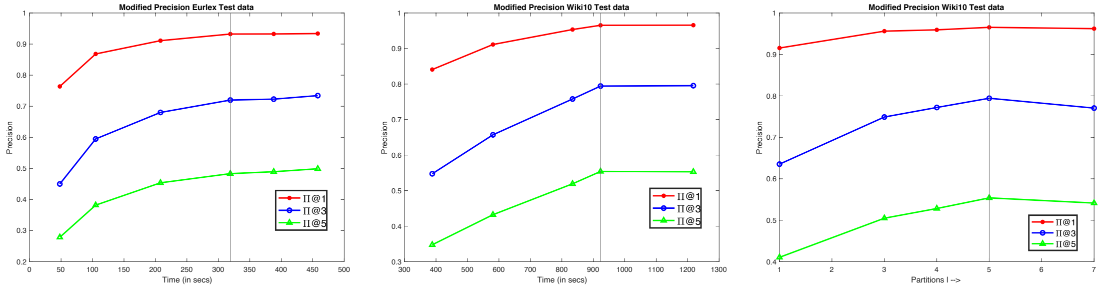
We thank the reviewers for their valuable time and thoughtful feedback. We are encouraged by the positive comments w.r.t. our novel ideas and application of group testing to multilabel classification (MLC)([R3],[R4]), our method's scalability ([R4]), impressive runtimes ([R2]), and efficiency ([R3]). Please see our response below:

[R1],[R2],[R3] **Novel Contributions:** Our main contributions are: (a) the development of a (non-trivial) data-dependent group testing scheme, that improves label grouping for MLGT significantly (vs. [35]), and can use the recently proposed log-time decoding algorithm; (b) the use of matrix reordering techniques to hierarchically partition the label space, so that we can apply MLGT to subsets of labels independently, in order to scale to very large label sets. These innovations lead to a significantly faster training algorithm (Table 3) compared to most existing methods ($\sim$ 50min vs. 370-730 hrs for DISMEC, which has the highest accuracy), yet yield comparable results. Note that (more accurate) OvA methods require $O(d)$ classifiers to be trained (taking many hours). The tree methods use k-means clustering to create label clusters (we use fast matrix reordering heuristics) and OvA classifiers at each leaf nodes (# classifiers $\sim o(d)$ vs. our $O(k \log d)$). Our method also has a provably log-time prediction algorithm, enabling almost real-time predictions.

[R1],[R2],[R3] **Similarity to [35]:** As we build on the MLGT algorithm of [35], where group testing was first proposed for MLC, the core idea of creating $O(k \log d)$ label groups is similar. However, the method in [35] yields poor accuracy on large datasets due to random label grouping, and does not scale to extreme settings. We overcome these issues by first developing a data-dependent grouping scheme (NMFGT) to improve the method's accuracy, and then use hierarchical partitioning to scale the method to very large problems. Sampling a group testing matrix that (a) captures the label correlations, (b) has distinctive columns, and (c) satisfies the SAFFRON construction, is non-trivial. We propose a technique that uses a normalized NMF basis (capturing label correlations) as a potential function for sampling columns of the GT matrix that are distinct and have fixed average sparsity c (left regular graph). Computing a good GT matrix via NMF for large d is difficult, and we use label partitioning in order to apply NMF-based MLGT to smaller problems.

[R1] **Weakness 1,2**: We develop the NMFGT and *also* the HE-NMFGT (NMFGT + hierarchical partitioning) to (a) tackle large datasets and (b) get better accuracy on large datasets, and thus describe both. **Weakness 3 - Experimental study**: We first show that NMFGT is better (See Fig 2. & suppl.) than earlier GT method in [35]. Also, SP-GT results do not match with [35] as the modified prediction algorithm in this paper is better (we get better accuracy than [35,Table 1]. We next use label partitioning to improve over NMF-GT for larger datasets (Table 2). Finally, we show that partitioning+NMFGT has *significantly faster* training and prediction times than other methods (Table 3). We believe that low training times (saving many hours) and fast predictions in return for a limited loss (few points) in accuracy will be critical in many "related search" applications. **Weakness 4**: We use HE-NMFGT only when # labels is too large to apply NMFGT. We do mention that for Mediamill and RCV1x there were no clear label partitions.

[R3], [R4] **Trade off and improvements:** We thank the reviewers for these suggestions. In the figures below we plot precisions ($\Pi@1, \Pi@3, \Pi@5$) versus runtimes (in secs) for Eurlex (left) and wiki10 (middle) datasets, by increasing # groups m in each partition. Indeed, we notice a clear trade-off: as we increase runtimes, accuracy improves. But beyond a point, the accuracy gain is limited as m is increased. In the paper, we chose smallest m (vertical line) for which our accuracy is close to the SOA tree methods. Improved accuracy can be achieved for higher runtimes (when m is much more than $k \log d$). We also plot $\Pi@k$ versus # partitions ℓ for Wiki10 (right). For smaller ℓ, it is hard to compute a good NMF for large matrices, and with many partitions, we will miss certain label correlations.



[R4] **Solution merger:** The output of MLGT will be a binary vector $\{0,1\}^{d_i}$, hence, comparing scores across disjoint subsets of labels will not an issue. For the shared labels across partitions, we indeed use weights for the label outputs such that these weights add to 1. Due to space constraints these details were only briefly discussed (in sec. 4). **Label partitions:** The matrix reordering method recursively partitions the labels, hence discovering a hierarchy. The code we use produces the partitions (and sizes), in addition to the permutations depicted in Fig. 1. So the process is automatic.

[R2] **Ensemble methods, missing details/comparisons:** The ensemble idea is an exciting direction we have not investigated! Linear SVM was used for classifiers (same as SOA tree methods). For large datasets the partition sizes were $\sim 40k$ labels. We had to defer the implementation details to supplement due to space constraints. We will include a comparison to AnnexML (Tagami, 17) if the paper is accepted.