We thank the authors for their careful reading of the paper. Below we repeat or paraphrase the reviewers' comments and questions and then offer our responses.

R1: Computational complexity, especially in light of integer linear programming. Bag pairing does indeed reduce to ILP. In the case of bags of equal size (the setting of our experiments), we show in the supplemental that the optimal algorithm is extremely simple: Pair the bags with highest and lowest LPs, the bags with next highest and next lowest, etc. For unequal bag sizes, this no longer holds, but there is a literature on scalable approximate algorithms for the weighted matching problem. We'll reference this in the final version.

R1: Tightness of bounds. While we have not studied this issue in depth, we anticipate that there are certain scenarios (e.g., worst case ones) where our bounds are tight, similar to conventional Rademacher complexity. In more typical scenarios, we expect that the bounds could be improved, e.g., by an appropriate analogue of local Rademacher complexity. However, we chose Rademacher complexity because we found that it led to a tractable upper bound that we were able to explicitly optimize to determine the weights for the different bag pairs.

R1: Distribution of LPs: This is a design choice in setting up the experiment. The user specifies the distribution of LPs, and we chose ours to be uniform on the given intervals.

R4: Restrictive assumptions (cf conditional independence): Under both our models (IIM and IBM), we allow the distribution of an instance to be dependent on which bag it is drawn from. Furthermore, under IBM, the instances within a bag (conditioned on the bag label proportion) can have arbitrary dependency structure. The literature on LLP frequently asserts the importance of these two settings, and our paper is the first to provide theoretical analysis wrt a classification performance measure under these assumptions. We have made every effort to make the assumptions as general as possible, and the assumptions needed for our results are indicated in our theorem statements.

R4: Generalisation bounds not formulated in terms of excess of risk. The term "generalization error bound" commonly refers to a bound on the deviation between true and empirical values of a performance measure. Given such a bound, there are standard arguments that yield consistency wrt the performance measure of interest, in our case BER. The reviewer may be asking about a "calibration" type excess risk bound. Such a bound would relate the excess risk for BER with loss ℓ, to excess risk for BER with 0-1 loss. If ℓ is calibrated wrt 0-1 loss (which is true for common losses like logistic), such bounds can easily be obtained for BER (see [5]). This aspect of our work is very similar to the analysis for the misclassification rate, and hence has been placed in the supplemental, although we will highlight it more prominently in the revision.

R4: Why focus on BER when experiments optimize AUC? The BER is needed for our theoretical analysis. The BER pairs naturally with MCMs, just like the misclassification rate pairs with the label flipping model for label noise [20]. BER is basically the frequentist analogue of misclassification rate, which assumes the class labels are governed by a prior distribution. The experiments use AUC because our competitors are designed wrt misclassification rate. We wanted a performance measure that did not favor one method over the other, so we chose to look at something other than BER or misclassification rate, and AUC seemed a natural choice.

R5: Extensions to multiclass / deep learning. We completely agree this is the next step. We wanted to present a thorough theoretical treatment and found it necessary to first understand the binary case. We have been working on the multiclass case and can confirm that while it is definitely a separate paper, many of the ideas from the binary case do extend with some interesting caveats.

R5: Competitors out of date. The two competitors are indeed somewhat old by ML standards, but they are, to our knowledge, the top-performing kernel methods that have appeared in mainstream machine learning publications. We will certainly develop a neural network implementation and compare to more recent methods in our future work on multiclass LLP.

R5: K-Merging schemes. Our experiment don't use K-merging schemes. We use Alg. 1, which is mentioned in line 221. How do K-merging schemes work? We would need to know more about your question. If you update your review with more details, we can address it in the next version of our paper.