CS 473 Project: Module 5 Proposal

We will use K-Means clustering for module 5. K-means clustering maintains exactly K clusters (represented as centroids), and assigns every datapoint to the nearest centroid. After every iteration, the centroids are recomputed until the sum of the squared distances between the data points and the centroids becomes as small as possible. This clustering method is useful in that it converges quickly and easily adapts to new examples. It can easily generalize clusters to different shapes and sizes based on the distribution of cluster points.

We will be incorporating the number of symbols (entities, weak entities, relationships, identifying relationships) to supplement our advanced-clustering method. The way we will do this is by saving the counts for the labeled symbols when pulling the text from each of the images. This will be done through the following algorithm:

Maintain a dictionary of symbol types and their counts for each image:
`dict_img1, dict_img2, …`

1) Loop through each image for clustering
    a) Loop through every label in the image
        i) Pull the type of symbol that the label has (entity, weak_entity, etc…)
        ii) Increment the count for that image and that corresponding symbol
            (1) `Example: dict_img['entity']+= 1`
        iii) Crop the image using the coordinates given by our labeled image
        iv) Pull the text from the cropped image
2) KMeans to cluster image

This way, with our advanced-clustering, along with comparing the text between the images to cluster them properly to train KMeans, we can also now compare the counts of certain symbols in images to cluster them together using KMeans. This is useful as if two images have a similar count for many of the different symbol types (Ex: img1 and img2 both have 4 weak entities and 2 relationships) and also have similar text, we can say with some confidence that these two images should be in the same clusters.

We can take this further by considering what text is present in certain symbols. A reason that this is useful is that we can identify common relationships and entity tables used by students to hopefully cluster common student submissions together. An example of this would be if we had many students using a Relationship called “Travels” and two Entities with “Airline” and “People” in the text, and we want to be able to cluster students together who use this similar text in the symbols. The functionality we are suggesting would help in this case. Adding this functionality to the algorithm we presented above would be relatively trivial, as instead of just comparing all of the text in images using KMeans, we would be comparing symbol text between images to better identify common submissions.

Taking these features we described above will give us a more accurate clustering than what we would be doing in module 4.