We thank all the reviewers for their insightful and encouraging feedback.

Initial points of acquisition function optimization Following the implementation details of Spearmint [1], COMBO adopts the idea of *spray points* [2] to promote exploitation, which is the same method as in the reference pointed by **R2**. Due to the discrete nature of COMBO's search space, the implementation detail is slightly different. In Spearmint, spray points are points around the best-evaluation point, after perturbing by a zero-mean Gaussian with user-specified variance (e.g 0.001^2). In contrast, in COMBO's combinatorial graphs, we have spray vertices. Spray vertices are randomly chosen vertices neighboring the best evaluated vertex, such that the shortest path distance to the best vertex is less than or equal to a user-specified distance (e.g, 2). As **R2** suggested, this heuristic promotes exploitation.

Using random vertices for exploration is similar to Spearmint. In Spearmint we use Sobol sequences, whereas in COMBO random vertices are chosen uniformly. For initial points for the acquisition function optimization, we probe acquisition values on 20,020 points consisted of 20 spray vertices and 20,000 random vertices, similar to Spearmint. The best 20 from 20,020 points are used as initial points for further optimization. In the camera-ready version we will give the full array of details regarding the exploitation-exploration trade-off, including the references pointed by **R2**.

Computational complexity In addition to the surrogate model fitting and acquisition function optimization in ordinary Bayesian optimization methods, COMBO has a one-off pre-processing step of eigendecomposition to compute Fourier basis. The complexity of the one-off eigendecomposition is *linear* with respect to the number of variables.

Surrogate model fitting. 1-dimensional slice sampling is used with the Gibbs sampler. The entire sampling procedure has $O(k^2d^2) + O(kd^3)$ complexity. Specifically, the 1-dimensional slice sampling requires *(a)* $O(kd^2)$ multiplications for k variables and d evaluations and *(b)* solving a linear system of cubic complexity $O(d^3)$. The $O(kd^2)$ multiplications, specifically, are due to the Kronecker product kernel that the diffusion kernel yields on a Graph Cartesian product.

Acquisition function optimization. We observe that the heuristic of local optimization with initial points selected from the pool of spray vertices and random vertices (similar to the description given by **R2**) works better than more complex solutions with simulated annealing as stated in the paper. We leave the trade-off between optimization and efficiency of other methods, such as, evolutionary search and genetic algorithm as future work.

Error measure The error measure in the experiment is the standard error. The evaluations in all experiments except NAS are noise-free. Thus, small yet still different from 0 standard errors indicate different optima.

Future works and limitations. COMBO currently handles continuous variables only if they are discretized. Mixing combinatorial and continues variables is challenging for Bayesian Optimization, not only for designing an appropriate surrogate model but also for the optimization of the acquisition function.

Besides addressing the aforementioned limitations, bridging the well-established graph theory with combinatorial optimization opens up lots of interesting research directions (**R1**). Examples are irregular structures not amenable to the graph Cartesian product or even learning the graph structure. Further, exploring problems, like joint neural architecture and hyperparameter search, traditionally dominated by low sample efficiency methods (*e.g.,* evolutionary search) is of great interest: the superior uncertainty quantification of the Bayesian Optimization surrogate model, like Gaussian Processes, would bring great advantages.

As pointed out by reviewers, we hope the graph theory framework on combinatorial optimization introduced by COMBO will attract attention to the under-explored problem of combinatorial Bayesian optimization. We appreciate the meticulous comments from all reviewers on the structure, presentation and typos. We will include all the clarifications in the camera-ready version and release all code, experiments and results upon acceptance.

[1] J. Snoek, H. Larochelle, R.P. Adams. Practical Bayesian Optimization of Machine Learning Algorithms. NeurIPS 2012. (`https://github.com/JasperSnoek/spearmint`)

[2] `https://github.com/JasperSnoek/spearmint/blob/b37a541be1ea035f82c7c82bbd93f5b4320e7d91/spearmint/spearmint/chooser/GPEIOptChooser.py#L235`