d). The second layer consists of 1000 hidden units. The next one is 500 hidden units. The third one is 50 and the fourth one is 10 units. The final unit can be either one or equals to the number of people in the system. Since the learning takes quite amount of time, for each user single DBN is trained, thus, for each user the final layer consists of 1 unit. The DBN is trained for 2000 iterations. The number of layers, units in each layer and number of iterations for training is chosen after several attempts. Further work should be done to find the optimum settings for layers and training.

3 Results

Figure 2, shows that, the system is able to learn to generate the signature similar to users.

4 Conclusion and Future Direction

In the very beginning of this project, the idea was to learn the handwriting of the users. Since the segmentation step of each character is whole another research topic, it takes quite amount of time. Thus, the project had to be narrowed down. How-

Figure 2: On the left an example of original signature, on the right generated version of it

ever, an amount of work shows that the system is able to learn the individual characters if each character is seperately provided to the system.

In previous sections, we mentioned that the system is able to decide whether the given image is an instance of the user's signature or not. Though system is capable of accomplishing this task, we could not provide such healthy results because of tight schedule.

Acknowledgments

I would like to thank Mehmet Ali Yatbaz, Taylan Cemgil, Metin Sezgin, my classmates and friends for comments on the topic. Unfortunately, this work is far away from what anyone expected due to some incidents.

References

Bengio, Y. Learning deep architectures for AI. *Foundations and Trends in Machine Learning*. 2009.

Hinton, G. Hinton, A practical guide to training restricted Boltzmann machines. 2010. *Momentum*

Bengio, Y., Lamblin, P., Popovici, D., Larochelle, H. Greedy layer-wise training of deep networks. , 19, 153. 2007. *Advances in neural information processing systems*

The Information Technology Act,2000 *http://www.dot.gov.in/Acts/itbill2000.pdf*.

Edwin Chen, Restircted Boltzmann Machines *https://github.com/echen/restricted-boltzmann-machines*.