CSCE 624 Homework I

Sketch Recognition - Arrow Detector

Aswin Jacob Thomas

UIN: 629009211

Septemeber 17, 2019

Contents

Chapter 1

Introduction

A gesture is referred to as the hand markings entered by a stylus or mouse[1]. Gestures issued with a pen is a desirable feature as gestures are more iconic and intuitive compared to commands entered through a keyboard. Since the command and the operand can be specified in a single stroke, the learning curve for the user is also fast[2]. [1] found an efficient way of gesture recognition using a toolkit that he developed, GRANDMA. A gesture starts when the user first presses the mouse and it ends when the user either release the mouse or stop moving the mouse for a given amount of time. Gestures are historically captured in a single stroke as it avoid the segmentation problem and allows longer timeouts. Poly-line gestures can also be recognized by first finding the primitives and then recombining them using geometrical properties to find the original gesture. A stroke consists of a series of x, y and time values as the pen crosses the screen.

The initial point of a n point stroke is taken as (x_0,y_0,t_0) and the final point is taken as (x_{n-1},y_{n-1},t_{n-1}). Each point in the stroke is referenced as

$$y_i = (x_i, y_i, t_i) \quad 0 <= i < n$$

A stroke can be visualized as a collection of vectors concatenated together, though it only gives the spacial information and not time information. Thus the length of the stroke can be found by adding the individual lengths of such vectors.

1.1 Problem Description

Given 1000 strokes, 500 strokes corresponding to arrow and remaining 500 of non-arrows, create a recognizer which classifies a given unknown stroke to arrow or non-arrow.

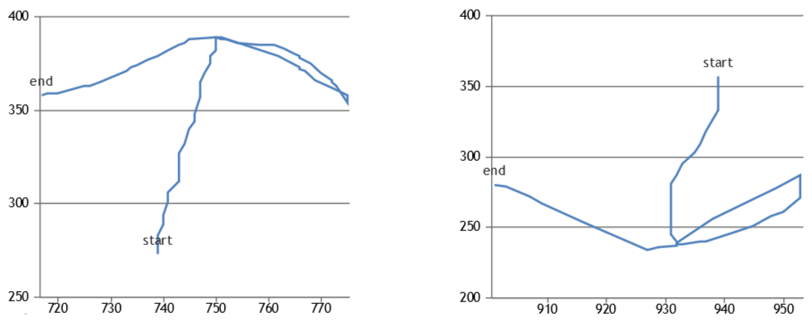

Figure 1.1: Arrow Gestures

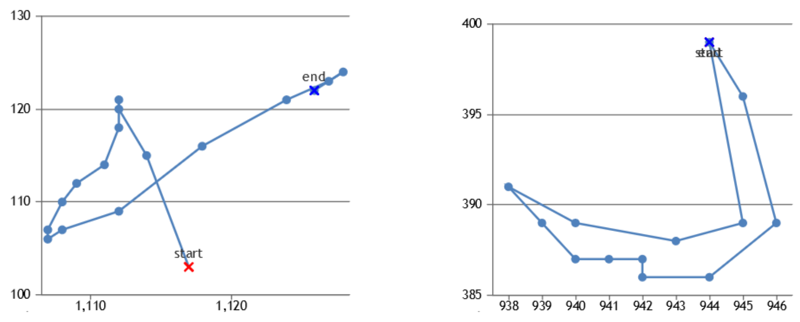

Figure 1.2: Non-Arrow Gestures

Chapter 2

Feature Explanation

Summary

A total of 30 features are used for gesture recognition. The first 13 features are Rubine[1] features, next 11 features are taken from Long[3].

2.1 Rubine's Features

1. Rubine Feature 1 - cosine of the starting angle

$$\cos(\alpha) = \frac{x_2 - x_0}{\sqrt{(x_2 - x_0)^2 + (y_2 - y_0)^2}}$$

   This feature is important as it gives information about the starting angle. The cosine of the angle is taken rather than taking the angle itself because angle is a discontinuous feature. Both cosine and sine curves are continuous, thus ensuring **similar shapes have similar values for the features**.

2. Rubine Feature 2 - sine of the starting angle

$$\sin(\alpha) = \frac{y_2 - y_0}{\sqrt{(x_2 - x_0)^2 + (y_2 - y_0)^2}}$$

Single cosine feature can correspond to 2 different angle. So both cosine and since of the angle are taken. **Signs of the combined angles is unique for each quadrant**.

The angle is taken between the first and the third point of the stroke as usually first and second point will be at the same location due to high sampling rate

3. Rubine Feature 3 - Length of the bounding box diagonal

$$f_3 = \sqrt{(x_{max} - x_{min})^2 + (y_{max} - y_{min})^2}$$

Bounding box is the smallest rectangle which surrounds the stroke. Features involve bounding box is crucial since it helps to identify the size of the stroke.

The length of the bounding box diagonal is chosen rather than the area of the bounding box because:

(a) Area does not scale well as diagonal.

(b) Similar shapes should have similar values. A horizontal line have an area close to 0. The same line, when rotated by an angle has a very large area. But, both the lines have the same bounding box diagonal lengths.

4. Rubine Feature 4 - Angle at the bounding box

$$f_4 = \arctan\frac{y_{max} - y_{min}}{x_{max} - x_{min}}$$

This feature directly uses the angles rather than sine and cosine values as the values ranges between 0 and $\pi/2$ and no two similar shapes have radically different f_4 value.

5. Rubine Feature 5 - Distance between end points of the stroke

$$f_5 = \sqrt{(x_{n-1} - x_0)^2 + (y_{n-1} - y_0)^2}$$

6. Rubine Feature 6 - cosine of the angle between end points

$$cos(\beta) = \arctan \frac{x_{n-1} - x_0}{f_5}$$

7. Rubine Feature 7 - sine of the angle between endpoints

$$sin(\beta) = \arctan \frac{y_{n-1} - y_0}{f_5}$$

8. Rubine Feature 8 - Stroke length

   Considering stroke as a concatenation of vectors, stroke length can be found by adding the vector lengths together. Lets define $\Delta x_i = x_i$-x_{i-1} and $\Delta y_i = y_i$-y_{i-1}

$$f_8 = \textstyle\sum_0^{n-2} \sqrt{(\Delta x_i^2 + \Delta x_i^2)}$$

An important recognizer is obtained by dividing f_5 by f_8. A stroke can be categorized as a line or not by using this feature.

Line Recognizer - $\frac{f_5}{f_8} > 0.95$

9. Rubine Feature 9 - Total curvature Lets define θ_p as

$$\theta_p = \frac{\Delta x_i \Delta y_{i-1} - \Delta x_{i-1} \Delta y_i}{\Delta x_i \Delta x_{i-1} - \Delta y_{i-1} \Delta y_i}$$

   Rubine's 9^{th} feature is the total summation of the θ_p

$$f_9 = \textstyle\sum_1^{n-2} \theta_p$$

10. Rubine Feature 10 - Total absolute curvature

$$f_{10} = \sum_{1}^{n-2} |\theta_p|$$

11. Rubine Feature 11 - Total squared curvature

$$f_{11} = \sum_{1}^{n-2} {\theta_p}^2$$

A line and a sine wave has the same value for f_9 but different value for f_{10}. An arc and V have same values for f_9 and $f10$ but they can be distinguished using f_{11}. f_{11} basically measures the sharpness of the stroke.

12. Rubine Feature 12 - Maximum speed

$$f_{12} = \max \frac{{\Delta x_i}^2 + {\Delta y_i}^2}{{\Delta t_i}^2}$$

Basic assumption is that long continuous strokes reach higher speeds.

13. Rubine feature 13 - Total duration of the stroke

$$f_{13} = t_{n-1} - t_0$$

Strokes may take more time if

(a) stroke is longer

(b) stroke is complicated

(c) stroke is less familiar

2.2 Long's Features

Long[3] used 22 features for gesture recognition. He used first 11 Rubine features other than speed and total duration. He wanted to remove the reliance on time and make the gesture recognition user independent.

1. Long Feature - Aspect ratio

$$l_1 = \left|\frac{\pi}{4} - f_4\right|$$

   This feature makes tall skinny features and long flat one mathematically similar.

2. Long Feature 2 - Curviness

$$l_2 = \textstyle\sum_1^{n-2} |\theta_i| \text{ such that } \theta_i < 19^o$$

   This only adds the absolute values of the angles which are not potential corners.

3. Long Feature 3 - Relative rotation

$$l_3 = \frac{\text{Total angle traversed}}{\text{Stroke Length}}$$

4. Long Feature 4 - Density metric 1

$$l_4 = \frac{\text{Total stroke length}}{\text{End Point length}} = \frac{f_8}{f_5}$$

5. Long Feature 5 - Density metric 2

$$l_5 = \frac{\text{Total stroke length}}{\text{Bounding Box diagonal length}} = \frac{f_8}{f_3}$$

6. Long Feature 6 - Non Subjective Openness

$$l_6 = \frac{\text{End Point Length}}{\text{Bounding Box diagonal length}} = \frac{f_5}{f_3}$$

f_3 is a subjective measure since it measures the distance between the start and the end point.

7. Long Feature 7 - Area of the bounding box Most intuitive feature but can result in mis-recognition

8. Long Feature 8 - Log of the area of bounding box

$$l_8 = \log l_7$$

9. Long Feature 9 - Rotational Change to Motion

$$l_9 = \frac{f_9}{f_{10}}$$

10. Long Feature 10 - Log of Stroke Length

$$l_{10} = \log f_8$$

11. Long Feature 11 - Log of Aspect Ratio

$$l_{10} = \log l_1$$

A linear classifier will try to combine various features by summation of features multiplied with coefficients. But linear classifier cannot perform logarithms and division of various features. Thus by projecting the features into a new feature space, Long tried to obtain better recognition rates.

2.3 Identified New Features

Apart Rubine's and Long's features, 7 new features were recognized for classification.

1. Normalized Distance Between Direction Extremes[4]

$$NDDE = \frac{\text{Stroke Length between Direction Extremes}}{\text{Total Stroke Length}}$$

2. Direction Change Ratio[4]

$$DCR = \frac{\text{Maximum Change in Direction}}{\text{Average Change in Direction}}$$

   The first and the last 5% of the points may be removed while calculating this feature as it can be attributed to the noise.

3. Log of the bounding box diagonal

$$\text{Log of bounding box diagonal=log } f_3$$

4. Circle variance[5]

$$CircleVariance = \frac{\sigma_R}{\mu_R}$$

   where μ_R and σ_R are the mean and standard deviation of the radial distance from centroid to the points.

5. Number of Corners
   Used ShortStraw algorithm[6] in finding the corners.

6. Compactness

$$Compactness = \frac{\text{Area of the bounding box}}{\text{Perimeter of bounding box}^2}$$

Chapter 3

Feature Motivation

1. **Normalized Distance Between Direction Extremes**

   (a) Identify the point with maximum direction change

   (b) Identify the point with minimum direction change

   (c) NDDE calculated as ratio of the distance between the above points to the total stroke length

   where direction change is defined as the change of y value over change of x value.

   NDDE essentially gives the percentage of stroke that occurs between the two direction extremes[4]. This feature can be used to differentiate between acrs and poly-lines. From 1.1, it is clear that an arrow can be considered as a poly-line stroke having a head and a shaft. For arcs, the maximum and minimum direction changes occur at the endpoints. But in case of poly-lines, there will be one or more spikes in the graph thus yielding a lower value for NDDE.

2. **Direction Change Ratio**

   This feature is defined as the ratio of maximum direction change to average direction change.

For poly-lines such as arrows, there will be sharp change between direction, but for lines and arcs, there will be little change between direction values. Thus DCR value will be more for poly-lines and less for lines and arcs.

The motivation behind the features 1 and 2 (NDDE and DCR) are taken from [4] which is used for differentiating various gestures like poly-lines, circles and ellipses.

3. **Log of the bounding box diagonal**

   Long used logarithms of area and stroke length as the numerical value grows quickly than perceptual value. Similar argument is used for taking the logarithm of the bounding box diagonal. Diagonal length difference becomes significant between small gestures, but is negligible between larger gestures. Thus taking the log value reduces this gap.

4. **Circle Variance**

   Distance d_i defined as the distance between a stroke point to the centroid point. Mean μ_R defined as the mean of d_i and σ_R defined as the standard deviation.

$$\mu_R = \frac{1}{N}\sum_1^{N-1} d_i \text{ and } \sigma_R = \sqrt{\frac{1}{N}\sum_1^{N-1}(d_i - \mu_R)^2}$$

   Circle variance is then calculated by dividing σ_R by μ_R.

$$CircleVariance = \frac{\sigma_R}{\mu_R}$$

Circle variance represents how a shape is similar to a circle. This feature are taken from [5]. These can help to differentiate between line gestures and arc gestures, thus good for arrow detection. It represents the spread of the points along the contour of the shape.

5. **Number of corners**

   This feature helps in identifying truss shaped figures, lines and geometrical figures from arrows. An arrow with a head and a shaft can have a maximum of 4 corners and minimum of 3 corners (since a single stroke is used for sketching). ShortStraw[6] is used for corner detection.

6. **Compactness**

   Compactness otherwise called circularity ratio is calculated as the ratio of the area of the bounding box to the square of the perimeter of the bounding box. A perfect circles will have a constant compactness. This feature distinguishes arc gestures from line gestures. It is another shape parameter apart circle variance, which compares the area and the perimeter of the shape and looks out for symmetrical figures.

Chapter 4

Weka Classifiers

Summary

First the whole stroke data is pre processed to remove points having the same time values and same spacial values. Pre processing is important as otherwise, various feature values can result to division by 0 exception or irrelevant values like infinity.

4.1 Preprocessing

4.1.1 Removing same spacial points

Points having same spacial values but different time values can occur due to sampling error or if the mouse was not moved for a certain time within a threshold. These points can be removed as they don't attribute to recognition rate. This is critical as these same points can lead to division by 0 error. The second point is removed if two points have the same spacial values values as the first point is the "true" stroke point. To eliminate the jiggle, points which are a threshold apart are also removed from the stroke[1]. Maximum instances are classified correctly when this threshold is selected as 1 pixel difference.

4.1.2 Removing same time points

Due to higher sampling rate of modern devices, points with same time values, but different spacial values may be captured. Empirically, removing the first point in such case can increase the number of correctly classified instances. A special case occurs when all the time values of the data points are the same. This occurs due to sampling issue of the pen. Instead of removing the whole stroke, a constant can be added incrementally to the time values of the points to make the points continuous in time.

4.2 Classifiers used for recognition

Each of the classifier is trained and tested with 2 different attribute sets:

1. Training using all the feature sets
2. Training using the feature set obtained after subset selection

4.2.1 Linear Regression

Linear regression tries to fit a line as the boundary between arrow points and other points. In WEKA, linear regression classifier requires the class label to be numeric. So a filter, NominalToBinary, is applied to the label class to convert the nominal value to binary value.

	All Attributes*	**Subset Selection***
Correlation	0.8743	0.852
MAE*	0.1937	0.2107
RMSE*	0.2428	0.2618
RAE*	38.6876	42.08
RRSE*	48.4941	52.27

MAE* - Mean Absolute Error

RMSE* - Root Mean Sqaured Error

RAE* - Relative Absolute Error

RRSE* - Root Relative Sqaured Error

All Attributes*: Metrics obtained by training and testing with all the attributes
Subset selection*: Metrics obtained by using the attributes obtained with subset selection

4.2.2 Logistic Regression

	All Attributes*	**Subset Selection***
Accuracy	0.974	0.982
Precision	0.974	0.982
Recall	0.974	0.982
FP Rate	0.026	0.018
F-measure	0.974	0.982

4.2.3 Multilayer Perceptron

	All Attributes*	**Subset Selection***
Accuracy	0.99	0.992
Precision	0.99	0.992
Recall	0.99	0.992
FP Rate	0.010	0.008
F-measure	0.99	0.992

4.2.4 J48

	All Attributes*	**Subset Selection***
Accuracy	0.989	0.994
Precision	0.989	0.994
Recall	0.989	0.994
FP Rate	0.011	0.006
F-measure	0.989	0.994

4.2.5 Random Tree

	All Attributes*	**Subset Selection***
Accuracy	0.984	0.99
Precision	0.984	0.99
Recall	0.984	0.99
FP Rate	0.016	0.01
F-measure	0.984	0.99

4.2.6 Random Forest

	All Attributes*	Subset Selection*
Accuracy	0.995	0.996
Precision	0.995	0.996
Recall	0.995	0.996
FP Rate	0.005	0.004
F-measure	0.995	0.996

4.2.7 Decision Table

	All Attributes*	Subset Selection*
Accuracy	0.984	0.989
Precision	0.984	0.989
Recall	0.984	0.989
FP Rate	0.016	0.011
F-measure	0.984	0.989

4.3 Result

Random Forest performed the best for this dataset when all attributes are taken as well as when a subset of the attribute is taken with a recognition rate of 99.5% accuracy and 99.6% accuracy respectively. Random Forest performs well for most of the classification problems. It uses an ensemble of various decision trees created out of the given instances and predicts the output class with the largest number of votes from the decision trees. It is also not prone to over fitting as the individual decision trees are least correlated with each other as they are trained with re-sampled chunck of instances each time with different combinations of features.

The recognition rate increased slightly when a subset of attributes is taken. This is because we are feeding only those attributes that are very relevant to the dataset. These attributes were obtained by performing correlation with the class and finding the individual information gain of the attribute.

Chapter 5

WEKA Subset Selection

WEKA has inbuilt support for subset selection. Algorithms used for attribute evaluation are InfoGainAttributeEval and CorrelationAttributeEval. The first one evaluates the attributes on the basis of entropy (information gain) and the second one ranks the algorithm on the basis of its correlation with the class. Search method used is Ranker algorithm. Ranker algorithm along with InfoGainAttributeEval ranks the attributes in order of their decreasing information gain. More the information contributed by the attribute, more is the information gain. Ranker algorithm along with CorrelationAttributeEval arranges the attributes in the decreasing order of their correlation with the class.

A total of 8 attributes are selected as the best fit attribute subset for recognition. Attributes are chosen empirically after running several machine learning algorithms on the top of the selected attributes. The selected attributes include

1. **Bounding box angle**
2. **Density Metric 1**
3. **Density Metric 2**
4. **Non-Subjective Openness**

5. **Area of the bounding box**
6. **Log of area of the bounding box**
7. **Log of bounding box diagonal**
8. **Circle Variance**

Chapter 6

Conclusion

Circle Variance, Area of the bounding box and the log of the area has the highest correlation with the class attribute. Circle Variance is a shape parameter, which compares the sketch to a perfect circle. In other words, it measures the distribution of the points in the shape. Perfect circle has 0 variance in terms of radial distance to the centroid, in which case is the center of the circle. This variance is much larger for arrow gestures, and less for non-arrow gestures. This can be due to fact that most of the non arrow gestures include trusses and rectangles which have less variance in terms of radial distance to the centre point than an arrow gesture.

Area of the bounding box will also be larger for non-arrow sketches like trusses compared to arrow sketches.

Density metrics 1 and 2 are the important features returned by information gain. Arrow gestures have low values for these metrics compared to non arrow gestures. This can be due to the fact that an arrow consists of a head and a shaft and the ratio of the stroke length to the end point length will be close to 1. But in case of trusses, the stroke length will be much larger compared to the end point length. Thus these attributes provide the maximum information in recognition.

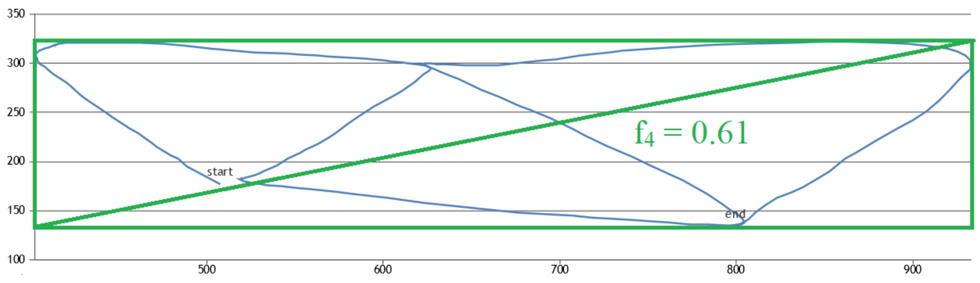

Figure 6.1: Non-Arrow Gesture - Truss

Another attribute that varies between arrows and non-arrows is the angle formed at the bounding box diagonal. For non-arrow gestures like trusses, the angle formed is lesser in comparison to the arrow gestures.

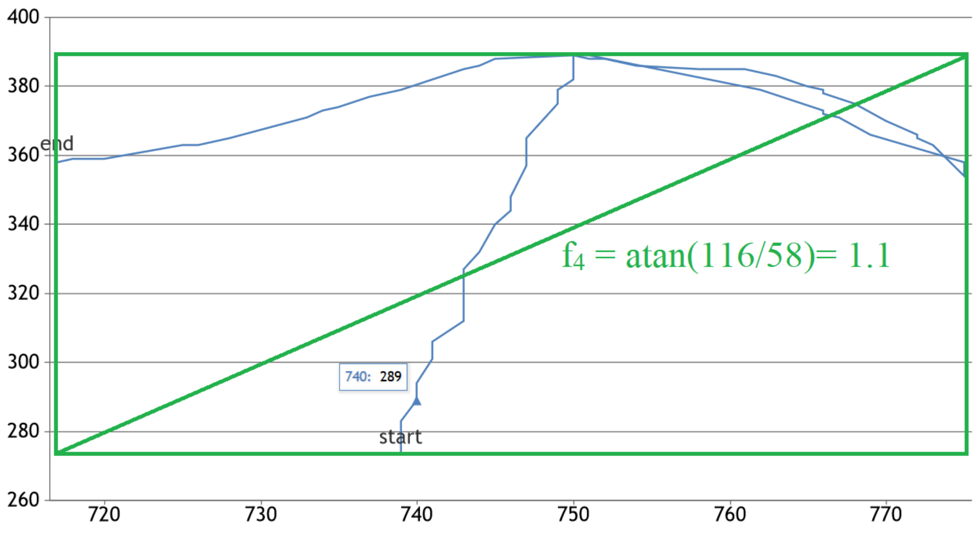

Figure 6.2: Arrow Gesture

References

[1] D. Rubine, "Specifying gestures by example," in *Proceedings of the 18th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '91, (New York, NY, USA), pp. 329–337, ACM, 1991. 1, 3, 13

[2] W. Buxton, "Human-computer interaction," ch. There's More to Interaction Than Meets the Eye: Some Issues in Manual, pp. 366–375, San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1987. 1

[3] A. C. Long, Jr., J. A. Landay, L. A. Rowe, and J. Michiels, "Visual similarity of pen gestures," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '00, (New York, NY, USA), pp. 360–367, ACM, 2000. 3, 7

[4] B. Paulson and T. Hammond, "Paleosketch: Accurate primitive sketch recognition and beautification," in *Proceedings of the 13th International Conference on Intelligent User Interfaces*, IUI '08, (New York, NY, USA), pp. 1–10, ACM, 2008. 9, 10, 11

[5] M. Yang, K. Kpalma, and J. Ronsin, "A Survey of Shape Feature Extraction Techniques," in *Pattern Recognition* (P.-Y. Yin, ed.), pp. 43–90, IN-TECH, Nov. 2008. 38 pages. 9, 11

[6] A. Wolin, B. Eoff, and T. Hammond, "Shortstraw: A simple and effective corner finder for polylines.," in *SBM*, pp. 33–40, 2008. 9, 12