Homework 2: Active Learning + GNNs

Naomi Derel 325324994, Gili Cohen 326280815, Renana Shachak 213920010

01.08.2024

1 Active Learning

1.1 Pipeline Structure

The pipeline we used for active learning is as follows:

1. Load the dataset and split it into train and test indices. Find the features and labels from the dataset.
2. For a set number of iterations:
   - Train a model on the labeled set.
   - Use the model to predict the labels of the unlabeled set, and calculate an accuracy score.
   - Select a number of samples from the unlabeled set according to a budget, using one of the selection methods as set. Update the labeled set with the new samples.
3. Output the accuracy scores for each iteration. Alternatively, output the final trained model.

1.2 Uncertainty-Based Selection

The measure we picked for uncertainty-based selection is entropy.

The entropy of a distribution is a measure of the uncertainty in that distribution, and so it is a classical measure for our case.

We used it by calculating the entropy of the output of the model for each sample in the pool, and defining them as an estimate for the uncertainty of the model on that sample. We then selected the samples with the highest entropy to be labeled next, since they are the samples the model is most uncertain about.

1.3 Custom Selection

The measure we picked for custom selection is the density selection measure we saw in class.

the density selection is based upon the intuition that the model will benefit the most from samples that extend the vector space the model has seen so far. We used it by calculating the distance of each sample in the pool from the samples the model has seen so far, and then selecting the samples with the lowest density.

For defining the density measure, we tried two approaches (where x_i are the samples the model has seen so far, and x is the sample we are calculating the density for):

1. The lowest sum of distances from the samples the model has seen: $\sum_{i=1}^{n} distance(x_i, x)$.
2. A density measure based on the sum of exponential distances from the samples the model has seen so far: $\sum_{i=1}^{n} \exp(-distance(x_i, x))$. This is based on a Gaussian kernel, and is a common measure for density.

The first approach ended up achieving identical or better results, so we used it for the final implementation.

We then selected the samples with the highest density to be labeled next.

1.4 Parameter Comparison

We compared the results of multiple combinations of models, iterations, and budget per iteration as instructed. We defined pairs of iterations and budget per iteration to be under the limit of 600.

The following tables contain the final accuracy score for each combination. In bold, we present the best results by model, and underlined are the best results overall per selection method.

Random Selection

	SVC	Random Forest	Logistic Regression
10 Iterations + 59 Budget	**0.686**	0.677	0.675
15 Iterations + 39 Budget	0.684	0.684	**0.695**
30 Iterations + 19 Budget	0.682	**<u>0.699</u>**	0.682

Table 1: Results of Parameter Comparison for Random Selection

Uncertainty-Based Selection

	SVC	Random Forest	Logistic Regression
10 Iterations + 59 Budget	**<u>0.699</u>**	**<u>0.699</u>**	**<u>0.699</u>**
15 Iterations + 39 Budget	0.688	0.688	0.688
30 Iterations + 19 Budget	0.695	0.695	0.695

Table 2: Results of Parameter Comparison for Uncertainty-Based Selection

Custom Selection

	SVC	Random Forest	Logistic Regression
10 Iterations + 59 Budget	0.733	0.733	0.733
15 Iterations + 39 Budget	0.729	0.729	0.729
30 Iterations + 19 Budget	**0.746**	**0.746**	**0.746**

Table 3: Results of Parameter Comparison for Custom Selection

Comparison by Selection

We now compare the training process for the model which achieved the best results of each selection method. For the selections with tied results, we chose to show the Random Forest model for consistency.

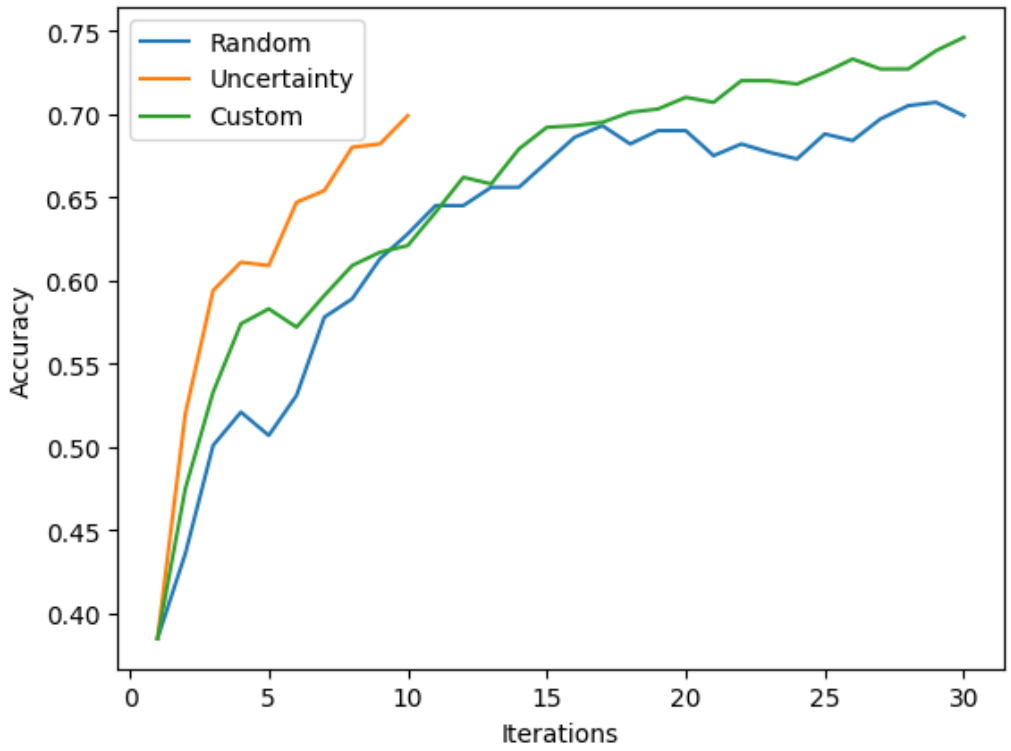

Figure 1.4.1: Comparison of Training Process for Best Results by Selection Method

Overall, we can see the custom criterion with 30 iterations and 19 budget per iteration achieved the best results, with an accuracy score of 0.746 (regardless of the model used).

Analysis and Discussion

The results show a few interesting points. Firstly, when using an advanced selection method (not random), the chosen model does not affect the results but the combination of iterations and budget per iteration does. This might mean that the models are equally good at learning from the data, but the selection method and iterations are key to the success of the active learning process.

Secondly, while the custom selection method achieved the best results with more iterations and a lower budget per iteration, the uncertainty-based selection method was directly opposite, achieving the best results with fewer iterations and a higher budget per iteration (but with a lower difference in accuracy scores). Consistently, the middle ground of the the tradeoff between iterations and budget per iteration was the worst for both selection methods.

Specifically for our custom selection method based on density, we can see that the model benefits from a lower budget per iteration, which seems to be better for an understanding of the data distribution. It benefits from more iterations, which allows the model more time to learn from the data. In contrast, the uncertainty-based selection method benefits from a higher budget per iteration, which allows the model to learn from more samples at once which might be more beneficial if the uncertainty signal is weak.

The random selection model, however, has changes in performance based both on the model used and the combination of iterations and budget. The winner among iterations per models changes with random effects, but the random forest model with most iterations seemed to consistently perform best overall. This is surprising, since overall this method has sees the lowest number of samples, but this could be due to the randomness of the selection.

Lastly, the plot of the training process for the best models per selection method shows that the uncertainty-based selection reaches better results faster (with more budget per iteration), but the custom selection method reaches better results with a more steady increase. The random method behaves more similarly to the custom method and sometimes even outperforms it during training, but eventually falls behind.