Response to Reviewer JexN

We sincerely thank the reviewer for the time and constructive comments. We find the comments all very helpful. We write in below responses to each of the comment. Each item starts with the original comment by the reviewer and follows with our response.

1 Responses to Comments on Weaknesses (Theory)

1. **Comment:** Regret bound is for when the true mapping $g(\theta)$, i.e. true network weights, are known and the network is not being trained. In this setting, the reward may be written as $f(x;\theta) = \sum_{i=1}^{m} \alpha_i(\theta)\boldsymbol{p}_i(x)$ where $\boldsymbol{p}_i$ are described by Equation (4) and $\alpha_i(\theta)$ are known. Therefore, Theorem 4.2 is a straightforward extension of [Srinivas et al. 2010] and [Krause Ong 2011]. The main challenge and the interesting part of this problem, would have been to propagate the error of the trained network in estimating $g(\theta)$ into the regret guarantee. Or to deal with the fact that the full kernel $\tilde{\mathcal{K}}$ is changing with t when g is not known since (then the network is being trained and) the weights are being updated at every step t.

   Response: We thank the reviewer for the constructive comment. We admit that the theoretical results are consistent with the NN-AGP-UCB algorithm described in **Algorithm 1**, where the error brought by neural networks is not considered in the acquisition function. Therefore, the assumption that the neural network is known is required.

   On the other hand, taking the neural network training error into consideration is an insightful suggestion and we really appreciate the suggestion. Although this is a challenging question, we provide a potential methodology to address it in the revised version or future work. The potential methodology requires 1) regarding the reward function as an element from a reproducing kernel Hilbert space while the associated kernel function is changing in different iterations; 2) "slightly" increasing the layers of nodes of the neural network as the data set is becoming larger; 3) enlarging the exploration term in the acquisition function of NN-AGP-UCB. In this way, the upper bound of the cumulative regret will include the term that results from the enlarged exploration term. This potential strategy is inspired by the universal approximation theorem of neural networks, misspecified Gaussian process bandit in Bogunovic and Krause (2021) and the martingale technologies in Chowdhury and Gopalan (2017).

2. **Comment:** It is required that the context set Θ and the action set $\mathcal{X}$ to be convex. Moreover, it is required that the scalar-valued kernel functions are smooth. As far as I know, in the kernelized contextual bandit literature, these assumptions are not necessary and can be lifted. I am guessing that these are required by theorem 4.2 since you are using a discretization argument to account for a compact (infinite) domain and then apply a rough union bound. This proof route is quite costly. Instead of this, you can use the martingale technique (or directly the confidence bound lemma) presented in [Chowdhury and Gopalan 2017] which lifts both of the assumptions on kernel and domain and even yields (slightly) better regret rates.

   Response: We thank the reviewer for the helpful comment. The martingale technique proposed in Chowdhury and Gopalan (2017) is a crucial contribution to the Gaussian process bandit problem. On the other hand, the regret analysis associated with IGP-UCB in their work assumes that the function is from an RKHS, while we assume that the function is a realization of a GP. We take this Bayesian view of the unknown function since it helps us better understand the construction of the acquisition function, which facilitates the generalization to other algorithms. We will include a discussion on the frequentist setting where NN-AGP is from an RKHS and applies the martingale technique to derive results as suggested.

3. **Comment:** Theorem 4.5 is a straightforward corollary of [Vakili et al. 2021], applied to kernels that take the form $k(x, \theta, x', \theta') = k_1(x, x')\, k_2(\theta, \theta')$. I can't recognize a new ingredient or original contribution there.

   Response: We thank the reviewer for the valuable comment. Our analysis of the maximal information gain builds upon the methodology proposed in Vakili et al. (2021). On the other hand, their methodology cannot be directly applied to a kernel that is a product of two general kernels that may both adopt infinite Mercer decomposition. Specifically, we show that the kernel function regarding the contextual variable θ is finite-rank and employs a finite Mercer decomposition. In this way, we can then rank the eigenvalues of the joint kernel $\tilde{\mathcal{K}}$ and derive Theorem 4.5. We agree that our result is largely built on Vakili et al. (2021) but hope to use this opportunity to point out the difference.

2 Responses to Comments on Weaknesses (Experiment)

1. **Comment:** The pricing experiment which has the most complex context (a 5-node graph) does not show good results in terms of how fast the cumulative reward grows.

   Response: We thank the reviewer for the constructive comment. While our experiments have shown that NN-AGP-UCB did not achieve outstanding results, it still outperformed the baseline CGP-UCB with various kernel selections. It is possible that the overparametrization of the GCN and simple reward functions may have contributed to these results, particularly given the small size of the 5-node graph used in our experiments. As such, we plan to conduct further experiments to explore the practicality and potential superiority of our approach.

2. **Comment:** One experiment in the appendix looks into how well the method scales with context dimension. This experiment uses the reward $f(x, \theta) = \sin \|x\|_2 + \cos \|\theta\|_2$, and shows that the algorithm scales well from a 3-dimensional θ to a 9-dimensional θ. (The plots for this experiment seem to be for a single run?) Another issue I had is that the problem setting is particularly interesting when $m \ll d'$, however in the experiments we seem to have $m = d' \pm 2$.

   Response: We thank the reviewer for the valuable comment. The plots for the experiments that are contained in Appendix are based on running 15 times of experiments. We did not contain the error bar since we would like to focus on the regret performance. We will contain the error bar in the revised version. In addition, we also conduct additional experiments with high-dimensional contextual variables. We consider that the observed contextual variable θ are randomly selected with equal probability from $\Theta = [-1/2, 1/2]^{50}$. That is $d' = 50$. Meanwhile, we select the reward function

   $$R_3(\mathbf{x}, \theta) = \sin\left(\|\mathbf{x}\|_2\right) \left|\cos\left(\|\theta_{\text{eff}}\|_2\right)\right|$$

   as in Appendix. Here, $\mathbf{x} \in [-\sqrt{2}, \sqrt{2}]^2$ and θ_{eff} denotes the first 20 dimensions of θ. That is, the remaining 30 dimensions of θ will not affect the reward function while the user does not know. The experimental results are contained in this link. The results on cumulative regrets indicate the superiority of our approach in high-dimensional scenarios, especially when the objective function adopts a sparse structure.

3. **Comment:** Issue may be that, the neural network will probably need more samples to learn an accurate mapping. One thing you can try is warm-starting the network, using a dataset from a similar problem.

   Response: We thank the reviewer for the constructive comment. Learning neural networks from insufficient data is challenging and this is the reason why NN-AGP-UCB may not achieve satisfactory results during the initial iterations. We mentioned in Section 6 that we will explore transfer learning technologies with NN-AGP between different but similar tasks, which is in line with your suggestion.

3 Responses to Comments on Questions (Problem Setting/ Theory)

1. **Comment:** Does equation (4) hold for MGPs with any kernel $\mathcal{K}$, or is it a model that you pick? What is Q ? Are $a_{l,q}$ assumed to be random or known weights? I suggest mentioning in the problem setting section that you only consider certain kinds of kernels (that adhere to Equation 4) to make the assumptions/limitations more clear.

   Response: We thank the reviewer for the constructive comment. Equation (4) represents the MGP model we specifically select in this work, which is consistent with the prevailing model selection of multi-output GP (Nguyen et al. 2014, Liu et al. 2018). This separate structure supports the analysis of both the regret bound and the information gain. On the other hand, we will consider other structures of $\mathbf{p}(\mathbf{x})$, for example, when $\mathbf{p}(\mathbf{x})$ is a convolutional MGP, where the summation of kernels in our manuscript is replaced by an integral of kernels; see Alvarez and Lawrence (2011). Here Q is a user-selected hyperparameter that determines the number of scalar GPs in MGP. A larger Q results in a more flexible MGP while bringing more computational burden. A suggested selection of Q is $\lceil d/3 \rceil$, where d is the dimension of decision variables. $a_{l,q}$ are deterministic hyperparameters that are learned in the first 20 iterations to attain surrogates, along with the neural network parameters. They will then be fixed in the remaining iterations. We will clarify these details in the revised version.

2. **Comment:** I did not understand the purpose of lines [229-248], particularly Proposition 4.1 and the explanations after. Why is the reward evaluated at a fixed point relevant and what does it mean to be closer to a deterministic function than a GP? Indeed for a fixed $x^\star$ or with a known $\boldsymbol{p}$, your reward model becomes a linear m-dimensional function of θ which can be learned from m (diverse enough) samples. But how is this justifying the NN-AGP setup? I suggest completely removing these lines, or re-writing from scratch.

   Response: We thank the reviewer for the valuable suggestion. Since NN-AGP is a joint GP with kernel function $\tilde{\mathcal{K}}$, we explain the difference between NN-AGP and a composite GP with additive or multiplicative kernels that are employed in the relevant literature. When fixing an $\mathbf{x}^*$, $\mathbf{p}(\mathbf{x}^*)$ reduces to an m-dimensional normal distribution based on the Gaussian assumption. In this way, the vector-valued random variable in Proposition 4.1 is n different linear transformations of a normal distribution and is therefore normal as well. The covariance matrix K^- is at most rank m, which means that when we have m observations (without noise) associated with $\mathbf{x}^*$ and exactly know $g(\theta)$, we then have the full knowledge of $f(\mathbf{x}^*;\theta), \forall\theta \in \Theta$. That is the reason why we said that $f(\mathbf{x}^*;\theta)$ is closer to a deterministic function. In comparison, for a composite GP with additive or multiplicative kernels, the objective function with a fixed decision variable remains a GP regarding the contextual variable.

   This difference is in line with the difference between the decision variables and contextual variables. Specifically, since decision variables are user-selected, we require a GP model to address the exploration-exploitation trade-off. On the other hand, the contextual variables are observed and cannot be decided, so we need an expressive deterministic mapping to capture how the objective function relies on θ. This difference motivates us to construct NN-AGP. We apologize for the unclarity of Proposition 4.1 and we will consider removing or revising it.

3. **Comment:** In Algorithm 1: Do you require a prior distribution over the parameters, or just an initial value? Seems like you only need to initialize these, since they are then estimated by maximizing loglikelihood, rather than calculating the posterior distribution given a prior. So I would suggest to make the input to the algorithm more clear.

   Response: We thank the reviewer for the valuable suggestion. These parameters are initial values that are updated by maximizing the likelihood function. We will revise it.

4 Responses to Comments on Questions (Experiments)

1. **Comment:** In the sensitivity experiment, it looks like that NN-AGP-UCB is rather sensitive to loss function, since Figure 5 shows a relatively worse regret than Figure 6. How do you conclude from this experiment that NN-AGP-UCB is not sensitive to reward? Also, are these average of multiple runs or a single run?

   Response: We thank the reviewer for the constructive comment. In Figure 5, we present the experiments on a multiplicative function $R_3(\mathbf{x}, \theta) = \sin(\|\mathbf{x}\|_2) |\cos(\|\theta\|_2)|$ and the result indicates that CGP-UCB with multiplicative kernels significantly outperforms CGP-UCB with additive kernels. On the other hand, we present the experiments on an additive function $R_4(\mathbf{x}, \theta) = \sin(\|\mathbf{x}\|_2) + \cos(\|\theta\|_2)$ and the result indicates that CGP-UCB with additive kernels outperforms CGP-UCB with multiplicative kernels in the first half iterations and achieves comparable results in the final stage. That is, the performance of CGP-UCB is sensitive to whether the structure of the reward function is consistent with the selected structure of joint kernels. In comparison, NN-AGP-UCB does not need to pre-specify a joint kernel and outperforms CGP-UCB in both structures of objective functions. In addition, these results are based on repeating the experiments 15 times and we will include the error bar in the revised version.

2. **Comment:** What's the reward function in section 5.3? Sum of $Y_i(t)$ as defined in appendix?

   Response: We thank the reviewer for the comment. In this experiment, we consider maximizing the total profit brought by users' service adoption in the network, and therefore the reward function is

   $$f(\mathbf{x}_t; \theta) = \mathbf{x}_t \times \mathbb{E}\left[\sum_i Y_i(t; \theta_t)\right],$$

   where x_t denotes the price of the service. An increase in prices is likely to have a negative impact on the adoption rate of service. We will clarify this reward function in the revised version.

3. **Comment:** Just curious, is there a reason why NN-UCB isn't benchmarked in Figure 2?

   Response: We thank the reviewer for the valuable comment. We apologize for the negligence and we include the experimental results in this online link. In our experiment, NN-UCB outperforms CGP-UCB while our NN-AGP-UCB achieves the best performance with the lowest cumulative regret.

   Besides, we will take the suggestions on assumption clarification, more cohesive notation, and revise the mentioned typos. We will also cite Zhu et al. (2022) to support the model selection of NN-AGP.

Again, we appreciate the valuable comments provided by the reviewer. We will make the necessary revisions to the manuscript based on your suggestions.

References

Alvarez, M. A. and Lawrence, N. D. (2011). Computationally efficient convolved multiple output gaussian processes. *The Journal of Machine Learning Research*, 12:1459–1500.

Bogunovic, I. and Krause, A. (2021). Misspecified gaussian process bandit optimization. *Advances in Neural Information Processing Systems*, 34:3004–3015.

Chowdhury, S. R. and Gopalan, A. (2017). On kernelized multi-armed bandits. In *International Conference on Machine Learning*, pages 844–853. PMLR.

Liu, H., Cai, J., and Ong, Y.-S. (2018). Remarks on multi-output gaussian process regression. *Knowledge-Based Systems*, 144:102–121.

Nguyen, T. V., Bonilla, E. V., et al. (2014). Collaborative multi-output gaussian processes. In *UAI*, pages 643–652. Citeseer.

Vakili, S., Khezeli, K., and Picheny, V. (2021). On information gain and regret bounds in gaussian process bandits. In *International Conference on Artificial Intelligence and Statistics*, pages 82–90. PMLR.

Zhu, Y., Foster, D. J., Langford, J., and Mineiro, P. (2022). Contextual bandits with large action spaces: Made practical. In *International Conference on Machine Learning*, pages 27428–27453. PMLR.