Penn Project Report

Machine Learning in Real Time Character Recognition

**Abstract**

This paper presents a MEMS based digital pen for handwriting recognition. While there are several devices which are capable of digitizing handwritten notes, all commercially available devices require secondary hardware to function. We propose a single device solution for digitizing text as it is written on a page. By calculating the dynamics of the pen from IMU measurements, we are able to estimate the pen tip accelerations and rotations as a function of time. We evaluated SVCs, decision trees, Ada-Boost, and logistic regression learning algorithms based on their ability to recognize handwritten characters and achieved 93.2 % accuracy using an SVC with a lineal kernel.

**1 Pen**

**1.1 Hardware**

% Mechanical design (copied from the previous report)

The pen layout and body was designed in CAD (pictured below) to ensure appropriate packing of components. The clear body was 3D printed in ABS in the University's prototyping labs.

% Electronics (copied from the previous report)

An ATMega32u4 microcontroller provides on-board processing and interfaces to the wireless module, IMU, switch, and indicator LEDs. Two Nordic nRF24LE1 transceivers handle wireless communication between the pen and base station computer. Inertial measurements are provided by an MPU6050 digital IMU with configurable gain triaxial accelerometers and gyroscopes. A Honeywell HMC5883L triaxial magnetometer provides a reference to earth's magnetic field for orientation. A switch behind the pen tip detects contact with the page for character segmentation. Three LEDs on the case display pen state. The pen also includes on-board batteries, power management, and an on-off switch.

**1.2 Filtering**

% Filtering

The raw data from the IMU pose issues for learning algorithms. Sources of variance can include posture, handed-ness, roll angle of the pen, and sensor drift. To minimize sources of error, we compute the trajectory of the pen tip on the page. Using various filtering and sensor fusion techniques we are able to compensate for gravity and perform attitude and position estimation. We use a Kalman Filter to estimate the orientation of the pen relative to gravity so that we can subtract the gravity components from the acceleration measurements. % Additional details here about: low & high passes, attitude estimation, coordinate systems?

% Coordinates & transforms ?

% Output signals

**2 Learning**

**2.1 Intro to Learning**

After all the data filtering and pre-processing we finally got some raw data (the one we are actually going to use to learn), and some images to see how well we were doing visually to recover a certain letter.

**2.1 Features**

% Feature vector

After a little processing of the data (Ask MIKE), we get the features we identified to be important for character recognition, which we put into a feature vector as: timestamp, x, y, z, v\_x, v\_y, v\_z, a\_x, a\_y, a\_z, roll, pitch, yaw, theta, phi, psi.

% Feature standardization

We then standardized these features to be able to compare different reading in different conditions with (equations for standardization)

**2.2 Learning Algorithms**

Once we had all the data processing part defined, we created a dataset to train and test different machine learning algorithms.

We created a data set that had repeated reading for all the letter in the alphabet and numbers from 0 to 9, with exception of: f, i, j, k, t, x.

% SVM with Kernel

We first tested Support Vector Machines with different Kernels to compare the results we were getting from them.

(Table from Excel Results)

As we can clearly see, while the lineal and polynomial kernel do a pretty good job to classify the characters, the rbf is VERY prune to subtle variations in the parameters, which makes it a non robust method for character recognition. On the other side, we can see that the accuracy obtained

**3 Performance**

**3.1 Tested Algorithms**

**3.2 Final performance**

% Accuracy

% Precision & recall, ROC, etc.

% Confusion matrix

**Acknowledgements**

**References**

**MIKE**

Rbf It doesn’t work well because the nature of the time series does not blend itself well.

ADA BOOST while it has the smoothing, the ideal decistion function is much smoother complex than its capable with axis parallel hyper rectangles