A NOTATIONS

Here is the list of notations used in this manuscript.

- $\mathcal{D}_{\text{pool}}$: The pool of unlabeled data.
- $\mathcal{D}^k_{\text{acq}}$: Acquired data at k-th AL cycle.
- $\mathcal{D}^k_{\text{train}}$: Cumulative training data after k-th cycle of AL.
- $\mathcal{D}_{\text{val}}$: Validation data.
- $\mathcal{D}_{\text{test}}$: Held-out test data.
- b: Acquisition batch size.
- K: Total number of AL cycles.
- $k = [1, 2, \dots, K]$: Index of the AL cycle.
- $[K] = [1, 2, \dots, K]$.
- $e \in \mathcal{E}$: Inherent noise (aleatoric uncertainty).
- $T \in \mathcal{T}$: Treatment variable.
- $\mathbb{E}[Y \mid X = x, do(T = t)]$: The conditional expected outcomes.
- $\hat{g}(t; \omega)$: The model parameterised by $\omega \in \Omega$ to estimate $\mathbb{E}[Y \mid X = x, do(T = t)]$.
- X: random variable X with distribution $X \sim F(X)$ and density $f(x)$.

B ACQUISITION FUNCTIONS CONT.

Coreset. Coreset acquisition looks to maximize the diversity of acquired samples. This is done by finding the data points in $\mathcal{D}^k_{\text{avail}}$ that are furthest from the labelled data points in $\mathcal{D}^{k-1}_{\text{cum}}$. The robust K-centers algorithm of Sener & Savarese (2017) approximates a solution to:

$$\alpha_{\text{CORESET}}(\widehat{g}^{k-1}(t), \mathcal{D}^k_{\text{avail}}) = \underset{\{t_1, \dots t_b\} \in \mathcal{D}^k_{\text{avail}}}{\operatorname{argmin}} \; \underset{t_i \in \mathcal{D}^k_{\text{avail}}}{\operatorname{argmax}} \; \underset{t_j \in \mathcal{D}^k_{\text{avail}} \cup \mathcal{D}^{k-1}_{\text{cum}}}{\operatorname{argmin}} \Delta(t_i, t_j). \tag{8}$$

Euclidean distances, $\Delta(t_i, t_j)$, are calculated between the output of the penultimate layer of $\widehat{g}(t; \omega)$.

Margin Sample. Margin sampling is designed for classifiers where selection is based on the distance of a sample from the classifiers decision boundary (Roth & Small, 2006). As a proxy, the difference between the predicted probability of the most and second most probable classes is used. The distance between the most probable and the second most probable classes for a multi-class classification problem can be seen as how confident the model is about the label of that class. However, The concept of a decision boundary is ill-defined for regression tasks. One option to approximate margin sampling could be to model the aleatoric uncertainty of the model by predicting the conditional variance of the outcome $\sigma^2(t; \omega)$ and select data based on the magnitude of this value. Here, we instead look at the difference in the maximum and minimum values of the predicted outcome as a measure of the model's confidence and select data based on the magnitude of this value. Formally, we have

$$\widehat{\mathcal{M}}(Y; \Omega \mid t_i, \mathcal{D}^{k-1}_{\text{cum}}) = \max_{j \in \{1, \dots m\}} (\widehat{g}(t; \omega^{k-1}_j)) - \min_{j \in \{1, \dots m\}} (\widehat{g}(t; \omega^{k-1}_j)), \tag{9}$$

and the acquisition function:

$$\alpha_{\text{Margin}}(\widehat{g}^{k-1}(t), \mathcal{D}^k_{\text{avail}}) = \underset{\{t_1, \dots t_b\} \in \mathcal{D}^k_{\text{avail}}}{\operatorname{argmax}} \sum_{i=1}^{b} \widehat{\mathcal{M}}(Y; \Omega \mid t_i, \mathcal{D}^{k-1}_{\text{cum}}). \tag{10}$$

Note that this approximation is similar to BALD under the assumption of a uniformly distributed outcome: $f(y \mid t, \omega) = \mathcal{U}(y \mid \widehat{g}(t; \omega))$.

Adversarial Basic Iteractive Method (AdvBIM). Some of the adversarial algorithms can act as active learning acquisition functions by nominating the adversarial samples. Here, we extended the famous Adversarial BIM method for our regression task as an example. BIM was introduced by (Kurakin et al., 2016) to iteratively perturb adversarial samples to maximize the cost function J subject to an l_p norm constraint as

$$\hat{t}^{(0)} = t, \hat{t}^{(i)} = \text{clip}_{t,e}(\hat{t}^{(i-1)} + \text{sign}(\nabla_{\hat{t}^{(i-1)}} J(\theta, \hat{t}^{(i-1)}, y))) \tag{11}$$

(intermediate results are clipped to stay in e-neighbourhood of the primary data point t). This technique bypasses the intractable problem of finding the distance from the decision boundary by iteratively perturbing the features until crossing the boundary (Tramèr et al., 2017). In our regression task, we perturb the features in the gradients' direction to increase the conditional variance of the outcome, i.e.,

$$\hat{t}^{(0)} = t, \hat{t}^{(i)} = \text{clip}_{t,e}(\hat{t}^{(i-1)} + \text{sign}(\nabla_{\hat{t}^{(i-1)}} \text{Var}_{\omega}(\hat{g}(t, \omega)))) \text{ for } i = \{1, \dots, m\}, \tag{12}$$

where $||\hat{t}_i - t||_2 < \gamma * ||t||_2$ with the hyperparameter γ. After creating adversarial samples for each data point in D^k_{avail}, $\alpha_{\text{AdversarialBIM}}$ acquires the samples by

$$\alpha_{\text{AdversarialBIM}}(\widehat{g}^{k-1}(t), \mathcal{D}^k_{\text{avail}}) = \bigcup_{t_i \in \mathcal{D}^k_{\text{avail}}} \underset{t_j \in \mathcal{D}^k_{\text{avail}}}{\text{argmin}} \Delta(\hat{t}_i^{(m)}, t_j), \tag{13}$$

where Δ is the euclidean distance.

k-means Sampling. This method nominates samples by returning the closest sample to each center of the unlabeled data clusters. In order to do so, one may run Kmeans++ clustering algorithm with the number of clusters equal to b over either the unlabeled data points D^k_{avail} or the output of the penultimate layer of $\widehat{g}(t; \omega)$. We refer to the former as `kmeansdata` and to the latter as `kmeansembed` in the experiments. Assuming $\{\mu_1, \dots, \mu_b\}$ are the centers of the clustering, we have

$$\alpha_{\text{Kmeans}}(\widehat{g}^{k-1}(t), \mathcal{D}^k_{\text{avail}}) = \bigcup_{i=1}^{b} \underset{t_j \in \mathcal{D}^k_{\text{avail}}}{\text{argmin}} \Delta(\mu_i, t_j), \tag{14}$$

where Δ is euclidean distance over the data points or the penultimate layer of $\widehat{g}(t; \omega)$.

C Detailed Experimental Results

C.1 Bayesian Neural Network (BNN) Model

We provide here detailed experimental results across all hyperparameter settings. The result of fig. 2 that was presented for 3 batch sizes are provided for 6 batch sizes in fig. 4. Similarly, the results of fig. 3 are provided for additional batch sizes in fig. 7. In addition, both fig. 2 and fig. 3 report the results for the STRING treatment descriptors. All experiments are repeated for two other sets of input treatment descriptors (Achilles and CCLE) whose results are provided in figs. 5, 6, 8 and 9.

C.2 Random Forest Model

In addition to the BNN model, we carried out thorough analyses for a different model class. The experiments are repeated for the random forest as an uncertainty aware ensemble model. The uncertainty in random forests, similar to other ensemble methods, is originated from the prediction made by each model instance in the ensemble. We use the random forest implementation in the Scikit-learn package (Pedregosa et al., 2011) with 100 trees and set the option max_depth=None so that the depth of the trees are determined automatically. The performance of the model trained over the active learning cycles can be seen in fig. 10 for different acquisition functions, different batch sizes, different target datasets, and the STRING treatment descriptors. Similarly, the hit ratio of the interesting genes for a random forest model is reported in fig. 12. The same experiment was repeated for CCLE treatment descriptors whose results are provided in fig. 11 and fig. 13. Notice that random forest experiments are done with a reduced set of acquisition functions that could be adjusted to the random forest model.

C.3 IN-DEPTH DESCRIPTION OF THE *Hit Ratio* EXPERIMENT

Here we elaborate more on the purpose and the message of the hit ratio experiment whose results are reported in figs. 7 to 9, 12 and 13 for various settings. The purpose of these experiments is to compare the performance of different acquisition functions in different settings of batch sizes and input/output datasets to hit the gene targets that are known to be interesting by genomics experts. To choose the set of interesting genes, we sort them based on their absolute target values. Then we choose the top 5% of this list that corresponds to both extremes of positive and negative values (both extremes are considered to be good targets by experts.) The experiments are repeated for 5 different random seeds to obtain the error bars.

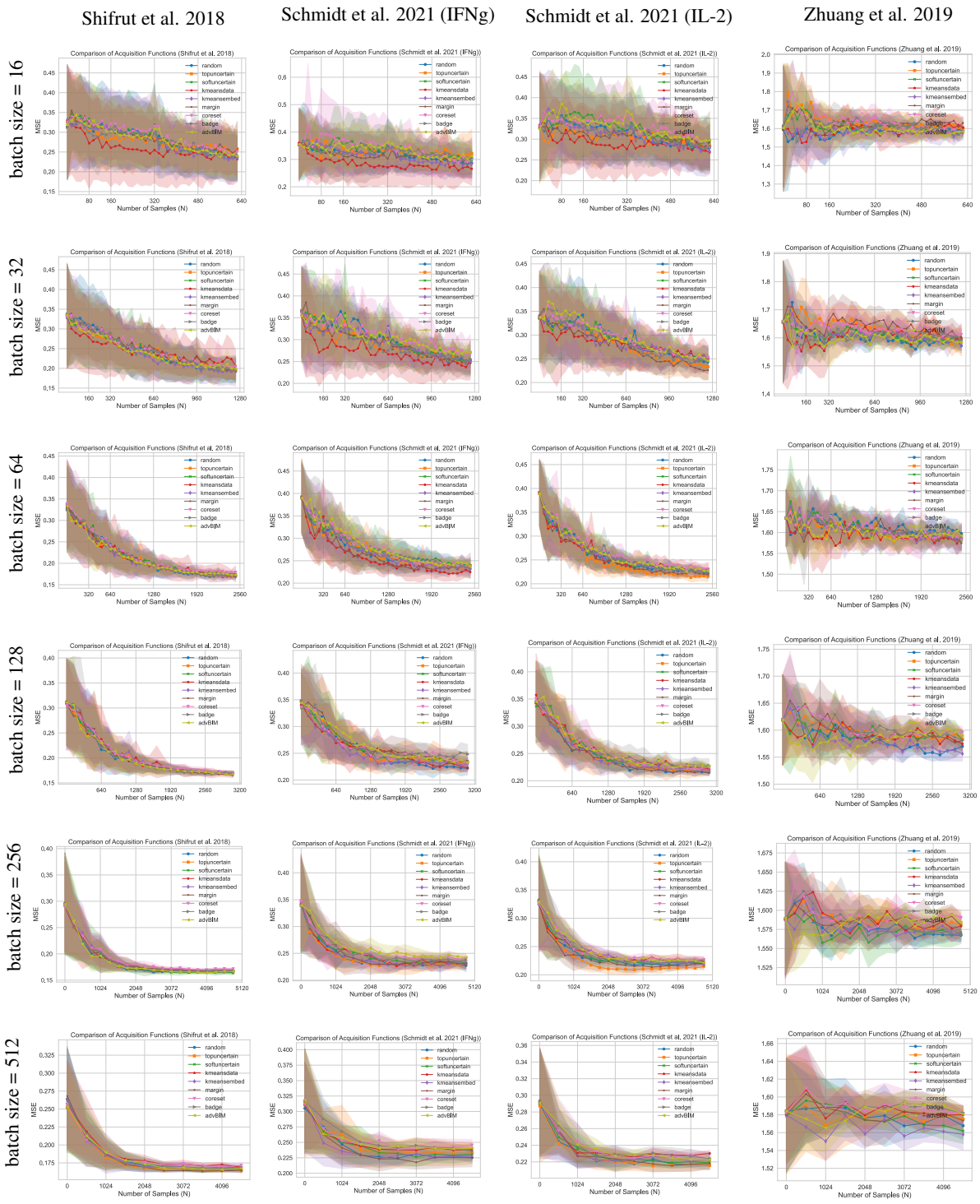

Figure 4: The evaluation of the model trained with STRING treatment descriptors at each active learning cycle for 4 datasets and 6 acquisition batch sizes. In each plot, the x-axis is the active learning cycles multiplied by the acquisition bath size that gives the total number of data points collected so far. The y-axis is the test MSE error evaluated on the test data.

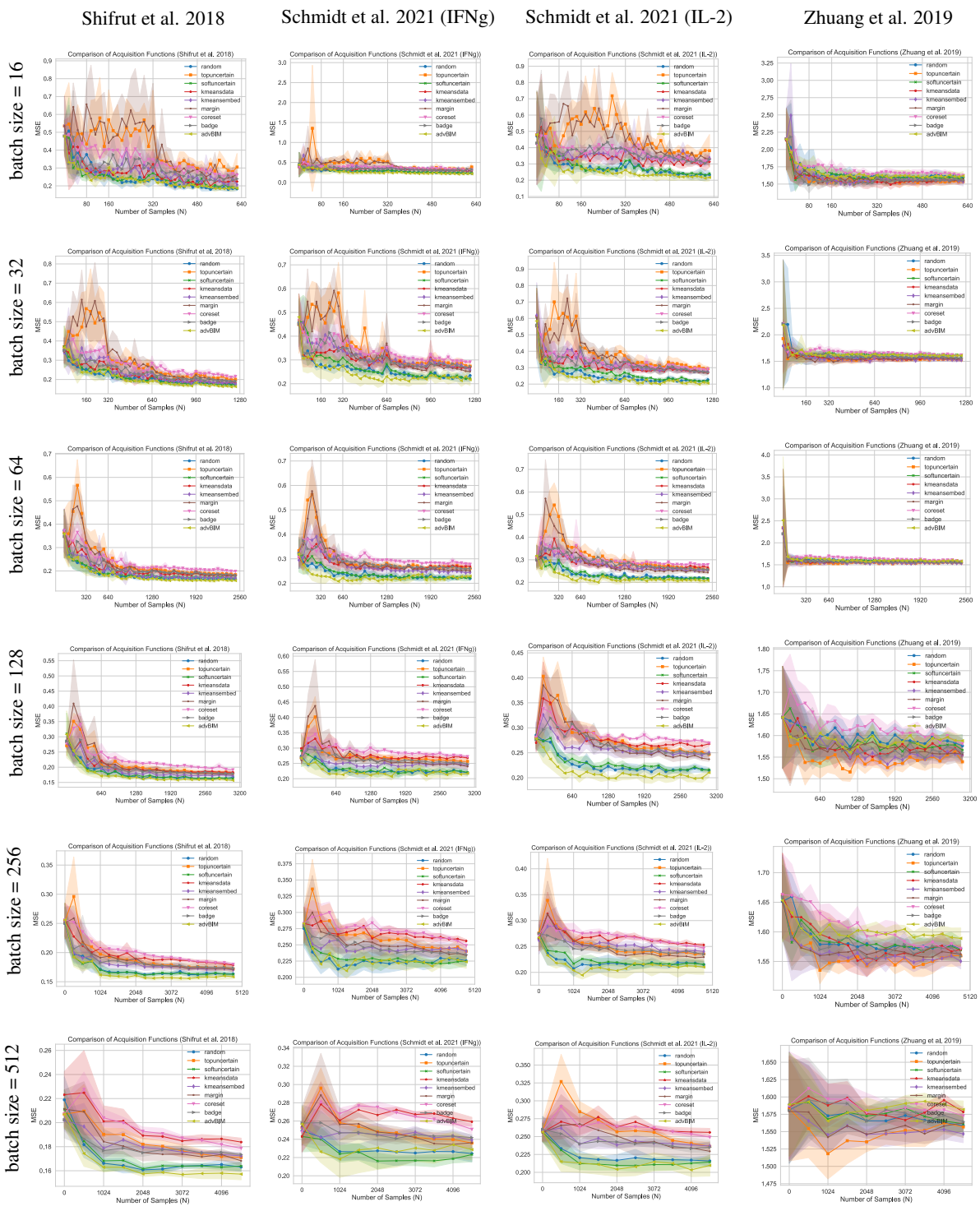

Figure 5: The evaluation of the model trained with Achilles treatment descriptors at each active learning cycle for 4 datasets and 6 acquisition batch sizes. In each plot, the x-axis is the active learning cycles multiplied by the acquisition bath size that gives the total number of data points collected so far. The y-axis is the test MSE error evaluated on the test data.

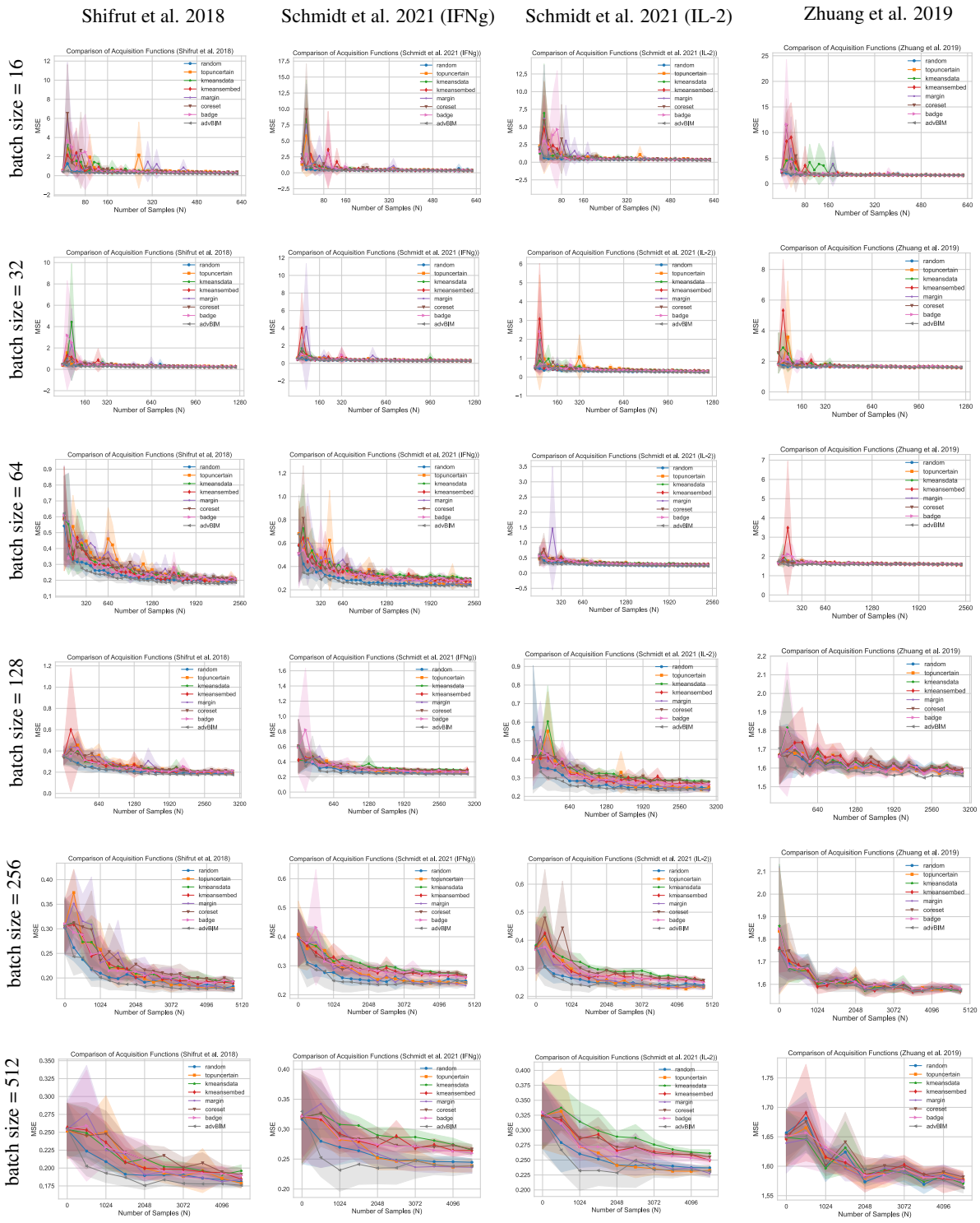

Figure 6: The evaluation of the model trained with CCLE treatment descriptors at each active learning cycle for 4 datasets and 6 acquisition batch sizes. In each plot, the x-axis is the active learning cycles multiplied by the acquisition bath size that gives the total number of data points collected so far. The y-axis is the test MSE error evaluated on the test data.

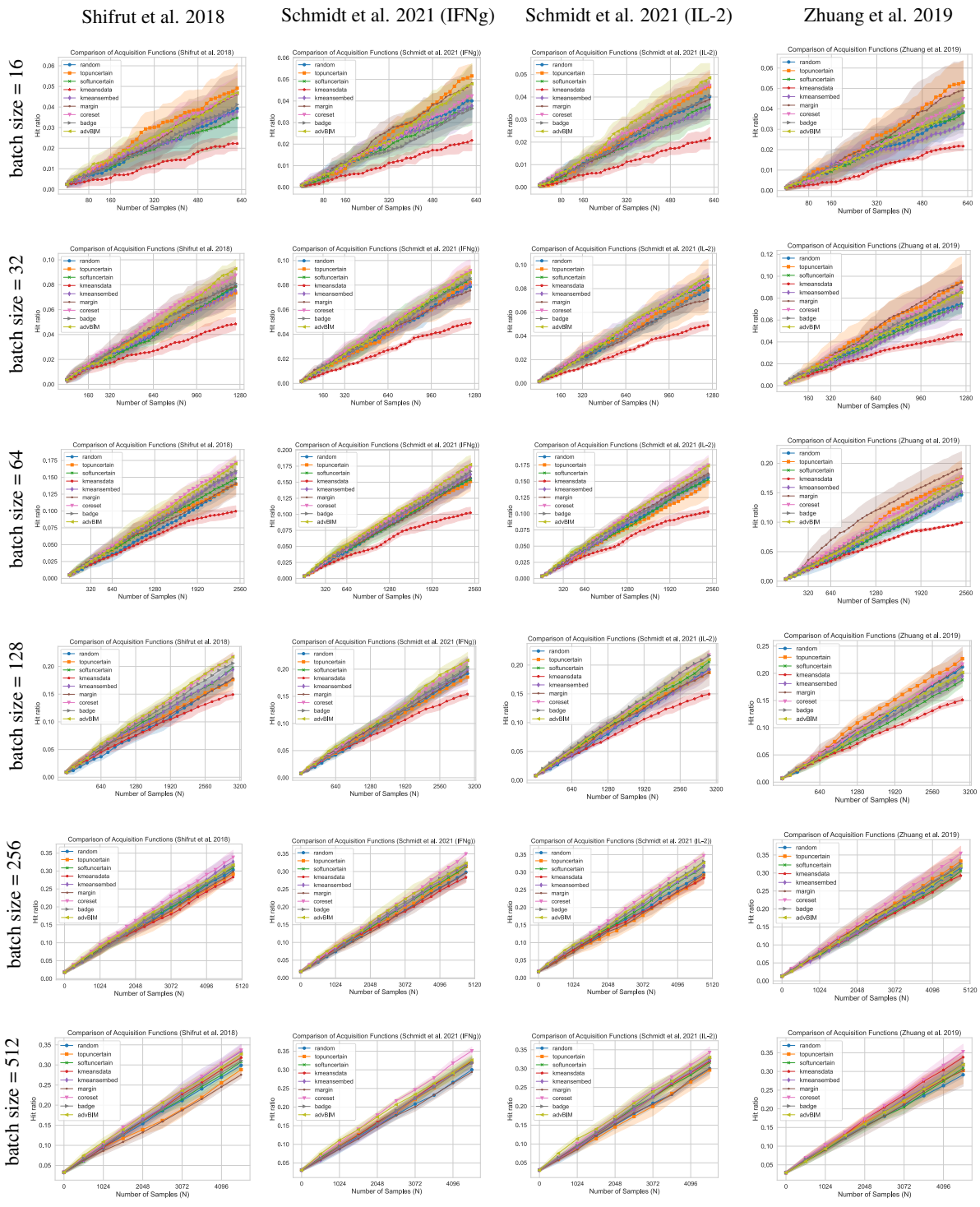

Figure 7: The hit ratio of different acquisition for BNN model, different target datasets, and different acquisition batch sizes. We use STRING treatment descriptors here. The x-axis shows the number of data points collected so far during the active learning cycles. The y-axis shows the ratio of the set of interesting genes that have been found by the acquisition function up until each cycle.

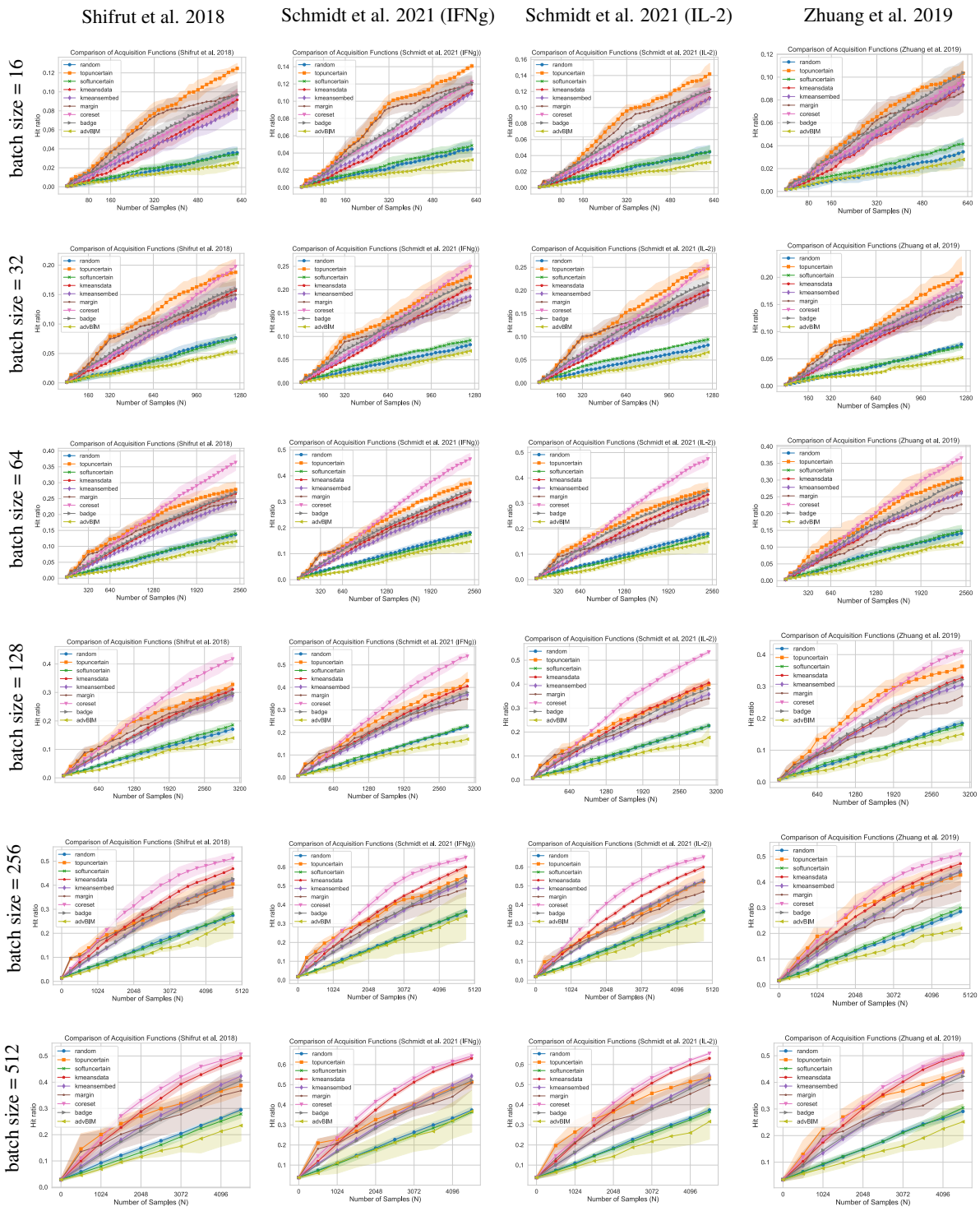

Figure 8: The hit ratio of different acquisition for BNN model, different target datasets, and different acquisition batch sizes. We use Achilles treatment descriptors here. The x-axis shows the number of data points collected so far during the active learning cycles. The y-axis shows the ratio of the set of interesting genes that have been found by the acquisition function up until each cycle.

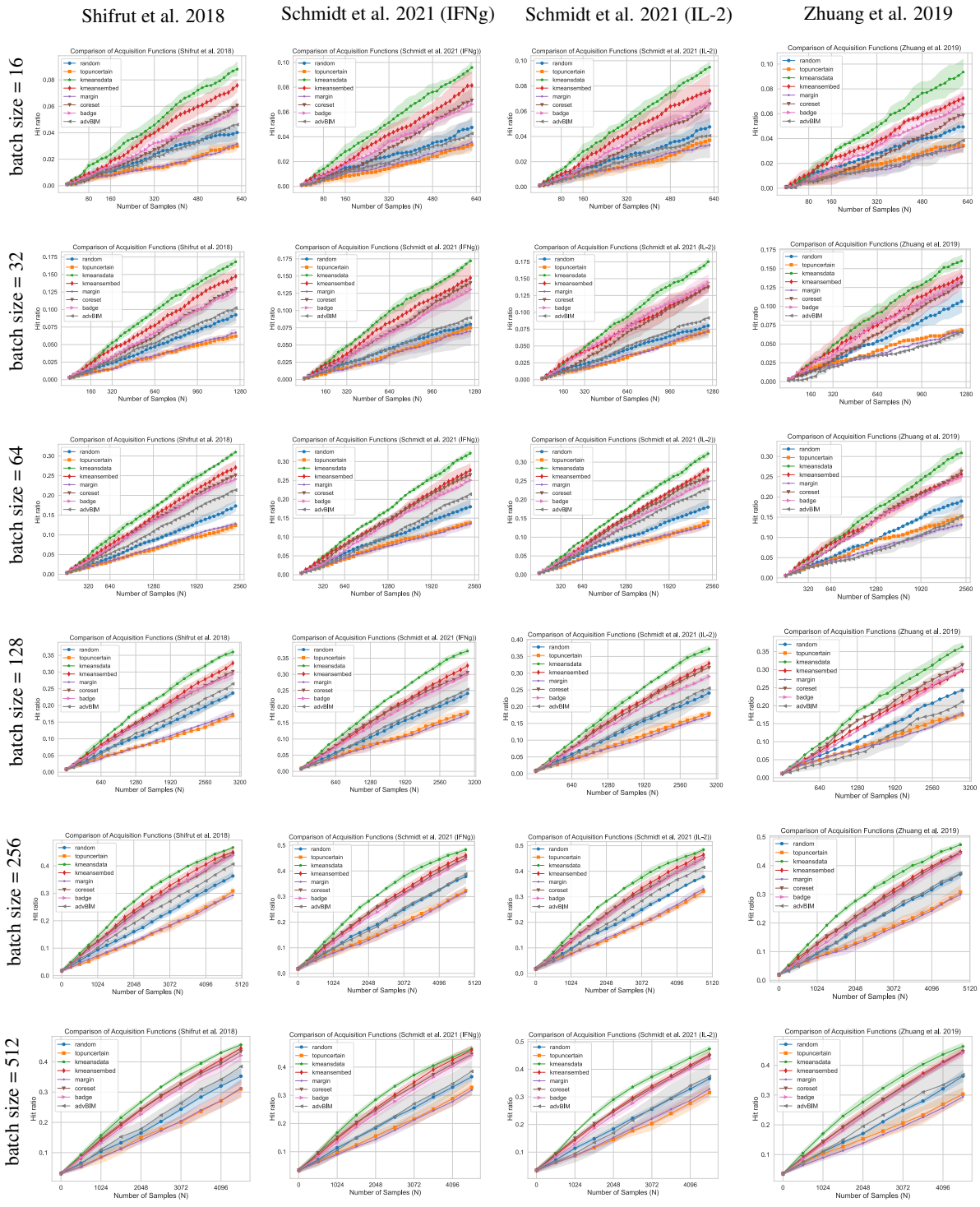

Figure 9: The hit ratio of different acquisition for BNN model, different target datasets, and different acquisition batch sizes. We use CCLE treatment descriptors here. The x-axis shows the number of data points collected so far during the active learning cycles. The y-axis shows the ratio of the set of interesting genes that have been found by the acquisition function up until each cycle.

Figure 10: The evaluation of the random forest model trained with STRING treatment descriptors at each active learning cycle for 4 datasets and 6 acquisition batch sizes. In each plot, the x-axis is the active learning cycles multiplied by the acquisition bath size that gives the total number of data points collected so far. The y-axis is the test MSE error evaluated on the test data.

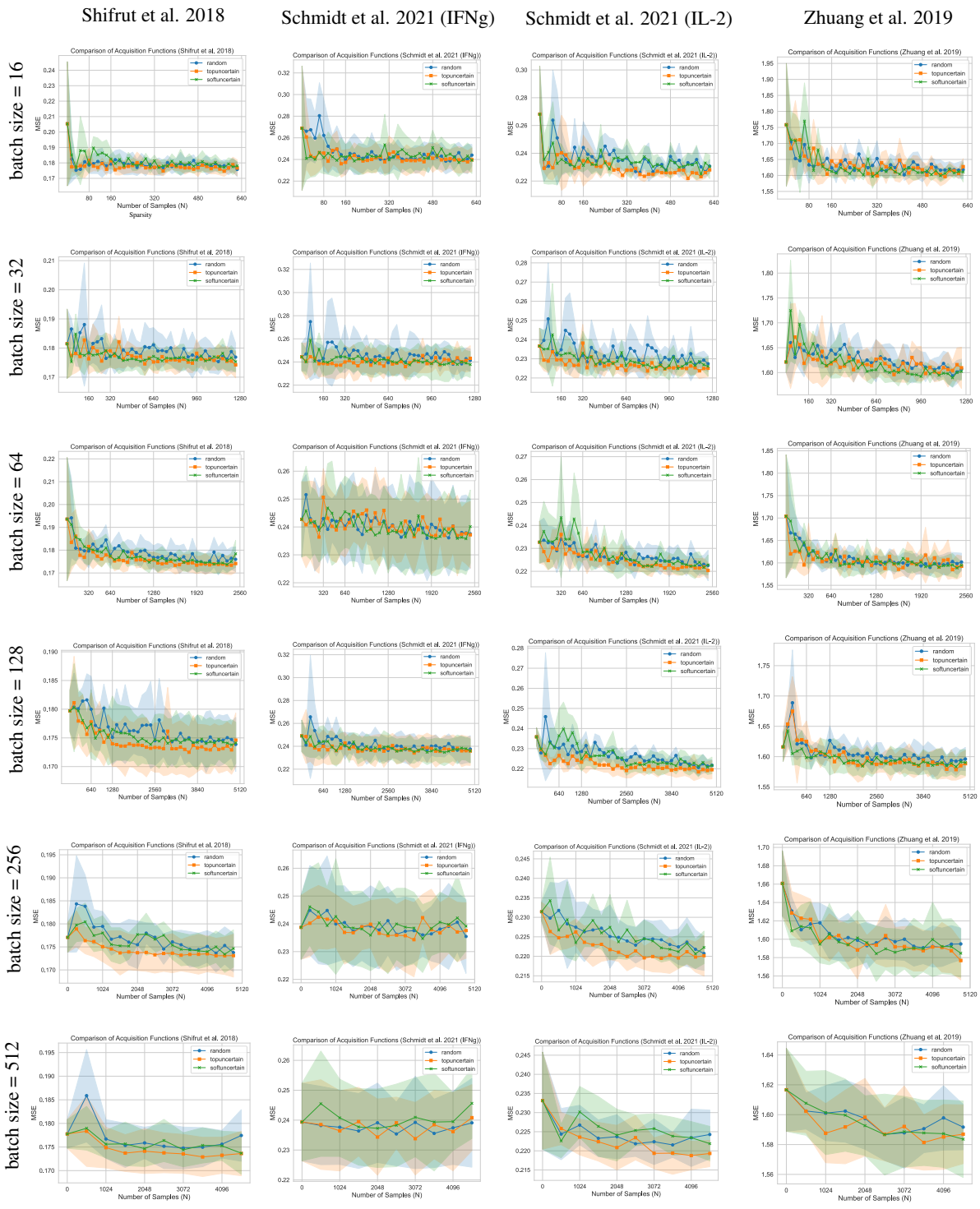

Figure 11: The evaluation of the random forest model trained with CCLE treatment descriptors at each active learning cycle for 4 datasets and 6 acquisition batch sizes. In each plot, the x-axis is the active learning cycles multiplied by the acquisition bath size that gives the total number of data points collected so far. The y-axis is the test MSE error evaluated on the test data.

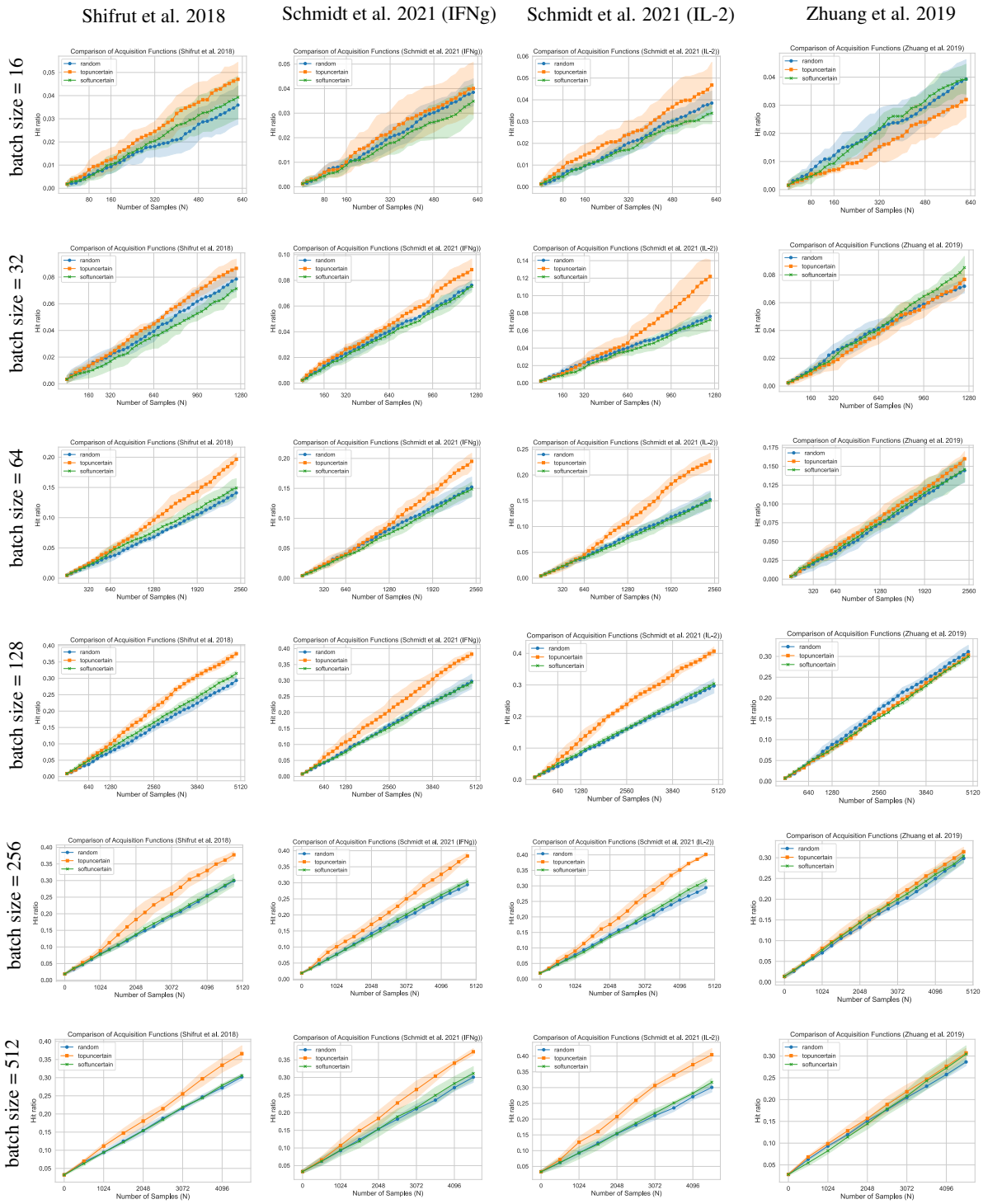

Figure 12: The hit ratio of different acquisition for random forest model, different target datasets, and different acquisition batch sizes. We use STRING treatment descriptors here. The x-axis shows the number of data points collected so far during the active learning cycles. The y-axis shows the ratio of the set of interesting genes that have been found by the acquisition function up until each cycle.

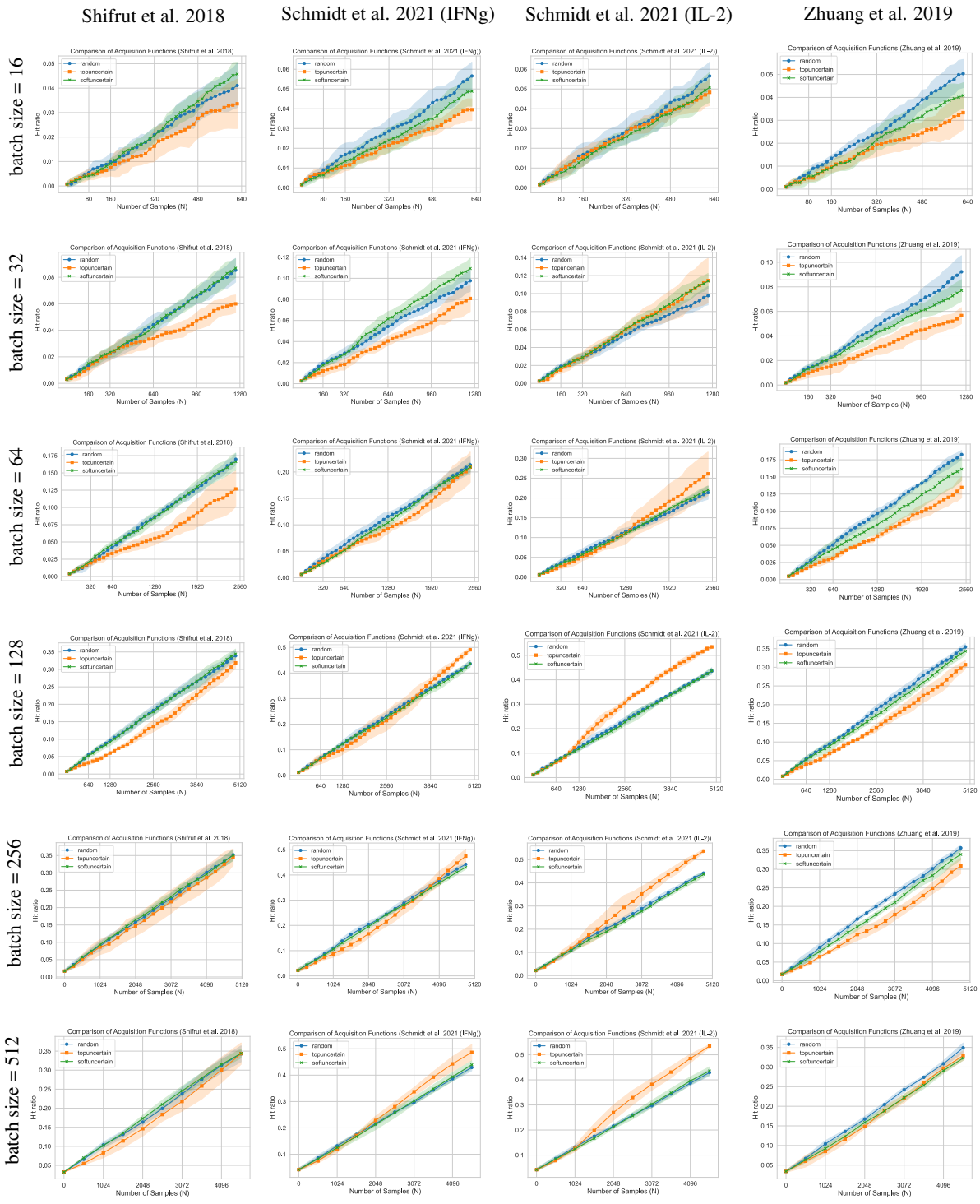

Figure 13: The hit ratio of different acquisition for random forest model, different target datasets, and different acquisition batch sizes. We use CCLE treatment descriptors here. The x-axis shows the number of data points collected so far during the active learning cycles. The y-axis shows the ratio of the set of interesting genes that have been found by the acquisition function up until each cycle.