tSNE

The project uses bh tsne which is stochastic neighborhood embedding to cluster 28x28 images of contours. The regular tnse algorithm reduces the dimensionality to 2 using the gaussian function with euclidean distance over the data points. Then it uses the gradient descent to minimize the convergence function. The bh tsne is a more efficient version of tsne that uses oct tree structure in the third dimension with n body formulations. My tsne example uses python script to display the clusters are in the Fig 1 below.

After the graph is displayed, the clicks are used to make a convex hull to classify all the contours in the convex hull with the key pressed. Then this is put into the classified files list.

My program uses an intermediate step using another python script to composite the classified clusters. Visually the mislabeled images are identified and corrected with a third script.
These helper scripts called ‘composite.py’ and ‘label_changer.py’ are supplied only to demonstrate the usefulness of the tsne clustering algorithm.

To run the tsne script execute the following from your client i.e MAC or WINDOWS terminal as follows:

python tsne_scatter_graph_clusters.py --contours_path=images_output.

The graph produced by the tsne script will look as in Fig 1 below

Press the key to classify the character and then click to make the convex hull around it. Then press the escape key to store the classified contours in the classified files txt.
For example, if I want to classify the < key then I press that first and draw the convex hull with the clicks. This is what it looks like:

Fig 1

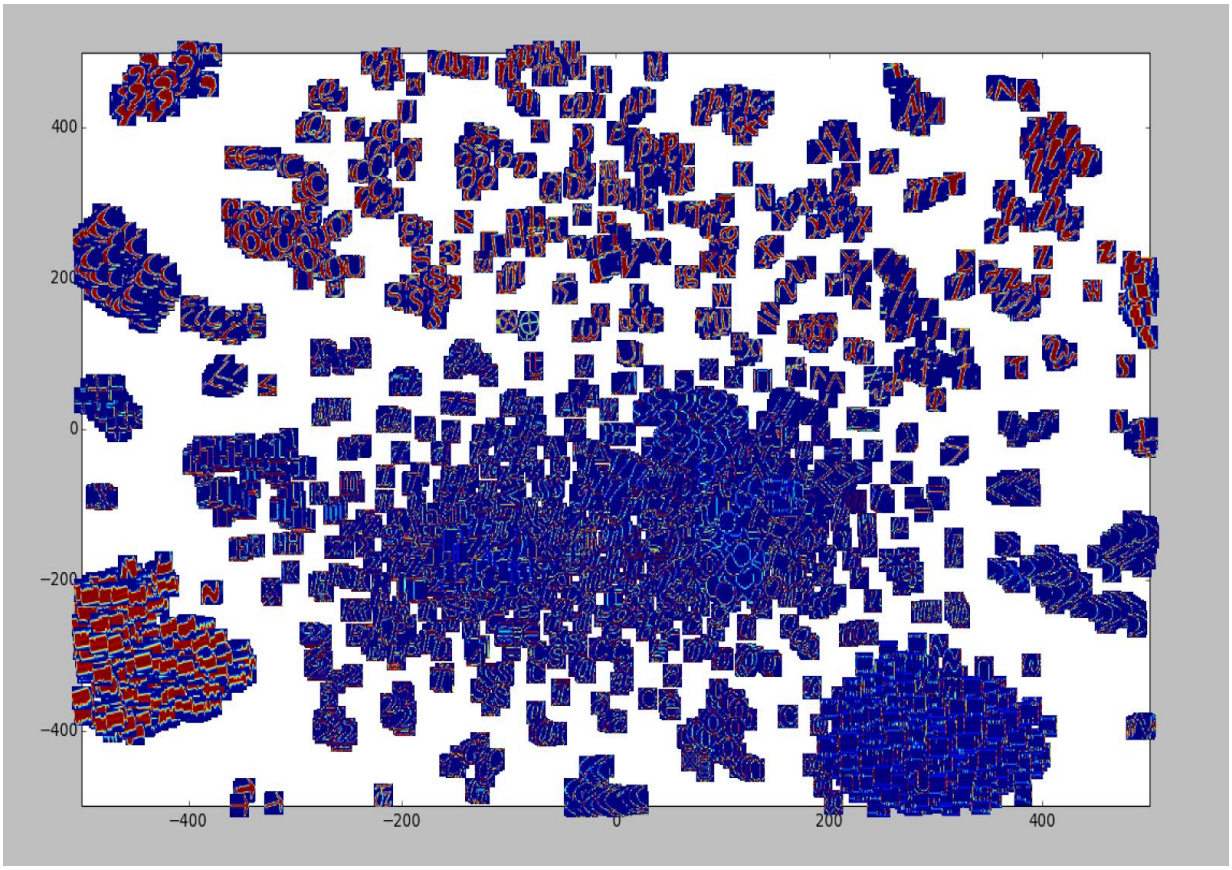

Fig 2

In Fig 2 above the vertices are the clicks and all the < characters are in the convex hull. After this, I press the escape key and it labels all the images in the convex hull with <. After the images have been classified, I use the composite.py program to see if there are any misclassifications. To run this , from your MAC or WINDOWS terminal run

python composite.py --classifier=<
To change the misclassified labels, use the label_changer.py program. The program needs the indices to change the misclassified labels. To run label_changer, do python label_changer.py --file_name=classified_files.txt --to_chr=< --from_chr=>.

Only a few samples of the 28x28 numpy array formatted images are supplied with the distribution. To run tsne one can supply their own images as long as the images comply with the

uint8 numpy ndarray that can be easily created from the font directories found in unix systems and with a simple python script.