State of Research - RefinerBayes

Joshua Ehrlich

July 24, 2009

The RefinerBayes is a BayesianClassifier used in the refiner project to classify (label) groups. Currently the classifier assumes that the grouping of substrokes is correct but makes no assumptions about its classification. The confusion matrix for the current classifier:

		ACTUAL LABEL									
		AND	LABEL	LABELBOX	NAND	NOR	NOT	NOTBUBBLE	OR	WIRE	XOR
P	AND	.81	0	0	.22	0	.04	0	.28	0	0
R	LABEL	.09	.93	.03	0	0	.04	0	.07	.03	0
E	LABELBOX	0	0	.97	0	0	0	0	0	0	0
D	NAND	.02	0	0	.28	0	0	0	0	0	0
I	NOR	0	0	0	.06	.37	.01	0	0	.03	.06
C	NOT	0	.02	0	.06	.16	.82	0	0	.02	0
T	NOTBUBBLE	.02	.01	0	0	0	0	.83	0	.01	0
E	OR	.0	.01	0	.39	.21	.01	0	.61	.01	.13
D	WIRE	.02	.02	0	0	0	0	.17	.02	.88	0
	XOR	0	.02	0	0	.26	.01	0	.03	.02	.81

1 Bus/Issues

There are no known bugs with the RefinerBayes. However, some of the features, in particular the substroke classification feature hurts the results more than it helps. A second issue is that it is not as general as would be desired. Finally, this classifier is a little slow, mostly due to running the current combination recognizer, which is also slow. This is not significant when classifying a single sketch but it makes training and testing slow. Another issue is

2 To Do

The main areas of improvement for the Bayes Classifier is modifying the features. We have currently implemented the three types of features. The first feature describes the neighboring shapes. This feature is currently implemented as a sorted list of the classifications of the neighboring shapes (Gate, Wire, and Label). The second feature describes the original recognizer results. This feature

is currently implemented as a triple consisting of the WireRecognizer result, the TextRecognizer probability, and the label returned by the GateRecognizer. The final set of features relates to errors made in single-stroke classification. There are four features used here. The first three are continuous features corresponding to the percentage of strokes classified in each class. The final feature is the number of substrokes in the shape.

Currently, the classification features are hurting recognition rather than helping. To improve the classifier new features could be added or existing one changed. It might be reasonble to use a single feature for the three percentages but that would require a three dimensional gaussian. When our connection data becomes more sophisticated we could expand that feature. Alternates we considered but have yet to implement include features related to the distance between the shape and its neighbors and differentiating between inputs and output connections.

3 How to use

The Bayes Classifier can either be trained or a pretrained classifier can be loaded from a file. To train, set the RETRAIN flag in the Refiner class to true. This will cause the classifier to train from a directory with sketches that have both the substrokes classified separately from the labels of the shapes, and the shapes connected properly to their neighbors. From this training data it generates the probabilities of each feature. After training the classifier is saved to a file rbayes.bc in the Refiner project. When loaded it initializes all the features that are not serializable, which consists of the shape recognizers and the continuous distributions. To test use the function TestingResults. This loads sketches from a separate set of files and runs the classifier on each shape and then generates a confusion matrix. It has three main functions for general use. The function that uses the bayesClassifier directly on the shape is called bayesClassification which takes a shape and a label and calculates the probability that the shape has the given label given its features. While this function should not be called directly (at least until one of the other functions is called) it is used in all of the other functions. BayesClassify(sketch, labels) is a function that takes a sketch and a list of potential labels and for each shape in the sketch classifies each shape with each label and applies the best label. Once this has been called once for a sketch you can use BayesClassification(shape, labels), which takes a single shape and a list of labels and returns the normalized probability that the shape has its current label.