Using Sketch Features to Recognize Arrows

Aniket Sanjiv Bonde
Department of Computer Science & Engineering
Texas A&M University
College Station, Texas 77843
bondeanikets@tamu.edu

Tracy Hammond
Department of Computer Science & Engineering
Texas A&M University
College Station, Texas 77843
hammond@cse.tamu.edu

***Abstract*— This project was developed as partial fulfillment of the requirements for the course CSCE 624, Sketch Recognition taught by Dr. Tracy Hammond during the Fall 2017 at Texas A&M University. This project uses sketch features to recognize arrows as opposed to other shapes. The main motto of this project is feature engineering for better recognition of arrows versus other shapes. Achieved a whopping 99.56% accuracy using 10-fold cross-validation and Random-Forest classifier on WEKA.**

I. Problem Statement

This project involves coding a variety of features to build an arrow recognizer, provided with a training data set to help build the best recognizer possible.

This project uses sketch features to recognize arrows as opposed to other shapes. The main motto of this project is feature engineering for better recognition of arrows versus other shapes.

Initial part of project consisted of importing the given data to the environment. Followed with this, engineering new and more efficient features for better recognition. Final step was to extract the results to a csv which can be loaded into WEKA software for application of various machine learning models for the actual recognition. ZeroR, J48, Random Forest, and one other classifier of our choice was permitted.

II. Pre-processing

As the data is real data, there persists a lot of discrepancies. So, it's wise choice to first pre-process the data using the resampling method.

The importance of this is that when the initial sampling was done on the real data, the time intervals are kept constant but as seen in figure [1], we see that the points lie on un-equal distances from each other, which is bad. Consider this situation:

- There might be some points overlapping since sketcher stayed at a point for some seconds without moving his pen. We don't want this to bias the model.
- What if the sketch was drawn too fast and there were too less points after the first sampling, so the resampling would interpolate and thus more data can be exploited

Hence in figure [2], a resampling method is deployed to overcome the situation described above. The red points indicate the sampled points and the blue line is the actual sketch drawn by the sketcher. The interpolation scheme is linear.

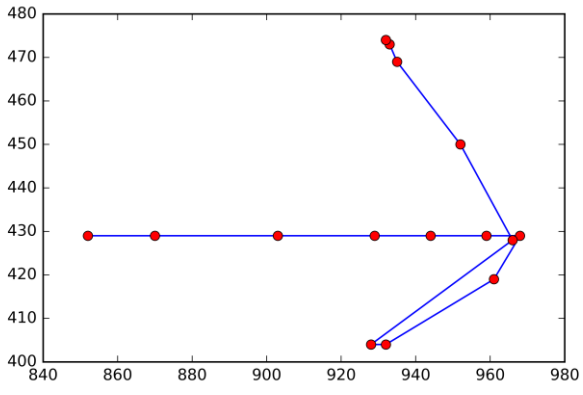

FIGURE. I

The resampling distance is taken to be the diagonal length of bounding box divided by a constant factor of 40. This is an empirically selected value, as a constant value for all the sketches would disregard the size of the sketch. Hence, a dynamic value which depends on the actual size of the sketch is selected for resampling.

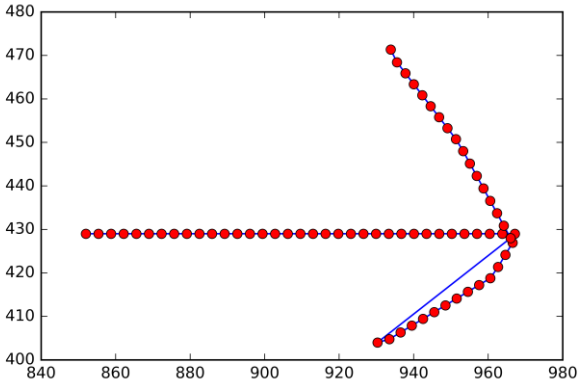

FIGURE. II

Another resampling method that I tried is just drop one of the overlapping points from the sketch to make it more realistic, but this doesn't prove to be too helpful as the adjacent points are not equidistant from each other.

The first resampling method mentioned in this paper takes care of both the scenarios mentioned above and hence proves helpful for this recognition task. Also, it was observed that this resampling reduces the overfitting, and hence provides a better generalization over other resampling method.

III. FEATURES

1) *Cosine of Initial Angle*

The angle between the first and the third point is measured and angle's cosine is used as features rather than the angle itself to avoid a discontinuity as the angle passes 2pi and wraps to 0.

Formula:

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

This makes sure that shapes with similar shapes have similar values. This proved to be one of the moderate features for this task. Although this may take a great deal of importance in the other sketch recognition tasks, for arrow recognition, it can take any values as arrows can be at any directions, hence this might be the actual justification why this feature didn't prove to be one of the important features.

2) *Sine of Initial Angle*

The angle between first and third point is measured and angle's sine is used as features rather than the angle itself to avoid a discontinuity as the angle passes 2pi and wraps to 0.

Formula:

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

Similar argument goes for this feature as well. This makes sure that shapes with similar shapes have similar values. This feature too was moderate. For arrow recognition, it can take any values as arrows can be at any directions, hence this might be the actual justification why this feature didn't prove to be one of the important features.

3) *Diagonal Length of Bounding Box*

The diagonal length of bounding box is calculated by drawing the bounding box such that all points on the strokes lie inside the bounding box.

Formula:

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

This proved to be an important feature in this task. The diagonal length of bounding box gave one of the highest single accuracies of 95.67 % alone, which is pretty strong for this task. The reason why I have kept this in my final sub-set is the same. For non-arrow shapes, this feature might be same for some cases, but looking at the dataset, it differed.

4) *Diagonal Angle of Bounding Box*

The diagonal angle of bounding box is calculated by drawing the bounding box such that all points on the strokes lie inside the bounding box and calculating its angle with horizontal.

Formula:

$$arc\tan\left(\frac{y_{max} - y_{min}}{x_{max} - x_{min}}\right)$$

Similar argument goes for this feature as well, but this was more than moderate feature for this task. As the arrows are facing in any direction, this feature value came out to be varying too much with the change of arrow direction. Still giving a good accuracy made this feature a feature in the final subset.

5) *Distance between Initial and Last Points*

The absolute distance between the first and last points is calculated by the flowing formula.

Formula:

$$\sqrt{\left(x_{p-1}-x_0\right)^2+\left(y_{\mathrm{p}-1}-y_0\right)^2}$$

This distance actually varies with user to user, that's why its performance was moderate. Some users tend to draw arrows in other fashion than the others so their starting and ending points are not representative of the shape. The others also had similar kind of distances between their initial and final points. Hence this feature might give out moderate results.

6) *Cosine of Angle Between Initial and Last Points*

The cosine of an angle is calculated by the flowing formula.

Formula:

$$\frac{\left(x_{p-1}-x_0\right)}{\sqrt{\left(x_{p-1}-x_0\right)^2+\left(y_{\mathrm{p}-1}-y_0\right)^2}}$$

Such features hugely depend on which direction the arrow is facing, hence again a moderate feature in my opinion. As the arrows are facing in any direction, this feature value came out to be varying too much with the change of arrow direction. Still giving a good accuracy made this feature a feature in the final subset.

7) *Sine of Angle Between Initial and Last Points*

The sine of an angle is calculated by the flowing formula.

Formula:

$$\frac{\left(y_{p-1}-y_0\right)}{\sqrt{\left(x_{p-1}-x_0\right)^2+\left(y_{\mathrm{p}-1}-y_0\right)^2}}$$

Similar argument goes for this feature as well. Such features hugely depend on which direction the arrow is facing, hence again a moderate feature in my opinion. This makes sure that shapes with similar shapes have similar values.

8) *Stroke Length*

The Stroke length as itself might not be a good metric by itself, but its needed for normalization of various feature values

Formula:

$$\sum_{p=0}^{P-2}(Length\ of\ individual\ strokes)$$

This is an important feature for this task. That's because even though this feature, by itself is not that great (as it depends on the user who is sketching), but this feature is used to normalize various metrics to remove this "user bias". Due to this, I feel that this feature is important. Hence, it's a part of my final subset as well.

9) *Total Angle*

The total angle is calculated as follows:

Formula:

$$\sum_{p=0}^{P-2}(\theta_p)$$

This has also proved to be an important feature by itself, giving a high accuracy when only this feature is used. For arrows, the total rotation lies in a range, even though they may point anywhere. Hence, this feature is not affected by the direction of arrow, which might be the reason why it gives so high accuracy let alone used. That's the reason why this feature made to my final feature list.

10) *Total Absolute Angle*

The total absolute angle is calculated as follows:

Formula:

$$\sum_{p=0}^{P-2}abs(\theta_p)$$

This feature measures the actual rotation of the sketch. Arrows typically have this feature is a range, where as for other figures, this might vary on a huge basis. This is the reason why its included in the feature list and is a very important feature to be considered for classification of arrows from other shapes.

11) Total Squared Angle (Sharpness)

The total angle is calculated as follows:

Formula:

$$\sum_{p=0}^{P-2} (\theta_p)^2$$

This metric grows up as the number of corners grow up, almost in an exponential way. This makes the feature to be a distinguishing feature of arrows from all the other shapes. As arrows have a definite number of corners (almost!), this feature proves to an important feature in the point of view of this classification task and hence makes it to the final subset.

12) Aspect

The aspect is defined in terms of the fourth feature and is calculated as follows:

Formula:

$$abs\left(45^o - arc\tan\left(\frac{y_{max} - y_{min}}{x_{max} - x_{min}}\right)\right)$$

This is a Long feature and it diminishes the difference between arrows with starting angles less than 45 and arrows with starting angles greater than 45, which seems to be essential in this task. But still, it counts as one of the moderate features as it does not take the shape of the sketch into consideration. This feature might give good results, but its prone to overfitting. This feature also makes place into final subset.

13) Area of Bounding Box

The area of the bounding box is calculated as follows.

Formula:

$$(x_{max} - x_{min}) * (y_{\max} - y_{\min})$$

The argument of Long that this feature is not that important as compared to others proved to be true in this task. Area also does not take the shape of the sketch into consideration, hence can be counted as a moderate feature. This feature might give good results, but its prone to overfitting. This feature also makes place into final subset.

14) Log of Area of Bounding Box

The logarithm of area of the bounding box and is calculated as follows.

Formula:

$$\log((x_{max} - x_{min}) * (y_{\max} - y_{\min}))$$

This feature makes a little difference in the classification task. As the logarithm is taken, the overall effect of total area is diminished, which is desirable. Hence, it can be said that this feature is more important than the previous one, yet moderately important. This feature also makes place into final subset.

15) Log of Aspect

This is the logarithm of the aspect and is calculated as follows:

Formula:

$$\log(abs\left(45^o - arc\tan\left(\frac{y_{max} - y_{min}}{x_{max} - x_{min}}\right)\right))$$

This is also a moderate feature. This feature does not take the actual shape of the sketch under consideration, and hence is prone to overfit the training data. Still, it gives pretty high accuracy when its solely used to classify arrows, which is the reason why this feature is kept in the final feature set.

16) Log of Stroke Length

The logarithm of Stroke length is calculated as follows

Formula:

$$\log\left\{\sum_{p=0}^{P-2} (Length\ of\ individual\ strokes)\right\}$$

Important feature. For this classification task it was seen that the stroke length was important but not linearly, but logarithmically. Higher stroke length does not mean that this feature must be weighted linearly, but a logarithm of that feature proved to be giving good results. Arrows tend to possess similar range of log stroke length values, hence this made to the final subset feature list.

17) Total Angle Divided by Absolute Angle

This is calculated as follows:

Formula:

$$\frac{\sum_{p=0}^{P-2}(\theta_p)}{\sum_{p=0}^{P-2} abs(\theta_p)}$$

This measures the approx. total number of non-absolute rotations, which is close in some range for arrows as opposed to other shapes, hence a vital feature. This makes sure that shapes with similar shapes have similar values. The single accuracy is very high for this feature as well, hence, this feature made it to the final set of features.

18) Total Angle Divided by Stroke Length

This is calculated as follows:

Formula:

$$\frac{\sum_{p=0}^{P-2}(\theta_p)}{\sum_{p=0}^{P-2}(Length\ of\ individual\ strokes)}$$

The normalization of the Total angle feature by stroke length proves to be important as it makes this feature invariant to different stroke length arrows as opposed to other shapes. Hence an important feature. This makes sure that shapes with similar shapes have similar values. The single accuracy is very high for this feature as well, hence, this feature made it to the final set of features.

19) Stroke Length Divided by Initial to Last Points Distance

This is a density metric and is calculated as follows

Formula:

$$\frac{\sum_{p=0}^{P-2}(Length\ of\ individual\ strokes)}{\sqrt{\left(x_{p-1}-x_0\right)^2+\left(y_{\mathrm{p-1}}-y_0\right)^2}}$$

The normalization of the initial to last points distance feature by stroke length proves to be important as it makes this feature invariant to different stroke length arrows as opposed to other shapes. Hence an important feature. The single accuracy is moderate for this feature as well, hence, this feature made it to the final set of features.

20) Stroke Length Divided by Diagonal Length

This is another density metric and is calculated as follows

Formula:

$$\frac{\sum_{p=0}^{P-2}(Length\ of\ individual\ strokes)}{\sqrt{(x_{max}-x_{min})^2+(y_{\max}-y_{\min})^2}}$$

The normalization of the diagonal distance feature by stroke length proves to be important as it makes this feature invariant to different stroke length arrows as opposed to other shapes. Hence an important feature. This makes sure that shapes with similar shapes have similar values. The single accuracy is moderate for this feature as well, hence, this feature made it to the final set of features.

21) Initial to Last Points Distance Divided by Diagonal Length

This is non-subjective openness metric and is calculated as follows

Formula:

$$\frac{\sqrt{\left(x_{p-1}-x_0\right)^2+\left(y_{\mathrm{p-1}}-y_0\right)^2}}{\sqrt{(x_{max}-x_{min})^2+(y_{\max}-y_{\min})^2}}$$

This normalization did not provide much classification of arrows as opposed to other shapes. Hence an important feature. The single accuracy is moderate for this feature as well, hence, this feature made it to the final set of features.

22) Curviness

The curviness (very important feature) is calculated as follows:

Formula:

$$\sum_{p=0}^{P-2} abs(\theta_p), \qquad (if\ abs(\theta_p) < 19^o)$$

This unique feature proved to be very important feature. The curviness of arrows is generally less than the other sketches given in this classification task which made this feature stand out on its own. This makes sure that shapes with similar shapes have similar values. The single accuracy is very high for this feature as well, hence, this feature made it to the final set of features.

23) Total Time

Although this depends on which user is sketching, but computed to see if it matters, its formula is:

Formula:

$$(t_{p-1} - t_0)$$

This is a misleading feature, as the time taken to draw a sketch is very much dependent on the user who is sketching, it performs worse. It can be seen form the table 1, this feature gives the lease accuracy when used solely to classify the arrows. Hence, this feature is excluded from the final feature subset.

24) Maximum Speed

Although this depends on which user is sketching, but computed to see if it matters, its formula is:

Formula:

$$max(s_0, s_1, \dots, s_{p-2})$$

Where,

$$s_p = \frac{\Delta x_p^2 + \Delta y_p^2}{\Delta t_p^2}$$

This is a misleading feature, as the time taken to draw a sketch is very much dependent on the user who is sketching, it performs worse. It can be seen form the table 1, this feature gives the lease accuracy when used solely to classify the arrows. This feature might give good results, but its prone to overfitting. Hence, this feature is excluded from the final feature subset.

25) Average Speed

Although this depends on which user is sketching, but computed to see if it matters, its formula is:

Formula:

$$mean(s_0, s_1, \dots, s_{p-2})$$

Where,

$$s_p = \frac{\Delta x_p^2 + \Delta y_p^2}{\Delta t_p^2}$$

This is a misleading feature, as the time taken to draw a sketch is very much dependent on the user who is sketching, it performs worse. This feature might give good results, but its prone to overfitting. It can be seen form the table 1, this feature gives the lease accuracy when used solely to classify the arrows. Hence, this feature is excluded from the final feature subset.

26) Entropy

Entropy measures the information content of the stroke; its formula is as follows:

Formula:

$$H(S) = -\sum_{i=1}^{m} P(s_i)\, log\, P(s_i)$$

Where,

'S' is a particular symbol according to the angles made by that point with adjacent ones.

This is the Bhat's Entropy calculated with a little different scheme than Bhat's. This feature had similar values for similar shapes, hence performed good on arrows vs the other sketches. As this was a distinguishing feature, it's one of the main features for this recognition task. Hence, this feature is included in the final feature subset.

27) Stroke Length Divided by Initial to Centroid Points Distance

This is a new feature which proved eminent in this problem, its formula is:

Formula:

$$\frac{\sum_{p=0}^{P-2}(Length\ of\ individual\ strokes)}{dist\{(x_0, y_0), (C_x, C_y)\}}$$

Where,

$(C_x, C_y) = Centroid(P_0, P_1, \dots, P_n)$, calculated

$$C_x = \sum_{p=0}^{P-1}(x_p), C_y = \sum_{p=0}^{P-1}(y_p)$$

$$dist\{(x_a, y_a), (x_b, y_b)\} = \sqrt{(x_a - x_b)^2 + (y_a - y_b)^2}$$

This is a new feature which proved eminent in this problem. The centroid can be defined as center of gravity of a particular symbol. This metric for arrow lie near the pointing head of the arrow. Hence, this is a distinguishing feature. For other shapes, this feature is just placed in the middle of the bounding box or random. Hence, the stroke length divided by initial to centroid points distance provides a good

normalization factor as well. Due to this, I feel that this feature is important. Hence, it's a part of my final subset as well.

28) Diagonal Length Divided by Initial to Centroid Points Distance

This is a new feature which proved eminent in this problem, its formula is:

Formula:

$$\frac{\sqrt{(x_{max} - x_{min})^2 + (y_{\max} - y_{\min})^2}}{distance\{(x_0, y_0), (C_x, C_y)\}}$$

Where,

$(C_x, C_y) = Centroid(P_0, P_1, \dots, P_n)$, calculated

$$C_x = \sum_{p=0}^{P-1}(x_p), C_y = \sum_{p=0}^{P-1}(y_p)$$

$$dist\{(x_a, y_a), (x_b, y_b)\} = \sqrt{(x_a - x_b)^2 + (y_{\mathrm{a}} - y_b)^2}$$

This is a new feature which proved eminent in this problem. For arrows, this feature lies in a range and for other shapes, this feature is just random. Hence, the diagonal length divided by initial to centroid Points distance provides a good normalization factor as well. Due to this, I feel that this feature is important. Hence, it's a part of my final subset as well.

29) Number of Times Direction Changes

This is a new feature which is defined as follows:

Formula:

$$\sum_{p=0}^{P-2} \boldsymbol{I}\{abs(\theta_p)\}, \qquad (if\ abs(\theta_p) > 90^o)$$

Where,

$$\boldsymbol{I}\{abs(\theta_p)\} = 1, \quad if\ abs(\theta_p) > 90^o$$

$$\boldsymbol{I}\{abs(\theta_p)\} = 0, \quad else$$

This unique new feature proved to be very important feature. The number of times direction changes of arrows is generally in a range than the other sketches given in this classification task which made this feature stand out on its own. The single accuracy is very high for this feature as well, hence, this feature made it to the final set of features.

30) Density of Points

This is a new feature which calculates the density of the sampled points for the given sketch, defined as follows:

Formula:

$$\frac{Total\ number\ of\ points}{\sum_{p=0}^{P-2}(Length\ of\ individual\ strokes)}$$

This feature is the normalized feature and hence proving to be moderate. The single number of points proved to be a bad feature as its too user-dependent. But the normalization has made this feature invariant to the length of strokes which is desired. The single accuracy is moderate for this feature as well, still this feature made it to the final set of features.

31) Normalized Centroid Angle

This is a new feature which is the angle centroid makes with the lower horizontal line of the bounding box. The arrow was rotated to calculate this feature, hence called as normalized. Its rotated to point the direction to the right of the arrow, by subtracting the initial angle from the resulting angle:

Formula:

$$Angle(line(\{X_{min}, Y_{min}\}, \{C_x, C_y\}), line(\{X_{min}, Y_{min}\}, \{X_{max}, Y_{min}\}))$$

Where,

$Angle(line\ 1, line\ 2)$ denotes the angle between the 'line 1' and 'line 2', and $Line(\{x, y\}, \{x, y\})$ represents line joined by the two points,

$$C_x = \sum_{p=0}^{P-1}(x_p), C_y = \sum_{p=0}^{P-1}(y_p)$$

Similar argument goes for this feature as well, but this was more than good feature for this task. This makes sure that shapes with similar shapes have similar values. As the arrows are facing in any direction, this feature value came out to be stable with the change of arrow direction, giving a good accuracy made this feature a feature in the final subset.

32) Aspect of Normalized Centroid Angle

This is a new feature which is the aspect of angle centroid makes with the lower horizontal line of the bounding box. The arrow was rotated to calculate this feature, hence called as normalized. Its rotated to point the direction to the right of the arrow, by subtracting the initial angle from the resulting angle:

Formula:

$$abs(45 - Angle(line(\{X_{min}, Y_{min}\}, \{C_x, C_y\}), line(\{X_{min}, Y_{min}\}, \{X_{max}, Y_{min}\})))$$

Where,

$Angle(line\ 1, line\ 2)$ denotes the angle between the 'line 1' and 'line 2', and $Line(\{x, y\}, \{x, y\})$ represents line joined by the two points,

$$C_x = \sum_{p=0}^{P-1}(x_p), C_y = \sum_{p=0}^{P-1}(y_p)$$

This unique new feature proved to be moderate feature. It diminishes the difference between arrows with starting angles less than 45 and arrows with starting angles greater than 45, which seems to be essential in this task. But still, it counts as one of the moderate features as it does not take the shape of the sketch into consideration. This makes sure that shapes with similar shapes have similar values. This feature might give good results, but its prone to overfitting. The single accuracy is moderate for this feature as well, still this feature made it to the final set of features.

33) Initial Point to Centroid Point Distance

This is a new feature which calculates the distance of centroid from the initial point where the stroke was drawn:

Formula:

$$\sqrt{(C_x - x_0)^2 + (C_y - y_0)^2}$$

Where,

$\{C_x, C_y\}$ is the centroid point,

$$C_x = \sum_{p=0}^{P-1}(x_p), C_y = \sum_{p=0}^{P-1}(y_p)$$

This is an important feature for this task. That's because even though this feature, depends on the user who is sketching, but still this feature proved vital. Hence, it's a part of my final subset as well.

34) Number of Corners

This is a new feature which calculates the number of corners by the following formula:

Formula:

$$\sum_{p=0}^{P-2} \boldsymbol{I}\{abs(\theta_p)\}, \quad (if\ abs(\theta_p) > 19^o)$$

Where,

$$\boldsymbol{I}\{abs(\theta_p)\} = 1, \quad if\ abs(\theta_p) > 19^o$$
$$\boldsymbol{I}\{abs(\theta_p)\} = 0, \quad else$$

This crude corner detection algorithm worked and gave a good individual accuracy for this feature. The number of corners of arrows is generally less than the other sketches given in this classification task which made this feature stand out on its own. This makes sure that shapes with similar shapes have similar values. Hence, this feature is included in the final feature subset.

35) Velocity

This is a new feature which calculates the velocity by the following formula:

Formula:

$$\frac{\sqrt{(x_{p-1} - x_0)^2 + (y_{p-1} - y_0)^2}}{(t_{p-1} - t_0)}$$

Where,

$$\boldsymbol{I}\{abs(\theta_p)\} = 1, \quad if\ abs(\theta_p) > 19^o$$
$$\boldsymbol{I}\{abs(\theta_p)\} = 0, \quad else$$

This is a misleading feature, as the time taken to draw a sketch is very much dependent on the user who is sketching, it performs worse. This feature might give good results, but its prone to overfitting. It can be seen form the table 1, this feature gives the lease accuracy when used solely to classify the arrows. Hence, this feature is excluded from the final feature subset.

IV. RESULTS

These are the results obtained from the classifier:

TABLE I

Feature Number	Accuracy Random Forest Classifier (Using Single Feature)	Included in Final Sub-Set?
1	77.46 %	Yes
2	78.28 %	Yes
3	95.84 %	Yes
4	83.82 %	Yes
5	72.87 %	Yes
6	78.23 %	Yes
7	84.65 %	Yes
8	83.97 %	Yes
9	91.43 %	Yes
10	93.44 %	Yes
11	94.82 %	Yes
12	72.82 %	Yes
13	79.45 %	Yes
14	89.75 %	Yes
15	93.77 %	Yes
16	85.74 %	Yes
17	89.54 %	Yes
18	92.78 %	Yes
19	87.45 %	Yes
20	89.45 %	Yes
21	82.56 %	Yes
22	79.94 %	Yes
23	68.43 %	No
24	71.56 %	No
25	77.67 %	No
26	91.05 %	Yes
27	94.53 %	Yes
28	86.44 %	Yes
29	92.67 %	Yes
30	88.94 %	Yes
31	90.18 %	Yes
32	86.83 %	Yes
33	88.84 %	Yes
34	91.32 %	Yes
35	72.54 %	No

TABLE II

Feature Number	Accuracy J48 (Using Single Feature)	Accuracy LMT (Using Single Feature)	Accuracy ZeroR (Using Single Feature)
1	73.67 %	73.95 %	50 %
2	72.34 %	71.84 %	50 %
3	91.58 %	92.46 %	50 %
4	81.23 %	80.73 %	50 %
5	71.34 %	74.12 %	50 %
6	74.96 %	77.39 %	50 %
7	81.54 %	83.28 %	50 %
8	79.34 %	81.38 %	50 %
9	90.3 %	91.50 %	50 %
10	91.46 %	89.37 %	50 %
11	92.32 %	90.23 %	50 %
12	71.39 %	70.36 %	50 %
13	73.56 %	77.24 %	50 %
14	84.75 %	82.83 %	50 %
15	91.27 %	90.28 %	50 %
16	84.71 %	89.72 %	50 %
17	87.04 %	86.49 %	50 %
18	91.80 %	92.23 %	50 %
19	82.49 %	80.89 %	50 %
20	79.32 %	81.28 %	50 %
21	84.38 %	86.77 %	50 %

22	78.73 %	80.46 %	50 %
23	64.33 %	69.23 %	50 %
24	70.06 %	72.72 %	50 %
25	72.46 %	70.23 %	50 %
26	90.70 %	92.37 %	50 %
27	92.75 %	94.82 %	50 %
28	86.98 %	87.27 %	50 %
29	90.69 %	89.12 %	50 %
30	84.39 %	82.82 %	50 %
31	89.61 %	91.72 %	50 %
32	83.83 %	85.22 %	50 %
33	90.84 %	89.28 %	50 %
34	90.82 %	91.66 %	50 %
35	68.75 %	65.01 %	50 %

TABLE III

10-Fold Cross-Validation scheme	Accuracy (All Features)	Accuracy (Subset of Features)
ZeroR	50 %	50 %
J48	99.08 %	99.075 %
Random Forest	99.56 %	99.54 %
LMT	99.10 %	99.00 %

In summarization, all the features with were dependent remotely on the 'time' feature proved to be a curse for the classifier (as seen from the table 1 and 2). It didn't ensure that shapes with similar shapes have similar values. They were more user dependent.

The reason why the results from subset features selection and all the features is because it's a 10-fold cross-validation scheme, hence the results from all the 10 folds are averaged to get the actual accuracy. Also, this is because I have tried to reduce the overfitting so that the generalization quotient is more. This means that when this model sees a new data, it is better at predicting the shapes of the new sketches. The 'time' related features made the model to overfit the training data which is highly undesirable.

I tried various experiments to check the generality of the features (Not to overfit the data). Following are the two scenarios I tried, which proves that I reduced over-fitting to a great extent.

A. A random 80% - 20% train - test split:

80% - 20% Train – Test Split	Test Accuracy (All Features)	Test Accuracy (Subset of Features)
ZeroR	50 %	50 %
J48	97.17 %	98.49 %
Random Forest	98.24 %	98.94 %
LMT	98.01 %	98.65 %

B. A random 70% - 30% train - test split:

70% - 30% Train – Test Split	Test Accuracy (All Features)	Test Accuracy (Subset of Features)
ZeroR	50 %	50 %
J48	94.08 %	96.85 %
Random Forest	95.56 %	97.88 %
LMT	94.10 %	97.21 %

This proves the argument that the picked subset was indeed the right one as it increased the test accuracy hugely in all the classifiers. For this particular problem, time-related features proved to be useless (This might not be the case for other classification tasks!).

All the features were developed keeping in mind only 1 thing: similar shapes should have similar feature values.

V. CONCLUSION AND FUTURE WORK

More feature engineering could be deployed to generalize the model not only for arrows but also for other types of sketch recognition tasks.

VI. CHALLENGES

- Due to the huge dataset, efficient cleaning/noise-removal was tricky.
- The edge cases needed to be taken care of, as some sub-strokes didn't have any points in them.
- A considerable amount of time had to be spent to achieve the statistical analysis of both the classes.

VII. PACKAGES USED (PYTHON)

(Also check requirements.txt for installing the packages)

1. NumPy: NumPy is the fundamental package for scientific computing in Python. It is a Python library that provides a multidimensional array object.
    - Project uses this library for speeding up the matrix operations.
2. json: json provides a toolbox with solid implementations of a bunch of state-of-the-art readers for json objects.
    - Project uses this library for importing the data.
3. codecs: It is used for serializing, de-serializing and decoding a Python object structure.
    - Project uses this library for decoding the data
4. Scipy: It is a Python-based ecosystem of open-source software for mathematics, science, and engineering.
    - Project uses this library for calculating stats
5. Pandas: This package provides fast, flexible, and expressive data structures designed to make working with "relational" or "labeled" data both easy and intuitive. It aims to be the fundamental high-level building block for doing practical, real world data analysis in Python.
    - Project uses this library for handling data frames.
6. copy: This module provides generic shallow and deep copy operations.
    - Project uses generating copies of lists
7. Matplotlib: It has functions that can generate plots, histograms, power spectra, bar charts, error charts, scatterplots, etc.
    - Project uses this library for plotting the sampled arrows on the dataset.

NOTE: See next page for appendix

ACKNOWLEDGMENT

Thankful to Prof. Tracy Hammond for teaching the wonderful on-going course on Sketch Recognition. Also, thankful to Paul Taele for providing dataset, instructions and constant help during the project.

REFERENCES

Dean Rubine. Specifying Gestures by Example, Proceedings of the 18th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '91), 1991.

A. Chris Long, Jr., James A. Landay, Lawrence A. Rowe, and Joseph Michiels. Visual Similarity of Pen Gestures, Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '91), 1991.

Brandon Paulson, Pankaj Rajan, Pedro Davalos, Ricardo Gutierrez-Osuna, and Tracy Hammond. What!?! No Rubine Features?: Using Geometric-based Features to Produce Normalized Confidence Values for Sketch Recognition. VL/HCC Workshop: Sketch Tools for Diagramming. 2008

Brandon Paulson and Tracy Hammond. PaleoSketch: Accurate Primitive Sketch Recognition and Beautification. Proceedings of the 13th international conference on Intelligent user interfaces. 2008.

Akshay Bhat and Tracy Anne Hammond. Using Entropy to Identify Shape and Text in Hand Drawn Diagrams. Proceedings of the Twenty-First International Joint Conference on Artificial Intelligence. 2009.

Tracy Hammond and Randall Davis. Tahuti: A Geometrical Sketch Recognition System for UML Class Diagrams. SIGGRAPH '06. 2006.

APPENDIX

Feature Number	Feature name in this paper	Feature name in python code
1	Cosine of Initial Angle	cosine_initial_angle
2	Sine of Initial Angle	sine_initial_angle
3	Diagonal Length of Bounding Box	diagonal_length_bounding_box
4	Diagonal Angle of Bounding Box	diagonal_angle_bounding_box
5	Distance between Initial and Last Points	initial_last_points_dist
6	Cosine of Angle Between Initial and Last Points	cosine_angle_initial_last_points
7	Sine of Angle Between Initial and Last Points	sine_angle_initial_last_points
8	Stroke Length	stroke_length
9	Total Angle	total_angle
10	Total Absolute Angle	total_absolute_angle
11	Total Squared Angle (Sharpness)	total_squared_angle
12	Aspect	aspect
13	Area of Bounding Box	area_bounding_box
14	Log of Area of Bounding Box	log_area_bounding_box
15	Log of Aspect	log_aspect
16	Log of Stroke Length	log_stroke_length
17	Total Angle Divided by Absolute Angle	total_by_absolute_angle
18	Total Angle Divided by Stroke Length	total_angle_by_stroke_length
19	Stroke Length Divided by Initial to Last Points Distance	stroke_length_by_initial_last_points_distance
20	Stroke Length Divided by Diagonal Length	stroke_length_by_diagonal_length
21	Initial to Last Points Distance Divided by Diagonal Length	initial_last_points_distance_by_diagonal_length

22	Curviness	curviness
23	Total Time	total_time
24	Maximum Speed	max_speed
25	Average Speed	avg_speed
26	Entropy	entropy
27	Stroke Length Divided by Initial to Centroid Points Distance	stroke_length_by_initial_centroid_points_dist
28	Diagonal Length Divided by Initial to Centroid Points Distance	diagonal_length_by_initial_centroid_points_dist
29	Number of Times Direction Changes	no_of_change_of_direction
30	Density of Points	density_of_points
31	Normalized Centroid Angle	normalized_centroid_angle
32	Aspect of Normalized Centroid Angle	aspect_normalized_centroid_angle
33	Initial Point to Centroid Point Distance	initial_centroid_points_dist
34	Number of Corners	no_of_corners
35	Velocity	velocity