Classification of Uppercase English Alphabets using Active Learning

-by Deepshikha Sharma

The task was to classify large number of black and white rectangular pixel displays into any of the 26 capital English alphabet. The dataset consisted of 16 features. There were total 20,000 data elements. The character images were based on 20 different fonts and each letter within these 20 fonts was randomly distorted to produce a file of 20,000 unique stimuli. Each stimulus was converted into 16 primitive numerical which were then scaled to fit into a range of integer values from 0 through 15.

Implemented the classification algorithm using both “pool-based” and “stream-based” active learning. In all the tasks, the code of pool-based is followed by the code of stream-based active learning.

The libraries and frameworks used in the model.

- Pandas: Pandas is a software library written for Python. It is basically used in data manipulation and analysis. It offers various data structures for this purpose. Used Pandas for reading the dataset. Used read_csv function of Pandas that helped import the dataset into the program and some variable takes it as its value.
- Numpy: It is a library in Python for handling and manipulating large arrays and matrices. Also used for handling arrays.
- Sklearn: It is a machine learning library in Python which has features for classification, clustering and regression algorithms. Used it for encoding the labels into numerical values(LabelEncoder), for implementing Random Forest Classifier ,for splitting training and test set, for k-means clustering for Gaussian process classifier.
- modAL: It is an active learning framework in Python3. It heps in building active learning workflows easily and with flexibility. Used ActiveLearner and Committee models from modAL. Used this framework for uncertainity sampling such as least confident, margin sampling and entropy sampling. Also used it for vote entropy and Kl divergence measures of disagreement.
- Matplotlib: It is a library in python for visualizations and for plotting. Used it for plotting graphs.

Used Pickle module to store results.

First of all, only 10% of labelled data was retained. “split” the dataset where 10% of the data was labelled only. Rest 90% was thus unlabelled in the pool.

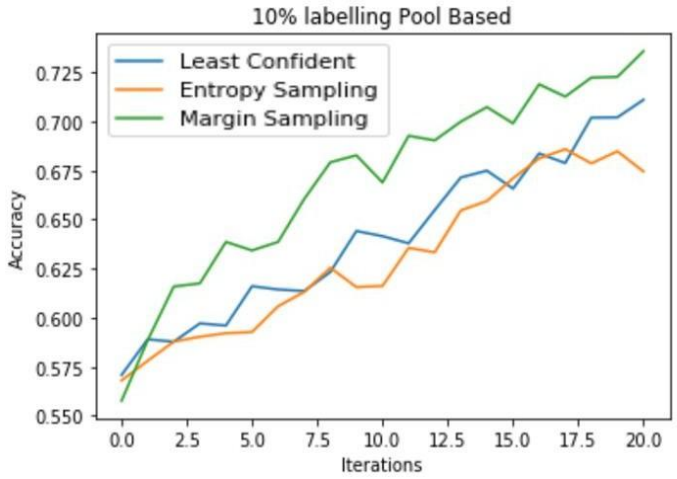
Then encoded the labels into numericals.

First used pool-based active learning and then stream-based active learning.

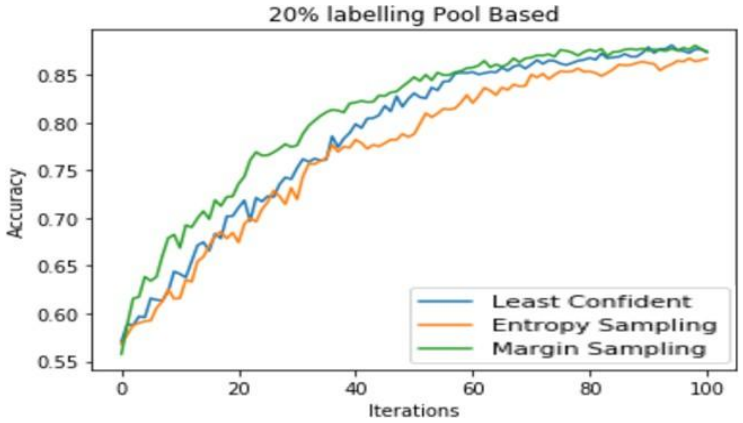
Then using the ActiveLearner model of the modal framework, implemented active learning on the dataset using least confident, margin and entropy sampling sampling individually and then compared them by plotting a graph. Uncertainity_sampling module gives least confident, margin_sampling module gives margin sampling and entropy_sampling module gives entropy sampling.

Repeated this process for additional 10%, 20%, 30%, 40% data points. The

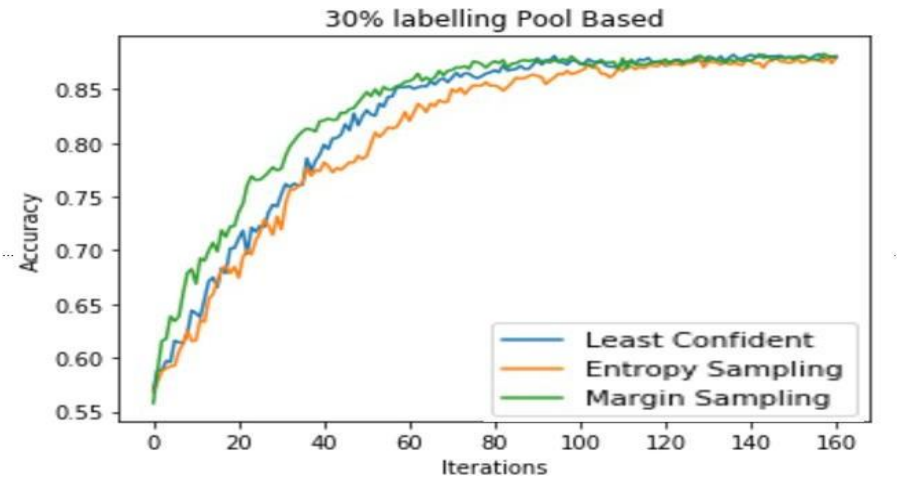
Results for 10% additional labelling using “pool-based” active learning is:



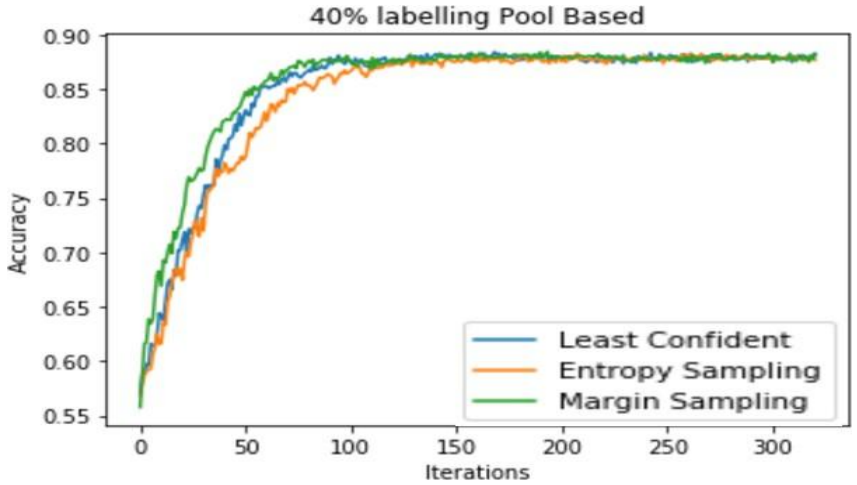
The results for 20% additional labelling using “pool-based” active learning is:



The results for 30% additional labelling using “pool-based” active learning is:



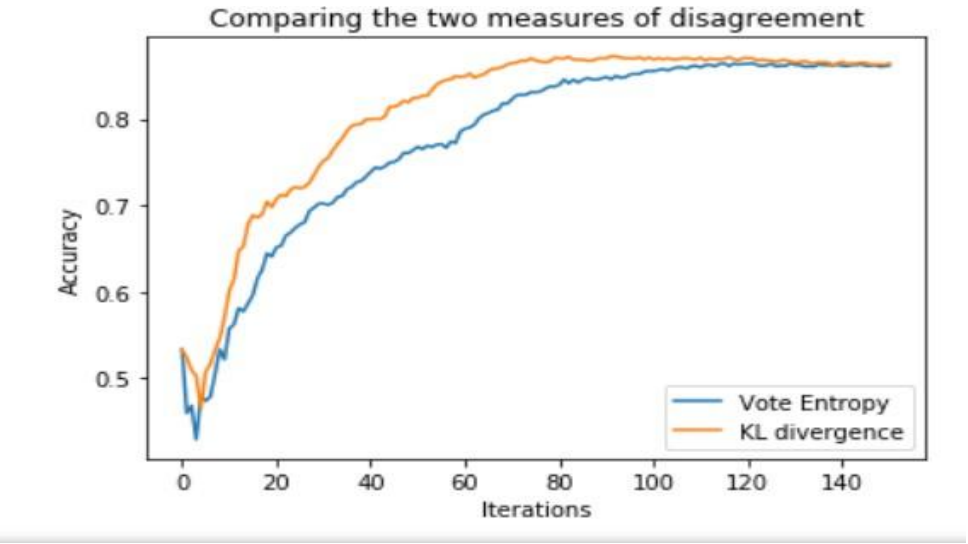
The results for 40% additional labelling using “pool-based” active learning is:



Next, used QBC (vote entropy and KL divergence) to label points. Used 6 committee members in our model. Used Committee model from modAL framework. First created a learner using Random Forest classifier and then apply Committee module to it. Then performed query by committee.

Max disagreement sampling was used to query the instance when disagreement is largest. Lastly applied vote entropy and KL max disagreement and compared the results.

The results are as follows:



The size of the version space was found next. For this we unused x and y were used values along with committee KL divergence from the previous results. Now using vote entropy, calculated the version space.

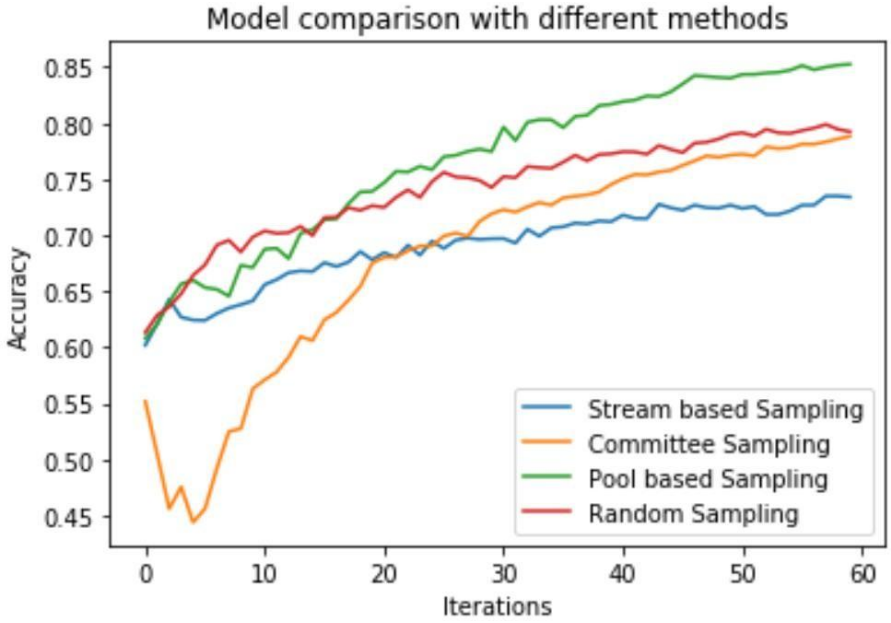
The version space that was coming out was 2478**.**

Further the points were sorted according to max entropy using the in-built sorted() function of the Python library.

To compare different models using additional data points 4 learners were created:

1. Stream-based sampling
2. Pool-based sampling
3. Committee based sampling
4. Random sampling

Looped for 20% training data and then trained dataset on all 4 models. Pool based and committee based sampling have already been discussed. For random sampling, use the same method as entropy sampling above. Used ActiveLearner model and used random_sampling module. In stream based learning, a learner receives one sample at a time and decides whether to ask for labels or not. So for implementing this, use classifier_uncertainty module. It measures uncertainty, After taking the mean of the resultant array of uncertainties, check if it is >0.3. If it is, use it for labelling, else reject it. Following is the result of the comparison between different methods:

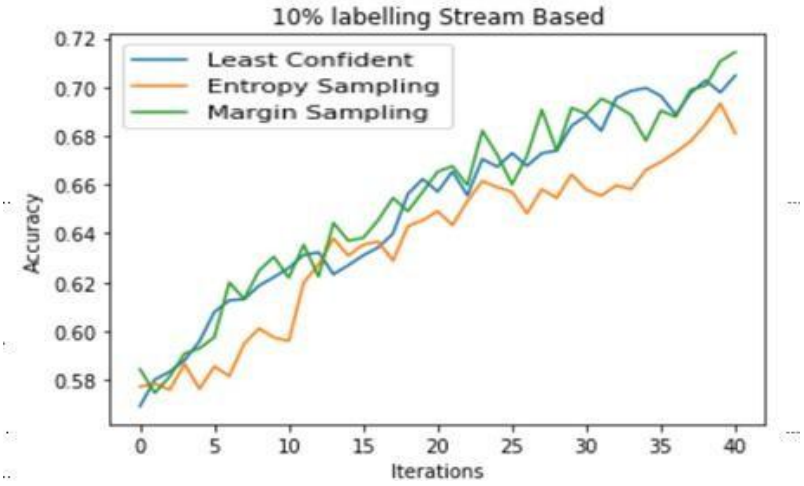


Pick 40% of the unlabelled data(90%) and use it for k-means clustering. In each cluster labelled 20% of the data. For implementing this, first used unlabelled data stored from the previous parts. Then split it into 40% training and test data. Then created a k-means clustering model with k=26. Used kMeans module from sklearn library. Then to find accuracy compared the predicted and true results. Cluster based labelling gave an accuracy of 68.132%. Total sample space was of 20,000. 90% of total space is unlabelled data,i.e.,18,000. Now 40% of 18,000 gives 7200. Now only 20% of it was labelled. 80% of it was unlabelled. Thus total unlabelled data is 5760.
Since each labelling cost was Rs.100, the total saving is Rs.57600.
Further since each labelling takes 1 hour, total time saves= 5760 hours.

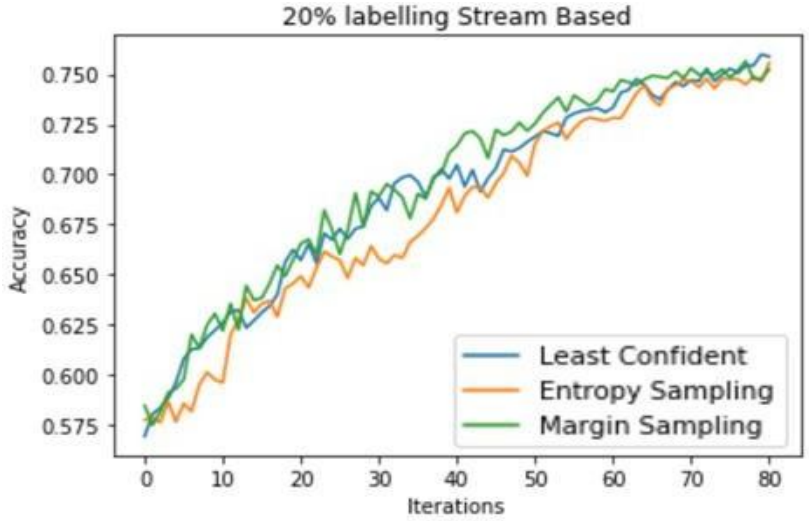
Next, implement stream-based sampling. For implementing this, used classifier_uncertainty module. It measures uncertainty, After taking the mean of the resultant array of uncertainties, checked if it is >0.3. If it is, use it for labelling, else reject it. 0.3 is the threshold that was taken. Taught the learner the labels when mean of array of uncertainties resulted in a value greater than 0.3.

Now using uncertainty sampling(least confident, margin and entropy sampling), compared these 3 models using stream-based active learning as well. Created 4 models for additional 10%, 20%, 30%, 40% points.

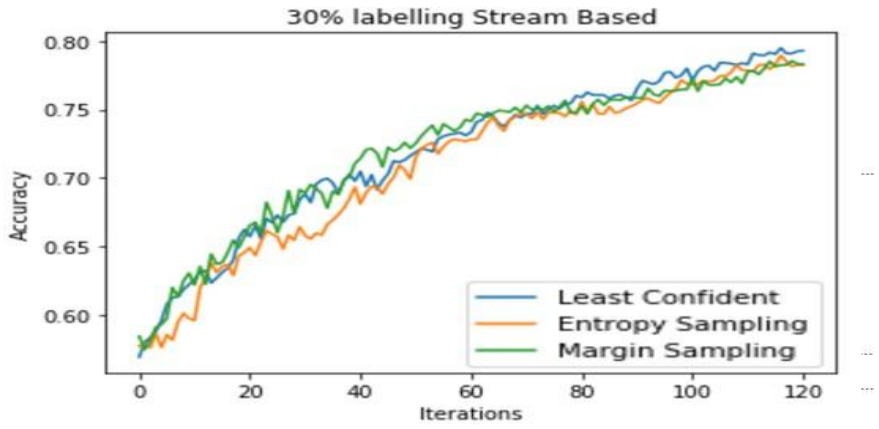
The results for 10% additional labelling using “stream-based” active learning is:



The results for 20% additional labelling using “stream-based” active learning is:



The results for 30% additional labelling using “stream-based” active learning is:



The results for 40% additional labelling using “stream-based” active learning is: