Bayesian Active Learning by Soft Mean Objective Cost of Uncertainty: Appendix

Guang Zhao[1], **Edward R. Dougherty**[1], **Byung-Jun Yoon**[1,3], **Francis J. Alexander**[3], **Xiaoning Qian**[1,2]
[1]Electrical & Computer Engineering, [2]Computer Science & Engineering, Texas A&M University
[3]Brookhaven National Laboratory

In this supplementary file, we provide the pseudo-code of our Soft-MOCU-based active learning method together with complexity analysis in Appendix A, an illustrative example to demonstrate the problem of MOCU-based active learning in Appendix B, and more detailed descriptions of our experiments, additional results, and discussions in Appendix C & D.

A. Soft-MOCU-based active learning & computational complexity

The pseudo-code of our Soft-MOCU-based active learning method in given in Algorithm 1 with the detailed descriptions of AcquisitionFun and smocu functions. We further estimate the computational complexity of our Soft-MOCU-based active learning.

Computational complexity Given the discrete feature space with the cardinality $N_x = |\mathcal{X}|$ and the uncertainty set of classifiers with $N_\theta = |\Theta|$ different models. For active learning, there are T iterations of queries as the total budget. We study the total complexity of the active learning method. In the smocu function, line 37 is called for $O(N_x N_\theta)$ times. In AcquisitionFun, smocu is called for constant times. Finally, in the main procedure, in each iteration, AcquisitionFun is called for each $\boldsymbol{x}$. Hence, the total complexity of Soft-MOCU-based active learning is $O(TN_x^2 N_\theta)$.

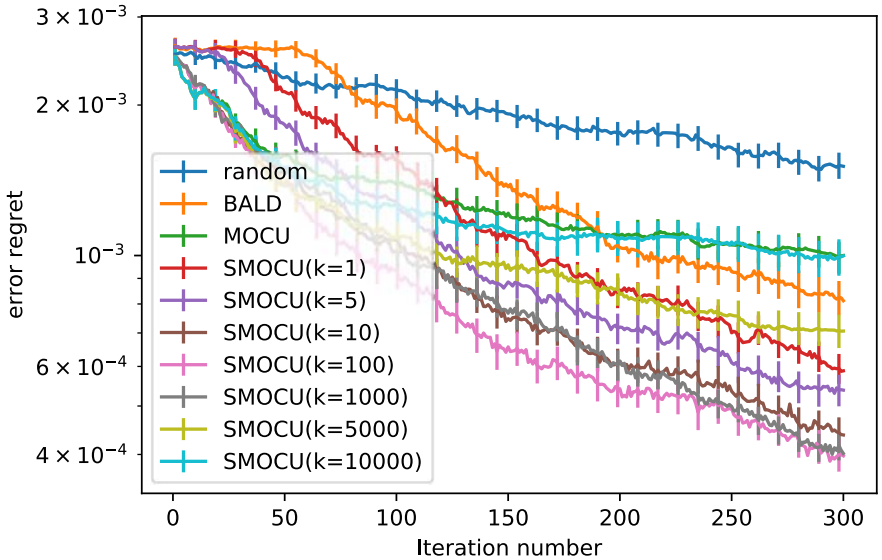


Figure S1. Comparison of the expected OBC error regret by SMOCU with different k values together with random sampling as well as MOCU- and BALD-based active learning methods.

Algorithm 1 Soft-MOCU-based active learning

function MAINPROCEDURE( )
  Set a discrete candidate set $\mathcal{X}$, the probability array p_x, and iteration number T
  Set the discrete parameter set Θ and the corresponding probability array π_θ
  Set approximation parameter k
  Initialize the data set $D = \emptyset$
  $\pi_{\theta|D} = \pi_\theta$
  for $t = 1$ to T **do**
    for x in $\mathcal{X}$ **do**
      Store ACQUISITIONFUN$(x, \pi_{\theta|D})$ to the array $U_{\mathcal{X}}$
    end for
    Optimize $U_{\mathcal{X}}$ and find the maximum point x^*
    Obtain the label y^* corresponds to x^* and update $D = D \cup \{x^*, y^*\}$
    for θ in Θ **do**
      Update $\pi_{\theta|D} \propto \pi_{\theta|D} \cdot p(y^*|x^*, \theta)$
    end for
    $\mathcal{X} = \mathcal{X}/\{x^*\}$
  end for
end function

function ACQUISITIONFUN$(x, \pi_{\theta|D})$
  $smocu_current =$SMOCU$(\pi_{\theta|D})$
  $smocu\_next = 0$
  for y in $\{0, 1, \ldots, M-1\}$ **do**
    for θ in Θ **do**
      Generate array $p(\theta, y|D, x) = \pi_{\theta|D} \cdot p(y|x, \theta)$
    end for
    $p(y|D, x) = \sum_\theta p(\theta, y|D, x)$
    $\pi_{\theta|D,x,y} = p(\theta, y|D, x)/p(y|D, x)$
    $smocu\_next = smocu\_next + p(y|D, x) \cdot$ SMOCU$(\pi_{\theta|D,x,y})$
  end for
  return $smocu\_current - smocu\_next$
end function

function SMOCU$(\pi_{\theta|D})$
  $smocu = 0$
  for x' in $\mathcal{X}$ **do**
    $bayesian\_max = 0$
    for θ in Θ **do**
      $bayesian\_max = bayesian\_max + \pi_{\theta|D} \cdot \max_{y'} p(y'|x', \theta)$
    end for
    $p(y'|D, x') = \sum_\theta \pi_{\theta|D} \cdot p(y'|x', \theta)$
    $G = bayesian\_max - \text{LogSumExp}[k \cdot p(y'|D, x')]/k$
    $smocu = smocu + p(x') \cdot G$
  end for
  return $smocu$
end function

B. A synthetic example where MOCU-based active learning gets stuck

We provide an example for intuitive illustration of how MOCU-based active learning can get stuck without converging to the true optimal classifier. Consider a binary classification problem with the uncertain class of two models $\Theta = \{\theta_1, \theta_2\}$ and the candidate pool with two candidates $\mathcal{X} = \{x_1, x_2\}$. We set the probabilistic model and the prior $\pi(\theta)$ as shown in Tables S1 and S2. In this setting, we can calculate the predictive probabilities,

which are also shown in Table S1, indicating the OBC $\psi_{\pi(\theta)}(x_1) = 1$ and $\psi_{\pi(\theta)}(x_2) = 1$. In the MOCU calculation, there is only one nonzero term $C_{\theta_2}(\psi_{\pi(\theta)}, x_2) - C_{\theta_2}(\psi_{\theta_2}, x_2)$ within the expectation in (2) in the main text. So the MOCU $\mathcal{M}(\pi(\theta)) = 0.01 > 0$, indicating the OBC has not converged to the true optimal classifier corresponding to either the underlying true model θ_1 or θ_2.

	$x = x_1$, $y = y_1$	$x = x_2$, $y = y_2$
$p(y\|x, \theta_1)$	(0.3, 0.7)	(0.4, 0.6)
$p(y\|x, \theta_2)$	(0.3, 0.7)	(0.6, 0.4)
$p(y\|x)$	(0.3, 0.7)	(0.44, 0.56)
$p(y\|x, y_1^* = 0)$	(0.3, 0.7)	(0.44, 0.56)
$p(y\|x, y_1^* = 1)$	(0.3, 0.7)	(0.44, 0.56)
$p(y\|x, y_2^* = 0)$	(0.3, 0.7)	(0.4546, 0.5454)
$p(y\|x, y_2^* = 1)$	(0.3, 0.7)	(0.4286, 0.5714)

Table S1. The probabilities of $p(y|x, \theta)$ and predictive probabilities.

	$\pi(\theta)$	$\pi(\theta\|x_1, y_1^* = 0)$	$\pi(\theta\|x_1, y_1^* = 0 \text{ or } 1)$	$\pi(\theta\|x_2, y_2^* = 0)$	$\pi(\theta\|x_2, y_2^* = 1)$
$\theta = \theta_1$	0.8	0.8	0.8	0.727	0.857
$\theta = \theta_2$	0.2	0.2	0.2	0.273	0.143

Table S2. The prior and posterior of $\pi(\theta)$.

Assume that the observation labels of x_1 and x_2 are y_1^* and y_2^* respectively. Based on the different observations of y_1^* or y_2^*, we can calculate the posterior probability of θ as shown in Table S2, and the posterior predictive probability as shown in Table S1. Since for x_1 the probability of y_1 is the same for both models, the posterior distribution of $\pi(\theta|x_1, y_1^*)$ does not change no matter what is the true label y_1^*. Therefore, querying x_1 cannot provide any information to the model and it is easy to verify from (5) in the main text that:

$$U^M(x_1, \pi(\theta)) = \mathcal{M}(\pi(\theta)) - \mathbb{E}_{p(y|\boldsymbol{x})}\mathcal{M}(\pi(\theta|x_1, y_1^*)) = 0. \tag{1}$$

On the other hand, querying y_2^* changes the posterior distribution of θ but it does not change the OBC for y_2^* being either 0 or 1, i.e. $\psi_{\pi(\theta)} = \psi_{\pi(\theta|x_2, y_2^*=0)} = \psi_{\pi(\theta|x_2, y_2^*=1)}$. As we discussed in the main text, that means $\pi(\theta)$ and $\pi(\theta|x_2, y_2^*)$ are within the same linear piece of MOCU, and in this case the acquisition function is 0. In fact, based on (6) in the main text, we have:

$$\begin{aligned} U^{\mathrm{M}}(x_2; \pi(\theta)) &= \mathbb{E}_x\{\mathbb{E}_{\pi(\theta)}[C_\theta(\psi_{\pi(\theta)}, x)]\} - \mathbb{E}_x\{\mathbb{E}_{p(y_2^*|x_2)}[\mathbb{E}_{\pi(\theta|x_2, y_2^*)}[C_\theta(\psi_{\pi(\theta|x_2, y_2^*)}, x)]]\}, \\ &= \mathbb{E}_x\{\mathbb{E}_{\pi(\theta)}[C_\theta(\psi_{\pi(\theta)}, x)]\} - \mathbb{E}_x\{\mathbb{E}_{p(y_2^*|x_2)}[\mathbb{E}_{\pi(\theta|x_2, y_2^*)}[C_\theta(\psi_{\pi(\theta)}, x)]]\} = 0, \end{aligned} \tag{2}$$

where the second line holds as $\pi(\theta) = \mathbb{E}_{p(y_2^*|x_2)}[\pi(\theta|x_2, y_2^*)]$. Therefore, although MOCU is larger than 0, the corresponding acquisition function is 0 for all the candidates. Hence, the MOCU-based active learning can get stuck in this case without identifying whether the true model is θ_1 or θ_2 to derive the true optimal classifier.

From the example above we can see that the MOCU-based method gets stuck when a single query of y_2^* does not change the label of the OBC even though it changes the posterior of $\pi(\theta)$. However, if we repetitively observe y_2^*, the posterior of $\pi(\theta_1)$ will converge to either 0 or 1 with reduced model uncertainty until identifying the true underlying model θ_1 or θ_2 and the MOCU will converge to 0. That shows the one-step-look-ahead strategy based on MOCU reduction cannot identify the long term effect of a single query. On the other hand, our Soft-MOCU-based acquisition function can capture the changes of posterior, even with small changes, due to the strict concavity of SMOCU. Therefore, SMOCU-based active learning does not have this problem and alleviates the myopic behavior as demonstrated in our experiments.

C. Additional synthetic experiments

In Figure S1, we show more results with different values of k for our Soft-MOCU-based active learning method on the example shown in Fig. 2 of the main text. From the figure, we can see as the value of k increases, the performance curve of SMOCU changes gradually, from the curve ($k = 1$) close to the performance curve of BALD, to the best performing curve ($k = 100$), then to the performance curve ($k = 10000$) close to that of

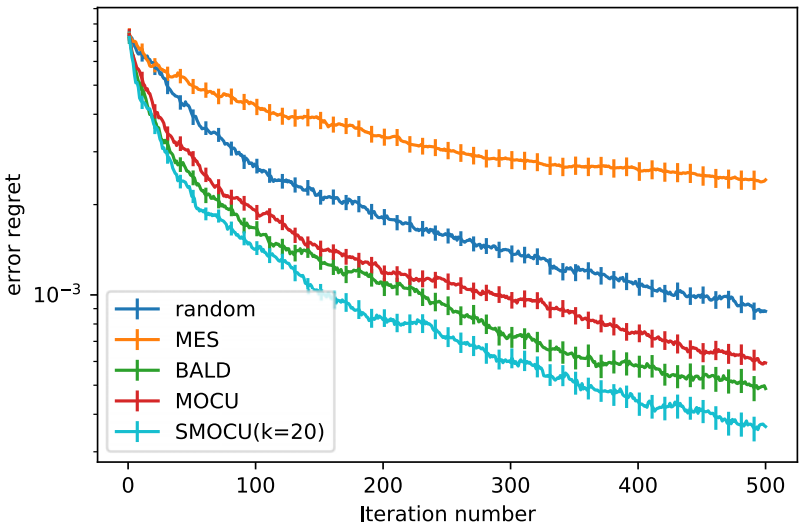


Figure S2. The expected OBC error comparison between different active learning methods on binary classification.

MOCU. The performance trends clearly show that SMOCU with appropriately chosen k values can significantly outperform the competing methods. Note that MOCU and SMOCU with large k values achieve the fastest convergence at the first 50 iterations as expected due to their local optimality. SMOCU-based methods with small k values ($k = 1$ or 5) have similar performance trends as BALD but perform better.

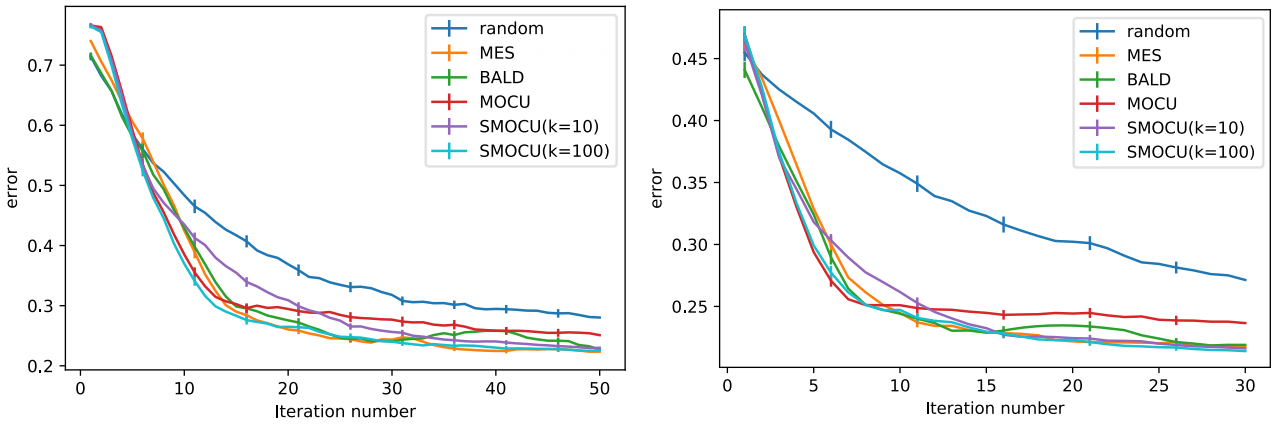
We further run the same synthetic experiment of Fig. 3 with a different prior setting: $w_1 \sim \mathcal{U}(0.3, 0.8)$, $w_2 \sim \mathcal{U}(-0.25, 0.25)$ and $b \sim \mathcal{U}(-0.25, 0.25)$. With this prior setting, the region near the x_2 axis where the decision boundary lies has similar uncertainty of $p(y|\boldsymbol{x}, \boldsymbol{w}, b)$ compared with the region far away from the x_2 axis. The results are shown in Figure S2. Under this setting, BALD performs well as expected since the model uncertainty affects the classification performance similarly, but our Soft-MOCU-based method still performs better than BALD and other competing methods.

D. Real-world benchmark experiments

We here present the complete results on the UCI User Knowledge dataset (Kahraman et al., 2013). In addition to the uncertainty class setup in the main text, we have tested two other setups of hyperparameter values. In the first setup (`Uniform prior`), uniform priors are adopted for all the 16 bins with $\boldsymbol{\alpha}^{(i)} = \mathbf{1}$, $1 \leq i \leq 16$. In the second setup (`Good prior`), we randomly choose 8 bins and set uniform priors on them with $\boldsymbol{\alpha}^{(i)} = \mathbf{1}$; for the other 8 bins, the priors are set as $\alpha_j^{(i)} = 10$ if j corresponds to the true label, and $\alpha_j^{(i)} = 1$ for the other labels. This setting is opposite to the setting in the main text.

We also randomly draw 50 samples from each class as the candidate pool and perform the same active learning methods. We repeat the whole procedure for 150 times and the averaged classification error is shown in Figure S3. In both prior settings, MOCU-based method gets stuck (degraded to random sampling) before converging to the optimal classifier, though it converges the fastest at the beginning (first 10 iterations). On the other hand, in both cases our Soft-MOCU-based method with $k = 100$ has the fast convergence both at the beginning and in the long run.

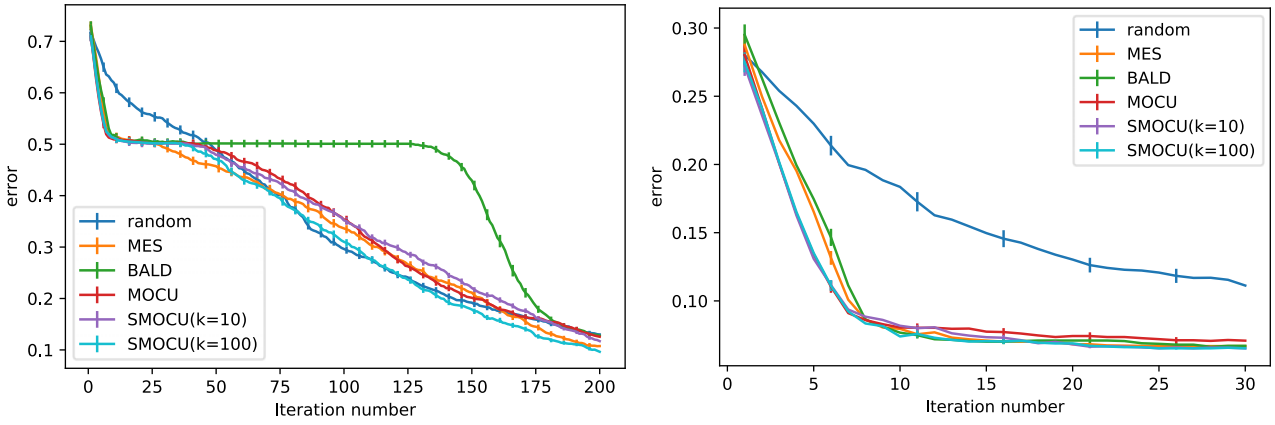
We also have made a binary classification problem from the UCI User Knowledge dataset for comparison between these different methods. We group these samples into two classes: 0: `High` or `Medium`; 1: `Low` or `Very Low`. With the same feature space and bin settings, we have tested two setups of hyperparameter values for the Dirichlet priors. The first setup (`Bad prior`) is the same as the one we have used to generate Fig. 5 of the main text, and the second setup (`Good prior`) is the same as the one adopted for Figure S3(b).



(a) `Uniform prior` (b) `Good prior`

Figure S3. Classification error rate comparison on the four-class classification problem with UCI User Knowledge dataset

We have randomly drawn 150 samples from each class as the candidate pool and perform the same comparison with different active learning methods. Again we repeat the whole procedure 150 times and the averaged classification error is shown in Figure S4. The performance trends are similar as those observed for the four-class classification problem. Our Soft-MOCU-based method with $k = 100$ has achieved better or similar performance compared with the best performing one among the competing methods.



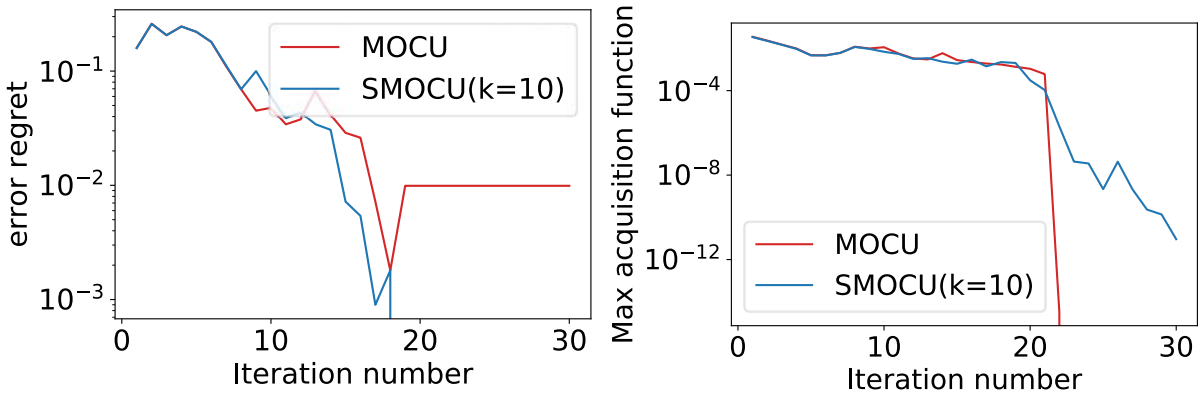
(a) `Bad prior` (b) `Good prior`

Figure S4. Classification error rate comparison on the binary classification problem with UCI User Knowledge dataset

E. Experiment illustrating that MOCU-based active learning may get stuck

In addition to the average performance comparison, we show explicitly that MOCU-based active learning can easily get stuck on a simple active learning problem while our Soft-MOCU-based active learning can find the true optimal classifier in about 20 iterations.

In this experiment, we assume a binary classification problem with a two dimensional feature space $\boldsymbol{x} = (x_1, x_2) \in [-0.5, 0.5]^2$. The probabilistic model is derived from a boundary in the feature space $f(\boldsymbol{x}) = ax_1^2 + bx_1 + c - x_2$, as $p(y = 1|\boldsymbol{x}, f) = 0.8 \times \mathbb{1}(f(\boldsymbol{x}) \geq 0) + 0.2 \times \mathbb{1}(f(\boldsymbol{x}) < 0)$. Here the quadratic function parameters (a, b, c) are uncertain and follow independent uniform distributions: $a \sim U(-4.3, 3.8)$, $b \sim U(-0.25, 0.25)$, $c \sim U(0, 1)$. Assume the true parameter values are $(a^* = -3, b^* = -1, c^* = 1.14)$. We apply MOCU- and Soft-MOCU-based methods in this setup to find the true optimal classifier. The performance of the two methods is shown in Fig. S5(a). In the figure, we can see that the error regret (the OBC error minus the true optimal classifier error) of



(a) Error regret

(b) Maximum of acquisition function

Figure S5. Comparison of MOCU and weighted MOCU on a simple active learning problem

the MOCU-based method does not change after 20 iterations, while the error regret of SMOCU reaches 0 after 18 iterations, indicating that the OBC approaches the true optimal classifier. Fig. S5(b) shows the changes of the maximum of the acquisition function during the active learning procedures. The acquisition function by MOCU decreases to 0 after 22 iterations, illustrating that MOCU-based active learning gets stuck. On the other hand, the maximum acquisition function by SMOCU is always positive to efficiently guide active learning.

References

H Tolga Kahraman, Seref Sagiroglu, and Ilhami Colak. The development of intuitive knowledge classifier and the modeling of domain dependent data. *Knowledge-Based Systems*, 37:283–295, 2013.