We thank reviewers for their efforts and insightful suggestions. In this work, we proposed an all-way Boolean Tensor Decomposition (BTD) method, GETF, by incorporating the geometric property of Boolean tensor data. We are encouraged that reviewers find our theoretical driven method effective and efficient, which provides a state-of-the-art way to solve an important problem. In the following, we answer the main questions and comments from the reviewers. We will also polish the paper writing to address all the minor issues pointed out by reviewers.

To **Reviewer #1, Q1** *[...why binary representation is better than count representation...]*. We think that both binary representation and count representation have their own merits and how to select the representation should depend on applications. In this work, we tackle the tensor decomposition problem in Boolean tensors and thus casting the crime dataset as a binary representation to indicate whether a crime has happened (1) or not (0) is the best choice to demonstrate our effectiveness. The proposed methods can also be applied to more general scenarios, such as knowledge graph, contextual recommender systems, search engine and high-dimensional spatial data modeling. **Q3** *[...What is the convergence tolerance?...]*. Task completion for experiments is defined as achieving the convergence criteria, presented in Appendix line 171. And yes, the number of patterns is fixed as 5. We will polish the writing in the revised version to make it clear. **Q4** *[...the y-axis in Figures Figure 7C and 7D is the same...]* The y-axis of Fig 7C-E are not the same. Definition of crime index/dates/counts are in the Appendix 4.2. GETF distinguishes the safe and dangerous regions by reconstructing the relational patterns of crimes, which forms a Boolean relational tensor data (region-date-year), on which GETF showed better performance over baselines.

To **Reviewer #1, Q2** *[...Why not showing the scalability of GETF and other baselines...]* and **Reviewer #4, Q1** *[...No comparisons with other methods were given...]*. In Fig 6 and Appendix Fig 4, we compared GETF with SOTA baselines on reconstruction error along with the increase of dimensions on different tensor scale, density (corresponds to non-zeros, Fig 6A-B,G-H) and noise. Due to the page limit, we have to put detailed scalability experiments on real world datasets in Appendix 4.1, 4.2 and 4.3 (Appendix Figure 7,10), and we apologize for leading to this misunderstanding.

To **Reviewer #2, Q1** *[...It would be good if the paper provides theoretical analysis...]*. Thanks for the suggestion, we will work on this direction in the future.

To **Reviewer #2, Q2** *[...none of the existing algorithms are designed to handle the HBTD problem for higher order tensors...]* and **Reviewer #3, Q4** *[...While alternating optimization methods have not been written...]* Thanks for pointing this out, and we agree that LOM can be theoretically extended to higher order. However, LOM's Bayesian fitting scheme is too heavy on computational cost for most higher-order tensors. Empirically, LOM took an hour to converge on 3D data with 500 features on each dimension (Appendix Fig 5D). Increasing data dimension usually results in 2 magnitude of increase in data size, yet LOM will fail to converge. In terms of the alternative optimization, besides the high space/computation cost by Khatri-Rao product for the matricization of higher order tensor, the updating process is already above O(n) complexity on each dimensionality. For the noisy tensor, this usually results in excessive updating for this NP-hard problem. Partition the matricization could save some computations, but with a price of increased space complexity (Park et al ICDE2017). For fair comparisons, we compared our methods on 2D with MP and 3D with LOM, representing SOTA methods. But for higher order (4D, 5D), we showed the performance of GETF without comparison as all baseline methods failed to converge on such moderate sized higher order tensors.

To **Reviewer #2, Q4** *[...Does it need all permutations...]* and **Reviewer #3, Q1** *[...finding the right permutations up to a (k-1) LTL tensor...]* The existence of (k-1)-LTL IRT for any tensor is given in line 154-156, and its uniqueness is supported by Lemma 3 (proof is in Appendix). Not all permutations of IRT are needed to achieve the (k-1)-LTL tensor.

To **Reviewer #3, Q2** *[...a (k-1) LTL tensor and then its closest flat 2-LTL tensor...The existence of such tensor for any tensor is not assured]*. GETF is designed based on the geometric property of Boolean tensor, where in a flat 2-LTL tensor, the largest pattern tensor resides on the $1/k$ segmentation point (Lemma1 Fig 2). Lemma 3 and 4 indicate the (k-1)-LTL IRT is the closest form to 2-LTL IRT. Even the 2-LTL IRT does not always exist, Lemma 2 indicates when a tensor is sparse and its largest pattern tensor is distinct, the pattern tensor can be sub-optimally detected by the flat 2-LTL tensor with largest solid overlap with the unique (k-1)-LTL IRT. Noted, there are at most n flat 2-LTL tensors needed to be considered, here n is the tensor size.

To **Reviewer #2, Q3** *[I actually have doubts on the efficiency of this step]* and **Reviewer #3, Q3** *[...it may be computationally demanding...]*. For a $n = m^k$ size tensor, since (k-1)-LTL IRT is unique and there are at most n flat 2-LTL tensors to be considered, the *2_LTL_project* is $O(n)$. The complexity cost of the Pattern_fiber_finding algorithm is $\frac{m^{k+1}-m}{m-1} + kmlog(m)$. On top of the identified pattern fiber, the *Geometric_folding* (Fig 5, main 3.4, Appendix 3.6) algorithm recovered the rank 1 tensor by applying *Pattern_fiber_finding* sequencially that has a complexity cost at $\frac{m^{k+2}-m^2}{(m-1)^2} - \frac{km}{m-1} + \frac{k(k+1)}{m-1} + \frac{k(k+1)}{2}mlog(m) \sim O(m^k)$, which explained the computational efficiency of GETF. Moreover, the additional space complexity for GETF is $\frac{m^k-m}{m-1}$, also $O(n)$. We highlighted the complexity analysis in main text Section 3.5 and will provide detailed derivation of complexity in the revised Appendix.